FUNCTIONAL REACTIVE JAVASCRIPT

FUNCTIONAL REACTIVE PROGRAMMING

variable	*stream*
snapshot of a value	all future values
at a specific time	over time

FUNCTIONAL REACTIVE PROGRAMMING

UI elements produce streams of values.

Functions combine and transform those streams into new streams.

Streams can be assigned to UI attributes to effect UI output.

our approach

for this evening


bacon.is

for bonus points:

TDD*!

* *Test-Driven Development*

Pete Hodgson
@ph1, Glorified Typist

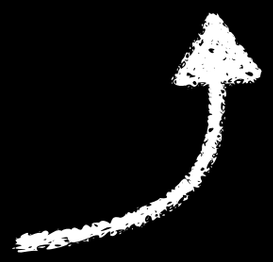
ThoughtWorks®

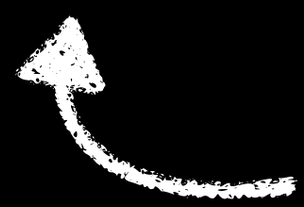
315 Montgomery Street, 16th Floor | San Francisco, California | 94104
T: +1 415 869 3000 C: +1 707 206 8897
E: phodgson@thoughtworks.com

example #1.

slider & label

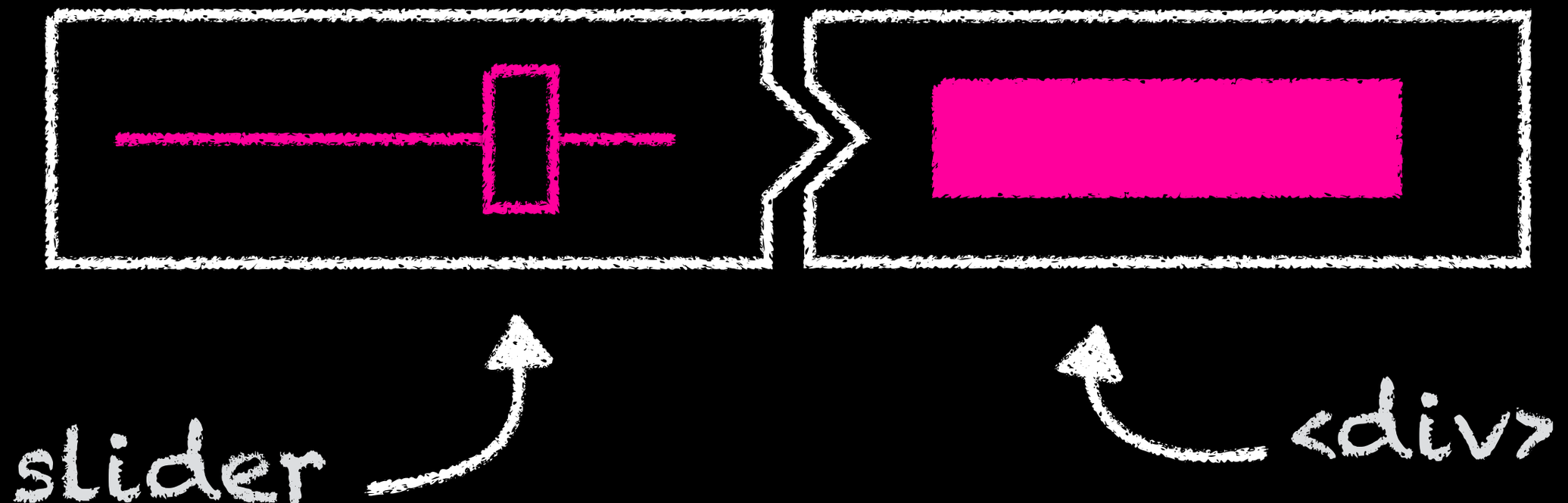
slider



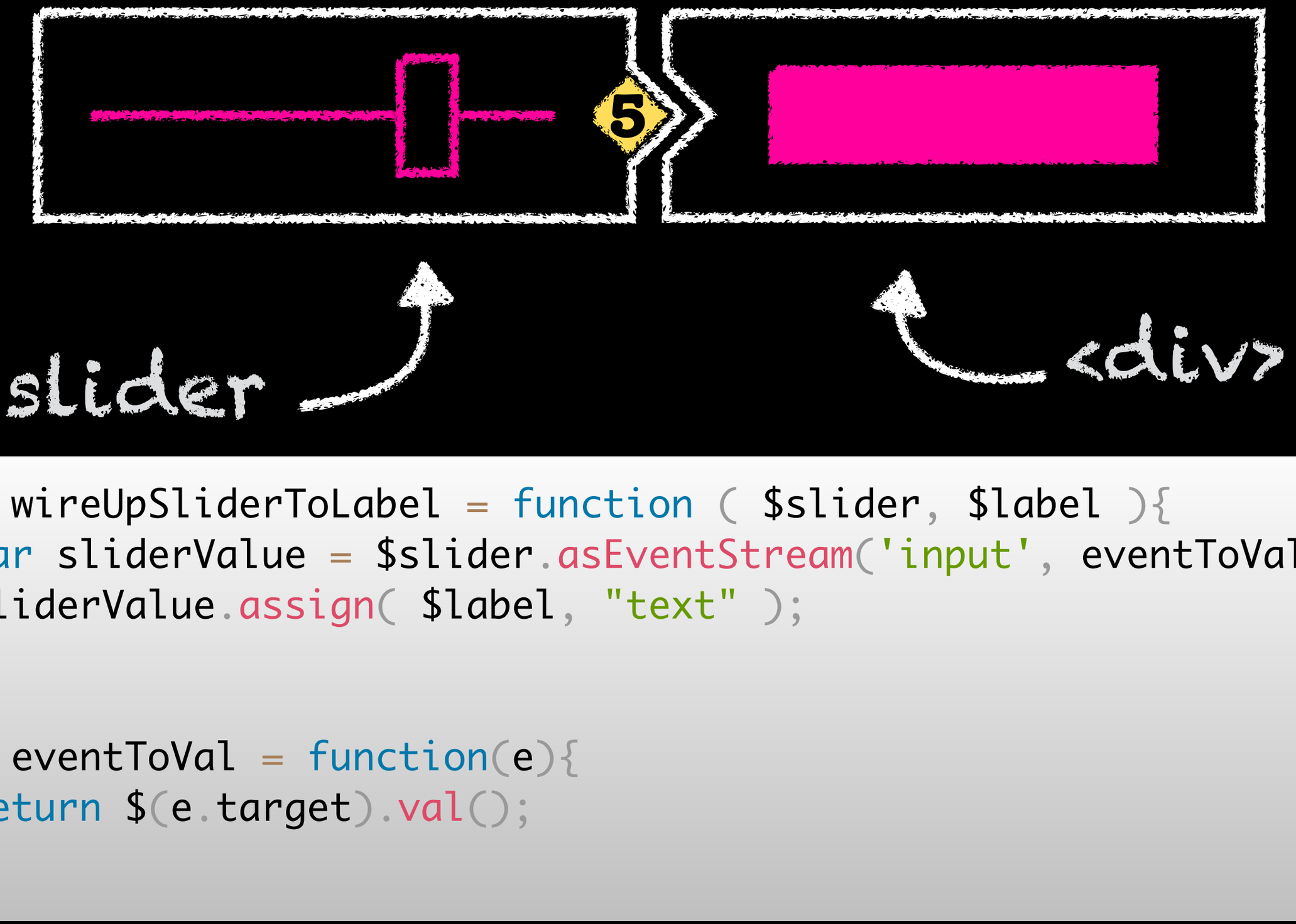


```
var wireUpSliderToLabel = function ( $slider, $label ){
  var sliderValue = $slider.asEventStream('input', eventToVal);
  sliderValue.assign( $label, "text" );
};

var eventToVal = function(e){
  return $(e.target).val();
};
```

