PHP Web Toolkit 1.0.4 Alpha (phpwebtk)

Brian Bisaillon
phpwebtk-1.0.4-alpha
Sat Nov 12 2016

Table of Contents

Table of contents

Namespace Index

Namespace List

Here is a list of all namespaces with brief descriptions:

Hierarchical Index

Class Hierarchy

This inheritance list is sorted roughly, but not completely, alphabetically:

Class Index

Class List

Here are the classes, structs, unions and interfaces with brief descriptions:

File Index

File List

Here is a list of all files with brief descriptions:

Namespace Documentation

phpwebtk Namespace Reference

Detailed Description

Id PHP Web Toolkit Version 1.0.4 Alpha

class **Client**

This class is responsible for building a **Request** object and sending the **Request** object to the front controller for processing. The front controller then gives the response back to the end user.

Author:

Brian Bisaillon `bisailb@myprivacy.ca`

Copyright:

Copyright (C) 2004-2016 by Brian Bisaillon

http

class **Command**

This class declares an interface for executing an operation.

Author:

Brian Bisaillon `bisailb@myprivacy.ca`

Copyright:

Copyright (C) 2004-2016 by Brian Bisaillon

http

class **ConfigDao**

This class declares an interface for a type of Data Access Object (DAO).

Author:

Brian Bisaillon `bisailb@myprivacy.ca`

Copyright:

Copyright (C) 2004-2016 by Brian Bisaillon

databases

class **Controller**

This class serves as a single entry point for handling all requests in the system. The front controller is responsible for delegating processes to various handlers while minimizing the coupling among these components by implementing flexible request handling mechanisms, and managing the choice of the next view to present to the end user.

Author:

Brian Bisaillon `bisailb@myprivacy.ca`

Copyright:

Copyright (C) 2004-2016 by Brian Bisaillon

http

class **Convert**

This class provides some common data conversion functions.

Author:
Brian Bisaillon bisailb@myprivacy.ca

Copyright:
Copyright (C) 2004-2016 by Brian Bisaillon

conversion

class **Crypt**

This class provides a simple interface to the mcrypt library. It can be used to encrypt and decrypt data. mcrypt was chosen because it supports a wide variety of block algorithms and cipher modes. For a complete list of supported algorithms and modes refer to the documentation of mcrypt.

Author:
Brian Bisaillon bisailb@myprivacy.ca

Copyright:
Copyright (C) 2004-2016 by Brian Bisaillon

cryptography

class **DaoFactory**

This class declares an interface for operations that create abstract Data Access Objects (DAOs).

Author:
Brian Bisaillon bisailb@myprivacy.ca

Copyright:
Copyright (C) 2004-2016 by Brian Bisaillon

databases

class **Digest**

This class provides a simple interface to the mhash library. It can be used to create both salted and unsalted message digests. mhash was chosen because it supports a wide variety of hash algorithms. For a complete list of supported hashes, refer to the documentation of mhash.

Author:
Brian Bisaillon bisailb@myprivacy.ca

Copyright:
Copyright (C) 2004-2016 by Brian Bisaillon

cryptography

class **Hash**

This class provides a simple interface to the mhash library. mhash was chosen because it supports a wide variety of hash algorithms. For a complete list of supported hashes, refer to the documentation of mhash.

Author:
Brian Bisaillon bisailb@myprivacy.ca

Copyright:
Copyright (C) 2004-2016 by Brian Bisaillon

cryptography

class **Hmac**

The **Hmac** class provides a simple interface to the mhash library. It can be invoked to create both salted and unsalted hashed message authentication codes (HMAC). mhash was chosen because it supports a

wide variety of hash algorithms. For a complete list of supported algorithms, refer to the documentation of mhash.

Author:
Brian Bisaillon `bisailb@myprivacy.ca`

Copyright:
Copyright (C) 2004-2016 by Brian Bisaillon

cryptography

class **HttpRequestBuilder**

This class constructs the parts of the **Request** by implementing the **RequestBuilder** interface, defines and keeps track of the representation it creates and provides an interface for retrieving the **Request** object.

Author:
Brian Bisaillon `bisailb@myprivacy.ca`

Copyright:
Copyright (C) 2004-2016 by Brian Bisaillon

http

Id PHP Web Toolkit Version 1.0.2 Alpha

class **HttpRequestHandler**

This class handles requests it is responsible for and can access its successor. If this class can handle the request, it does so; otherwise it it forwards the request to its successor.

Author:
Brian Bisaillon `bisailb@myprivacy.ca`

Copyright:
Copyright (C) 2004-2016 by Brian Bisaillon

textprocessing

class **Invoker**

This class asks the command to carry out the request.

Author:
Brian Bisaillon `bisailb@myprivacy.ca`

Copyright:
Copyright (C) 2004-2016 by Brian Bisaillon

http

class **KsesRequestHandler**

This class handles requests it is responsible for and can access its successor. If this class can handle the request, it does so; otherwise it forwards the request to its successor.

Author:
Brian Bisaillon `bisailb@myprivacy.ca`

Copyright:
Copyright (C) 2004-2016 by Brian Bisaillon

textprocessing

class **MysqlDaoFactory**

This class implements the **DaoFactory**'s operations that create concrete MySQL Data Access Objects (DAOs).

Author:

Brian Bisaillon `bisailb@myprivacy.ca`

Copyright:

Copyright (C) 2004-2016 by Brian Bisaillon

databases

class **MysqliDaoFactory**

This class implements the **DaoFactory**'s operations that create concrete MySQL Data Access Objects (DAOs).

Author:

Brian Bisaillon `bisailb@myprivacy.ca`

Copyright:

Copyright (C) 2004-2016 by Brian Bisaillon

databases

class **MysqliSampleDao**

This class defines a Data Access Object to be created by the corresponding **MysqliDaoFactory** and implements the **SampleDao** interface.

This class contains all MySQL specific code and SQL statements. The implementation details are hidden from the end user.

Author:

Brian Bisaillon `bisailb@myprivacy.ca`

Copyright:

Copyright (C) 2004-2016 by Brian Bisaillon

databases

class **MysqlSampleDao**

This class defines a Data Access Object to be created by the corresponding **MysqlDaoFactory** and implements the **SampleDao** interface.

This class contains all MySQL specific code and SQL statements. The implementation details are hidden from the end user.

Author:

Brian Bisaillon `bisailb@myprivacy.ca`

Copyright:

Copyright (C) 2004-2016 by Brian Bisaillon

databases

class **MysqltDaoFactory**

This class implements the **DaoFactory**'s operations that create concrete MySQL Data Access Objects (DAOs).

Author:

Brian Bisaillon `bisailb@myprivacy.ca`

Copyright:

Copyright (C) 2004-2016 by Brian Bisaillon

databases

class **MysqltSampleDao**

This class defines a Data Access Object to be created by the corresponding **MysqltDaoFactory** and implements the **SampleDao** interface.

This class contains all MySQL specific code and SQL statements. The implementation details are hidden from the end user.

Author:

Brian Bisaillon `bisailb@myprivacy.ca`

Copyright:

Copyright (C) 2004-2016 by Brian Bisaillon

databases

class **PException**

This class is responsible for exception handling and inherits from the internal Exception class.

Author:

Brian Bisaillon `bisailb@myprivacy.ca`

Copyright:

Copyright (C) 2004-2016 by Brian Bisaillon

debugging

class **Postgres7DaoFactory**

This class implements the **DaoFactory**'s operations that create concrete PostgreSQL Data Access Objects (DAOs).

Author:

Brian Bisaillon `bisailb@myprivacy.ca`

Copyright:

Copyright (C) 2004-2016 by Brian Bisaillon

databases

class **Postgres8DaoFactory**

This class implements the **DaoFactory**'s operations that create concrete PostgreSQL Data Access Objects (DAOs).

Author:

Brian Bisaillon `bisailb@myprivacy.ca`

Copyright:

Copyright (C) 2004-2016 by Brian Bisaillon

databases

class **Prng**

This class provides a simple interface to two character devices, the rand() function and the mt_rand() function. The /dev/random character device is suitable for use when very high quality randomness is desired. The /dev/urandom character device will result in randomness that is merely cryptographically strong. The main difference between the two is that /dev/random is blocking and /dev/urandom is non-blocking. The rand() function uses the libc random number generator. However, mt_rand() is a drop-in replacement for rand() that uses a random number generator with known characteristics using the Mersenne Twister, that will produce randomnumbers four times faster than what the average libc rand() provides.

Author:

Brian Bisaillon `bisailb@myprivacy.ca`

Copyright:

Copyright (C) 2004-2016 by Brian Bisaillon

mathematics

class **Request**

This class represents the complex **Request** object under construction. The **RequestBuilder** builds the **Request** object's internal representation and defines the process by which it's assembled. This class includes classes that define the constituent parts, includng interfaces for assembling the parts into the final result.

Author:

Brian Bisaillon `bisailb@myprivacy.ca`

Copyright:

Copyright (C) 2004-2016 by Brian Bisaillon

http

class **RequestBuilder**

This class specifies an abstract interface for creating parts of a **Request** object.

Author:

Brian Bisaillon `bisailb@myprivacy.ca`

Copyright:

Copyright (C) 2004-2016 by Brian Bisaillon

http

class **RequestDirector**

This class constructs a **Request** object using the **RequestBuilder** interface.

