
PHPMarkup

Mar 29, 2021

Project Information

1	Installation	3
2	Contribute	5
3	Navigation	7
4	Indices and tables	31
	Index	33

PHPMarkup is an PHP implementation of a LHTML parser.

It is a powerful and flexible way to build dynamic web pages.

```
<?php
use Ouxsoft\PHPMarkup\Factory\ProcessorFactory;

$processor = ProcessorFactory::getInstance();

$processor->addElement([
    'xpath' => '//partial',
    'class_name' => 'Partial\{name}'
]);

$processor->addRoutine([
    'method' => 'onRender',
    'execute' => 'RETURN_CALL'
]);

$processor->parseBuffer();
?>
<html lang="en">
<partial name="Alert" type="success">
    This is a success alert.
</partial>
</html>
```


CHAPTER 1

Installation

Get an instance of PHPMarkup up and running in less than 5 minutes.

PHPMarkup is available on Packagist.

Install with Composer:

```
composer require ouxsoft/phpmarkup
```

That's it. You're done! Please take the rest of the time to read our docs.

CHAPTER 2

Contribute

- Issue Tracker: <https://github.com/ouxsoft/PHPMarkup/issues>
- Source Code: <https://github.com/ouxsoft/PHPMarkup>

3.1 Processor

The `Processor` is the primary object instantiated and used within this library.

3.1.1 `ParseBuffer()` Example

Shows a `Processor` with `Elements` and `Routines` defined during runtime processing the output buffer of the file.

```
<?php
use Ouxsoft\PHPMarkup\Factory\ProcessorFactory;

$processor = ProcessorFactory::getInstance();

$processor->addElement([
    'xpath' => '//partial',
    'class_name' => 'Partial\{name}'
]);

$processor->addRoutine([
    'method' => 'onRender',
    'execute' => 'RETURN_CALL'
]);

$processor->parseBuffer();
?>
<html lang="en">
    <partial name="Alert" type="success">
        This is a success alert.
    </partial>
</html>
```

Outputs:

3.1.2 ParseBuffer() Inside Router Example

A server-side markup abstraction layer example. Shows using ParseBuffer inside a third party router to prevent the need to declare the Processor within each file.

```
<?php

use Ouxsoft\PHPMarkup\Factory\ProcessorFactory;
use Ouxsoft\Hoopless\Router;

require_once '../vendor/autoload.php';

// define common directories
define('ROOT_DIR', dirname(__DIR__, 1) . '/');
define('PUBLIC_DIR', ROOT_DIR . 'public/');
define('ASSET_DIR', ROOT_DIR . 'assets/');
define('IMAGE_DIR', ASSET_DIR . 'images/');
define('CONFIG_DIR', ROOT_DIR . 'config/');

// set include path
set_include_path(ROOT_DIR);

// instantiate processor with configuration and set to parse buffer
global $processor;
$processor = ProcessorFactory::getInstance();
$processor->loadConfig(CONFIG_DIR . 'config.dist.json');
$processor->parseBuffer();

// Route traffic to a specific file
$router = new Router();
$router->response();

// if response is a blank document chances are the page is missing a root element
```

3.1.3 ParseFile() Example

Shows a Processor defined with Elements and Routines defined in a loaded config and a parse file containing markup.

```
<?php
use Ouxsoft\PHPMarkup\Factory\ProcessorFactory;

$processor = ProcessorFactory::getInstance();

$processor->loadConfig('config.json');
$processor->parseFile('index.html')
```

Outputs:

3.1.4 ParseString() Example

Shows a Processor with Configuration Elements and Routines manually defined parsing a string.

```
<?php
use Ouxsoft\PHPMarkup\Factory\ProcessorFactory;

$processor = ProcessorFactory::getInstance();

$processor->addElement([
    'xpath' => '//partial',
    'class_name' => 'Partial\{name}'
]);

$processor->addRoutine([
    'method' => 'onRender',
    'execute' => 'RETURN_CALL'
]);

$processor->parseString('<html lang="en">
    <partial name="Alert" type="success">
        This is a success alert.
    </partial>
</html>');
```

Outputs:

3.2 Routines

Routines are methods that are automatically called by the processor during run time.

During a routine call all instantiated elements featuring the routine's method are called.

3.2.1 Prefix

It is recommended to establish a naming convention for routines that distinguish them from other methods.

Often packages choose to prefix Routine methods with the word *before*, *on*, or *after* followed by an explanation of the stage of execute.

For example:

Parameter	Comments
beforeLoad	Execute before object data load
onLoad	Execute during object data load
afterLoad	Execute after object data loaded
beforeRender	Execute before object render
onRender	Execute during object render
afterRender	Execute after object rendered

3.3 Elements

Elements are the working bees of PHPMarkup.

Elements are instantiated DOMElements. Much of how an DOMElement is processed is determined by the `class_name` defined to process them.

3.3.1 Args

During the Element's construction, the Engine sends arguments that were found in both the DOMELEMENT's attributes and child `arg` DOMELEMENTs.