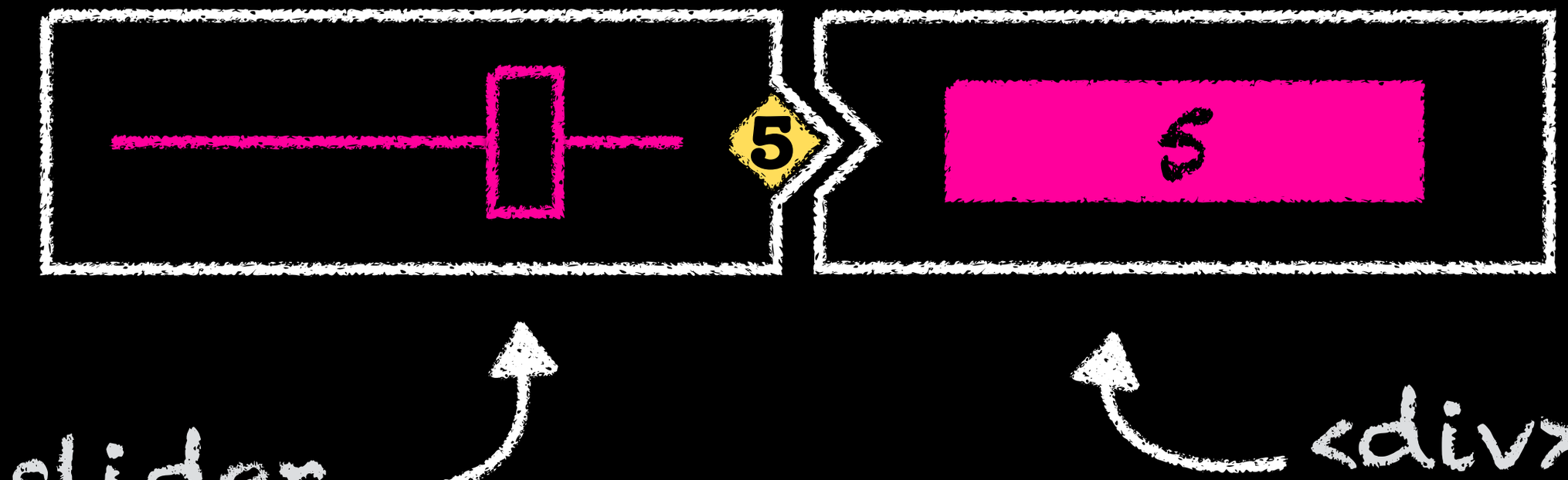

```
var wireUpSliderToLabel = function ( $slider, $label ){
  var sliderValue = $slider.asEventStream('input', eventToVal);
  sliderValue.assign( $label, "text" );
};

var eventToVal = function(e){
  return $(e.target).val();
};
```




```
var wireUpSliderToLabel = function ( $slider, $label ){
  var sliderValue = $slider.asEventStream('input', eventToVal);
  sliderValue.assign( $label, "text" );
};

var eventToVal = function(e){
  return $(e.target).val();
};
```




```
var wireUpSliderToLabel = function ( $slider, $label ){
  var sliderValue = $slider.asEventStream('input', eventToVal);
  sliderValue.assign( $label, "text" );
};

var eventToVal = function(e){
  return $(e.target).val();
};
```




```
var wireUpSliderToLabel = function ( $slider, $label ){
  var sliderValue = $slider.asEventStream('input', eventToVal);
  sliderValue.assign( $label, "text" );
};

var eventToVal = function(e){
  return $(e.target).val();
};
```


```
var wireUpSliderToLabel = function ( $slider, $label ){
  var sliderValue = $slider.asEventStream('input', eventToVal);
  sliderValue.assign( $label, "text" );
};

var eventToVal = function(e){
  return $(e.target).val();
};
```


```
var wireUpSliderToLabel = function ( $slider, $label ){
  var sliderValue = $slider.asEventStream('input', eventToVal);
  sliderValue.assign( $label, "text" );
};

var eventToVal = function(e){
  return $(e.target).val();
};
```