Author:

Brian Bisaillon `bisailb@myprivacy.ca`

Copyright:

Copyright (C) 2004-2016 by Brian Bisaillon

http

abstract class **RequestHandler**

This class defines an interface for handling the requests and optionally implements the successor link.

Author:

Brian Bisaillon `bisailb@myprivacy.ca`

Copyright:

Copyright (C) 2004-2016 by Brian Bisaillon

textprocessing

class **SampleDao**

Declares an interface for a type of Data Access Object (DAO).

Author:

Brian Bisaillon `bisailb@myprivacy.ca`

Copyright:

Copyright (C) 2004-2016 by Brian Bisaillon

databases

class **SampleView**

This class represents and displays information to the end user. The information that is used in a dynamic display is retrieved from a model. Handlers support views by encapsulating and adapting a model for use in a display.

Author:
Brian Bisaillon `bisailb@myprivacy.ca`

Copyright:

templates

class **Session**

This class enables data to be preserved across subsequent requests invoked by the end user.

Additional measures must be taken to actively protect the integrity of a session.

Author:
Brian Bisaillon `bisailb@myprivacy.ca`

Copyright:

usermanagement

Id PHP Web Toolkit Version 1.0.3 Alpha

class **SlashesRequestHandler**

This class handles requests it is responsible for and can access its successor. If this class can handle the request, it does so; otherwise it forwards the request to its successor.

Author:
Brian Bisaillon `bisailb@myprivacy.ca`

Copyright:

textprocessing

class StreamIO

This class provides stream input and output operations and supports buffering for write operations. PHP will search for a protocol handler (also known as a wrapper) for schemes in the form of "scheme://..." unless URL-aware fopen wrappers and other wrappers are disabled.

Author:
Brian Bisaillon `bisailb@myprivacy.ca`

Copyright:

filesandfolders

class **View**

This class knows how to perform the operation(s) associated with carrying out the request.

Author:
Brian Bisaillon `bisailb@myprivacy.ca`

Copyright:

http

class **ViewCommand**

This class defines a binding between a **View** object and an action, and implements Execute by invoking the corresponding operation(s) on the **View**.

Author:

Brian Bisaillon `bisailb@myprivacy.ca`

Copyright:

Copyright (C) 2004-2016 by Brian Bisaillon

http

class **XmlConfigDao**

This class defines a Data Access Object (DAO) to be created by the corresponding **XmlDaoFactory** and implements the **ConfigDao** interface.

This class contains all XML specific code and XPath statements. implementation details are hidden from the end user.

Author:

Brian Bisaillon `bisailb@myprivacy.ca`

Copyright:

Copyright (C) 2004-2016 by Brian Bisaillon

databases

class **XmlDaoFactory**

This class implements the DAO Factory's operations that create concrete XML Data Access Objects (DAOs).

Author:

Brian Bisaillon `bisailb@myprivacy.ca`

Copyright:

Copyright (C) 2004-2016 by Brian Bisaillon

databases

Class Documentation

Client Class Reference

Public Member Functions

- **SendRequest** ()

Member Function Documentation

Client::SendRequest ()

function SendRequest

This method builds and sends the **Request** object to the front controller via the ProcessRequest method for processing. The front controller then gives a response back to the end user.

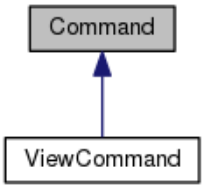
public

The documentation for this class was generated from the following file:

- **client.class.php**

Command Class Reference

Inheritance diagram for Command:



Public Member Functions

- **Execute (Request $Request)**

Member Function Documentation

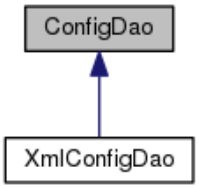
Command::Execute (Request ***$Request*****)** `[abstract]`

The documentation for this class was generated from the following file:

- **command.class.php**

ConfigDao Class Reference

Inheritance diagram for ConfigDao:



Public Member Functions

- **GetElementsByPath** ($expression)
- **SetElementByPath** ($expression, $data, $fileName)

Member Function Documentation

ConfigDao::GetElementsByPath (*$expression*) `[abstract]`

ConfigDao::SetElementByPath (*$expression*, *$data*, *$fileName*) `[abstract]`

The documentation for this class was generated from the following file:

- **configdao.class.php**

Controller Class Reference

Public Member Functions

- **ProcessRequest** (**Request** $**Request**)

Member Function Documentation

Controller::ProcessRequest (Request *$Request*)

function ProcessRequest

This method initiates a session to preserve specific information across subsequent requests; adds or strips slashes from the HTTP GET or HTTP POST information; executes customized operations; and transfers the modified **Request** object to the Dispatch method along with the name of the view to present to the end user.

public

Parameters:

Request	**Request** object

The documentation for this class was generated from the following file:

- **controller.class.php**

Convert Class Reference

Public Member Functions

- **Hex2Bin** ($hexData)

Member Function Documentation

Convert::Hex2Bin (*$hexData*)

function Hex2Bin

This method converts hexadecimal data into a binary representation.

public

Parameters:

hexData	Hexadecimal encoded data

Returns:

string Binary encoded data

The documentation for this class was generated from the following file:

- **convert.class.php**

Crypt Class Reference

Public Member Functions

- **__construct** ()
- **SetEncryptionKey** ($plaintext)
- **Encrypt** ($plaintext)
- **Decrypt** ($ciphertext)

Constructor & Destructor Documentation

Crypt::__construct ()

function __construct

This method is executed when an object is instantiated from this class. Preprocessing can be done here before the object is put into service.

public

Member Function Documentation

Crypt::Decrypt (*$ciphertext*)

function Decrypt

This method initializes all buffers needed for decryption, decrypts data and deinitializes a decryption module.

public

Parameters:

ciphertext	Ciphertext

Returns:

string Stripped plaintext

Crypt::Encrypt (*$plaintext*)

function Encrypt

This method initializes all buffers needed for encryption, encrypts data and deinitializes an encryption module.

public

Parameters:

plaintext	Plaintext

Returns:

string Hexadecimal encoded ciphertext

Crypt::SetEncryptionKey (*$plaintext*)

function SetEncryptionKey

This method creates a encryption key according to the maximum supported key size of the opened mode and stores it in the XML configuration file.

public

Parameters:

plaintext	Plaintext password

Returns:

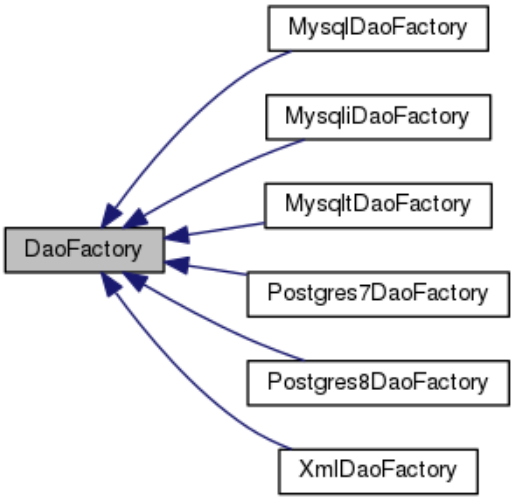
string Encryption key

The documentation for this class was generated from the following file:

- **crypt.class.php**

DaoFactory Class Reference

Inheritance diagram for DaoFactory:



Static Public Member Functions

- static **GetDaoFactory** ($driver)

Member Function Documentation

static DaoFactory::GetDaoFactory (*$driver*) `[static]`

function GetDaoFactory

This method builds a Data Source Name (DSN) and invokes the appropriate DAO Factory with the DSN as a parameter.

public

Returns:

mixed object instance|FALSE

The documentation for this class was generated from the following file:

- **daofactory.class.php**

Digest Class Reference

Public Member Functions

- **__construct** ()
- **GetDigest** ($plaintext)
- **IsValidDigest** ($ciphertext, $plaintext)

Constructor & Destructor Documentation

Digest::__construct ()

function __construct

This method is executed when an object is instantiated from this class. Preprocessing can be done here before the object is put into service.

public

Member Function Documentation

Digest::GetDigest (*$plaintext*)

function GetDigest

This method generates a salted or unsalted message digest.

public

Parameters:

plaintext	Plaintext

Returns:

string Base64 encoded ciphertext

Digest::IsValidDigest (*$ciphertext*, *$plaintext*)

function IsValidDigest

This method validates a salted or unsalted message digest.

public

Parameters:

ciphertext	Ciphertext
plaintext	Plaintext

Returns:

boolean TRUE|FALSE

The documentation for this class was generated from the following file:

- **digest.class.php**

Hash Class Reference

Public Member Functions

- **GetHashInfo** ()
- **GetBlockSize** ($hashId)

Static Public Member Functions

- **GetSalt** ($bytes, $source)
- **CompareCiphertextData** ($ciphertext, $genCiphertext)

Member Function Documentation

Hash::CompareCiphertextData (*$ciphertext*, *$genCiphertext*) `[static]`

function CompareCiphertextData

This method compares the original ciphertext to the generated ciphertext.

