TwitalBundle Documentation

Release 1.0-alpha

Asmir Mustafic

Jan 18, 2019

Contents

TwitalBundle is a Symfony bundle that integrates the Twital template engine into Synfony2. This enables you to use all Twig/Symfony functionalities with the Twital template engine language syntax.

To learn more about Twital, you can read the dedicated documentation.

CHAPTER 1

Install

The recommended way to install TwitalBundle is via Composer:

```
composer require 'goetas/twital-bundle'
```

If you are using SymfonyFlex, the bundle will be automatically enabled and configured, otherwise follow the next steps.

CHAPTER 2

Enable the bundle

To enable the bundle in the kernel:

```php
<?php
// app/AppKernel.php

public function registerBundles()
{
    $bundles = array(
        // ...
        new Goetas\TwitalBundle\GoetasTwitalBundle(),
        //..
    );
}
```

CHAPTER 3

Configure

If you are using SymfonyFlex, the bundle is auto-configured and you can skip this step.

If you are using the symfony templating component (available in Symfony 2.x and 3.x), you have to enable the `twital` template engine inside your `config.yml`.

```
framework:
    templating:
        engines: ['twig', 'twital']
```

CHAPTER 4

Optional Configurations

Here are some optional configurations for the bundle.

4.1 Source Adapters

By default Twital parses `*.html.twital`, `*.xml.twital` and `*.xhtml.twital` files. If you want you can customize the file types automatically parsed by Twital.

```
goetas_twital:
    # extra file extension matching
    source_adapter:
        - { service: twital.source_adapter.xml, pattern: ['/\.xml\.twital$/', '/\.
↪atom\.twital$/'] }
        - { service: twital.source_adapter.html5, pattern: ['/\.html\.twital$/', '/\.
↪htm\.twital$/'] }
        - { service: twital.source_adapter.xhtml, pattern: ['/\.xhtml\.twital$/'] }
```

Twital comes with the following source adapters that you can use to parse your template files:

- `twital.source_adapter.html5`: used for most of the HTML-style templates
- `twital.source_adapter.xml`: used for strictly XML-compliant templates
- `twital.source_adapter.xhtml`: used for strictly XHTML-compliant templates (similar to XML but with some XHTML customizations)

4.2 Full Twig Compatibility

The following template is a valid Twig template, but is not a valid HTML5 document (the `div` tag can contain only attributes). Because of it, the Twital source adapters will not be able to parse the template.

```
<div {% if foo %} class="row" {% endif %}>
    Hello World
</div>
```

You can also enable a full Twig compatibility mode to allow this kind of templates.

```
goetas_twital:
    full_twig_compatibility: true
```

CHAPTER 5

Integration

`TwitalBundle` comes with all features that are already supported by TwigBundle (forms, translations, assetic, routing, etc).

CHAPTER 6

Contributing

This is an open source project - contributions are welcome. If your are interested, you can contribute to documentation, source code, test suite or anything else!

To start contributing right now, go to https://github.com/goetas/twital-bundle and fork it!

You can read some tips to improve your contributing experience looking into https://github.com/goetas/twital-bundle/blob/master/CONTRIBUTING.md present inside the root directory of Twital GIT repository.

CHAPTER 7

Contents

7.1 Add your own Twital extension

The recommended way to add your TwitalExtension to `Twital` instance is registering it using the Symfony2 dependency injection system.

You have to add your extensions as services and tag them with the `twital.extension` tag.

Depending on your preferences, you can choose which syntax to adopt.

Using XML:

```
<service id="my.extension" class="...myExtensionClass...">
    <tag name="twital.extension" />
</service>
```

Using YAML:

```
services:
    my.extension:
        class: ...myExtensionClass...
        tags:
            - { name: twital.extension }
```

Using PHP:

```
<?php

 $container
    ->register('my.extension', '...myExtensionClass...')
    ->addTag('twital.extension')
;
```

Once you have added one of this configurations to your bundle, your extension should be available.

7.2 Add your own source adapter

The recommended way to add your TwitalSourceAdapter to `TwitalLoader` is registering it using the Symfony2 dependency injection system.

Depending on your preferences, you can choose which syntax to adopt.

Using XML:

```
<service id="my.source_adapter" class="...mySourceAdapterClass...">

</service>
```

Using YAML:

```
services:
    my.source_adapter:
        class: ...mySourceAdapterClass...
```

Using PHP:

```
<?php

 $container
    ->register('my.source_adapter', '...mySourceAdapterClass...')
;
```

Once you have added one of this configurations to your bundle, choose which file name pattern will activate the loader. To do this you have to edit your `config.yml`.

```
goetas_twital:
    source_adapter:
        my.source_adapter: ['/\.myext1\.twital$/', '/\.myext2\.twital$/']
```

7.2.1 Alternative way to add your source adapter

If you prefer to use the Symfony2 service tagging system, you can also use the following method:

You have to add your adapters as services and tag them with `twital.source_adapter`, and you also have to specify the `pattern` attribute.