3.3.2 Element Development

It is simple to make new Elements.

1. Create a class that extends the abstract class `\PHPMarkup\Element\AbstractElement`.
2. Add an Element to the Processor.
3. Run the Processor with Markup containing the DOMELEMENT

Hello World Example

```
<?php
namespace Ouxsoft\PHPMarkup\Elements;

class HelloWorld extends Ouxsoft\PHPMarkup\Element
{
    public function onRender()
    {
        return 'Hello, World';
    }
}
```

Bootstrap 4 Alert Example

```
<?php
namespace Partial;

class Alert extends Ouxsoft\PHPMarkup\Element\AbstractElement
{
    public function onRender()
    {
        switch($this->getArgByName('type')){
            case 'success':
                $class = 'alert-success';
                break;
            case 'warning':
                $class = 'alert-warning';
                break;
            default:
                $class = 'alert-info';
                break;
        }
        return "<div class=\"alert {$class}\" role=\"alert\">{$this->innerText()}</div>";
    }
}
```

3.3.3 LHTML Elements

To allow for library reuse PHPMarkup comes packaged with only LHTML Elements used for test. For additional Elements, see:

- [Hoopless](https://github.com/ouxsoft/hoopless)

3.4 Configuration

The Configuration class is responsible for the instructions that explain to the Builder how to build the LHTML Document. These instructions can be set by modifying the config file that is loaded.

3.4.1 Config File

Configurations can be loaded from a file or created using an object. Below is an example of a config file.

Example

```
{
  "version": 3,
  "elements": [
    {
      "xpath": "//bitwise",
      "class_name": "PHPMarkup\\Test\\Bitwise"
    }
  ],
  "routines": [
    {
      "method": "beforeLoad",
      "description": "Execute before object data is loaded"
    },
    {
      "method": "onLoad",
      "description": "Execute when object data is loading"
    },
    {
      "method": "afterLoad",
      "description": "Execute after object data is loaded"
    },
    {
      "method": "beforeRender",
      "description": "Execute before object is rendered"
    },
    {
      "method": "onRender",
      "description": "Execute while object is rendering",
      "execute": "RETURN_CALL"
    },
    {
      "method": "afterRender",
      "description": "Execute after object is rendered"
    }
  ]
}
```

Autoload

If a config file has been specified during construction, it will be loaded.

If a config file has not been specified the `Configuration` class tries to load a `config.json` file if present.

If the `config.json` is not present, the `Configuration` will try

to load the packaged config `config.dist.json` file.

Parameters

Parameter	Comments
<code>version</code>	Indicates the file structure to the <code>Configuration</code> for stability purposes.
<code>elements</code>	An array containing the types of elements to load at runtime. Each type contains contain an array with a <code>name</code> , a <code>class_name</code> , and a <code>xpath</code> expression.
<code>element:*</code>	Defines what the elements is named.
<code>element:*</code>	Specifies exactly how find <code>DOMElements</code> to initialize as elements. Xpath expressions are a powerful syntax for searching within a the <code>Document</code> for <code>DOMElements</code> .
<code>element:*</code>	Specifies which class to instantiate the <code>DOMElement</code> as. The <code>class_name</code> provided must refer to a class that extends the abstract <code>Element</code> class. The class name may feature a <code>{name}</code> variable which is automatically populated by the <code>DOMElement</code> 's <code>name</code> attribute during runtime.
<code>routines</code>	An array containing automated method calls that will be made to all <code>Elements</code> during runtime. The order of items in this array determines the order of execution.
<code>routines:*</code>	The exact name of the method being executed.
<code>routines:*</code>	An explanation of what the method is doing that indicates its order.
<code>routines:*</code>	Determines whether the method should be ran differently. Currently, the following commands are supported * <code>RETURN_CALL</code> - The output of the method will replace the <code>DOMElement</code> in the <code>DOMDocument</code> . Is optional
<code>markup:</code>	String containing the actual LHTML that will be parsed by the <code>Builder</code> . This field is typically omitted from the config file and is instead appended to <code>Configuration</code> during runtime, often by the <code>Autoloader</code> .

3.5 Security

3.5.1 Escaping HTML and XSS

It is the responsibility of the library client to escape HTML to avoid XSS. This library itself will not alter its input.