public

Parameters:

ciphertext	Original ciphertext
genCiphertext	Generated ciphertext

Returns:

boolean TRUE|FALSE

Hash::GetBlockSize (*$hashId*)

function GetBlockSize

This method retrieves the block size of the specified hash.

public

Parameters:

hashId	**Hash** identifier

Returns:

mixed Block size|FALSE

Hash::GetHashInfo ()

function GetHashInfo

This method displays the hash id, the algorithm and the block size for each hash algorithm supported by the mhash library.

public

Returns:

string **Hash** identifier, algorithm and output size

Hash::GetSalt (*$bytes*, *$source*) `[static]`

function GetSalt

This method retrieves random bits from a Pseudo-Random Number Generator (PRNG).

public

Parameters:

bytes	Size of salt in bytes
source	Random source of entropy

Returns:

string Binary encoded salt

The documentation for this class was generated from the following file:

- **hash.class.php**

Hmac Class Reference

Public Member Functions

- **__construct** ()
- **SetHmacKey** ($plaintext)
- **GetHmac** ($plaintext)
- **IsValidHmac** ($ciphertext, $plaintext)

Constructor & Destructor Documentation

Hmac::__construct ()

function __construct

This method is executed when an object is instantiated from this class. Preprocessing can be done here before the object is put into service.

public

Member Function Documentation

Hmac::GetHmac (*$plaintext*)

function GetHmac

The GetHmac method gets random bits from the Pseudo-Random Number Generator (PRNG), invokes the Salted S2k algorithm to further randomize the salt, hashes the plaintext including the salt and appends the salt to the end of the resultant ciphertext.

public

Parameters:

plaintext	Plaintext password

Returns:

string Base64 encoded ciphertext

Hmac::IsValidHmac (*$ciphertext*, *$plaintext*)

function IsValidHmac

The isValidHmac method validates a salted or unsalted message authentication code (HMAC).

public

Parameters:

ciphertext	Ciphertext
plaintext	Plaintext

Returns:

boolean TRUE|FALSE

Hmac::SetHmacKey (*$plaintext*)

function SetHmacKey

The SetHmacKey method gets random bits from the Pseudo-Random Number Generator (PRNG), invokes the Salted S2K algorithm to further randomize the salt, hashes the plaintext including the salt to create an HMAC key and stores it in the XML configuration file.

public

Parameters:

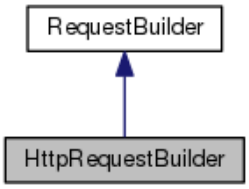
plaintext	Plaintext password

The documentation for this class was generated from the following file:

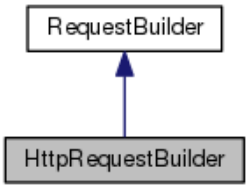
- **hmac.class.php**

HttpRequestBuilder Class Reference

Inheritance diagram for HttpRequestBuilder:



Collaboration diagram for HttpRequestBuilder:



Public Member Functions

- **__construct** ()
- **BuildHttpAccept** ()
- **BuildHttpAcceptCharset** ()
- **BuildHttpAcceptEncoding** ()
- **BuildHttpAcceptLanguage** ()
- **BuildHttpConnection** ()
- **BuildHttpGet** ()
- **BuildHttpHost** ()
- **BuildHttpPost** ()
- **BuildHttpReferer** ()
- **BuildHttpUserAgent** ()
- **BuildQueryString** ()
- **BuildRemoteAddress** ()
- **BuildRemoteHost** ()
- **BuildRemotePort** ()
- **BuildRemoteProxyAddr** ()
- **BuildRemoteProxyHost** ()
- **BuildRequestMethod** ()
- **BuildRequestUri** ()
- **BuildServerProtocol** ()
- **GetRequest** ()

Static Public Member Functions

- **GetInstance** ()

Constructor & Destructor Documentation

HttpRequestBuilder::__construct ()

function __construct

This method is executed when an object is instantiated from this class. Preprocessing can be done here before the object is put into service.

public

Member Function Documentation

HttpRequestBuilder::BuildHttpAccept ()

function BuildHttpAccept

This method sets the contents of the Accept: header from the current request, if there is one.

public

HttpRequestBuilder::BuildHttpAcceptCharset ()

function BuildHttpAcceptCharset

This method sets the contents of the Accept-Charset: header from the current request, if there is one.

public

HttpRequestBuilder::BuildHttpAcceptEncoding ()

function BuildHttpAcceptEncoding

This method sets the contents of the Accept-Encoding: header from the current request, if there is one.

public

HttpRequestBuilder::BuildHttpAcceptLanguage ()

function BuildHttpAcceptLanguage

This method sets the contents of the Accept-Language: header from the current request, if there is one.

public

HttpRequestBuilder::BuildHttpConnection ()

function BuildHttpConnection

This method sets the contents of the Connection: header from the current request, if there is one.

public

HttpRequestBuilder::BuildHttpGet ()

function BuildHttpGet

This method sets variables provided to the script via HTTP GET.

public

HttpRequestBuilder::BuildHttpHost ()

function BuildHttpHost

This method sets the contents of the Host: header from the current request, if there is one.

public

HttpRequestBuilder::BuildHttpPost ()

function BuildHttpPost

This method sets variables provided to the script via HTTP POST.

public

HttpRequestBuilder::BuildHttpReferer ()

function BuildHttpReferer

This method sets the address of the page (if any) which referred the user agent to the current page. This is set by the user agent. Not all user agents will set this, and some provide the ability to modify HTTP_REFERER as a feature. In short, it cannot really be trusted.

public

HttpRequestBuilder::BuildHttpUserAgent ()

function BuildHttpUserAgent

This method sets the contents of the User-Agent: header from the current request, if there is one. This is a string denoting the user agent being which is accessing the page.

public

HttpRequestBuilder::BuildQueryString ()

function BuildQueryString

This method sets the query string, if any, via which the page was accessed.

public

HttpRequestBuilder::BuildRemoteAddress ()

function BuildRemoteAddress

This method sets the IP address from which the user is viewing the current page.

public

HttpRequestBuilder::BuildRemoteHost ()

function BuildRemoteHost

This method sets the Host name from which the user is viewing the current page. The reverse dns lookup is based off the REMOTE_ADDR of the user.

public

HttpRequestBuilder::BuildRemotePort ()

function BuildRemotePort

This method sets the port being used on the user's machine to communicate with the web server.

public

HttpRequestBuilder::BuildRemoteProxyAddr ()

function BuildRemoteProxyAddr

This method sets the proxy IP address from which the user is being forwarded.

public

HttpRequestBuilder::BuildRemoteProxyHost ()

function BuildRemoteProxyHost

This method sets the Host name from which the user is being forwarded. The reverse dns lookup is based off the REMOTE_PROXY_ADDR of the user.

public

HttpRequestBuilder::BuildRequestMethod ()

function BuildRequestMethod

This method sets the request method that was used to access the page.

public

HttpRequestBuilder::BuildRequestUri ()

function BuildRequestUri

This method retreives the URI which was given in order to access this page.

public

HttpRequestBuilder::BuildServerProtocol ()

function BuildServerProtocol

This method sets the name and revision of the information protocol via which the page was requested.

public

HttpRequestBuilder::GetInstance () `[static]`

function GetInstance

This method instantiates a new object from this class; more specifically, it's a singleton instance.

public

Returns:

HttpRequestBuilder object instance

HttpRequestBuilder::GetRequest ()

function GetRequest

This method returns the **Request** object to the calling method.

public

Returns:

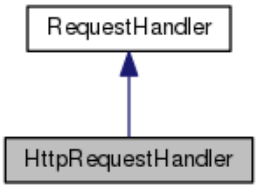
Request object instance

The documentation for this class was generated from the following file:

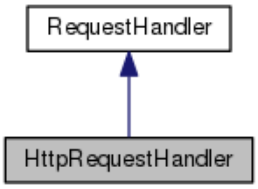
- **httprequestbuilder.class.php**

HttpRequestHandler Class Reference

Inheritance diagram for HttpRequestHandler:



Collaboration diagram for HttpRequestHandler:



Public Member Functions

- **HandleRequest (Request $Request)**

Additional Inherited Members

Member Function Documentation

HttpRequestHandler::HandleRequest (Request *$Request*)

function HandleRequest

This method processes the request if certain conditions are met.

public

Parameters:

Request	**Request** object

The documentation for this class was generated from the following file:

- **httprequesthandler.class.php**

Invoker Class Reference

Public Member Functions

- **SetCommand** (**Command** $**Command**)
- **ExecuteCommand** (**Request** $**Request**)

Member Function Documentation

Invoker::ExecuteCommand (Request *$Request*)

function SetCommand

This method executes the command and returns the result to the receiver.

private

Invoker::SetCommand (Command *$Command*)

function SetCommand

This method sets the command.

