

如果你看到了这里,那末我承认,你是一个英雄!你可能没想到,你仍然处于前三章的入门阶段!!但是你的努力很快就会得到很好的回报。前三部分太深入的框架结构。因为前三章让Symfony从众多的框架中脱因而出,接下来,让我们深入探索Symfony的架构。

了解目录结构

一个Symfony的应用的目录结构是相当灵活的,但推荐的结构如下:

app/	应用程序的配置,模板和翻译
src/	该项目的php代码
vendor/	第三方的依赖
web/	该网站的根目录

The web/ Directory

该网站的根目录是存放如图像,样式表和JavaScript文件中的所有公共和静态文件的。这也是每个前端控制器的生活,比如这里显示的生产控制:

```

1.  // web/app.php
2.  require_once __DIR__.'../../app/bootstrap.php.cache';
3.  require_once __DIR__.'../../app/AppKernel.php';
4.
5.  use Symfony\Component\HttpFoundation\Request;
6.
7.  $kernel = new AppKernel('prod', false);
8.  $kernel->loadClassCache();
9.  $request = Request::createFromGlobals();
10. $response = $kernel->handle($request);
11. $response->send();

```

控制器首先引导程序使用内核类(AppKernel 在这种情况下)。然后,它会创建使用PHP的全局变量 Request 对象,并将其传递给内核。最后一步是发送由内核向用户返回的响应内容。

The app/ Directory

如上所述的 AppKernel 类是应用程序的配置的主要入口,因此,它被存储在该 app/ 目录。

这个类必须实现两个方法:

registerBundles()	必须以数组方式返回一组运行应用程序所需要的一组数据包(这将在下节讲到)
registerContainerConfiguration()	载入应用配置

自动加载通过Composer自动处理,这意味着你可以使用任何PHP类而无须做其他事情!所有依赖存储 vendor/ 目录下,但是这仅仅是一个约定。你可以将他们存储在任何你想要的地方,存储在全球范围内的服务器上或本地的项目中都行。

了解捆绑系统

本节介绍Symfony最强大的一个特征,捆绑系统。

包(bundle)有点儿像其他软件插件。那么,为什么它被称为一个包,而不是一个插件?这是因为从核心架构特性来编写你的应用程序的代码时,在Symfony中一切都是捆绑的。

所有你写你的应用程序中的代码被组织成捆(bundles)。用Symfony的来表述就是,一个包(bundle)是一套结构化的文件(PHP文件,样式表,JavaScript的,图像,...),用来实现一个功能(博客,论坛,...),并且易于与其他开发人员共享。

在Symfony的世界里 **Bundles** 是一等公民。这使您能够灵活的使用第三方的 **Bundles** 和分发分发你自己 **Bundles** 。它可以很容易地在你的应用程序中挑选功能，并按照你想要的方式优化他们。和在一天结束时，你的应用程序代码也同样和核心框架本身一样重要。

Symfony的已经包括**AppBundle**，您可以直接使用Symfony来开发应用程序。如果您需要将应用程序划分为可重用的组件，你可以创建自己的包。

注册一个包 (Registering a Bundle)

一个应用程序由众多的包组成,这些包通过 **AppKernel** 类的方法 **registerBundles()** 来定义。每个包是包含描述一个单一的捆绑类的目录

```
1. // app/AppKernel.php
2. public function registerBundles()
3. {
4.     $bundles = array(
5.         new Symfony\Bundle\FrameworkBundle\FrameworkBundle(),
6.         new Symfony\Bundle\SecurityBundle\SecurityBundle(),
7.         new Symfony\Bundle\TwigBundle\TwigBundle(),
8.         new Symfony\Bundle\MonologBundle\MonologBundle(),
9.         new Symfony\Bundle\SwiftmailerBundle\SwiftmailerBundle(),
10.        new Symfony\Bundle\DoctrineBundle\DoctrineBundle(),
11.        new Symfony\Bundle\AsseticBundle\AsseticBundle(),
12.        new Sensio\Bundle\FrameworkExtraBundle\SensioFrameworkExtraBundle(),
13.        new AppBundle\AppBundle();
14.    );
15.
16.    if (in_array($this->getEnvironment(), array('dev', 'test'))) {
17.        $bundles[] = new Symfony\Bundle\WebProfilerBundle\WebProfilerBundle();
18.        $bundles[] = new Sensio\Bundle\DistributionBundle\SensioDistributionBundle();
19.        $bundles[] = new Sensio\Bundle\GeneratorBundle\SensioGeneratorBundle();
20.    }
21.
22.    return $bundles;
23. }
```