3.5.2 Disable Entity Loader

You may want to choose to disable external entities.

```
libxml_disable_entity_loader(true);
```

For more information, see [PHP Security Injection Attacks](#)

3.6 API documentation

3.6.1 Builder namespace

DynamicPageBuilder

Qualified name Ouxsoft\PHPMarkup\Builder\DynamicPageBuilder

Implements *BuilderInterface*

class DynamicPageBuilder

public __construct (EngineInterface \$engine, ConfigurationInterface \$config)

Parameters

- **\$engine** (EngineInterface) –
- **\$config** (ConfigurationInterface) –

public createObject ()

Instantiate elements and call routines inside engine

Returns void

public getObject () → Engine

Gets Page object

Returns *Engine* –

SearchIndexBuilder

Qualified name Ouxsoft\PHPMarkup\Builder\SearchIndexBuilder

Implements *BuilderInterface*

class SearchIndexBuilder

public __construct (EngineInterface \$engine, ConfigurationInterface \$config)

Parameters

- **\$engine** (EngineInterface) –
- **\$config** (ConfigurationInterface) –

public createObject ()

Creates Page object using parameters supplied omits elements with search_engine = false

Returns void

public getObject () → Engine

Gets Page object

Returns *Engine* –

StaticPageBuilder

Qualified name Ouxsoft\PHPMarkup\Builder\StaticPageBuilder

Implements *BuilderInterface*

class StaticPageBuilder

public **__construct** (*EngineInterface* \$engine, *ConfigurationInterface* \$config)

Parameters

- **\$engine** (*EngineInterface*) –
- **\$config** (*ConfigurationInterface*) –

public **createObject** ()

Creates Page object using parameters supplied

Returns void

public **getObject** () → Engine

Gets Page object

Returns *Engine* –

3.6.2 Contract namespace

AbstractFactoryInterface

Qualified name Ouxsoft\PHPMarkup\Contract\AbstractFactoryInterface

interface AbstractFactoryInterface

public **makeBuilder** (*Container* & \$container) → *BuilderInterface*

Parameters

- **& \$container** (*Container*) –

Returns *BuilderInterface* –

public **makeConfig** (*Container* & \$container) → *Configuration*

Parameters

- **& \$container** (*Container*) –

Returns *Configuration* –

public **makeDocument** (*Container* & \$container) → *Document*

Parameters

- **& \$container** (*Container*) –

Returns *Document* –

public **makeElementPool** ()

Returns *ElementPool*

public **makeEngine** (*Container* & \$container) → *Engine*

Parameters

- **& \$container** (*Container*) –

Returns *Engine* –

public makeKernel (*Container* & *\$container*) → *Kernel*

Parameters

- **& \$container** (*Container*) –

Returns *Kernel* –

BuilderInterface

Qualified name Ouxsoft\PHPMarkup\Contract\BuilderInterface

interface BuilderInterface

public __construct (*EngineInterface* ' \$engine, *ConfigurationInterface* ' \$config)

Parameters

- **\$engine** (*EngineInterface* ') –
- **\$config** (*ConfigurationInterface* ') –

public createObject ()

public getObject ()

ConfigurationInterface

Qualified name Ouxsoft\PHPMarkup\Contract\ConfigurationInterface

interface ConfigurationInterface

DocumentInterface

Qualified name Ouxsoft\PHPMarkup\Contract\DocumentInterface

interface DocumentInterface

ElementPoolInterface

Qualified name Ouxsoft\PHPMarkup\Contract\ElementPoolInterface

interface ElementPoolInterface

public add (*AbstractElement* ' \$element)

Parameters

- **\$element** (*AbstractElement* ') –

public callRoutine (*string* \$routine)

Parameters

- **\$routine** (*string*) –

```
public count ()
public getById ([ ])
```

Parameters

- **\$element_id** (*?string*) – Default: null

```
public getIterator ()
public getPropertiesById (string $element_id)
```

Parameters

- **\$element_id** (*string*) –

EngineInterface

Qualified name Ouxsoft\PHPMarkup\Contract\EngineInterface

interface EngineInterface

```
public __construct (DocumentInterface $document, ElementPoolInterface $element_pool)
```

Parameters

- **\$document** (*DocumentInterface*) –
- **\$element_pool** (*ElementPoolInterface*) –

```
public __toString ()
public callRoutine (array $routine)
```

Parameters

- **\$routine** (*array*) –

```
public getDomElementByPlaceholderId (string $element_id)
```

Parameters

- **\$element_id** (*string*) –

```
public getElementAncestorProperties (string $element_id)
```

Parameters

- **\$element_id** (*string*) –

```
public getElementArgs (DOMElement $element)
```

Parameters

- **\$element** (*DOMElement*) –

```
public getElementInnerXML (string $element_id)
```

Parameters

- **\$element_id** (*string*) –

```
public instantiateElements (array $html_element)
```

Parameters

- `$lhtml_element` (*array*) –

```
public queryFetch (string $query[, DOMELEMENT $node])
```

Parameters

- `$query` (*string*) –
- `$node` (*DOMELEMENT*) – Default: null

```
public queryFetchAll (string $query[, DOMELEMENT $node])
```

Parameters

- `$query` (*string*) –
- `$node` (*DOMELEMENT*) – Default: null

```
public removeElements (array $lhtml_element)
```

Parameters

- `$lhtml_element` (*array*) –

```
public renderElement (string $element_id)
```

Parameters

- `$element_id` (*string*) –

```
public replaceDomElement (DOMELEMENT $element, string $new_xml)
```

Parameters

- `$element` (*DOMELEMENT*) –
- `$new_xml` (*string*) –

```
public setType ([ ])
```