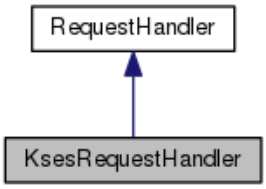
private

The documentation for this class was generated from the following file:

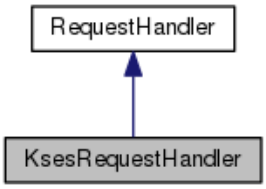
- **invoker.class.php**

KsesRequestHandler Class Reference

Inheritance diagram for KsesRequestHandler:



Collaboration diagram for KsesRequestHandler:



Public Member Functions

- **BasicTags** (Kses5 $Kses5)
- **CharacterFormatTags** (Kses5 $Kses5)
- **OutputTags** (Kses5 $Kses5)
- **BlockTags** (Kses5 $Kses5)
- **LinkTags** (Kses5 $Kses5)
- **FrameTags** (Kses5 $Kses5)
- **InputTags** (Kses5 $Kses5)
- **ListTags** (Kses5 $Kses5)
- **ImageTags** (Kses5 $Kses5)
- **TableTags** (Kses5 $Kses5)
- **StyleTags** (Kses5 $Kses5)
- **MetaInformationTags** (Kses5 $Kses5)
- **ProgrammingTags** (Kses5 $Kses5)
- **HandleRequest** (**Request** $**Request**)

Additional Inherited Members

Member Function Documentation

KsesRequestHandler::BasicTags (Kses5 *$Kses5*)

function BasicTags

This method allows and/or disallows XHTML's basic tags.

public

KsesRequestHandler::BlockTags (Kses5 *$Kses5*)

function BlockTags

This method allows and/or disallows XHTML's block tags.

public

KsesRequestHandler::CharacterFormatTags (Kses5 *$Kses5*)

function CharacterFormatTags

This method allows and/or disallows XHTML's character format tags.

public

KsesRequestHandler::FrameTags (Kses5 *$Kses5*)

function FrameTags

This method allows and/or disallows XHTML's frame tags.

public

KsesRequestHandler::HandleRequest (Request *$Request*)

function HandleRequest

This method processes the request if certain conditions are met.

public

Parameters:

Request	**Request** object

KsesRequestHandler::ImageTags (Kses5 *$Kses5*)

function ImageTags

This method allows and/or disallows XHTML's image tags.

public

KsesRequestHandler::InputTags (Kses5 *$Kses5*)

function InputTags

This method allows and/or disallows XHTML's input tags.

public

KsesRequestHandler::LinkTags (Kses5 *$Kses5*)

function LinkTags

This method allows and/or disallows XHTML's link tags.

public

KsesRequestHandler::ListTags (Kses5 *$Kses5*)

function ListTags

This method allows and/or disallows XHTML's list tags.

public

KsesRequestHandler::MetaInformationTags (Kses5 *$Kses5*)

function MetaInformationTags

This method allows and/or disallows XHTML's meta information tags.

public

KsesRequestHandler::OutputTags (Kses5 *$Kses5*)

function OutputTags

This method allows and/or disallows XHTML's output tags.

public

KsesRequestHandler::ProgrammingTags (Kses5 *$Kses5*)

function ProgrammingTags

This method allows and/or disallows XHTML's programming tags.

public

KsesRequestHandler::StyleTags (Kses5 *$Kses5*)

function StyleTags

This method allows and/or disallows XHTML's style tags.

public

KsesRequestHandler::TableTags (Kses5 *$Kses5*)

function TableTags

This method allows and/or disallows XHTML's table tags.

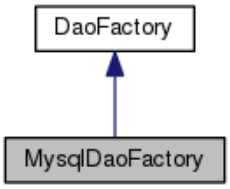
public

The documentation for this class was generated from the following file:

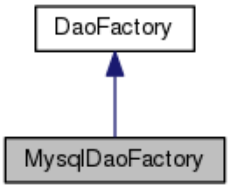
- **ksesrequesthandler.class.php**

MysqlDaoFactory Class Reference

Inheritance diagram for MysqlDaoFactory:



Collaboration diagram for MysqlDaoFactory:



Public Member Functions

- **__construct** ()
- **GetSampleDao** ()

Static Public Member Functions

- static **GetInstance** ()
- static **CreateConnection** ($dsn)

Constructor & Destructor Documentation

MysqlDaoFactory::__construct ()

function __construct

This method is executed when an object is instantiated from this class. Preprocessing can be done here before the object is put into service.

public

Member Function Documentation

static MysqlDaoFactory::CreateConnection (*$dsn*)`[static]`

function CreateConnection

This method creates a database connection object using the provided Data Source Name (DSN).

public

Parameters:

dsn	Data Source Name (DSN)

Returns:

Database object instance

static MysqlDaoFactory::GetInstance () `[static]`

function GetInstance

This method instantiates a new object from this class; more specifically, it's a singleton instance.

public

Returns:

MysqlDaoFactory object instance

MysqlDaoFactory::GetSampleDao ()

function GetSampleDao

This method creates a new object of class **MysqlSampleDao**.

public

Returns:

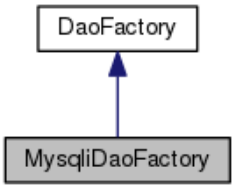
MysqlSampleDao object instance

The documentation for this class was generated from the following file:

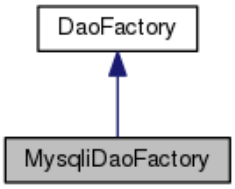
- **mysqldaofactory.class.php**

MysqliDaoFactory Class Reference

Inheritance diagram for MysqliDaoFactory:



Collaboration diagram for MysqliDaoFactory:



Public Member Functions

- **__construct** ()
- **GetSampleDao** ()

Static Public Member Functions

- static **GetInstance** ()
- static **CreateConnection** ($dsn)

Constructor & Destructor Documentation

MysqliDaoFactory::__construct ()

function __construct

This method is executed when an object is instantiated from this class. Preprocessing can be done here before the object is put into service.

public

Member Function Documentation

static MysqliDaoFactory::CreateConnection (*$dsn*) `[static]`

function CreateConnection

This method creates a database connection object using the provided Data Source Name (DSN).

public

Parameters:

dsn	Data Source Name (DSN)

Returns:

Database object instance

static MysqliDaoFactory::GetInstance () `[static]`

function GetInstance

This method instantiates a new object from this class; more specifically, it's a singleton instance.

public

Returns:

MysqliDaoFactory object instance

MysqliDaoFactory::GetSampleDao ()

function GetSampleDao

This method creates a new object of class **MysqliSampleDao**.

public

Returns:

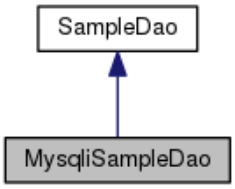
MysqliSampleDao object instance

The documentation for this class was generated from the following file:

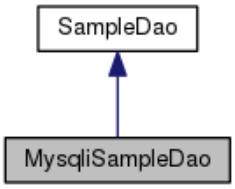
- **mysqlidaofactory.class.php**

MysqliSampleDao Class Reference

Inheritance diagram for MysqliSampleDao:



Collaboration diagram for MysqliSampleDao:



Public Member Functions

- **__construct** ($dsn)
- **InsertSample** ()
- **DeleteSample** ()
- **FindSample** ()
- **UpdateSample** ()
- **SelectSampleRS** ()
- **SelectSampleTO** ()
- **GetData** ($RecordSet)
- **GetDataTO** ($RecordSet)

Constructor & Destructor Documentation

MysqliSampleDao::__construct (*$dsn*)

function __construct

This method is executed when an object is instantiated from this class. Preprocessing can be done here before the object is put into service.

public

Member Function Documentation

MysqliSampleDao::DeleteSample ()

function DeleteSample

This method retrieves a database connection object, starts a transaction, executes SQL delete statements and catches any exceptions thrown.

public

MysqliSampleDao::FindSample ()

function FindSample

This method retrieves a database connection object, executes SQL select statements, parses recordsets and catches any exceptions thrown.

public

MysqliSampleDao::GetData (*$RecordSet*)

function GetData

This method uses PEAR style data retrieval to retrieve arrays containing the current row. FetchRow() internally moves to the next record after returning the current row.

public

MysqliSampleDao::GetDataTO (*$RecordSet*)

function GetDataTO

This method uses PEAR style data retrieval to retrieve the current row as an object. FetchNextObject() internally moves to the next row automatically.

public

MysqliSampleDao::InsertSample ()

function InsertSample

This method retrieves a database connection object, starts a transaction, executes SQL insert statements and catches any exceptions thrown.

public

MysqliSampleDao::SelectSampleRS ()

function SelectSampleRS

This method retrieves a database connection object, executes SQL select statements, parses recordsets and catches any exceptions thrown.

public

MysqliSampleDao::SelectSampleTO ()

function SelectSampleTO

This method retrieves a database connection object, executes SQL select statements, parses recordsets and catches any exceptions thrown.

public

MysqliSampleDao::UpdateSample ()

function UpdateSample

This method retrieves a database connection object, starts a transaction, executes SQL update statements and catches any exceptions thrown.

