Symfony2框架实战教程——第二天：创建新页面

18条回复

昨天我们已经成功初始化一个Symfony项目，今天我们将在此基础上继续添砖加瓦。

在此之前，我们先明确一下我们要实现什么功能，在开始开展具体的代码工作前先明确自己的目标是一个节省时间的好习惯。我们的需求如下：

1. 用户可以使用新浪微博或者QQ帐号登录。
2. 用户可以投稿，需要填写的内容为“标题”+“正文”
3. 用户可以对某篇文章进行评论

在以上的基础上，我们继续具体化我们的需求：

1. 用户的内容都将使用markdown格式
2. 评论内容不可超过140个字
3. 新闻内容不可超过5000个字
4. 标题内容不可超过70个字

好，到这里我们的需求已经很明确了，虽然简单得似乎不可能会有人用的样子～

修改配置文件

此时项目里还有很多不需要的文件。比如昨天我们看到的Symfony自带的演示页面。这些文件都应该被清理。不过清理之前请等一会儿，演示页面里有一个“configure”功能我们还可以利用一下。点击了“configure”按钮以后我们可以对数据库、以及网站的密钥进行配置。其中第二步骤是设置网站的“密钥”，先不用管“密钥”的用途，只管点“下一步”就行。

注意：由于我们此时需要使用Mysql来作为我们的数据存储方案，所以这里我们需要安装Mysql，并且在第一步里将driver设置为`PDO_Mysql`。由于我们这里只介绍Symfony2，关于Mysql的安装，不太熟悉的同学可以自行查询，相关文档也是非常之多，这里就略过了。当然，如果对存储层非常熟悉的同学，你们也可以选择熟悉的驱动，Mysql并不是唯一方案。

当两个步骤都完成之后，Symfony会提示你，新的配置已经取代了旧的配置。此时就可以把此演示项目从我们

的项目代码里清理掉了。

刚才的步骤，其实只是修改了`app/config/parameter.yml`文件而已，直接修改也是可以的。此文件里包含了一些显而易见的配置，如果你英语还过关的话，应该一看就明白。

注意：如果安装的是*2.7*以及以上版本*Symfony*，只能直接修改*app/config/parameter.yml*文件

清理不需要的文件和代码

首先需要清理的是`app/AppKernel.php`文件。在这里首先说一个Symfony世界里的“术语”：

什么是**Bundle**

我认为Bundle并没有明确的定义，一般来说，如果某几个功能，相互之间有关联，并且相互影响，或者都属于某一类功能，这些功能就可以捆绑在一起成为一个Bundle。Bundle是一个目录，里面包含了此Bundle的相关代码以及配置文件。“Bundle”本来就有“一捆”的意思。比如Symfony自带演示项目Acme，就是一个Bundle的例子。在AcmeDemoBundle里的所有代码，都是为了演示用的，所以大家可以随意删除此Bundle而不用担心会影响现有的项目。但为了大家的入门速度以及不把大家搞晕，这里先不多说了。

现在打开`app/AppKernel.php`

```
1  // ...
2  class AppKernel extends Kernel
3  {
4      public function registerBundles()
5      {
6          $bundles = array(
7              new Symfony\Bundle\FrameworkBundle\FrameworkBundle(),
8              new Symfony\Bundle\SecurityBundle\SecurityBundle(),
9              new Symfony\Bundle\TwigBundle\TwigBundle(),
10             new Symfony\Bundle\MonologBundle\MonologBundle(),
11             new Symfony\Bundle\SwiftmailerBundle\SwiftmailerBundle
12             new Symfony\Bundle\AsseticBundle\AsseticBundle(),
13             new Doctrine\Bundle\DoctrineBundle\DoctrineBundle(),
14             new Sensio\Bundle\FrameworkExtraBundle\SensioFramework
15             new AppBundle\AppBundle(),
16         );
```

```
17
18          if (in_array($this->getEnvironment(), array('dev', 'test')
19              $bundles[] = new Symfony\Bundle\DebugBundle\DebugBundl
20              $bundles[] = new AcmeDemo\Bundle\AcmeDemoBundle();
21              $bundles[] = new Symfony\Bundle\WebProfilerBundle\WebP
22              $bundles[] = new Sensio\Bundle\DistributionBundle\Sens
23              $bundles[] = new Sensio\Bundle\GeneratorBundle\SensioG
24          }
25
26          return $bundles;
27      }
28      // ...
29  }
30
```