Parameters

- `$value` – Default: null
- `$type` – Default: 'string'

KernelInterface

Qualified name Ouxsoft\PHPMarkup\Contract\KernelInterface

```
interface KernelInterface
```

```
public __construct (EngineInterface $engine, BuilderInterface $builder, ConfigurationInterface $config)
```

Parameters

- `$engine` (*EngineInterface*) –
- `$builder` (*BuilderInterface*) –
- `$config` (*ConfigurationInterface*) –

```
public build ()
```

```
public getBuilder ()
```

```
public getConfig ()
```

```
public setBuilder (string $builder_class)
```

Parameters

- **\$builder_class** (*string*) –

```
public setConfig (ConfigurationInterface $config)
```

Parameters

- **\$config** (*ConfigurationInterface*) –

3.6.3 Element namespace

AbstractElement

Qualified name Ouxsoft\PHPMarkup\Element\AbstractElement

class AbstractElement

```
public __construct ([array])
```

Element constructor

Parameters

- **\$args** (*ArgumentArray*) – Default: null

```
public __invoke (string $method) → bool
```

Invoke wrapper call to method if exists

Parameters

- **\$method** (*string*) –

Returns bool –

```
public __toString () → string
```

Call onRender if exists on echo / output

Returns string –

```
public getArgByName (string $name)
```

Get arg by name

Parameters

- **\$name** (*string*) –

Returns mixed|null

```
public getArgs () → ArgumentArray
```

Get all args

Returns *ArgumentArray* –

```
public getId ()
```

Gets the ID of the Element, useful for *ElementPool*

Returns intlstring

```
public innerText ()
```

Get innerText

Returns string|null

public onRender() → mixed

Abstract output method called by magic method The extending class must define this method

Returns mixed –

ElementPool

Qualified name Ouxsoft\PHPMarkup\Element\ElementPool

Implements *ElementPoolInterface*

class ElementPool

public add(*AbstractElement* \$element)

Add new element to pool

Parameters

- **\$element** (*AbstractElement*) –

public callRoutine(*string* \$routine)

Invoke a method if present in each element

Parameters

- **\$routine** (*string*) –

public count() → int

Returns a count of number of elements in collection

Returns int –

public getById(*[]*)

Get Element by placeholder id

Parameters

- **\$element_id** (*?string*) – Default: null

Returns AbstractElement|null

public getIterator()

Iterator to go through element pool

Returns ArrayIterator

public getPropertiesById(*string* \$element_id) → array

Get the public properties of a element using the elements ID

Parameters

- **\$element_id** (*string*) –

Returns array –

3.6.4 Exception namespace

Exception

Qualified name Ouxsoft\PHPMarkup\Exception\Exception

class **Exception**

public **__construct** (**[]**)

Exception constructor.

Parameters

- **\$log** – Default: null

public **getLog** ()

Returns log

Returns string|null

3.6.5 Factory namespace

ConcreteFactory

Qualified name Ouxsoft\PHPMarkup\Factory\ConcreteFactory

Implements *AbstractFactoryInterface*

class **ConcreteFactory**

public **makeBuilder** (*Container* & *\$container*)

Parameters

- & **\$container** (*Container*) –

Returns *BuilderInterface*

public **makeConfig** (*Container* & *\$container*) → *Configuration*

Parameters

- & **\$container** (*Container*) –

Returns *Configuration* –

public **makeDocument** (*Container* & *\$container*) → *Document*

Parameters

- & **\$container** (*Container*) –

Returns *Document* –

public **makeElementPool** ()

Returns *ElementPool*

public **makeEngine** (*Container* & *\$container*) → *Engine*

Parameters

- & **\$container** (*Container*) –

Returns *Engine* –

public **makeKernel** (*Container* & *\$container*) → *Kernel*

Parameters

- & **\$container** (*Container*) –

Returns *Kernel* –

ContainerFactory

Qualified name Ouxsoft\PHPMarkup\Factory\ContainerFactory

class ContainerFactory

static buildContainer (*AbstractFactoryInterface* \$abstractFactory)

Parameters

- **\$abstractFactory** (*AbstractFactoryInterface*) –

Returns Container

ProcessorFactory

Qualified name Ouxsoft\PHPMarkup\Factory\ProcessorFactory

class ProcessorFactory

static getInstance → Processor

Returns *Processor* –

3.6.6 ArgumentArray

Qualified name Ouxsoft\PHPMarkup\ArgumentArray

class ArgumentArray

public count () → int

Returns count of containers

Returns int –

public current () → mixed

Returns mixed –

public get () → array

Return container property

Returns array –

public key ()

Returns bool|float|int|mixed|string|null

public merge (\$array)

Merge array passed with container property

Parameters

- **\$array** –

public next ()

Returns bool|mixed|void

public offsetExists (*\$offset*) → bool

Check if item exists inside container

Parameters

- **\$offset** –

Returns bool –

public offsetGet (*\$offset*) → mixed

Get item from container

Parameters

- **\$offset** –

Returns mixed –

public offsetSet (*\$offset*, *\$value*)

