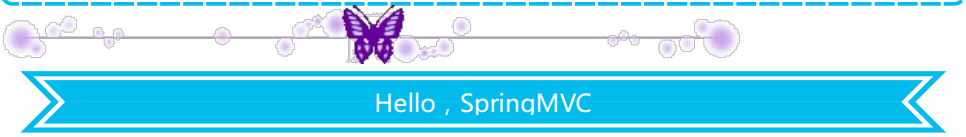
狂神说SpringMVC02：第一个MVC程序

秦疆 狂神说 2020-03-31



在上一节中，我们讲解了 什么是SpringMVC以及它的执行原理！
狂神说SpringMVC01：什么是SpringMVC
现在我们来看看如何快速使用SpringMVC编写我们的程序吧！

配置版

1、新建一个Moudle ， springmvc-02-hello ， 添加web的支持！

2、确定导入了SpringMVC 的依赖！

3、配置web.xml ， 注册DispatcherServlet

```
<?xml version="1.0" encoding="UTF-8"?><web-app 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_4_0.xsd"        version="4.0">   <!--1.注册DispatcherServlet-->   <servlet>       <servlet-name>springmvc</servlet-name>       <servlet-class>org.springframework.web.servlet.DispatcherServlet</servlet-class>       <!--关联一个springmvc的配置文件:【servlet-name】-servlet.xml-->       <init-param>           <param-name>contextConfigLocation</param-name>           <param-value>classpath:springmvc-servlet.xml</param-value>       </init-param>       <!--启动级别-1-->       <load-on-startup>1</load-on-startup>   </servlet>   <!--/ 匹配所有的请求；（不包括.jsp）-->   <!--/* 匹配所有的请求；（包括.jsp）--><servlet-mapping>       <servlet-name>springmvc</servlet-name>       <url-pattern>/</url-pattern>   </servlet-mapping></web-app>
```

4、编写SpringMVC 的 配置文件！名称：springmvc-servlet.xml ：[servletname]-servlet.xml

说明，这里的名称要求是按照官方来的

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

5、添加 处理映射器

```
<bean class="org.springframework.web.servlet.handler.BeanNameUrlHandlerMapping"/>
```

6、添加 处理器适配器

```
<bean class="org.springframework.web.servlet.mvc.SimpleControllerHandlerAdapter"/>
```

7、添加 视图解析器

```
<!--视图解析器:DispatcherServlet给他的ModelAndView--><bean class="org.springfr
amework.web.servlet.view.InternalResourceViewResolver" id="InternalResource
ViewResolver">   <!--前缀-->   <property name="prefix" value="/WEB-INF/jsp/"
/>   <!--后缀-->   <property name="suffix" value=".jsp"/></bean>
```

8、编写我们要操作业务Controller ，要么实现Controller接口，要么增加注解；需要返回一个ModelAndView，装数据，封视图；

```
package com.kuang.controller;
import org.springframework.web.servlet.ModelAndView;import org.springframew
ork.web.servlet.mvc.Controller;
import javax.servlet.http.HttpServletRequest;import javax.servlet.http.Http
ServletResponse;
//注意：这里我们先导入Controller接口public class HelloController implements Cont
roller {
   public ModelAndView handleRequest(HttpServletRequest request, HttpServle
tResponse response) throws Exception {        //ModelAndView 模型和视图
```

```
ModelAndView mv = new ModelAndView();
        //封装对象，放在ModelAndView中。Model        mv.addObject("msg","HelloSpr
ingMVC!");        //封装要跳转的视图，放在ModelAndView中        mv.setViewName("he
llo"); //: /WEB-INF/jsp/hello.jsp        return mv;   }    }
```

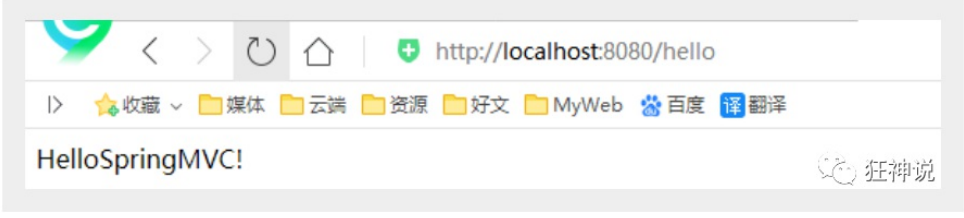
9、将自己的类交给SpringIOC容器，注册bean

```
<!--Handler--><bean id="/hello" class="com.kuang.controller.HelloController'
/>
```

10、写要跳转的jsp页面，显示ModelandView存放的数据，以及我们的正常页面；

```
<%@ page contentType="text/html;charset=UTF-8" language="java" %><html><heac
>   <title>Kuangshen</title></head><body>${msg}</body></html>
```

11、配置Tomcat 启动测试！



可能遇到的问题：访问出现404，排查步骤：

1. 查看控制台输出，看一下是不是缺少了什么jar包。
2. 如果jar包存在，显示无法输出，就在IDEA的项目发布中，添加lib依赖！
3. 重启Tomcat 即可解决！

小结：看这个估计大部分同学都能理解其中的原理了，但是我们实际开发才不会这么写，不然就疯了，还学这个玩意干嘛！我们来看个注解版实现，这才是SpringMVC的精髓，到底有多么简单，看这个图就知道了。

