Pimple Config Service Provider Documentation

Release 1.3

NunoPress LLC

Nov 01, 2017

Contents

Pimple Config Service Provider based on Illuminate Config package for Silex Microframework or any Pimple Container project's.

Tip: The Service Provider is installable with Composer:

```
composer require nunopress/pimple-config-service-provider
```

CHAPTER 1

Parameters

1.1 config.path

Path defined to find every php files inside for loading.

1.2 config.environment (*optional*)

Search before in the defined path and then in the environment path (`config.path/config.environment` *format*). The service use `array_replace_recursive` for help the developers to change only what you need in the different environment instead to write again all the configuration set.

Here a simple example:

```php
<?php

// config/view.php
return [
    'twig' => [
        'path' => realpath(__DIR__ . '/../views'),
        'options' => [
            'debug' => false,
            'cache' => realpath(__DIR__ . '/../../storage/cache/twig')
        ]
    ]
];

// config/development/view.php
return [
    'twig' => [
        'options' => [
            'debug' => true,
            'cache' => false
        ]
```

```
    ]
];

// RESULT
[
    'twig' => [
        'path' => realpath(__DIR__ . '/../views'),
        'options' => [
            'debug' => true,
            'cache' => false
        ]
    ]
]
```

1.3 config.merge_factory (*optional*)

You can configure your merge method instead to use the default merge factory `array_replace_recursive`:

```
<?php

$app['config.merge_factory'] = $app->share($app->protect('config.merge_factory',␣
↪function (array $old, array $new) {
    return array_merge($old, $new);
}));
```

CHAPTER 2

Services

For access to config keys you need to use the `filename` (*without extension*) before every config keys, example:

```php
<?php

// config/view.php
return [
    'test' => 'yep'
];

// Access to test key
$app['config']->get('view.test'); // Result: yep
```

2.1 config

The `Illuminate\Config\Repository` instance. The main way to interact with Config.

CHAPTER 3

Registering

```php
<?php

$app->register(new NunoPress\Pimple\Config\Provider\ConfigServiceProvider(), [
    'config.path' => __DIR__ . '/config',
    'config.environment' => ($app['debug']) ? 'dev' : 'prod'
]);
```

CHAPTER 4

Usage

The Config provider provides a `config` service:

```php
<?php

$app->get('/hello', function () use ($app) {
    $name = $app['config']->get('app.name', 'NunoPress');

    return 'Hello ' . $name . '!!';
});
```

Note: Read the Config reference for the Illuminate Config document to learn more about the various Config functions.

CHAPTER 5

Traits

`NunoPress\Pimple\Config\Application\ConfigTrait` adds the following shortcuts:

5.1 config

Access to Config object for retrieve the `key` requested, for the second param you can define a default value.

```
<?php

$name = $app->config('app.name', 'NunoPress');
```

Define this trait in your `Application` class:

```
<?php

class App extends \Silex\Application
{
    use \NunoPress\Pimple\Config\Application\ConfigTrait;
}

$app = new App();

$name = $app->config('app.name', 'NunoPress');
```

CHAPTER 6

Customization

You can configure the Config object before using it by extending the `config` service:

```php
<?php

$app['config'] = $app->share($app->extend('config', function ($config, $app) {
    // Instead to have separate the config items you can share it in the current container
    $items = $config->all();

    foreach ($items as $name => $item) {
        $app[$name] = $item;
    }

    return $config;
}));
```