Adds new item to array, if only one item in array then it will be a string

Parameters

- **\$offset** –
- **\$value** –

public offsetUnset (*\$offset*)

Remove item from container

Parameters

- **\$offset** –

public rewind()

public valid() → bool

Returns bool –

3.6.7 Configuration

Qualified name Ouxsoft\PHPMarkup\Configuration

Implements *ConfigurationInterface*

class Configuration

public __construct (*DocumentInterface* *\$document* [, *?string* *\$config_file_path*])
Configuration constructor

Parameters

- **\$document** (*DocumentInterface*) –
- **\$config_file_path** (*?string*) – Default: null

public addElement (*array* *\$element*)

Adds a element

Parameters

- **\$element** (*array*) –

public addElements (*array* *\$elements*)

Adds multiple elements at once

Parameters

- **\$elements** (*array*) –

public addRoutine (*array \$routine*)
Adds a routine

Parameters

- **\$routine** (*array*) –

public addRoutines (*array \$routines*)
Adds multiple routines at once

Parameters

- **\$routines** (*array*) –

public clearConfig ()
Clear config

public getElements () → *array*
Get elements

Returns *array* –

public getMarkup () → *string*
Get source

Returns *string* –

public getRoutines () → *array*
Get routines

Returns *array* –

public loadFile ([*]*)
load a configuration file

Parameters

- **\$filepath** (*string*) – Default: null

Returns *void*

public setConfig (*array \$config*)
Set entire config at once

Parameters

- **\$config** (*array*) –

public setMarkup (*string \$markup*)
Set LHTML source/markup

Parameters

- **\$markup** (*string*) –

3.6.8 Document

Qualified name Ouxsoft\PHPMarkup\Document

Implements *DocumentInterface*

class Document

public __construct ()

Document constructor.

public loadSource (string \$source) → bool

Loads source, which is in LHTML format, as DomDocument

A custom load page wrapper is required for server-side HTML5 entity support. Using `$this->loadHTMLFile` will removes HTML5 entities, such as

param string \$source

returns bool –

3.6.9 Engine

Qualified name Ouxsoft\PHPMarkup\Engine

Implements *EngineInterface*

class Engine

public __construct (DocumentInterface \$document, ElementPoolInterface \$element_pool)

Engine constructor.

Parameters

- **\$document** (*DocumentInterface*) –
- **\$element_pool** (*ElementPoolInterface*) –

public __toString () → string

Returns DomDocument property as HTML5

Returns string –

public callRoutine (array \$routine)

Call Hooks

Parameters

- **\$routine** (*array*) –

Returns bool-

public getDomElementByPlaceholderId (string \$element_id)

Gets DOMEelement using element_id provided

Parameters

- **\$element_id** (*string*) –

Returns DOMEelement|null

public getElementAncestorProperties (string \$element_id) → array

Get a Element ancestors' properties based on provided element_id DOMEelement's ancestors

Parameters

- **\$element_id** (*string*) –

Returns array –

public getElementArgs (*DOMElement \$element*) → *ArgumentArray*

Get DOMElement's attribute and child <args> elements and return as a single list items within the list are called args as they are passed as parameters to element methods

Parameters

- **\$element** (*DOMElement*) –

Returns *ArgumentArray* –

public getElementInnerXML (*string \$element_id*) → *string*

Get Element inner XML

Parameters

- **\$element_id** (*string*) –

Returns *string* –

public instantiateElements (*array \$lhtml_element*) → *bool*

Instantiates elements from DOMElement's found during Xpath query against DOM property

Parameters

- **\$lhtml_element** (*array*) –

Returns *bool* –

public queryFetch (*string \$query* [, *DOMElement \$node*]) → *mixed*

XPath query for class \$this->DOM property that fetches only first result

Parameters

- **\$query** (*string*) –
- **\$node** (*DOMElement*) – Default: null

Returns *mixed* –

public queryFetchAll (*string \$query* [, *DOMElement \$node*]) → *mixed*

XPath query for class \$this->DOM property that fetches all results as array

Parameters

- **\$query** (*string*) –
- **\$node** (*DOMElement*) – Default: null

Returns *mixed* –

public removeElements (*array \$lhtml_element*)

Removes elements from the DOM

Parameters

- **\$lhtml_element** (*array*) –

Returns *void*

public renderElement (*string \$element_id*) → *bool*

Within DOMDocument replace DOMElement with Element->class: __toString() output

Parameters

- **\$element_id** (*string*) –

Returns *bool* –

public replaceDomElement (*DOMElement \$element, string \$new_xml*)

Replaces DOMElement from property DOM with contents provided

Parameters

- **\$element** (*DOMElement*) –
- **\$new_xml** (*string*) –

public setType ([])

Set a value type to avoid Type Juggling issues and extend data types

Parameters

- **\$value** – Default: null
- **\$type** – Default: 'string'