除了一个已经谈到的 **appbundle**，注意内核还支持其他的包,这些包也是Symfony的组成部分，如 **FrameworkBundle**，**DoctrineBundle**，**SwiftmailerBundle** 和 **AsseticBundle**。

配置包(Configuring a Bundle)

每个包可以通过写在YAML，XML或PHP配置文件来进行定制。看一看默认的Symfony配置示例：

```

1.  # app/config/config.yml
2.  imports:
3.      - { resource: parameters.yml }
4.      - { resource: security.yml }
5.      - { resource: services.yml }
6.
7.  framework:
8.      #esi:                ~
9.      #translator:        { fallbacks: ["%locale%"] }
10.     secret:              "%secret%"
11.     router:
12.         resource: "%kernel.root_dir%/config/routing.yml"
13.         strict_requirements: "%kernel.debug%"
14.     form:                 true
15.     csrf_protection: true
16.     validation:          { enable_annotations: true }
17.     templating:          { engines: ['twig'] }
18.     default_locale:      "%locale%"
19.     trusted_proxies: ~
20.     session:             ~
21.
22. # Twig Configuration
23. twig:
24.     debug:                "%kernel.debug%"
25.     strict_variables:     "%kernel.debug%"
26.
27. # Swift Mailer Configuration
28. swiftmailer:
29.     transport: "%mailer_transport%"
30.     host:      "%mailer_host%"
31.     username:  "%mailer_user%"
32.     password:  "%mailer_password%"
33.     spool:     { type: memory }
34.
35. # ...

```

扩展一个包(Extending a Bundle)

除了作为一个很好的方式来组织和配置您的代码，一个包可以扩展另一个包。捆绑继承允许你以自定义的控制器，模板，或任何文件覆盖现有的包，。

逻辑文件名

当你想从一个包引用文件，使用这个符号：`@BUNDLE_NAME/path/to/file`；Symfony的将解析 `@BUNDLE_NAME` 到真实路径包。例如，该逻辑路径 `@AppBundle/Controller/DefaultController.php` 将被转换为 `src/AppBundle/Controller/DefaultController.php`，因为symfony知道 `appbundle` 的位置。

逻辑控制器名称

对于控制器，需要引用 `BUNDLE_NAME:CONTROLLER_NAME:ACTION_NAME` 格式后动作。例如，`AppBundle:Default:index` 映射到来自 `AppBundle\Controller\DefaultController` 类的方
法 `indexAction`

我的主页:<http://aifei8.net>:



爱妃科技

主页 服务 案例 联系我们 我的博客

服务项目.

专注于移动互联网以及嵌入式设备的研发



android应用定制开发

使用google官方的java语言开发框架，效率高，占用资源少，运行速度快。也可采用最近风靡一时的node技术。



IOS应用定制开发

使用node和html5,开发速度快，运行速度快，低耦合，低成本，几乎接近C语言的运行速度



PC桌面应用定制开发

支持windows、linux以及apple的Mac Book三大主流操作系统，良好的跨平台开发技术，给你不一样的体验



美工设计

采用html5超文本标记语言和css样式表控制页面内容，ECMAScript与用户进行交互，高效的ajax技术实现实时更新站点内容。



网站建设

采用开源免费的linux、apache、mysql、php的lamp架构。占用资源少，运行速度快



SEO优化

搜索引擎爆炸式的成长是Internet的代表，用户每次对搜索引擎的访问都潜在的产生了对特定厂商的业务。

分享

我的博客:<http://blog.aifei8.net>

github主页:<http://github.com/Pengfei-Gao>

[Pull requests](#) [Issues](#) [Gist](#)

Love Princess Studio

Pengfei-Gao

Love Princess Studio
吉林省延吉市延边大学
net.aifei8@gmail.com
http://aifei8.net
Joined on 30 Dec 2015

0

Followers

4

Starred

0

Following

[Contributions](#)[Repositories](#)[Public activity](#)[Edit profile](#)

Popular repositories

YBUInformation 一个延边大学urp教务系统学生信息查询的示例	1 ★
develop-reference-data 一些常用的开发文档	1 ★
ionic Advanced HTML5 mobile development framework and SDK. Build incredible mobile apps with web technologies y...	0 ★
source-Insight-3-for-centos7 centos7 下集成的source Insight 3	0 ★
Symfony-doc-translate Symfony的中文文档	0 ★

Contributions