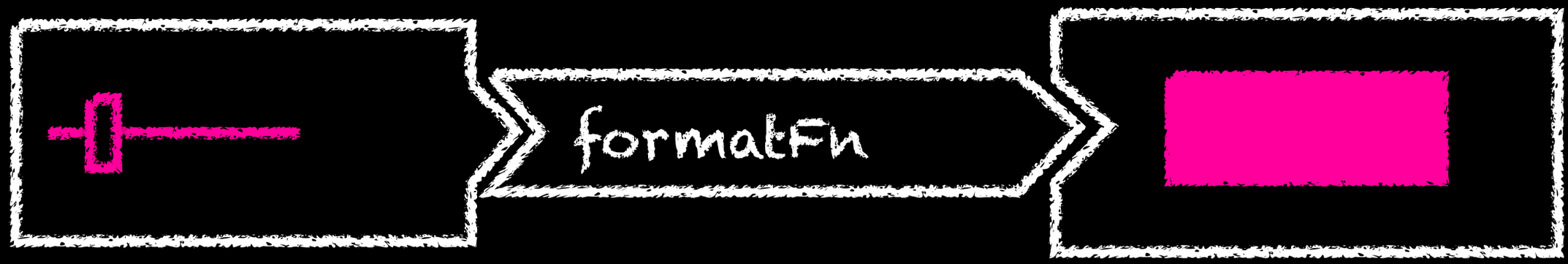
epiphany

declarative,

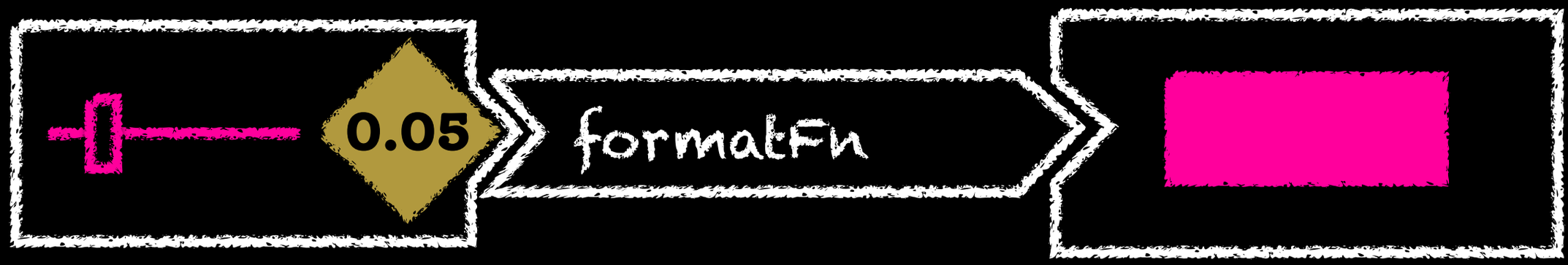
not imperative

example #2.

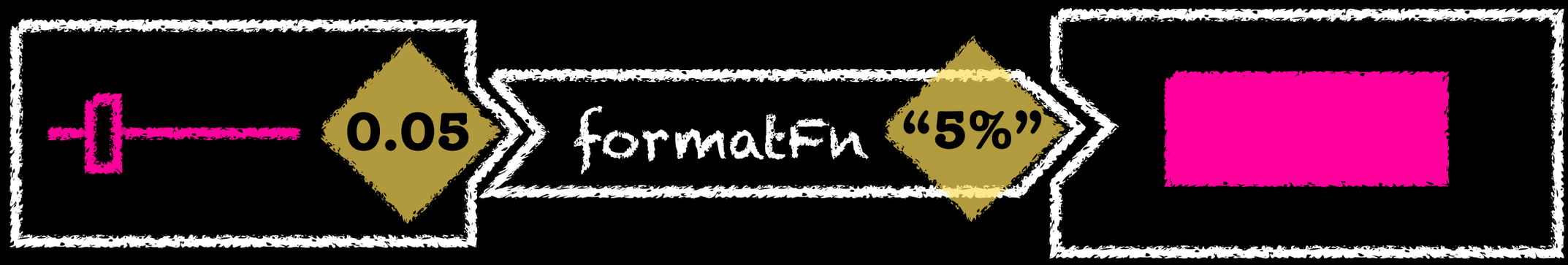
slider & label w. formatting



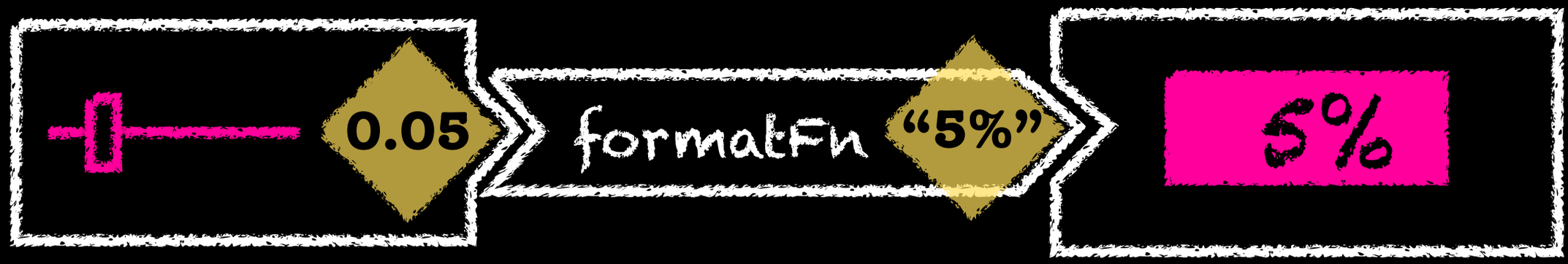

```
var wireUpSliderToFormattedLabel = function ( $slider, $label ){
  var sliderValue = $slider.asEventStream('input', eventToVal);
  var formattedValue = sliderValue.map( ratioToPercentage );
  formattedValue.assign( $label, "text" );
};

var ratioToPercentage = function(ratio){
  return "" + Math.round(ratio*100) + "%";
}

var eventToVal = function(e){
  return $(e.target).val();
};
```




```
var wireUpSliderToFormattedLabel = function ( $slider, $label ){
  var sliderValue = $slider.asEventStream('input', eventToVal);
  var formattedValue = sliderValue.map( ratioToPercentage );
  formattedValue.assign( $label, "text" );
};

var ratioToPercentage = function(ratio){
  return "" + Math.round(ratio*100) + "%";
}

var eventToVal = function(e){
  return $(e.target).val();
};
```




```
var wireUpSliderToFormattedLabel = function ( $slider, $label ){
  var sliderValue = $slider.asEventStream('input', eventToVal);
  var formattedValue = sliderValue.map( ratioToPercentage );
  formattedValue.assign( $label, "text" );
};

var ratioToPercentage = function(ratio){
  return "" + Math.round(ratio*100) + "%";
}

var eventToVal = function(e){
  return $(e.target).val();
};
```




```
var wireUpSliderToFormattedLabel = function ( $slider, $label ){
  var sliderValue = $slider.asEventStream('input', eventToVal);
  var formattedValue = sliderValue.map( ratioToPercentage );
  formattedValue.assign( $label, "text" );
};

var ratioToPercentage = function(ratio){
  return "" + Math.round(ratio*100) + "%";
}

var eventToVal = function(e){
  return $(e.target).val();
};
```