仔细观察`registerBundles`(注册Bundles)方法，可以看到此方法有许多名为`XxxBundle`的实例。所有的Bundle都需要一个类作为此Bundle的“代表”。只有Bundle的代表在此方法里被注册了，才可以使用此Bundle里的配置以及服务。大家可以看到此方法里已经注册了许多Bundle。假如我们不使用twig作为我们的模板引擎，我们是完全可以把`TwigBundle`那行直接删除掉的。支持Symfony框架的第三方库，都是以Bundle的方式存在。后面我们便会举例子如何注册第三方Bundle。

之前已经透露，演示项目的Bundle名叫“AcmeDemoBundle”，我们直接把`AcmeDemoBundle`那一行删掉。注意`AcmeDemoBundle`是在`if`里的，意思是如果当前环境是`dev`或者`test`，才注册AcmeDemoBundle。这个判断是什么意思？大家打开`web`目录，可以发现有`app.php`以及`app_dev.php`两个文件，大家先不要急着看这两个文件的内容，只用知道这两个文件都是入口文件就行了。所有的请求，都会首先执行它们中的其中一个，其实看文件名大家也应该明白app_dev.php是开发的时候用的入口文件。昨天使用`php app/console server:start localhost:8000`命令开启web server的时候，此命令已经自动将`app_dev.php`作为了入口文件。当我们的项目完成上线，就应该使用Nginx、Apache等Web Server把`app.php`作为入口文件。

开发环境不同会导致什么不同？再来看看`if`里的代码：不仅`AcmeDemoBundle`，在这个`if`里的所有Bundle都是为了开发而存在的，比如DebugBundle是为了在代码里使用`dump()`方法打印格式化好的变量信息，`WebProfilerBundle`则是为了收集开发所需要的调试数据，并且有好看的页面可以显示这些数据，是我们开发的好帮手。我们看到演示页面最下面的那条工具栏，就是`WebProfilerBundle`里包含的，后面我们将经常使用此工具栏做开发调试分析工作。

从`app/AppKernel.php`删除了`AcmeDemoBundle`那行代码之后，刷新页面，发现报错：

> *FileLoaderLoadException in FileLoader.php line 120: Bundle "AcmeDemoBundle" does not exist or it is not enabled. Maybe you forgot to add it in the registerBundles() method of your AppKernel.php file? in @AcmeDemoBundle/Resources/config/routing.yml (which is being imported from "/…/app/config/routing_dev.yml"). Make sure the "AcmeDemoBundle" bundle is correctly registered and loaded in the application kernel class.*

意思是说，你是否忘记在`AppKernel.php`文件里注册`AcmeDemoBundle`？因为在`app/config/routing_dev.yml`里还有AcmeDemoBundle的相关配置。

错误信息给我们提供了一个线索：`app/config/routing_dev.yml`文件里也有需要删除的代码。我们将此文件打开，将一下代码也删除掉（代码里也有注释说"以下代码应该被删除"）

```
1 # AcmeDemoBundle routes (to be removed)
2 _acme_demo:
3     resource: "@AcmeDemoBundle/Resources/config/routing.yml"
4
```

此文件是一个"YAML"格式文件，关于此文件格式这里也不多说，大家也不用有太多压力，因为YAML格式非常好懂，Symfony2的大多数配置文件，都是使用的YAML格式。Symfony2支持多种文件格式作为配置文件，常见的还有XML文件，以及直接使用PHP来写配置文件。对于配置文件的选择来说，XML可以使用DTD来做校验，所以一些风格严谨的第三方Bundle内部的配置文件是用的XML，但XML有写起来繁琐的问题，而编写方便正是YAML的优点。在这里我们遵守Symfony installer帮我们做的选择：使用YAML。需要注意：YAML和XML不是一个Bundle只能用一种，他们是可以混用的，而且从性能速度来说，没有丝毫差别，因为最终都会被编译成PHP文件。除了以上说的几种格式外，还有一种叫做"注解"的格式，后面再介绍。

`routing***.yml`定义了访问我们的项目的访问路径。Symfony2框架使用的"前端控制器"模式来处理HTTP访问的需求，即每一种形式的路径，都将映射到某个类的某个方法来处理。比如说当用户访问`/hello/{name}`路径的时候，我们可以将此路径映射到AppBundle里的`HelloController`（一般称为"控制器类"）的`helloAction($name)`（一般称为控制器方法）方法来处理。`routing***.yml`文件就是用来描述这种映射关系的。并且`routing***.yml`文件可以有包含关系，我们可以看到`routing_dev.yml`包含了`routing.yml`文件。另外，我们从文件名也可以看出，`routing_dev.yml`是开发环境才加载的配置。

从`routing_dev.yml`里删除掉`acme`相关的路由以后，再访问首页。我们可以发现错误信息变了：

No route found for “GET /”