Returns boolmixedstringnull

private instantiateElement (*DOMElement \$element, string \$class_name*) → bool

Instantiate a DOMElement as a Element using specified class_name

Parameters

- **\$element** (*DOMElement*) –
- **\$class_name** (*string*) –

Returns bool –

3.6.10 Entities

Qualified name Ouxsoft\PHPMarkup\Entities

class Entities

public fetchArray () → array

Download and encode entities from url

Returns array –

public fetchString () → string

Fetches entity string for use in DomDocument Doctype declaration

Returns string –

public getURL () → string

Get url of fetch point

Returns string –

static get → string

Get list of entities to pass to DOM. These will prevent the character from causing parse errors

Returns string –

3.6.11 Kernel

Qualified name Ouxsoft\PHPMarkup\Kernel

Implements *KernelInterface*

class Kernel

public __construct (*EngineInterface* \$engine, *BuilderInterface* \$builder, *ConfigurationInterface* \$config)
Kernel constructor.

Parameters

- **\$engine** (*EngineInterface*) –
- **\$builder** (*BuilderInterface*) –
- **\$config** (*ConfigurationInterface*) –

public build () → *Engine*
Calls Builder using parameters supplied

Returns *Engine* –

public getBuilder ()
Get builder

Returns *BuilderInterface*

public getConfig ()
Get config

Returns *ConfigurationInterface*

public setBuilder (*string* \$builder_class)
Set builder

Parameters

- **\$builder_class** (*string*) –

public setConfig (*ConfigurationInterface* \$config)
Set config

Parameters

- **\$config** (*ConfigurationInterface*) –

3.6.12 Processor

Qualified name Ouxsoft\PHPMarkup\Processor

class Processor

public __construct (*KernelInterface* \$kernel, *ConfigurationInterface* \$config)
Processor constructor.

Parameters

- **\$kernel** (*KernelInterface*) –
- **\$config** (*ConfigurationInterface*) –

public addElement (*array* \$element)
Add definition for processor LHTML element

Parameters

- **\$element** (*array*) –

```

public addElements (array $elements)
    Add definition for processor LHTML element

    Parameters

    • $elements (array) –

public addRoutine (array $routine)
    Add definition for processor LHTML routine

    Parameters

    • $routine (array) –

public addRoutines (array $routines)
    Add definition for processor LHTML routine

    Parameters

    • $routines (array) –

public getBuilder ()
    Get builder

    Returns BuilderInterface

public getConfig () → Configuration
    Get config

    Returns Configuration –

public getStatus () → bool
    Gets whether process runs or does not run

    Returns bool –

public loadConfig (string $filepath)
    Load config

    Parameters

    • $filepath (string) –

public parseBuffer ()
    Process output buffer

public parseFile (string $filepath) → string
    Process a file

    Parameters

    • $filepath (string) –

    Returns string –

public parseString (string $source) → string
    Process string

    Parameters

    • $source (string) –

    Returns string –

public setBuilder (string $builder_class)
    Set builder

    Parameters

```



```

    • $builder_class (string) –

public setConfig (ConfigurationInterface $config)
    Set config

Parameters

    • $config (ConfigurationInterface) –

Returns void

public setStatus (bool $status)
    Set whether process runs or does not run

Parameters

    • $status (bool) –

private parse () → string
    Parse using a Kernel to build an Engine

Returns string –

```

3.7 Code of Conduct

3.7.1 Our Pledge

In the interest of fostering an open and welcoming environment, we as contributors and maintainers pledge to making participation in our project and our community a harassment-free experience for everyone, regardless of age, body size, disability, ethnicity, sex characteristics, gender identity and expression, level of experience, education, socio-economic status, nationality, personal appearance, race, religion, or sexual identity and orientation.

3.7.2 Our Standards

Examples of behavior that contributes to creating a positive environment include:

- Using welcoming and inclusive language
- Being respectful of differing viewpoints and experiences
- Gracefully accepting constructive criticism
- Focusing on what is best for the community
- Showing empathy towards other community members

Examples of unacceptable behavior by participants include:

- The use of sexualized language or imagery and unwelcome sexual attention or advances
- Trolling, insulting/derogatory comments, and personal or political attacks
- Public or private harassment
- Publishing others' private information, such as a physical or electronic address, without explicit permission
- Other conduct which could reasonably be considered inappropriate in a professional setting

3.7.3 Our Responsibilities

Maintainers are responsible for clarifying the standards of acceptable behavior and are expected to take appropriate and fair corrective action in response to any instances of unacceptable behavior.

Maintainers have the right and responsibility to remove, edit, or reject comments, commits, code, wiki edits, issues, and other contributions that are not aligned to this Code of Conduct, or to ban temporarily or permanently any contributor for other behaviors that they deem inappropriate, threatening, offensive, or harmful.

3.7.4 Scope

This Code of Conduct applies both within project spaces and in public spaces when an individual is representing the project or its community. Examples of representing a project or community include using an official project e-mail address, posting via an official social media account, or acting as an appointed representative at an online or offline event. Representation of a project may be further defined and clarified by project maintainers.