epiphany

compose-ability

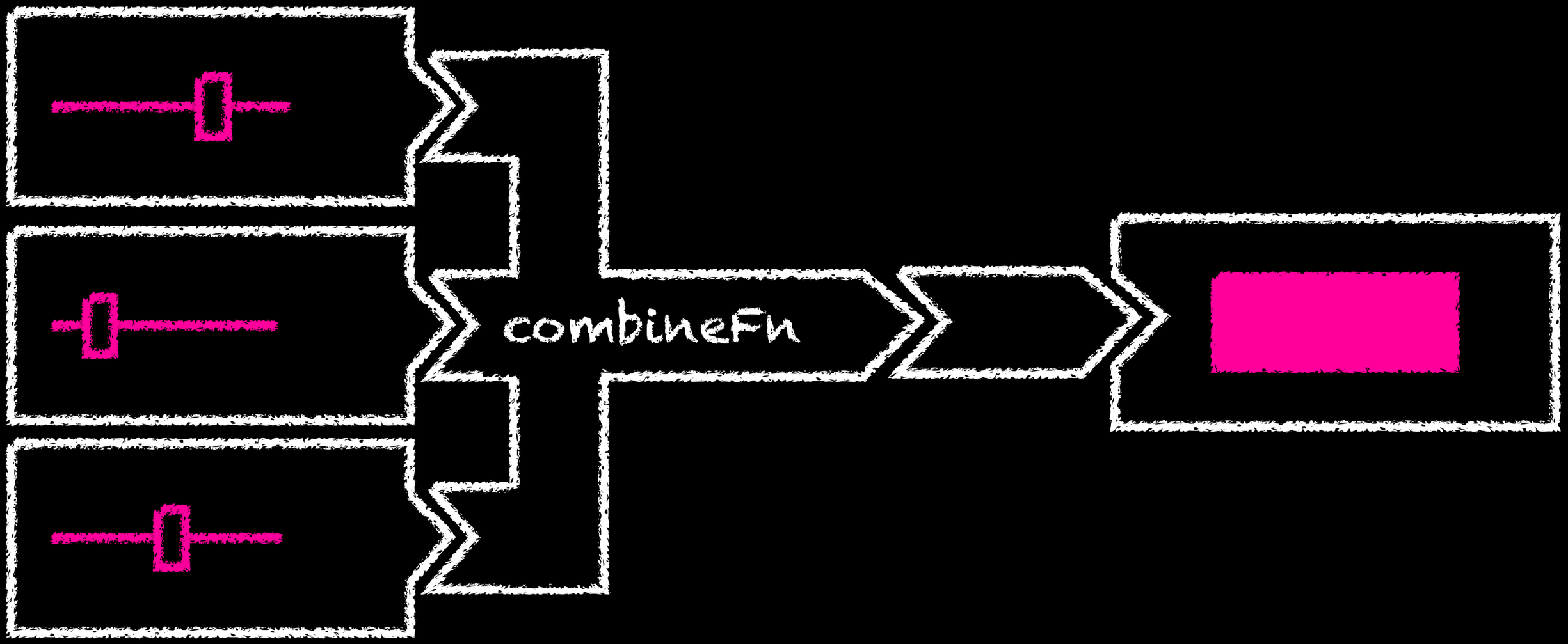
(pipes & filters)