Using XML:

```
<service id="my.source_adapter" class="...mySourceAdapterClass...">
    <tag name="twital.source_adapter" pattern="/\.xml\.twtal$/i" />
</service>
```

Using YAML:

```
services:
    my.source_adapter:
        class: ...mySourceAdapterClass...
        tags:
            - { name: twital.source_adapter, pattern: '/\.xml\.twtal$/i' }
```

Using PHP:

```
<?php

 $container
    ->register('my.source_adapter', '...mySourceAdapterClass...')
    ->addTag('twital.source_adapter', array('pattern' => '/\.xml\.twtal$/i'))
;
```

7.3 Tags provided

Here you can find the documentation for all attributes provided by `TwitalBundle` for a better integration of Symfony2 Functionalities.

7.3.1 `trans`

The `t:trans` attribute is an alias of the `trans` Symfony tag and allows you to translate the content of a node.

Let's see how does it work:

```
<div t:trans="">
    Hello world
</div>
```

This option will allow Symfony to extract and translate the "Hello world" sentence.

Of course, you can also use any variable inside your text.

```
<div t:trans="{'%name%':'John'}">
    Hello %name%
</div>
```

You can also specify different domains for your translations.

```
<div t:trans="{'%name%':'John'}, 'app'">
    Hello %name%
</div>
```

Tip: See here http://symfony.com/it/doc/current/book/translation.html to learn more about the Symfony translation system.

7.3.2 `trans-n`

The `t:trans-n` attribute is an alias of the `transchoice` Symfony tag and allows you to translate the content of a node with plurals.

Let's see how does it work:

```
<div t:trans-n="applesCount">
    {0} There are no apples|{1} There is one apple|]1,Inf] There are %count% apples
</div>
```

Of course, you can also use variables in your text.

```
<div t:trans-n="applesCount, {'%name%':'John'}">
    {0} %name% don't like apples|{1} %name% is eating one apple|]1,Inf] %name% is␣
↪eating %count% apples
</div>
```

You can also specify different domains for your translations.

```
<div t:trans-n="applesCount, {'%name%':'John'}, 'app'">
    {0} %name% don't like apples|{1} %name% is eating one apple|]1,Inf] %name% is␣
↪eating %count% apples
</div>
```

Tip: See here http://symfony.com/it/doc/current/book/translation.html to learn more about the Symfony translation system.

7.3.3 trans-attr

The `t:trans-attr` attribute is an alias of the `trans` Symfony tag, but it works only with HTML/XML attributes, and allows you to translate the content of one or more attributes.

The main advantage of `t:trans-attr` is the preservation of the original document structure: you do not need to change the *value* attribute with dirty code.

Let's see how does it work:

```
<inpiut value="Apple" t:trans-attr="value"/>
```

This option will allow Symfony to extract and translate the "Apple" word.

Of course, you can also use variables inside your text.

```
<inpiut value="The pen is on the %place%" t:trans-attr="value:{'%place%':'table'}"/>
```

You can also translate more than one attribute on the same node.

```
<inpiut
    value="The pen is on the %place%"
    title="My favorite color is %color%"
    t:trans-attr="value:{'%place%':'table'}, title:{'%color%':'red'}"/>
```

You can also specify different domains for your translations.

```
<inpiut
    value="The pen is on the %place%"
    title="My favorite color is %color%"
    t:trans-attr="value:[{'%place%':'table'}, 'app'], title:[{'%color%':'red'},
↪'colors']"/>
```

Tip: See here http://symfony.com/it/doc/current/book/translation.html to learn more about the Symfony translation system.

7.3.4 trans-attr-n

The `t:trans-attr-n` attribute is an alias of the `transchoice` Symfony tag, but it works only with HTML/XML attributes, and allows you to translate the content of one or more attribute using also plural forms.

Let's see how does it work:

```
<inpiut
    value="There is one apple|There are %count% apples"
    t:trans-attr-n="value:[3, {'%count%':3}]" />
```

This option will allow Symfony to extract and translate the "There is one apple", "There are %count% apples", and "%count% apples" sentences.

Of course, you can also use variables in your text.

```
<inpiut
    value="%name% is eating one apple|%name% is eating %count% apples"
    t:trans-attr-n="value:[3, {'%count%':3, '%name%':'John'}]" />
```

You can also translate more than one attribute on the same node.

```
<inpiut
    value="%name% is eating one apple|%name% is eating %count% apples"
    title="There is one apple|There are %count% apples"
    t:trans-attr-n="value:[3, {'%count%':3, '%name%':'John'}], title:[3, {'%count%':3}
↪]" />
```

You can also specify different domains for your translations.

```
<inpiut
    value="%name% is eating one apple|%name% is eating %count% apples"
    t:trans-attr-n="value:[3, {'%count%':3, '%name%':'John'}, 'app']" />
```

You can also combine plural and non-plural translations

```
<inpiut
    value="%name% is eating one apple|%name% is eating %count% apples"
    title="The pen is on the %place%"

    t:trans-attr="title:{'%place%':'table'}"
    t:trans-attr-n="value:[3, {'%count%':3, '%name%':'John'}]" />
```

Tip: See here http://symfony.com/it/doc/current/book/translation.html to learn more about the Symfony translation system.

CHAPTER 8

Note

I'm sorry for the *terrible* english fluency used inside the documentation, I'm trying to improve it. Pull Requests are welcome.