手把手教你搭建SpringMVC——最小化配置 - xingoo - 博客园

为什么需要Spring MVC

最开始接触网页的时候，是纯的html/css页面，那个时候还是用Dreamweaver来绘制页面。

随着网站开发的深入，开始学习servlet开发，记得最痛苦的就是servlet返回网页的内容是字符串拼接的html页面，整不好就无法显示....

再到后来开学学习SSH，庞大的架构眼花缭乱。Struts繁杂的标签、hibernate搞不清楚的数据表，Spring不知道哪里搞错的bean。

最后随着发展，前端开始占有一席之地，nodejs风生水起，很多业务逻辑开始前置。再也看不到当初的bo、dao了，取而代之的是各种框架的mvvm，后台减轻压力只负责一些必要的逻辑。

到现在，好像web开发又发展到了一个阶段——前端由于Nodejs的作用，可以支撑一部分业务逻辑，通过转发代理，统一发往后台。后台通过url实现mvc，对性持久化、更深入的逻辑操作等等。Spring MVC在这里就起了很关键的作用....它通过Url拦截请求，自定义业务逻辑，可以返回自定义的view或者模型数据。

当然，上面的鬼扯都是片面的，不代表行业的发展，只是博主管中窥豹而已。

下面步入正题，说说Spring MVC的最小化配置，给入门的朋友引个路。

Spring MVC的最小化配置

需要的jar包

- Spring framework spring-context
- Spring framework spring-mvc

具体可以参考maven中的引用：

```
<dependency><groupId>org.springframeworkgroupId><artifactId>spring-webmvcartifactId>
<version>4.2.4.RELEASEversion>dependency><dependency><groupId>org.springframeworkgroupId>
<artifactId>spring-contextartifactId><version>4.2.4.RELEASEversion>dependency>
```

web.xml配置

```
<web-
appversion="3.1"xmlns="http://xmlns.jcp.org/xml/ns/javaee"xmlns:xsi="http://www.w3.org/2001/XMLSchema-
instance"xsi:schemaLocation="http://xmlns.jcp.org/xml/ns/javaee
http://xmlns.jcp.org/xml/ns/javaee/web-app_3_1.xsd"><context-param><param-
name>contextConfigLocationparam-name><param-value>/WEB-INF/applicationContext.xmlparam-value>context-
param><listener><listener-class>            org.springframework.web.context.ContextLoaderListener
listener-class>listener><servlet><servlet-name>SpringMVCservlet-name><servlet-
class>org.springframework.web.servlet.DispatcherServletservlet-class><init-param><param-
name>contextConfigLocationparam-name><param-value>/WEB-INF/SpringMVC-servlet.xmlparam-value>init-param>
<load-on-startup>1load-on-startup>servlet><servlet-mapping><servlet-name>SpringMVCservlet-name><url-
pattern>/url-pattern>servlet-mapping>web-app>
```

其中，必要的配置就是指定servlet和listener.

- ContextLoaderListener指定了IOC容器初始化的方法
- DispatcherServlet则定义了mvc的相关内容，并配置拦截的url，如上面所示，所有/开头的请求，都会通过SpringMVC这个servlet进行处理。

他们都需要一个xml文件，默认位置上面已经说过了。

applicationContext.xml

空的，反正咱也没用什么bean。

```
<beansxmlns="http://www.springframework.org/schema/beans"xmlns:xsi="http://www.w3.org/2001/XMLSchema-
instance"xmlns:aop="http://www.springframework.org/schema/aop"xmlns:context="http://www.springframework.o
rg/schema/context"xmlns:tx="http://www.springframework.org/schema/tx"xsi:schemaLocation="http://www.sprin
gframework.org/schema/beans                    http://www.springframework.org/schema/beans/spring-
beans-4.0.xsd                    http://www.springframework.org/schema/tx
```

```
http://www.springframework.org/schema/tx/spring-tx-4.0.xsd
http://www.springframework.org/schema/aop
http://www.springframework.org/schema/aop/spring-aop-4.0.xsd
http://www.springframework.org/schema/context
http://www.springframework.org/schema/context/spring-context-4.0.xsd">beans>
```

SpringMVC-servlet.xml

里面放一个扫描controller的配置即可。

```
<beansxmlns="http://www.springframework.org/schema/beans"xmlns:xsi="http://www.w3.org/2001/XMLSchema-
instance"xmlns:mvc="http://www.springframework.org/schema/mvc"xmlns:context="http://www.springframework.o
rg/schema/context"xmlns:aop="http://www.springframework.org/schema/aop"xmlns:tx="http://www.springframewo
rk.org/schema/tx"xsi:schemaLocation="http://www.springframework.org/schema/beans
http://www.springframework.org/schema/beans/spring-beans-4.0.xsd
http://www.springframework.org/schema/mvc
http://www.springframework.org/schema/mvc/spring-mvc-4.0.xsd
http://www.springframework.org/schema/context
http://www.springframework.org/schema/context/spring-context-4.0.xsd
http://www.springframework.org/schema/aop
http://www.springframework.org/schema/aop/spring-aop-4.0.xsd
http://www.springframework.org/schema/tx
http://www.springframework.org/schema/tx/spring-tx-4.0.xsd"><context:component-scanbase-package="hello"
/>beans>
```

controller文件

```
packagehello;importorg.springframework.stereotype.Controller;importorg.springframework.web.bind.annotatio
n.RequestMapping;importorg.springframework.web.bind.annotation.ResponseBody;@Controllerpublic class
HelloController {        @RequestMapping("/hello")    public @ResponseBody String test() {        return
"hello, world! Thiscomfromspring!";    }}
```

总结一下：

1 两个maven依赖，spring-context；spring-mvc。maven就会自动下载所有关联的jar包，包括

- spring-webmvc
- spring-beans
- spring-core
- spring-expression
- spring-web
- spring-context
- spring-aop
- aopalliance
- commons-logging

2 一个web.xml文件，配置了listener和servlet

3 两个spring相关的文件，applicationContext.xml和servletName-servlet.xml

4 一个controller文件，配置了拦截的url处理代码

有了这些准备工作，运行后输入

http://localhost:8080/SpringTest/hello

就能得到

```
hello, world! This com from spring!
```

这样的信息，恭喜你的SpringMVC搭起来了！

最后附送源码的度盘连接