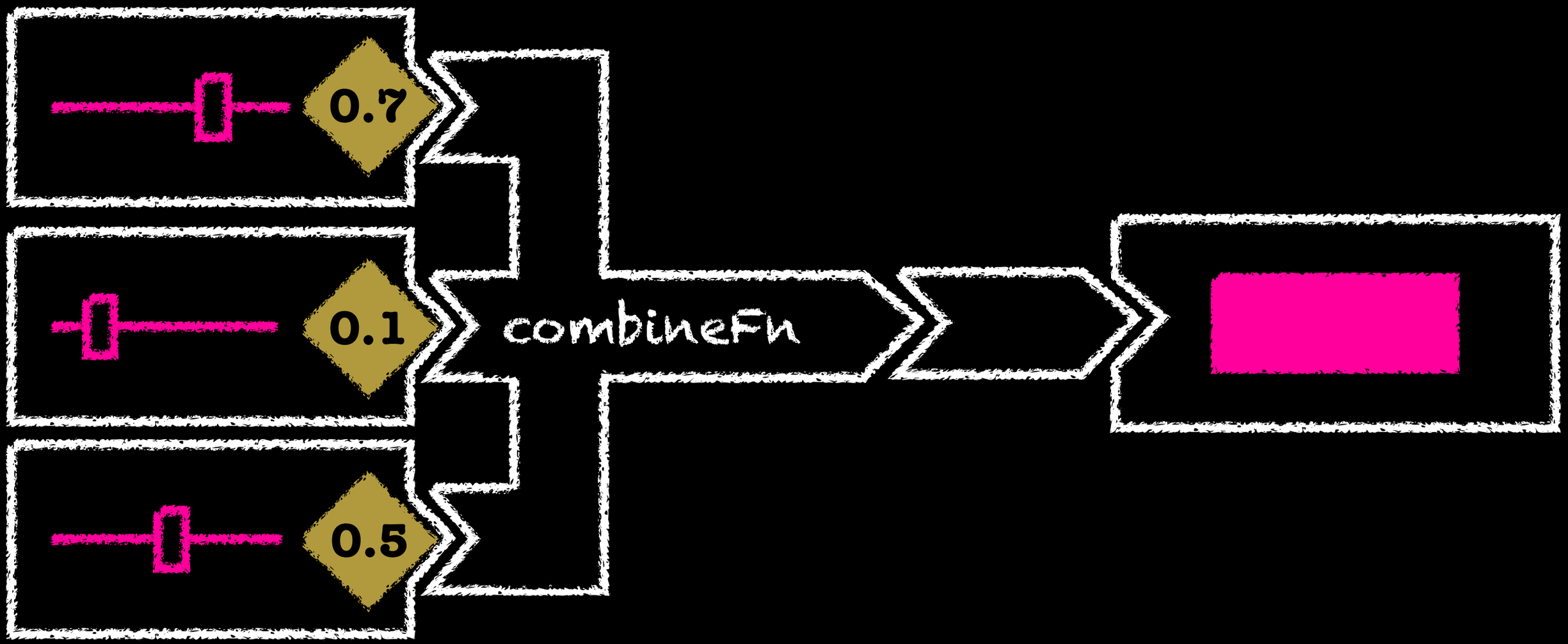
example #3.

