Defining multiple layouts

Most applications use a single layout for all their views. However, there are situations when multiple layouts are needed. For example, an application can use different layouts on different pages: two additional columns for blogs, one additional column for articles, and no additional columns for portfolios.

Getting ready

Create a new application using the Composer package manager, as described in the official guide at [http://www.yiiframework.com/doc-2.0/guide-startinstallat]'on.html](http://www.yiiframework.com/doc-2.0/guide-startinstallation.html).

How to do it...

1. Create two layouts in views/layouts: blog and articles. Blog will contain the following code:

<?php $this->beginContent('//layouts/main')?>

<div>

<?= $content ?>

</div>

<div class="sidebar tags">

<ul>

<li><a href="#php">PHP</a></li>

<li><a href="#yii">Yii</a></li>

</ul>

</div>

<div class="sidebar links">

<ul>

<li><a href="<http://yiiframework.com/>">

Yiiframework</a></li>

<li><a href="[http://php.net/">PHP</a></li](http://php.net/%22%3ePHP%3c/a%3e%3c/li)>

</ul>

</div>

<?php $this->endContent()?>

1. Articles will contain the following code:

<?php

/\* @var $this yii\web\View \*/

?>

<?php $this->beginContent('@app/views/layouts/main.php'); ?>

<div class="container">

<div class="col-xs-8">

<?= $content ?>

</div>

<div class="col-xs-4">

<h4>Table of contents</h4>

<ol>

<li><a href="#intro">Introduction</a></li>

<li><a href="#quick-start">Quick start</a></li>

<li>..</li>

</ol>

</div>

</div>

<?php $this->endContent() ?>

1. Create a view file, views/site/content. php, as follows:

<h1>Title</h1>

<p>Lorem ipsum dolor sit amet, consectetur adipisicing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur.</p>

1. Create three controllers named BlogController, ArticleController, and PortfolioController, with index actions in all three. The content of the controllers/BlogController. php file is as follows:

<?php

namespace app\controllers;

use yii\web\Controller;

class BlogController extends Controller {

public $layout = 'blog';

public function actionIndex()

{

return $this->render('//site/content');

}

}

1. The content of the controllers/ArticleController. php file is as follows:

<?php

namespace app\controllers; use yii\web\Controller;

class ArticleController extends Controller {

public $layout = 'articles';

public function actionIndex()

{

return $this->render('//site/content');

}

}

1. The content of the controllers/PortfolioController . php file is as follows:

<?php

namespace app\controllers; use yii\web\Controller;

class PortfolioController extends Controller {

public function actionIndex()

