Using regular expressions in URL rules

One of the hidden features of the Yii URL router is that you can use regular expressions that are pretty powerful for handling strings.

Getting ready

1. Create a new application using the Composer package manager, as described in the official guide at <http://www.yiiframework.com/doc-2.0/guide-start-installation.html>.
2. In your @app/controllers directory, create Postcontroller. php using the following:

<?php

namespace app\controllers;

use yii\helpers\Html; use yii\web\Controller;

class PostController extends Controller {

public function actionView($alias)

{

return $this->renderContent(Html::tag('h2',

'Showing post with alias ' . Html::encode($alias)

));

}

public function actionIndex($type = 'posts', $order = 'DESC')

{

return $this->renderContent(Html::tag('h2',

'Showing ' . Html::encode($type) . ' ordered ' . Html::encode($order)

));

}

public function actionHello($name)

{

return $this->renderContent(Html::tag('h2',

'Hello, ' . Html::encode($name) . '!'

));

}

}

This is our application controller that we are going to access using our custom URLs.

1. Configure your application server to use clean URLs. If you are using Apache with mod\_rewrite and AllowOverride turned on, then you should add the following lines to the . htaccess file under your @web folder:

Options +FollowSymLinks IndexIgnore \*/\*

RewriteEngine on

* if a directory or a file exists, use it directly RewriteCond %{REQUEST\_FILENAME} !-f RewriteCond %{REQUEST\_FILENAME} !-d
* otherwise forward it to index.php RewriteRule . index.php

How to do it...

We want our Postcontroller action to accept parameters according to some specified rules and give the 404 not found HTTP response for all parameters that do not match. In addition, post/index should have an alias URL archive.

Add the following config of the urlManager component to @app/config/web . php:

'components' => [

// ..

'urlManager' => [

'enablePrettyUrl' => true,

'rules' => [

'post/<alias:[-a-z]+>' => 'post/view',

'<type:(archive|posts)>' => 'post/index',

'<type:(archive|posts)>/<order:(DESC|ASC)>' => 'post/index',

'sayhello/<name>' => 'post/hello',

]

],

// ..

],

The following URLs will be successful:

* <http://yii-book.app/post/test>
* <http://yii-book.app/posts>
* <http://yii-book.app/archive>
* <http://yii-book.app/posts/ASC>
* <http://yii-book.app/sayhello>

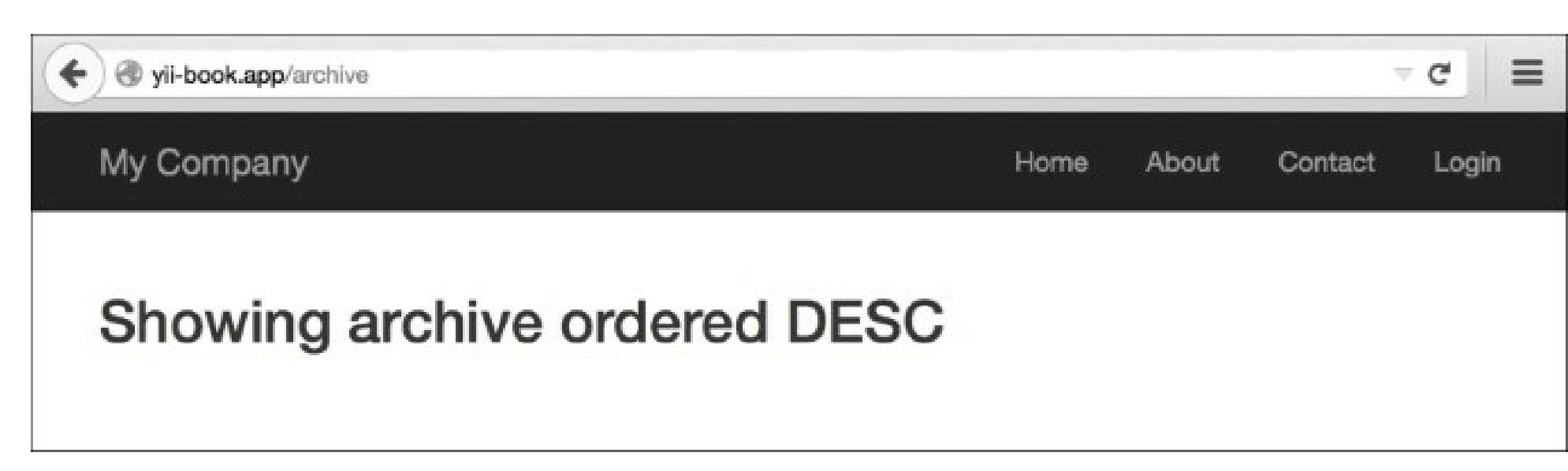
The following URLs will fail:

* http: //yii-book.app/archive/test
* http: //yii-book.app/post/another\_post

The following screenshot shows that the URL http: //yii-book. app/post/test has run successfully:



The following screenshot shows that the URL http: //yii-book. app/archive has run successfully too:



The following screenshot shows that the URL http: //yii-book. app/archive/test did not run successfully and encountered an error:



How it works...

You can use regular expressions in both the parameter definition and the rest of the rule. Let’s read our rules one by one:

'post/<alias:[-a-z]+>' => 'post/view',

The alias parameter should contain one or more English letters or a dash. No other symbols are allowed. '(posts|archive)' => 'post/index',

'(posts|archive)/<order:(DESC|ASC)>' => 'post/index',

Both posts and archive lead to post/index. The order parameter can only accept two values—desc

and asc:

'sayhello/<name>' => 'post/hello',

You should specify the name part but there are no restrictions on what characters are allowed. Note that regardless of the rule used, the developer should never assume that the input data is safe.



There’s more...

To learn more about regular expressions, you can use the following sources:

* <http://www.php.net/manual/en/reference.pcre.pattern.syntax.php>
* ***Mastering Regular Expressions, Jeffrey Fried!*** available at <http://regex.info/>.

See also

* The ***Configuring URL rules*** recipe