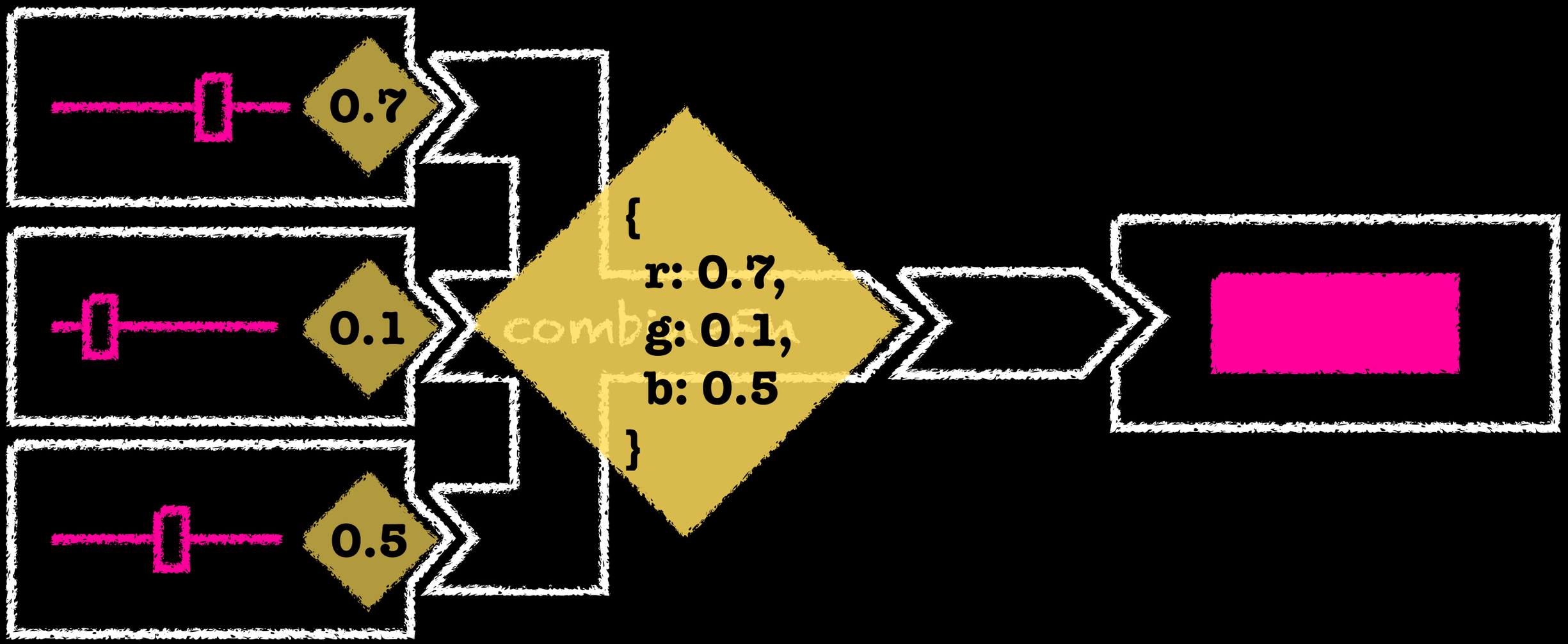
slider & grayscale swatch

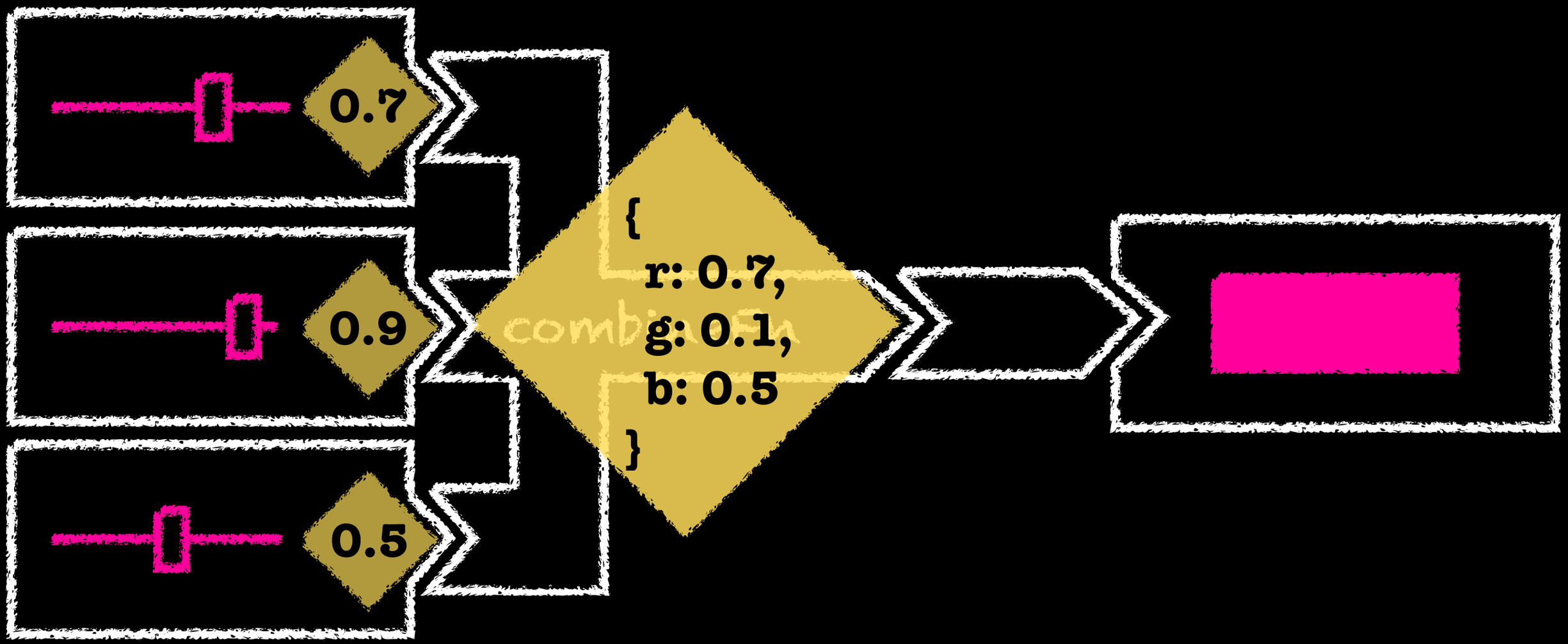
example #4.