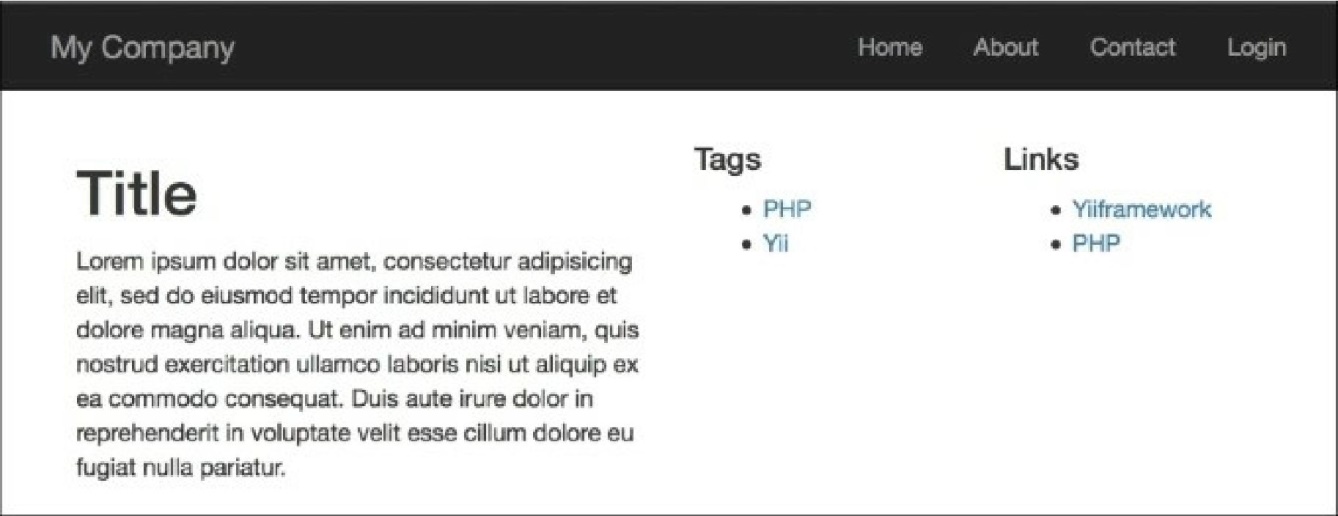
{

return $this->render('//site/content');

}

}

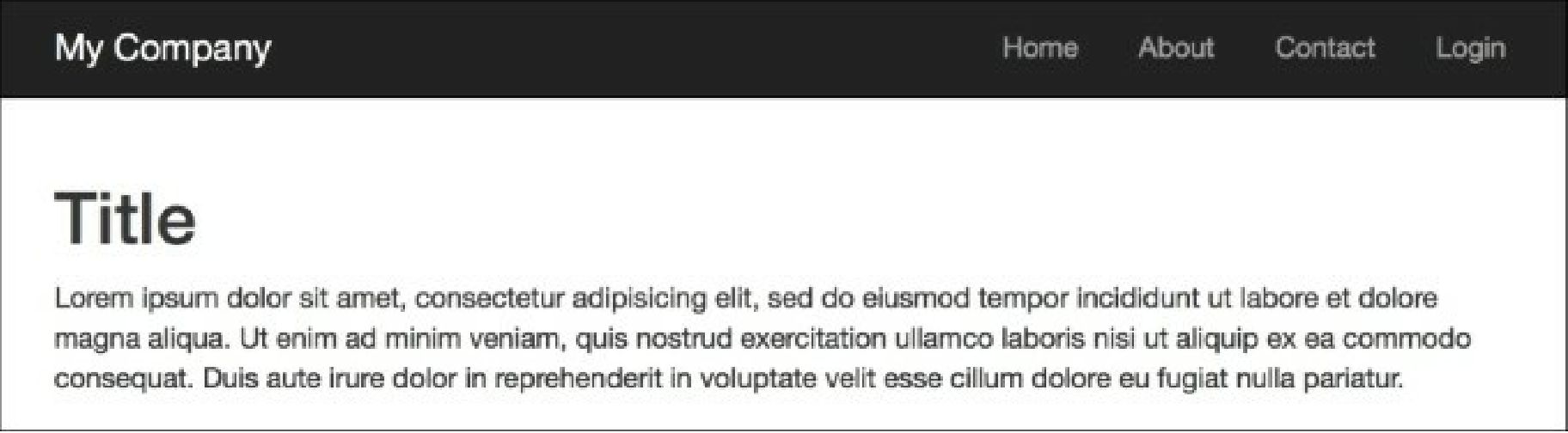
1. Now try running http : //yii-book . app/?r=blog/index:



8. Then try running http://yii-book.app/?r=article/index:



9. Finally, try running http: //yii-book . app/?r=portfolio/index:



How it works...

We defined two additional layouts for the blog and articles. As we don’t want to copy and paste common parts from the main layout, we apply additional layout decorators using $this->beginContent and $this->endContent.

So, we use a view rendered inside the articles layout as the main layout’s $content.

See also

* The [http://www.yiiframework.com/doc-2.0/guide-structure-views.html#nested-layouts](http://www.yiiframework.com/doc-2.0/guide-structure-views.html%23nested-layouts) URL provides more details about layouts.
* The Using the controller context in a view recipe
* The Using decorators recipe