Clastic Documentation

Release latest

April 13, 2016

Contents

Clastic is a PHP CMS. It is written on top of symfony. It allows developers to build better applications without the fuss of creating a backoffice.

Clastic can be used as a backend for any type of application, from personal blog to high traffic mobile app api.

CHAPTER 1

User guide

1.1 Overview

1.1.1 Requirements

1. PHP 5.4.0
2. Composer
3. nodejs and npm
4. gulp

Note: For detailed instructions on how to install the dependencies. Please refer to google.

1.1.2 Installation

The recommended way to start your projects with Composer. Composer is a dependency management tool for PHP that allows you to declare the dependencies your project needs and installs them into your project.

If you get an error, remove the word 'php' in front of the line. This depends on your php installation.

```
$ composer create-project clastic/standard-edition path/to/install -s dev
```

If you want more information. See the Installation documentation.

Development

You can now start your developement environment.

```
$ make dev
```

You can now access your website at http://127.0.0.1:8000/ and the backoffice at http://127.0.0.1:8000/admin/. You can logon using admin as username and secret as password. Note you should immediately change this.

Have fun!

1.1.3 License

Licensed using the MIT license.

> Copyright (c) 2015 Dries De Peuter
>
> Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense, and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:
>
> The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.
>
> THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

1.2 Installation

1.2.1 Preparations

Clastic has a few dependencies to get the project up-and-running.

- PHP ~5.4
- Composer
- node.js
- npm
- gulp
- bower

For detail information about the dependencies see the *Installing dependencies* section.

1.2.2 Create a project

The recommended way to start your projects with Composer. Composer is a dependency management tool for PHP that allows you to declare the dependencies your project needs and installs them into your project.

If you get an error, remove the word 'php' in front of the line. This depends on your php installation.

```
$ composer create-project clastic/standard-edition path/to/install -s dev
```

Note: If the installation fails, check if the `composer` executable works. You might need to use `php composer.phar create-project path/to/install -s dev` if you installed composer locally.

During the installation you will be asked to fill in some parameters. Make sure all the values are correct.

You should now have a project located in `path/to/install`. Please go to this folder and continue.

```
$ cd path/to/install
```

Clastic needs multiple assets to be installed in the correct place. The standard project contains a *Makefile* to simplify this process.

```
make install
```

Note: Now is a good time to have a first commit!

Development

You can now start your development environment.

```
$ make dev
```

You can now access your website at http://127.0.0.1:8000/ and the backoffice at http://127.0.0.1:8000/admin/. You can logon using admin as username and secret as password. Note you should immediately change this.

Have fun!

1.2.3 Production

For any information about setting up a production server. Please refer to the symfony documentation.

1.2.4 Installing dependencies

PHP

To install PHP refer to your option of choice. Installation instructions are different for every platform. See the PHP Installation Guide. Make sure your PHP version is higher than *5.4.0*.

Composer

Composer is a dependency manager for PHP.

Use the following code to install, see the Official installation documentation for detailed information.

```
$ curl -sS https://getcomposer.org/installer | php
```

Node.js

Install node.js using the official installers available at https://nodejs.org/download/.

NPM

Npm comes included with node.js.

Gulp

Gulp is a streaming build system. It is used to build assets.

Use the following code to install, see the Official installation documentation for detailed information.

```
$ npm install --global gulp
```

Bower

Use the following code to install, see the Official installation documentation for detailed information.

```
$ npm install -g bower
```

1.3 Entities

1.3.1 Overview

Clastic uses doctrine to handle all data. It also give you an *Node* approach. Nodes are entities that have basic information.

- title (string)
- user (User)
- create (DateTime)
- changed (DateTime)
- publication (NodePublication)
- available (boolean)
- publishedFrom (DateTime)
- publishedTill (DateTime)

1.3.2 Create

You can use whatever approach you want to generate the base entity. The simplest way is to use the doctrine generator.

```
$ php app/console doctrine:generate:entity
```

Fill in all the field you need.

Note: You don't need to define fields like title, author, ... when you choose to make them Nodes.

Once you created your entity you need to alter the generated code.

Object

Change the *NodeReferenceInterface* to your freshly generator entity. You can also use the *NodeReferenceTrait* which helps you implement the interface.

ex:

```
<?php
namespace Clastic\BlogBundle\Entity;

use Clastic\NodeBundle\Entity\Node;
use Clastic\NodeBundle\Node\NodeReferenceInterface;

class Blog implements NodeReferenceInterface
{
    use NodeReferenceTrait;

    // ...
}
```

Definition

Add the following code to your xml definition.

```
<many-to-one field="node" target-entity="Clastic\NodeBundle\Entity\Node">
  <cascade><cascade-all/></cascade>
  <join-column name="node_id" referenced-column-name="id" />
</many-to-one>
```

Generate

Now your entity is ready to generate a table. Do so by calling the following command.

```
$ php app/console doctrine:schema:update [--force]
```

1.4 Modules

1.4.1 Overview

Clastic uses modules to expose data to the backend. To generate a module you must first create an entity.

See Entities for more information on the entities.

Note: The generator only supports Node modules at the moment.

```
$ php app/console clastic:generate:module
```

After generation your modules will be available in the backoffice.

1.5 Bundles

1.5.1 Overview

Clastic is composed of different modules all serving a single purpose.

1.5.2 Base Bundles

Core

The core bundle is heart of the framework. It defines the basic flow and dependencies.

Backoffice

The backoffice bundle contains the basic features of the backoffice including the form types.

- DatePicker
- EntityHidden
- EntityMultiSelect
- Fieldset
- Link
- MultiSelect
- Tree
- Wysiwyg

User

The user bundle provides everything to manage users.

Security

The security bundle adds security to the backoffice.

Node

The node bundle provides extra metadata for your data. It allows other bundles to dynamically extend your data.

Front

The front bundle provides basic features for the public side of your application.

1.5.3 Feature Bundles

Alias

The alias bundle provides aliases for your nodes. This will allow you to create a user-friendly url for every node.

Block

The block bundle provides an easy interface for developers to allow administrators to change non-content with ease.

Taxonomy

The taxonomy bundle provides basic tools to simplify creation of nested and sortable content.

1.5.4 Module Bundles

Blog

The blog bundle provides a simple module to create a very basic blog.

Media

The media bundle provides basic tools to handle media in your application.

Menu

The menu bundle provides a module to handle your menu structures.

Text

The text bundle provides a module to create simple content pages.

1.5.5 Tool Bundles

Generator

The generator bundle provides a generator to increase development times.

1.5.6 Contrib Bundles

You can find more bundles in the Contrib Bundles.

1.6 Contrib Bundles

1.6.1 Overview

Clastic is composed of different bundles all serving a single purpose. Not all features are provided by the core installation. Clastic has a set of bundles that are not for the general public, but may be useful for you.

You can find these bundles at the clastic contrib page.

1.6.2 Publishing

Sharing bundles with others is a nice thing to do. If you would like your bundle to be featured in the contrib list, create an issue in the issue queue. Your bundle will than be evaluated by the Clastic team. If the bundle is approved you can tranfer ownership of the bundle. You will than be part of the Clastic team and will be promoted as maintainer of the bundle.

Naming

Note: Please don't use `clastic` as the vendor name before your package has been approved.

Say you are building a catalog module. Don't use `clastic/catalog-bundle` in the `composer.json`. Keep your package under your own vendor name (`myname/catalog-bundle`). Once accepted you can choose to change or keep the name of the package.