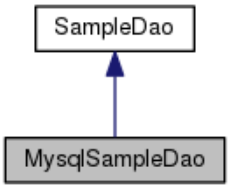
public

The documentation for this class was generated from the following file:

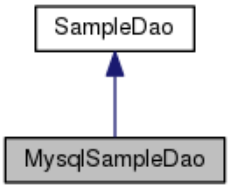
- **mysqlisampledao.class.php**

MysqlSampleDao Class Reference

Inheritance diagram for MysqlSampleDao:



Collaboration diagram for MysqlSampleDao:



Public Member Functions

- **__construct** ($dsn)
- **InsertSample** ()
- **DeleteSample** ()
- **FindSample** ()
- **UpdateSample** ()
- **SelectSampleRS** ()
- **SelectSampleTO** ()
- **GetData** ($RecordSet)
- **GetDataTO** ($RecordSet)

Constructor & Destructor Documentation

MysqlSampleDao::__construct (*$dsn*)

function __construct

This method is executed when an object is instantiated from this class. Preprocessing can be done here before the object is put into service.

public

Member Function Documentation

MysqlSampleDao::DeleteSample ()

function DeleteSample

This method retrieves a database connection object, starts a transaction, executes SQL delete statements and catches any exceptions thrown.

public

MysqlSampleDao::FindSample ()

function FindSample

This method retrieves a database connection object, executes SQL select statements, parses recordsets and catches any exceptions thrown.

public

MysqlSampleDao::GetData (*$RecordSet*)

function GetData

This method uses PEAR style data retrieval to retrieve arrays containing the current row. FetchRow() internally moves to the next record after returning the current row.

public

MysqlSampleDao::GetDataTO (*$RecordSet*)

function GetDataTO

This method uses PEAR style data retrieval to retrieve the current row as an object. FetchNextObject() internally moves to the next row automatically.

public

MysqlSampleDao::InsertSample ()

function InsertSample

This method retrieves a database connection object, starts a transaction, executes SQL insert statements and catches any exceptions thrown.

public

MysqlSampleDao::SelectSampleRS ()

function SelectSampleRS

This method retrieves a database connection object, executes SQL select statements, parses recordsets and catches any exceptions thrown.

public

MysqlSampleDao::SelectSampleTO ()

function SelectSampleTO

This method retrieves a database connection object, executes SQL select statements, parses recordsets and catches any exceptions thrown.

public

MysqlSampleDao::UpdateSample ()

function UpdateSample

This method retrieves a database connection object, starts a transaction, executes SQL update statements and catches any exceptions thrown.

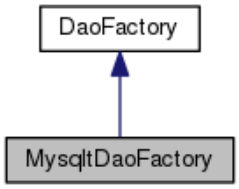
public

The documentation for this class was generated from the following file:

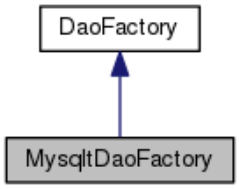
- **mysqlsampledao.class.php**

MysqltDaoFactory Class Reference

Inheritance diagram for MysqltDaoFactory:



Collaboration diagram for MysqltDaoFactory:



Public Member Functions

- **__construct** ()
- **GetSampleDao** ()

Static Public Member Functions

- static **GetInstance** ()
- static **CreateConnection** ($dsn)

Constructor & Destructor Documentation

MysqltDaoFactory::__construct ()

function __construct

This method is executed when an object is instantiated from this class. Preprocessing can be done here before the object is put into service.

public

Member Function Documentation

static MysqltDaoFactory::CreateConnection (*$dsn*) `[static]`

function CreateConnection

This method creates a database connection object using the provided Data Source Name (DSN).

public

Parameters:

dsn	Data Source Name (DSN)

Returns:

Database object instance

static MysqltDaoFactory::GetInstance () `[static]`

function GetInstance

This method instantiates a new object from this class; more specifically, it's a singleton instance.

public

Returns:

MysqltDaoFactory object instance

MysqltDaoFactory::GetSampleDao ()

function GetSampleDao

This method creates a new object of class **MysqltSampleDao**.

public

Returns:

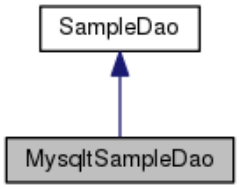
MysqltSampleDao object instance

The documentation for this class was generated from the following file:

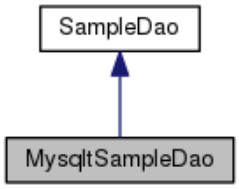
- **mysqltdaofactory.class.php**

MysqltSampleDao Class Reference

Inheritance diagram for MysqltSampleDao:



Collaboration diagram for MysqltSampleDao:



Public Member Functions

- **__construct** ($dsn)
- **InsertSample** ()
- **DeleteSample** ()
- **FindSample** ()
- **UpdateSample** ()
- **SelectSampleRS** ()
- **SelectSampleTO** ()
- **GetData** ($RecordSet)
- **GetDataTO** ($RecordSet)

Constructor & Destructor Documentation

MysqltSampleDao::__construct (*$dsn*)

function __construct

This method is executed when an object is instantiated from this class. Preprocessing can be done here before the object is put into service.

public

Member Function Documentation

MysqltSampleDao::DeleteSample ()

function DeleteSample

This method retrieves a database connection object, starts a transaction, executes SQL delete statements and catches any exceptions thrown.

public

MysqltSampleDao::FindSample ()

function FindSample

This method retrieves a database connection object, executes SQL select statements, parses recordsets and catches any exceptions thrown.

public

MysqltSampleDao::GetData (*$RecordSet*)

function GetData

This method uses PEAR style data retrieval to retrieve arrays containing the current row. FetchRow() internally moves to the next record after returning the current row.

public

MysqltSampleDao::GetDataTO (*$RecordSet*)

function GetDataTO

This method uses PEAR style data retrieval to retrieve the current row as an object. FetchNextObject() internally moves to the next row automatically.

public

MysqltSampleDao::InsertSample ()

function InsertSample

This method retrieves a database connection object, starts a transaction, executes SQL insert statements and catches any exceptions thrown.

public

MysqltSampleDao::SelectSampleRS ()

function SelectSampleRS

This method retrieves a database connection object, executes SQL select statements, parses recordsets and catches any exceptions thrown.

public

MysqltSampleDao::SelectSampleTO ()

function SelectSampleTO

This method retrieves a database connection object, executes SQL select statements, parses recordsets and catches any exceptions thrown.

public

MysqltSampleDao::UpdateSample ()

function UpdateSample

This method retrieves a database connection object, starts a transaction, executes SQL update statements and catches any exceptions thrown.

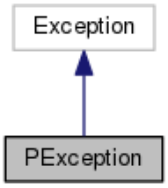
public

The documentation for this class was generated from the following file:

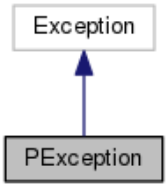
- **mysqltsampledao.class.php**

PException Class Reference

Inheritance diagram for PException:



Collaboration diagram for PException:



Static Public Member Functions

- static **Display** (Exception $Exception)

Member Function Documentation

static PException::Display (Exception *$Exception*) `[static]`

function Display

This method formats the error message appropriately and displays it.

public

Parameters:

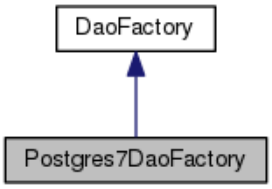
Exception	Exception object

The documentation for this class was generated from the following file:

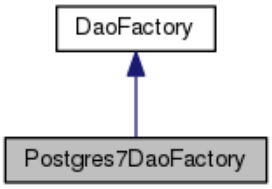
- **pexception.class.php**

Postgres7DaoFactory Class Reference

Inheritance diagram for Postgres7DaoFactory:



Collaboration diagram for Postgres7DaoFactory:



Public Member Functions

- **__construct** ()
- **GetSampleDao** ()

Static Public Member Functions

- static **GetInstance** ()
- static **CreateConnection** ($dsn)

Constructor & Destructor Documentation

Postgres7DaoFactory::__construct ()

function __construct

This method is executed when an object is instantiated from this class. Preprocessing can be done here before the object is put into service.

public

Member Function Documentation

static Postgres7DaoFactory::CreateConnection (*$dsn*) `[static]`

function CreateConnection

This method creates a database connection object using the provided Data Source Name (DSN).

public

Parameters:

dsn	Data Source Name (DSN)

Returns:

Database object instance

static Postgres7DaoFactory::GetInstance () `[static]`

function GetInstance

This method instantiates a new object from this class; more specifically, it's a singleton instance.

public

Returns:

Postgres7DaoFactory object instance

Postgres7DaoFactory::GetSampleDao ()

function GetSampleDao

This method creates a new object of class Postgres7SampleDao.

public

Returns:

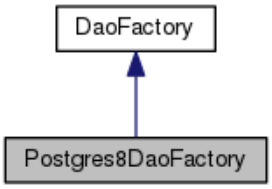
Postgres7SampleDao object instance

The documentation for this class was generated from the following file:

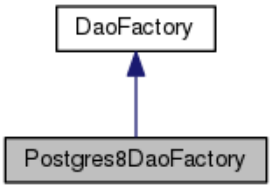
- **postgres7daofactory.class.php**

Postgres8DaoFactory Class Reference

Inheritance diagram for Postgres8DaoFactory:



Collaboration diagram for Postgres8DaoFactory:



Public Member Functions

- **__construct** ()
- **GetSampleDao** ()

Static Public Member Functions

- static **GetInstance** ()
- static **CreateConnection** ($dsn)

Constructor & Destructor Documentation

Postgres8DaoFactory::__construct ()

function __construct

This method is executed when an object is instantiated from this class. Preprocessing can be done here before the object is put into service.

public

Member Function Documentation

static Postgres8DaoFactory::CreateConnection (*$dsn*) `[static]`

function CreateConnection

This method creates a database connection object using the provided Data Source Name (DSN).