3.7.5 Enforcement

Instances of abusive, harassing, or otherwise unacceptable behavior may be reported by contacting the Code of Conduct Committee at <conduct@ouxsoft.com>. All complaints will be reviewed and investigated and will result in a response that is deemed necessary and appropriate to the circumstances. The Code of Conduct Committee is obligated to maintain confidentiality with regard to the reporter of an incident. Further details of specific enforcement policies may be posted separately.

3.7.6 Attribution

This Code of Conduct is adapted from the Contributor Covenant, version 1.4, available at <https://www.contributor-covenant.org/version/1/4/code-of-conduct.html>

CHAPTER 4

Indices and tables

- `genindex`
- `search`

Symbols

__construct() (AbstractElement method), **18**
 __construct() (BuilderInterface method), **15**
 __construct() (Configuration method), **22**
 __construct() (Document method), **24**
 __construct() (DynamicPageBuilder method), **13**
 __construct() (Engine method), **24**
 __construct() (EngineInterface method), **16**
 __construct() (Exception method), **20**
 __construct() (Kernel method), **27**
 __construct() (KernelInterface method), **17**
 __construct() (Processor method), **27**
 __construct() (SearchIndexBuilder method), **13**
 __construct() (StaticPageBuilder method), **14**
 __invoke() (AbstractElement method), **18**
 __toString() (AbstractElement method), **18**
 __toString() (Engine method), **24**
 __toString() (EngineInterface method), **16**

A

AbstractElement (class), **18**
 AbstractFactoryInterface (interface), **14**
 add() (ElementPool method), **19**
 add() (ElementPoolInterface method), **15**
 addElement() (Configuration method), **22**
 addElement() (Processor method), **27**
 addElements() (Configuration method), **22**
 addElements() (Processor method), **27**
 addRoutine() (Configuration method), **23**
 addRoutine() (Processor method), **28**
 addRoutines() (Configuration method), **23**
 addRoutines() (Processor method), **28**
 ArgumentArray (class), **21**

B

build() (Kernel method), **27**
 build() (KernelInterface method), **17**
 buildContainer() (ContainerFactory method), **21**
 BuilderInterface (interface), **15**

C

callRoutine() (ElementPool method), **19**
 callRoutine() (ElementPoolInterface method), **15**
 callRoutine() (Engine method), **24**
 callRoutine() (EngineInterface method), **16**
 clearConfig() (Configuration method), **23**
 ConcreteFactory (class), **20**
 Configuration (class), **22**
 ConfigurationInterface (interface), **15**
 ContainerFactory (class), **21**
 count() (ArgumentArray method), **21**
 count() (ElementPool method), **19**
 count() (ElementPoolInterface method), **16**
 createObject() (BuilderInterface method), **15**
 createObject() (DynamicPageBuilder method), **13**
 createObject() (SearchIndexBuilder method), **13**
 createObject() (StaticPageBuilder method), **14**
 current() (ArgumentArray method), **21**

D

Document (class), **23**
 DocumentInterface (interface), **15**
 DynamicPageBuilder (class), **13**

E

ElementPool (class), **19**
 ElementPoolInterface (interface), **15**
 Engine (class), **24**
 EngineInterface (interface), **16**
 Entities (class), **26**
 Exception (class), **19**

F

fetchArray() (Entities method), **26**
 fetchString() (Entities method), **26**

G

get() (ArgumentArray method), **21**
 get() (Entities method), **26**

[getArgByName\(\)](#) (AbstractElement method), [18](#)
[getArgs\(\)](#) (AbstractElement method), [18](#)
[getBuilder\(\)](#) (Kernel method), [27](#)
[getBuilder\(\)](#) (KernelInterface method), [17](#)
[getBuilder\(\)](#) (Processor method), [28](#)
[getById\(\)](#) (ElementPool method), [19](#)
[getById\(\)](#) (ElementPoolInterface method), [16](#)
[getConfig\(\)](#) (Kernel method), [27](#)
[getConfig\(\)](#) (KernelInterface method), [17](#)
[getConfig\(\)](#) (Processor method), [28](#)
[getDomElementByPlaceholderId\(\)](#) (Engine method), [24](#)
[getDomElementByPlaceholderId\(\)](#) (EngineInterface method), [16](#)
[getElementAncestorProperties\(\)](#) (Engine method), [24](#)
[getElementAncestorProperties\(\)](#) (EngineInterface method), [16](#)
[getElementArgs\(\)](#) (Engine method), [24](#)
[getElementArgs\(\)](#) (EngineInterface method), [16](#)
[getElementInnerXML\(\)](#) (Engine method), [25](#)
[getElementInnerXML\(\)](#) (EngineInterface method), [16](#)
[getElements\(\)](#) (Configuration method), [23](#)
[getId\(\)](#) (AbstractElement method), [18](#)
[getInstance\(\)](#) (ProcessorFactory method), [21](#)
[getIterator\(\)](#) (ElementPool method), [19](#)
[getIterator\(\)](#) (ElementPoolInterface method), [16](#)
[getLog\(\)](#) (Exception method), [20](#)
[getMarkup\(\)](#) (Configuration method), [23](#)
[getObject\(\)](#) (BuilderInterface method), [15](#)
[getObject\(\)](#) (DynamicPageBuilder method), [13](#)
[getObject\(\)](#) (SearchIndexBuilder method), [13](#)
[getObject\(\)](#) (StaticPageBuilder method), [14](#)
[getPropertiesById\(\)](#) (ElementPool method), [19](#)
[getPropertiesById\(\)](#) (ElementPoolInterface method), [16](#)
[getRoutines\(\)](#) (Configuration method), [23](#)
[getStatus\(\)](#) (Processor method), [28](#)
[getURL\(\)](#) (Entities method), [26](#)

