之前我们看Bundle文档添加了在src目录下添加了Controller目录，如果你的Bundle对外提供路由，那么可以编写自己的Controller类，和Teebblog项目里的Controller类一样，需要继承自`AbstractController`

```
class TestController extends AbstractController
{
    public function test():Response
    {
        return new Response('Hello Bundle');
    }
}
```

`TestController`是一个服务类，我们需要添加xml配置文件，将Controller添加到容器里，`test()`方法是一个路由对应的controller方法，同样的我们要写配置文件把路由也添加一下。

```
<?xml version="1.0" encoding="utf-8" ?>
<container xmlns="http://symfony.com/schema/dic/services"
           xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
           xsi:schemaLocation="http://symfony.com/schema/dic/services
http://symfony.com/schema/dic/services/services-1.0.xsd">
    <services>
        <service id="teebb.upload.controller.test_controller"
class="Teebb\UploadBundle\Controller\TestController" public="true">
            <!-- 这里可以查看其他的Controller添加tag -->
            <tag name="controller.service_arguments"/>
            <tag name="container.service_subscriber"/>
        </service>
    </services>
</container>
```

最后让`Extension`类加载xml文件。查看容器里是否有`TestController`。

编写路由的配置文件：

```
<?xml version="1.0" encoding="UTF-8"?>
<routes xmlns="http://symfony.com/schema/routing"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
        xsi:schemaLocation="http://symfony.com/schema/routing
http://symfony.com/schema/routing/routing-1.0.xsd">

    <route id="teebb_upload_test" path="/test">
        <default
key="_controller">Teebb\UploadBundle\Controller\TestController::test</defa
ult>
        <default key="route">teebb_upload_test</default>
    </route>

</routes>
```

然后回到Teebblog项目，在config/routes目录下把bundle的路由配置添加进来

```
#config/routes/teebb_upload.yaml

_teebb_upload:
    resource: "@TeebbUploadBundle/Resources/config/routing/route.xml"
    prefix: /teebb
```

查看项目的路由： symfony console debug:route teebb，现在就可以访问了，如果想为路由统一添加前缀可以在teebb_upload.yaml添加prefix: /teebb，这样route.xml文件里的路由就会统一加上/teebb前缀了。

如果你的Bundle里有一些js、css、图片资源文件，你可以添加到src/Resources/public目录里，这个目录里的所有文件，都可以使用symfony console assets:install 命名行安装到项目的public目录下。

```
symfony console assets:install public --symlink
```

这样就可以Bundle的模板文件里直接使用了资源文件了。

现在项目的config目录下，我们有两个关于我们自己的Bundle的配置文件，我们可以把Bundle的默认配置文件做成recipe，提交到Symfony的recipe贡献仓库。

```
symfony官方组件、bundle的仓库
https://github.com/symfony/recipes

第三方组件、Bundle的recipe仓库
https://github.com/symfony/recipes-contrib
```

在Github上fork仓库，git clone下来后，把我们自己的包添加到创建再提交，最后提交PR.

我们来写自己的recipe配置：

创建teebbstudios/upload-bundle目录： 这个目录名称和Bundle发布的名称是一致的，然后目录下是主版本号，我们使用1.0版本。

在1.0目录下有一个manifest.json文件，这个文件告诉flex组件，在composer安装完包怎么添加recipe配置 创建这个文件：

```
// manifest.json

{
    "bundles": {
```

```
        "Teebb\\UploadBundle\\TeebbUploadBundle": ["all"]   //Bundle可以运行
的环境，可以设置为prod dev test 这要看你的需求，这里我们设置为all
    },
    //这项配置就是配置文件从仓库直接下载后复制到哪里，
    "copy-from-recipe": {
        "config/": "%CONFIG_DIR%/" // 这里的意思是把当前recipe下的所有config目录
复制到项目的`config`目录，
    }
}
```

然后在`teebbstudios/upload-bundle/1.0`创建config目录，把Teebblog项目里的配置文件移动过来到对应的目录里。 路由配置是必需有的，bundle配置文件是可以省略的，因为我们的bundle，现在不需要默认的配置，所有的配置是根据用户需要进行配置的，可以删除。

> 注意：
> 1.要把配置文件里的所有注释都删除掉
> 2.配置文件中应该只有自己的Bundle设置，不要有其他Bundle的配置。

最后提交PR，等待审核就可以了。

再把Bundle在`https://packagist.org/`进行发布，发布的版本号要和recipe的主版本号对应。

现在如果有用户在Symfony项目中安装了我们，就会自动安装对应的Recipe。

我们的Bundle都是在Teebblog项目中进行开发的，也没有写测试代码。现在我们要在Bundle项目里进行独立的测试。