public

Parameters:

dsn	Data Source Name (DSN)

Returns:

Database object instance

static Postgres8DaoFactory::GetInstance () `[static]`

function GetInstance

This method instantiates a new object from this class; more specifically, it's a singleton instance.

public

Returns:

Postgres8DaoFactory object instance

Postgres8DaoFactory::GetSampleDao ()

function GetSampleDao

This method creates a new object of class Postgres8SampleDao.

public

Returns:

Postgres8SampleDao object instance

The documentation for this class was generated from the following file:

- **postgres8daofactory.class.php**

Prng Class Reference

Public Member Functions

- **GetPseudoRandomValue** ($source, $length=8)

Member Function Documentation

Prng::GetPseudoRandomValue (*$source*, *$length* = 8)

function **GetPseudoRandomValue()**

This method retrieves random bits of entropy using a Pseudo-Random Number Generator (PRNG) device or function. The format of the random bits is determined by first converting them to hexadecimal format and then converting them to decimal format byte by byte for /dev/random and /dev/urandom. Furthermore, since values are converted from binary to decimal one at a time, the RAND_MAX (2147483647) constraint does not limit our ability to generate very long random numbers.

public

Parameters:

source	Random source of entropy
length	Length of entropy in bytes

Returns:

mixed Random number|FALSE

The documentation for this class was generated from the following file:

- **prng.class.php**

Request Class Reference

Static Public Member Functions

- **SetProperty** ($propertyName, $requestPart)

Member Function Documentation

Request::SetProperty (***$propertyName*****,** ***$requestPart*****)** `[static]`

function SetProperty

This method sets dynamic properties of this class.

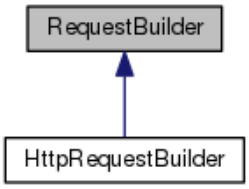
public

The documentation for this class was generated from the following file:

- **request.class.php**

RequestBuilder Class Reference

Inheritance diagram for RequestBuilder:



Public Member Functions

- **BuildHttpAccept** ()
- **BuildHttpAcceptCharset** ()
- **BuildHttpAcceptEncoding** ()
- **BuildHttpAcceptLanguage** ()
- **BuildHttpConnection** ()
- **BuildHttpGet** ()
- **BuildHttpHost** ()
- **BuildHttpPost** ()
- **BuildHttpReferer** ()
- **BuildHttpUserAgent** ()
- **BuildQueryString** ()
- **BuildRemoteAddress** ()
- **BuildRemoteHost** ()
- **BuildRemotePort** ()
- **BuildRemoteProxyAddr** ()
- **BuildRemoteProxyHost** ()
- **BuildRequestMethod** ()
- **BuildRequestUri** ()
- **BuildServerProtocol** ()
- **GetRequest** ()

Member Function Documentation

RequestBuilder::BuildHttpAccept () `[abstract]`

RequestBuilder::BuildHttpAcceptCharset () `[abstract]`

RequestBuilder::BuildHttpAcceptEncoding () `[abstract]`

RequestBuilder::BuildHttpAcceptLanguage () `[abstract]`

RequestBuilder::BuildHttpConnection () `[abstract]`

RequestBuilder::BuildHttpGet () `[abstract]`

RequestBuilder::BuildHttpHost () `[abstract]`

RequestBuilder::BuildHttpPost () `[abstract]`

RequestBuilder::BuildHttpReferer () `[abstract]`

RequestBuilder::BuildHttpUserAgent () `[abstract]`

RequestBuilder::BuildQueryString () `[abstract]`

RequestBuilder::BuildRemoteAddress () `[abstract]`

RequestBuilder::BuildRemoteHost () `[abstract]`

RequestBuilder::BuildRemotePort () `[abstract]`

RequestBuilder::BuildRemoteProxyAddr () `[abstract]`

RequestBuilder::BuildRemoteProxyHost () `[abstract]`

RequestBuilder::BuildRequestMethod () `[abstract]`

RequestBuilder::BuildRequestUri () `[abstract]`

RequestBuilder::BuildServerProtocol () `[abstract]`

RequestBuilder::GetRequest () `[abstract]`

The documentation for this class was generated from the following file:

- **requestbuilder.class.php**

RequestDirector Class Reference

Public Member Functions

- **ConstructRequest** (**RequestBuilder** $**RequestBuilder**)

Member Function Documentation

RequestDirector::ConstructRequest (RequestBuilder *$RequestBuilder*)

function ConstructRequest

This method assembles the specified parts (below) of the **Request**.

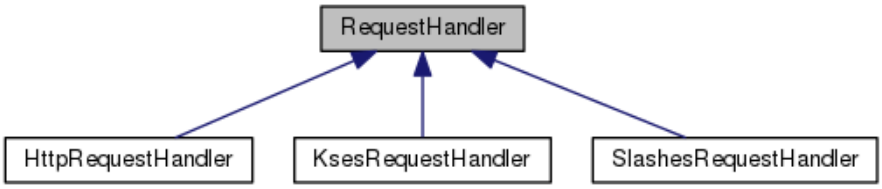
public

The documentation for this class was generated from the following file:

- **requestdirector.class.php**

RequestHandler Class Reference

Inheritance diagram for RequestHandler:



Public Member Functions

- **HandleRequest** (**Request** $**Request**)
- **SetSuccessor** (**RequestHandler** $Successor)

Protected Attributes

- **$Successor**

Member Function Documentation

RequestHandler::HandleRequest (Request *$Request*) `[abstract]`

RequestHandler::SetSuccessor (RequestHandler *$Successor*)

function SetSuccessor

This method sets the Successor.

public

Parameters:

object	*$Successor*	object of type Handler

Member Data Documentation

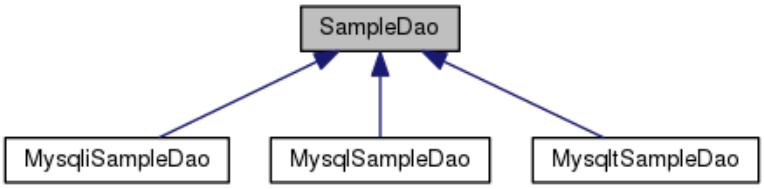
RequestHandler::$Successor `[protected]`

The documentation for this class was generated from the following file:

- **requesthandler.class.php**

SampleDao Class Reference

Inheritance diagram for SampleDao:



Public Member Functions

- **InsertSample** ()
- **DeleteSample** ()
- **FindSample** ()
- **UpdateSample** ()
- **SelectSampleRS** ()
- **SelectSampleTO** ()
- **GetData** ($RecordSet)
- **GetDataTO** ($RecordSet)

Member Function Documentation

SampleDao::DeleteSample () `[abstract]`

SampleDao::FindSample () `[abstract]`

SampleDao::GetData (*$RecordSet*) `[abstract]`

SampleDao::GetDataTO (*$RecordSet*) `[abstract]`

SampleDao::InsertSample () `[abstract]`

SampleDao::SelectSampleRS () `[abstract]`

SampleDao::SelectSampleTO () `[abstract]`

SampleDao::UpdateSample () `[abstract]`

The documentation for this class was generated from the following file:

- **sampledao.class.php**

SampleView Class Reference

Public Member Functions

- **__construct** (**Request** $**Request**)

Static Public Member Functions

- **Display** ()

Constructor & Destructor Documentation

SampleView::__construct (Request *$Request*)

function __construct

This method is executed when an object is instantiated from this class. Preprocessing can be done here before the object is put into service.

public

Member Function Documentation

SampleView::Display () `[static]`

function Display

This method displays the complete view to the user.

public

The documentation for this class was generated from the following file:

- **sampleview.class.php**

Session Class Reference

Public Member Functions

- **__construct** ()
- **GetCacheExpire** ()
- **GetCacheLimiter** ()
- **GetCookieParameters** ()
- **GetId** ()
- **GetName** ()
- **GetSavePath** ()
- **OpenSession** ()
- **CloseSession** ()

Constructor & Destructor Documentation

Session::__construct ()

function __construct

This method is executed when an object is instantiated from this class. Preprocessing can be done here before the object is put into service.

private

Member Function Documentation

Session::CloseSession ()

function **CloseSession()** {

This method starts a session if it does not exist, destroys all data registered to the session, and expires the session cookie.

public

Session::GetCacheExpire ()

function GetCacheExpire

This method retrieves the current cache expire setting.

public

Returns:

integer Current cache expire

Session::GetCacheLimiter ()

function GetCacheLimiter

This method retrieves the current cache limiter setting.

public

Returns:

string Current cache limiter

Session::GetCookieParameters ()

function GetCookieParameters

This method retrieves the session cookie parameters.

public

Returns:

array **Session** cookie parameters

Session::GetId ()

function getId

This method retrieves the current session id.

public

Returns:

string Current session id

Session::GetName ()

function GetName

This method retrieves the current session name.

public

Returns:

string Current session name

Session::GetSavePath ()

function GetSavePath

This method retrieves the current session save path.

public

Returns:

string Current session save path

Session::OpenSession ()

function OpenSession

This method sets the current session save path, the session name, the session cookie parameters, the current cache limiter, the current cache expire and starts a session.