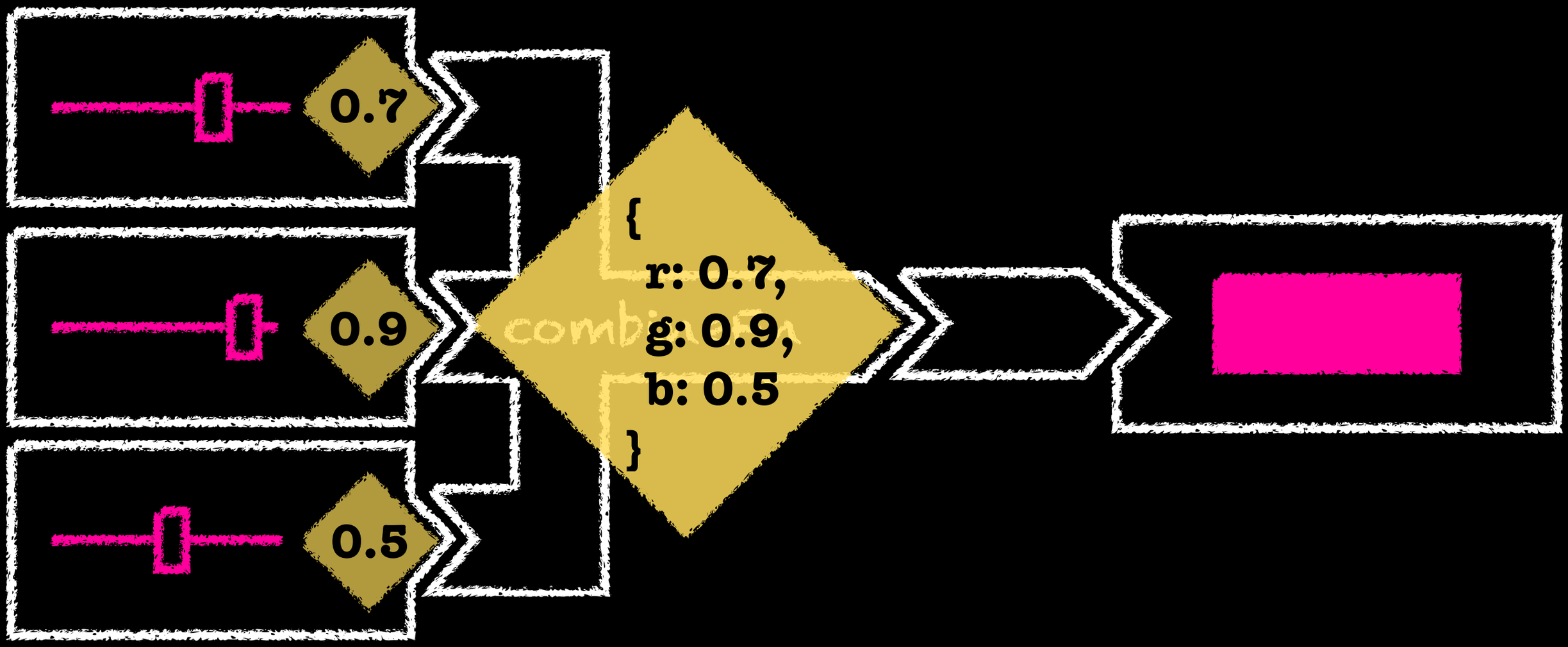
sliders & color swatch


combineFn


0.7
0.1
combineFn
0.5


0.7
0.1
0.5
{
r: 0.7,
g: 0.1,
b: 0.5
}


0.7
0.9
0.5
{
r: 0.7,
g: 0.1,
b: 0.5
}


0.7
0.9
0.5
{
r: 0.7,
g: 0.9,
b: 0.5
}

epiphany

most recent value is always available

epiphany

very little need to manage *mutable state*

back to theory

variable	*stream*
snapshot of a value	all future values
at a specific time	over time

```
var a = 12
var b = a + 10    // b == 22
a = 20            // b == 22
```

```
var a = 12
var b = a + 10    // b == 22
a = 20            // b == 22
```

```
var a = aStreamWith(12)
var b = a.map( -> + 10 )
a.updateTo(20)
```

we're just
getting started

frp-talk-examples.thepete.net

thanks!

questions?

@ph1

phodgson@thoughtworks.com

.....

frp-talk-examples.thepete.net

blog.thepete.net

slides.thepete.net