I

[innerText\(\)](#) (AbstractElement method), [18](#)
[instantiateElement\(\)](#) (Engine method), [26](#)
[instantiateElements\(\)](#) (Engine method), [25](#)
[instantiateElements\(\)](#) (EngineInterface method), [16](#)

K

[Kernel](#) (class), [26](#)
[KernelInterface](#) (interface), [17](#)
[key\(\)](#) (ArgumentArray method), [21](#)

L

[loadConfig\(\)](#) (Processor method), [28](#)
[loadFile\(\)](#) (Configuration method), [23](#)
[loadSource\(\)](#) (Document method), [24](#)

M

[makeBuilder\(\)](#) (AbstractFactoryInterface method), [14](#)
[makeBuilder\(\)](#) (ConcreteFactory method), [20](#)
[makeConfig\(\)](#) (AbstractFactoryInterface method), [14](#)
[makeConfig\(\)](#) (ConcreteFactory method), [20](#)
[makeDocument\(\)](#) (AbstractFactoryInterface method), [14](#)
[makeDocument\(\)](#) (ConcreteFactory method), [20](#)
[makeElementPool\(\)](#) (AbstractFactoryInterface method), [14](#)
[makeElementPool\(\)](#) (ConcreteFactory method), [20](#)
[makeEngine\(\)](#) (AbstractFactoryInterface method), [14](#)
[makeEngine\(\)](#) (ConcreteFactory method), [20](#)
[makeKernel\(\)](#) (AbstractFactoryInterface method), [15](#)
[makeKernel\(\)](#) (ConcreteFactory method), [20](#)
[merge\(\)](#) (ArgumentArray method), [21](#)

N

[next\(\)](#) (ArgumentArray method), [21](#)

O

[offsetExists\(\)](#) (ArgumentArray method), [21](#)
[offsetGet\(\)](#) (ArgumentArray method), [22](#)
[offsetSet\(\)](#) (ArgumentArray method), [22](#)
[offsetUnset\(\)](#) (ArgumentArray method), [22](#)
[onRender\(\)](#) (AbstractElement method), [18](#)

P

[parse\(\)](#) (Processor method), [29](#)
[parseBuffer\(\)](#) (Processor method), [28](#)
[parseFile\(\)](#) (Processor method), [28](#)
[parseString\(\)](#) (Processor method), [28](#)
[Processor](#) (class), [27](#)
[ProcessorFactory](#) (class), [21](#)

Q

[queryFetch\(\)](#) (Engine method), [25](#)
[queryFetch\(\)](#) (EngineInterface method), [17](#)
[queryFetchAll\(\)](#) (Engine method), [25](#)
[queryFetchAll\(\)](#) (EngineInterface method), [17](#)

R

[removeElements\(\)](#) (Engine method), [25](#)
[removeElements\(\)](#) (EngineInterface method), [17](#)
[renderElement\(\)](#) (Engine method), [25](#)
[renderElement\(\)](#) (EngineInterface method), [17](#)
[replaceDomElement\(\)](#) (Engine method), [25](#)
[replaceDomElement\(\)](#) (EngineInterface method), [17](#)
[rewind\(\)](#) (ArgumentArray method), [22](#)

S

[SearchIndexBuilder](#) (class), [13](#)
[setBuilder\(\)](#) (Kernel method), [27](#)
[setBuilder\(\)](#) (KernelInterface method), [17](#)

[setBuilder\(\) \(Processor method\), 28](#)
[setConfig\(\) \(Configuration method\), 23](#)
[setConfig\(\) \(Kernel method\), 27](#)
[setConfig\(\) \(KernelInterface method\), 18](#)
[setConfig\(\) \(Processor method\), 29](#)
[setMarkup\(\) \(Configuration method\), 23](#)
[setStatus\(\) \(Processor method\), 29](#)
[setType\(\) \(Engine method\), 26](#)
[setType\(\) \(EngineInterface method\), 17](#)
[StaticPageBuilder \(class\), 14](#)

V

[valid\(\) \(ArgumentArray method\), 22](#)