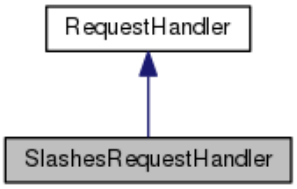
public

The documentation for this class was generated from the following file:

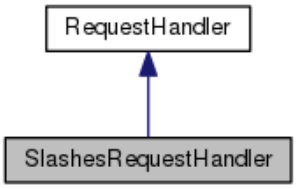
- **session.class.php**

SlashesRequestHandler Class Reference

Inheritance diagram for SlashesRequestHandler:



Collaboration diagram for SlashesRequestHandler:



Public Member Functions

- **HandleRequest (Request $Request)**

Additional Inherited Members

Member Function Documentation

SlashesRequestHandler::HandleRequest (Request *$Request*)

function HandleRequest

This method processes the request if certain conditions are met.

public

Parameters:

Request	**Request** object

The documentation for this class was generated from the following file:

- **slashesrequesthandler.class.php**

Streamlo Class Reference

Public Member Functions

- **__construct** ($fileOrUrl)
- **Open** ($mode)
- **Read** ()
- **ReadLength** ($length)
- **ReadStream** ()
- **ReadStreamLength** ($length)
- **Write** ($data)
- **WriteLength** ($data, $length)
- **Close** ()

Constructor & Destructor Documentation

Streamlo::__construct (*$fileOrUrl*)

function __construct

This method is executed when an object is instantiated from this class. Preprocessing can be done here before the object is put into service.

public

Member Function Documentation

Streamlo::Close ()

function Close

This method closes a file or a stream.

public

Returns:

resource File pointer

Streamlo::Open (*$mode*)

function Open

This method opens a file or stream.

public

Parameters:

mode	Type of access

Returns:

resource File pointer

Streamlo::Read ()

function Read

This method reads all bytes from the file pointer (binary-safe).

public

Returns:

string File contents

StreamIo::ReadLength (*$length*)

function ReadLength

This method reads a specific number of bytes from the file pointer (binary-safe).

public

Parameters:

length	Length in bytes

Returns:

string File contents

StreamIo::ReadStream ()

function ReadStream

This method reads all bytes from the stream pointer (binary-safe).

public

Returns:

string Stream contents

StreamIo::ReadStreamLength (*$length*)

function ReadStreamLength

This method reads a specific number of bytes from the stream pointer (binary-safe).

public

Parameters:

length	Length in bytes

Returns:

string Stream contents

StreamIo::Write (*$data*)

function Write

This method writes all data to the file pointer (binary-safe).

public

Parameters:

data	Data to write

StreamIo::WriteLength (*$data*, *$length*)

function WriteLength

This method writes a specific number of bytes to the file pointer (binary-safe).

public

Parameters:

data	Data to write

length	Length in bytes

The documentation for this class was generated from the following file:

- **streamio.class.php**

View Class Reference

Public Member Functions

- **Action (Request $Request)**

Member Function Documentation

View::Action (Request *$Request*)

function Action

This method defines the operation(s) to be executed by the invoker.

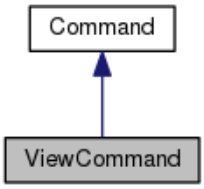
private

The documentation for this class was generated from the following file:

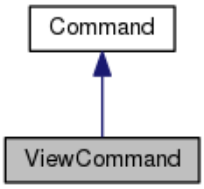
- **view.class.php**

ViewCommand Class Reference

Inheritance diagram for ViewCommand:



Collaboration diagram for ViewCommand:



Public Member Functions

- **__construct** (**View** $Receiver)
- **Execute** (**Request** $**Request**)

Protected Attributes

- **$Receiver**

Constructor & Destructor Documentation

ViewCommand::__construct (View *$Receiver*)

function __construct

This method is executed when an object is instantiated from this class. Preprocessing can be done here before the object is put into service.

private

Member Function Documentation

ViewCommand::Execute (Request *$Request*)

function Execute

This method executes the action and returns the result to the invoker.

public

Member Data Documentation

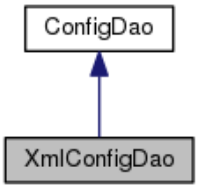
ViewCommand::$Receiver `[protected]`

The documentation for this class was generated from the following file:

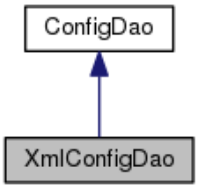
- **viewcommand.class.php**

XmlConfigDao Class Reference

Inheritance diagram for XmlConfigDao:



Collaboration diagram for XmlConfigDao:



Public Member Functions

- **__construct** (DomDocument $DomDocument)
- **GetElementsByPath** ($expression)
- **SetElementByPath** ($expression, $data, $fileName)

Constructor & Destructor Documentation

XmlConfigDao::__construct (DomDocument *$DomDocument*)

function __construct

This method is executed when an object is instantiated from this class. Preprocessing can be done here before the object is put into service.

private

Member Function Documentation

XmlConfigDao::GetElementsByPath (*$expression*)

function GetElementsByPath

This method queries an XML document for a specific node(s) matching some criteria and returns a collection of elements and their associated values.

public

Parameters:

expression	XPath expression

Returns:

array XML element value list

XmlConfigDao::SetElementByPath (*$expression*, *$data*, *$fileName*)

function SetElementByPath

This method queries an XML document for a specific node(s) matching some criteria and replaces the original text content with new text content.

public

Parameters:

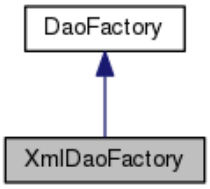
expression	XPath expression
data	Data to write
fileName	XML file to save

The documentation for this class was generated from the following file:

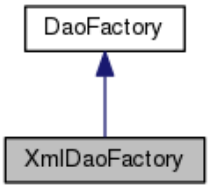
- **xmlconfigdao.class.php**

XmlDaoFactory Class Reference

Inheritance diagram for XmlDaoFactory:



Collaboration diagram for XmlDaoFactory:



Public Member Functions

- **LoadXmlFile** ($fileName, $schemaName, $whiteSpace=FALSE)
- **GetConfigDao** (DomDocument $DomDocument)

Static Public Member Functions

- static **GetInstance** ()

Member Function Documentation

XmlDaoFactory::GetConfigDao (DomDocument *$DomDocument*)

function GetConfigDao

The GetConfigDao method creates a new object of class **XmlConfigDao**.

public

Parameters:

DomDocument	DomDocument object

Returns:

XmlConfigDao object instance

static XmlDaoFactory::GetInstance () `[static]`

function GetInstance

This method instantiates a new object from this class; more specifically, it's a singleton instance.

public

Returns:

XmlDaoFactory object instance

XmlDaoFactory::LoadXmlFile (*$fileName*, *$schemaName*, *$whiteSpace* = `FALSE`)

function LoadXmlFile

This method loads XML from a file, validates its associated XML Schema and optionally preserves white space.

public

Parameters:

fileName	XML file
schemaName	XML Schema file
whiteSpace	Preserve white space

Returns:

DomDocument object instance

The documentation for this class was generated from the following file:

- **xmldaofactory.class.php**

File Documentation

client.class.php File Reference

Classes

- class **Client**

Namespaces

- **phpwebtk**

command.class.php File Reference

Classes

- class **Command**

Namespaces

- **phpwebtk**

configdao.class.php File Reference

Classes

- class **ConfigDao**

Namespaces

- **phpwebtk**

configuration/constants.php File Reference

Variables

- const **CLASS_PATH** getenv ('DOCUMENT_ROOT') . '/phpwebtk/'
- const **CONFIG_FILE CLASS_PATH** . 'configuration/phpwebtk.xml'
- const **SCHEMA_FILE CLASS_PATH** . 'configuration/phpwebtk.xsd'
- const **ADODB_PERF_NO_RUN_SQL** 0
- const **SMARTY_DIR CLASS_PATH** . 'addons/smarty-3.1.21/'

Variable Documentation

const ADODB_PERF_NO_RUN_SQL 0

Disable the "Run SQL" link for the web-based user interface for performance monitoring.

const CLASS_PATH getenv ('DOCUMENT_ROOT') . '/phpwebtk/'

The system path to the location of the PHP Web Toolkit class files.

const CONFIG_FILE CLASS_PATH . 'configuration/phpwebtk.xml'

The system path to the location of the configuration XML file.

const SCHEMA_FILE CLASS_PATH . 'configuration/phpwebtk.xsd'

The system path to the location of the configuration XML Schema file.

const SMARTY_DIR CLASS_PATH . 'addons/smarty-3.1.21/'

The system path to the location of the Smarty class files.

controller.class.php File Reference

Classes

- class **Controller**

Namespaces

- **phpwebtk**

convert.class.php File Reference

Classes

- class **Convert**

Namespaces

- **phpwebtk**

crypt.class.php File Reference

Classes

- class **Crypt**

Namespaces

- **phpwebtk**

daofactory.class.php File Reference

Classes

- class **DaoFactory**

Namespaces

- **phpwebtk**

digest.class.php File Reference

Classes

- class **Digest**

Namespaces

- **phpwebtk**

hash.class.php File Reference

Classes

- class **Hash**

Namespaces

- **phpwebtk**

hmac.class.php File Reference

Classes

- class **Hmac**

Namespaces

- **phpwebtk**

httprequestbuilder.class.php File Reference

Classes

- class **HttpRequestBuilder**

Namespaces

- **phpwebtk**

httprequesthandler.class.php File Reference

Classes

- class **HttpRequestHandler**

Namespaces

- **phpwebtk**

index.php File Reference

Variables

- **$Client** = new **Client** ()