“route”即“路由映射规则”，出现此错误说明已经没有关于“GET /”路径相关的配置了。这说明至少从配置上，我们已经把AcmeDemoBundle删除掉了。

接下来我们再删除掉src/Acme目录，清理工作便告一段落，说是一个段落的原因是还有一个比较关键的配置文件我们没有提到，但不影响接下来的开发。以后会说。

添加页面

每次访问首页都报错，不是个好兆头，怎么着我们得有一个首页，无论有多简单。

按我们之前所说，要想首页能够被访问，我们得在routing***.yml里添加有关首页的配置。

由于首页是无论开发环境还是生产环境都需要访问的，我们就把配置写在routing.yml里好了，注意之前说过routing_dev.yml已经包含了routing.yml了。

我们打开app/config/routing.yml文件，可以看到已经有如下内容：

```
1 app:
2     resource: @AppBundle/Controller/
3     type:     annotation
4
```

这里的resource指向的不是一个文件，而是一个路径：@AppBundle我想不用多说，一定是指AppBundle所在的目录，也就是src/AppBundle。是的，resource可以不用指定某个具体的文件，目录也行，意思是：此目录内的所有配置文件都包含进来，甚至包括子目录的配置。

然后type的值为annotation，这就是之前说的“注解”格式。

我们打开@AppBundle/Controller/里随便什么文件，看看注解格式到底是个啥。

@AppBundle/Controller/目录里目前只包含一个DefaultController.php，打开此PHP文件：

```
// ...
class DefaultController extends Controller
{
    /**
     * @Route("/app/example", name="homepage")
     */
    public function indexAction()
    {
        return $this->render('default/index.html.twig');
    }
}

```

仔细观察，像是配置的代码，很明显是@Route("/app/example", name="homepage")那一行。原来“注解”格式，即是写在代码里的注释。使用此格式的配置有一些其他格式不可比拟的好处，比如方便重构（举个栗子：控制器名字改了，改1个文件VS改2个文件）。

既然Symfony installer帮我们生成了此文件，我们也继续沿用。我们要做的只是把路径改成首页。然后路由名改短一点，路由名是个什么，后面会说到：

```
/**
 * @Route("/", name="home")
 */
public function indexAction()

```

这个时候我们再访问首页，会出现写着“Homepage.”的页面。

这个“Homepage.”又是从哪儿来的？看代码里面给我们提供的线索：indexAction里返回了一个$this->render('default/index.html.twig')，如果英语过关，很容易看出来此方法做的事情，就是“渲染”了一个模板文件default/index.html.twig。

模板文件默认目录位于app/Resources/views/，所产我们要找的文件位于app/Resources/views/default/index.html.twig，打开此文件可以看到里面有一行Homepage.，以及一些其他的代码。现在我们不用管其他的代码。

为了方便开发我们的“新闻”板块。我们先给首页加一个链接/news。

我们可以直接将Homepage.替换成新闻。但是这样做有一个不好的地方：如果想改路径，比如改成/新闻，那么每个连接到“新闻”的href="/news"都需要改一遍。Symfony2以及Laravel或者ROR等现代框架，几乎都支持“双向路由”，即在路径和对应的控制器方法之间，再加一个路由名。此路由名可对应一个路径，又对应一个控制器方法。前端模板里不再直接写路径，而是写路由名：

```
1 <a href="{{ path('news_index') }}">新闻</a>
2
```

以后想修改路径，都只用改一下配置文件即可。

修改好以上代码，再访问首页，又会出现以下错误：

> *An exception has been thrown during the rendering of a template (“Unable to generate a URL for the named route “news_index” as such route does not exist.”) in default/index.html.twig at line 4.*

意思是，目前还没有一个叫news的路由规则。OK我们现在来创建它：

我们把新闻相关的路径都放在AppBundle\Controller\NewsController类里：

```
1  <?php
2
3  // src/AppBundle/Controller/NewsController.php
4  namespace AppBundle\Controller;
5
6  use Sensio\Bundle\FrameworkExtraBundle\Configuration\Route;
7  use Symfony\Bundle\FrameworkBundle\Controller\Controller;
8
9  /**
10  * @Route("/news")
11  */
12 class NewsController extends Controller
13 {
14     /**
```

```
15       * @Route("/", name="news_index")
16       */
17      public function indexAction()
18      {
19          return $this->render('news/index.html.twig');
20      }
21  }
22
```

并且为其创建模板文件app/Resources/views/news/index.html.twig，目前就是空文件就行。

此时再刷新首页，错误消失。并且“新闻“已经可以点击。

创建文件时需要大家注意的是，一定要确认代码的文件的编码格式是UTF-8，否则会出现不可预料的字符乱码问题。