Variable Documentation

$Client = new Client ()

invoker.class.php File Reference

Classes

- class **Invoker**

Namespaces

- **phpwebtk**

ksesrequesthandler.class.php File Reference

Classes

- class **KsesRequestHandler**

Namespaces

- **phpwebtk**

mysqldaofactory.class.php File Reference

Classes

- class **MysqlDaoFactory**

Namespaces

- **phpwebtk**

mysqlidaofactory.class.php File Reference

Classes

- class **MysqliDaoFactory**

Namespaces

- **phpwebtk**

mysqlisampledao.class.php File Reference

Classes

- class **MysqliSampleDao**

Namespaces

- **phpwebtk**

mysqlsampledao.class.php File Reference

Classes

- class **MysqlSampleDao**

Namespaces

- **phpwebtk**

mysqltdaofactory.class.php File Reference

Classes

- class **MysqltDaoFactory**

Namespaces

- **phpwebtk**

mysqltsampledao.class.php File Reference

Classes

- class **MysqltSampleDao**

Namespaces

- **phpwebtk**

pexception.class.php File Reference

Classes

- class **PException**

Namespaces

- **phpwebtk**

postgres7daofactory.class.php File Reference

Classes

- class **Postgres7DaoFactory**

Namespaces

- **phpwebtk**

postgres8daofactory.class.php File Reference

Classes

- class **Postgres8DaoFactory**

Namespaces

- **phpwebtk**

prng.class.php File Reference

Classes

- class **Prng**

Namespaces

- **phpwebtk**

request.class.php File Reference

Classes

- class **Request**

Namespaces

- **phpwebtk**

requestbuilder.class.php File Reference

Classes

- class **RequestBuilder**

Namespaces

- **phpwebtk**

requestdirector.class.php File Reference

Classes

- class **RequestDirector**

Namespaces

- **phpwebtk**

requesthandler.class.php File Reference

Classes

- class **RequestHandler**

Namespaces

- **phpwebtk**

sampledao.class.php File Reference

Classes

- class **SampleDao**

Namespaces

- **phpwebtk**

sampleview.class.php File Reference

Classes

- class **SampleView**

Namespaces

- **phpwebtk**

session.class.php File Reference

Classes

- class **Session**

Namespaces

- **phpwebtk**

slashesrequesthandler.class.php File Reference

Classes

- class **SlashesRequestHandler**

Namespaces

- **phpwebtk**

streamio.class.php File Reference

Classes

- class **StreamIo**

Namespaces

- **phpwebtk**

testscripts/controllertest.php File Reference

Variables

- **$start** = microtime (1)
- **$Client** = new **Client** ()
- **$stop** = microtime (1)
- **$time** = $stop - $start

Variable Documentation

$Client = new Client ()

$start = microtime (1)

$stop = microtime (1)

$time = $stop - $start

testscripts/createkeys.php File Reference

Variables

- **$start** = microtime (1)
- **$Crypt** = new **Crypt** ()
- **$Hmac** = new **Hmac** ()
- **$hmackey** = $**Hmac**->SetHmacKey ('foo')
- **$encryptionkey** = $**Crypt**->SetEncryptionKey ($**Hmac**->GetHmac ('bar'))
- **$stop** = microtime (1)
- **$time** = $stop - $start

Variable Documentation

$Crypt = new Crypt ()

$encryptionkey = $Crypt->SetEncryptionKey ($Hmac->GetHmac ('bar'))

$Hmac = new Hmac ()

$hmackey = $Hmac->SetHmacKey ('foo')

$start = microtime (1)

$stop = microtime (1)

$time = $stop - $start

testscripts/crypttest.php File Reference

Variables

- **$start** = microtime (1)
- **$plaintext** = 'what do ya want for nothing?'
- **$Crypt** = new **Crypt** ()
- **$ciphertext** = $**Crypt**->Encrypt ($plaintext)
- **$output** = $**Crypt**->Decrypt ($ciphertext)
- **$stop** = microtime (1)
- **$time** = $stop - $start

Variable Documentation

$ciphertext = $Crypt->Encrypt ($plaintext)

$Crypt = new Crypt ()

$output = $Crypt->Decrypt ($ciphertext)

$plaintext = 'what do ya want for nothing?'

$start = microtime (1)

$stop = microtime (1)

$time = $stop - $start

testscripts/digesttest.php File Reference

Variables

- **$start** = microtime (1)
- **$plaintext** = 'what do ya want for nothing?'
- **$Digest** = new **Digest** ()
- **$ciphertext** = $**Digest**->GetDigest ($plaintext)
- if(FALSE!==$**Digest**->IsValidDigest($ciphertext, $plaintext)) **else**
- **$stop** = microtime (1)
- **$time** = $stop - $start

Variable Documentation

$ciphertext = $Digest->GetDigest ($plaintext)

$Digest = new Digest ()

$plaintext = 'what do ya want for nothing?'

$start = microtime (1)

$stop = microtime (1)

$time = $stop - $start

if (FALSE!==$Digest->IsValidDigest($ciphertext, $plaintext)) else

```
Initial value:{
    print ('<br />Invalid Signature')
```

testscripts/encryptionkey.php File Reference

Variables

- **$start** = microtime (1)
- **$Crypt** = new **Crypt** ()
- **$Hmac** = new **Hmac** ()
- **$encryptionkey** = $**Crypt**->SetEncryptionKey ($**Hmac**->GetHmac ('bar'))
- **$stop** = microtime (1)
- **$time** = $stop - $start

Variable Documentation

$Crypt = new Crypt ()

$encryptionkey = $Crypt->SetEncryptionKey ($Hmac->GetHmac ('bar'))

$Hmac = new Hmac ()

$start = microtime (1)

$stop = microtime (1)

$time = $stop - $start

testscripts/hmackey.php File Reference

Variables

- **$start** = microtime (1)
- **$Hmac** = new **Hmac** ()
- **$hmackey** = $**Hmac**->SetHmacKey ('foo')
- **$stop** = microtime (1)
- **$time** = $stop - $start

Variable Documentation

$Hmac = new Hmac ()

$hmackey = $Hmac->SetHmacKey ('foo')

$start = microtime (1)

$stop = microtime (1)

$time = $stop - $start

testscripts/hmactest.php File Reference

Variables

- **$start** = microtime (1)
- **$plaintext** = 'what do ya want for nothing?'
- **$Hmac** = new **Hmac** ()
- **$ciphertext** = $**Hmac**->GetHmac ($plaintext)
- if(FALSE!==$**Hmac**->IsValidHmac($ciphertext, $plaintext)) **else**
- **$stop** = microtime (1)
- **$time** = $stop - $start

Variable Documentation

$ciphertext = $Hmac->GetHmac ($plaintext)

$Hmac = new Hmac ()

$plaintext = 'what do ya want for nothing?'

$start = microtime (1)

$stop = microtime (1)

$time = $stop - $start

if (FALSE!==$Hmac->IsValidHmac($ciphertext, $plaintext)) else

```
Initial value:{
    print ('<br />Invalid Signature')
```

testscripts/mysqlsample.php File Reference

Variables

- **$start** = microtime (1)
- **$DaoFactory** = **DaoFactory::GetDaoFactory** ("mysqli")
- **$SampleDao** = $**DaoFactory**->GetSampleDao ()
- **$stop** = microtime (1)
- **$time** = $stop - $start

Variable Documentation

$DaoFactory = DaoFactory::GetDaoFactory ("mysqli")

$SampleDao = $DaoFactory->GetSampleDao ()

$start = microtime (1)

$stop = microtime (1)

$time = $stop - $start

testscripts/prngtest.php File Reference

Variables

- **$start** = microtime (1)
- **$Prng** = new **Prng** ()
- **$random** = $**Prng**->GetPseudoRandomValue (0)
- **$urandom** = $**Prng**->GetPseudoRandomValue (1)
- **$stop** = microtime (1)
- **$time** = $stop - $start

Variable Documentation

$Prng = new Prng ()

$random = $Prng->GetPseudoRandomValue (0)

$start = microtime (1)

$stop = microtime (1)

$time = $stop - $start

$urandom = $Prng->GetPseudoRandomValue (1)

testscripts/sessiontest.php File Reference

Variables

- **$start** = microtime (1)
- **$Session** = new **Session** ()
- **$stop** = microtime (1)
- **$time** = $stop - $start

Variable Documentation

$Session = new Session ()

$start = microtime (1)

$stop = microtime (1)

$time = $stop - $start

view.class.php File Reference

Classes

- class **View**

Namespaces

- **phpwebtk**

viewcommand.class.php File Reference

Classes

- class **ViewCommand**

Namespaces

- **phpwebtk**

xmlconfigdao.class.php File Reference

Classes

- class **XmlConfigDao**

Namespaces

- **phpwebtk**

xmldaofactory.class.php File Reference

Classes

- class **XmlDaoFactory**

Namespaces

- **phpwebtk**

Index

INDEX