注解版

1、新建一个Moudle，springmvc-03-hello-annotation 。添加web支持！

2、由于Maven可能存在资源过滤的问题，我们将配置完善

```
<build>   <resources>       <resource>           <directory>src/main/java</
directory>           <includes>               <include>**/*.properties</inc
lude>               <include>**/*.xml</include>           </includes>
     <filtering>false</filtering>       </resource>       <resource>
    <directory>src/main/resources</directory>           <includes>
      <include>**/*.properties</include>               <include>**/*.xml</i
nclude>           </includes>           <filtering>false</filtering>
</resource>   </resources></build>
```

3、在pom.xml文件引入相关的依赖：主要有Spring框架核心库、Spring MVC、servlet , JSTL等。我们在父依赖中已经引入了！

4、配置web.xml

注意点：

```
<?xml version="1.0" encoding="UTF-8"?><web-app 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_4_0.xsd"        version="4.0">
   <!--1.注册servlet-->   <servlet>       <servlet-name>SpringMVC</servlet-n
```

```
ame>        <servlet-class>org.springframework.web.servlet.DispatcherServlet
</servlet-class>        <!--通过初始化参数指定SpringMVC配置文件的位置，进行关联-->
     <init-param>           <param-name>contextConfigLocation</param-name>
          <param-value>classpath:springmvc-servlet.xml</param-value>
</init-param>       <!-- 启动顺序，数字越小，启动越早 -->       <load-on-startup>
</load-on-startup>   </servlet>
   <!--所有请求都会被springmvc拦截 -->   <servlet-mapping>       <servlet-name>
SpringMVC</servlet-name>       <url-pattern>/</url-pattern>   </servlet-map
ping>
</web-app>
```

/ 和 /* 的区别：< url-pattern > / </ url-pattern > 不会匹配到.jsp， 只针对我们编写的请求；即：.jsp 不会进入spring的 DispatcherServlet类 。< url-pattern > /* </ url-pattern > 会匹配 *.jsp，会出现返回 jsp视图 时再次进入spring的DispatcherServlet 类，导致找不到对应的controller所以报404错。

- 注意web.xml版本问题，要最新版！
- 注册DispatcherServlet
- 关联SpringMVC的配置文件
- 启动级别为1
- 映射路径为 / 【不要用/*，会404】

5、添加Spring MVC配置文件

在resource目录下添加springmvc-servlet.xml配置文件，配置的形式与Spring容器配置基本类似，为了支持基于注解的IOC，设置了自动扫描包的功能，具体配置信息如下：

```
<?xml version="1.0" encoding="UTF-8"?><beans xmlns="ht
tp://www.springframework.org/schema/beans"      xmlns:
xsi="http://www.w3.org/2001/XMLSchema-instance"      x
mlns:context="http://www.springframework.org/schema/co
ntext"      xmlns:mvc="http://www.springframework.org/
schema/mvc"      xsi:schemaLocation="http://www.spring
framework.org/schema/beans       http://www.springfram
ework.org/schema/beans/spring-beans.xsd       http://w
ww.springframework.org/schema/context       https://ww
w.springframework.org/schema/context/spring-context.xsd
       http://www.springframework.org/schema/mvc
 https://www.springframework.org/schema/mvc/spring-mvc
.xsd">
   <!-- 自动扫描包，让指定包下的注解生效,由IOC容器统一管理 -->
  <context:component-scan base-package="com.kuang.cont
roller"/>   <!-- 让Spring MVC不处理静态资源 -->   <mvc:de
fault-servlet-handler />   <!--    支持mvc注解驱动
在spring中一般采用@RequestMapping注解来完成映射关系        要
想使@RequestMapping注解生效        必须向上下文中注册DefaultA
nnotationHandlerMapping        和一个AnnotationMethodHan
dlerAdapter实例        这两个实例分别在类级别和方法级别处理。
    而annotation-driven配置帮助我们自动完成上述两个实例的注入。
    -->   <mvc:annotation-driven />
   <!-- 视图解析器 -->   <bean class="org.springframewor
k.web.servlet.view.InternalResourceViewResolver"
   id="internalResourceViewResolver">       <!-- 前缀 -
->       <property name="prefix" value="/WEB-INF/jsp/"
```

```
 />       <!-- 后缀 -->       <property name="suffix" v
alue=".jsp" />   </bean>
</beans>
```

在视图解析器中我们把所有的视图都存放在/WEB-INF/目录下，这样可以保证视图安全，因为这个目录下的文件，客户端不能直接访问。

- 让IOC的注解生效
- 静态资源过滤 ：HTML . JS . CSS . 图片， 视频 .....
- MVC的注解驱动
- 配置视图解析器

6、创建Controller

编写一个Java控制类：com.kuang.controller.HelloController , 注意编码规范

-
-
-
-
-
-
-
-
-
-
-
-
-
-
-
-
-
-
-
-

```
package com.kuang.controller;
import org.springframework.stereotype.Controller;import
 org.springframework.ui.Model;import org.springframewo
rk.web.bind.annotation.RequestMapping;
@Controller@RequestMapping("/HelloController")public c
lass HelloController {
   //真实访问地址 : 项目名/HelloController/hello   @Reques
tMapping("/hello")   public String sayHello(Model mode
l){       //向模型中添加属性msg与值，可以在JSP页面中取出并渲染
      model.addAttribute("msg","hello,SpringMVC");
   //web-inf/jsp/hello.jsp       return "hello";  }}
```

- @Controller是为了让Spring IOC容器初始化时自动扫描到；
- @RequestMapping是为了映射请求路径，这里因为类与方法上都有映射所以访问时应该是/HelloController/hello；
- 方法中声明Model类型的参数是为了把Action中的数据带到视图中；
- 方法返回的结果是视图的名称hello，加上配置文件中的前后缀变成WEB-INF/jsp/**hello**.jsp。

7、创建视图层

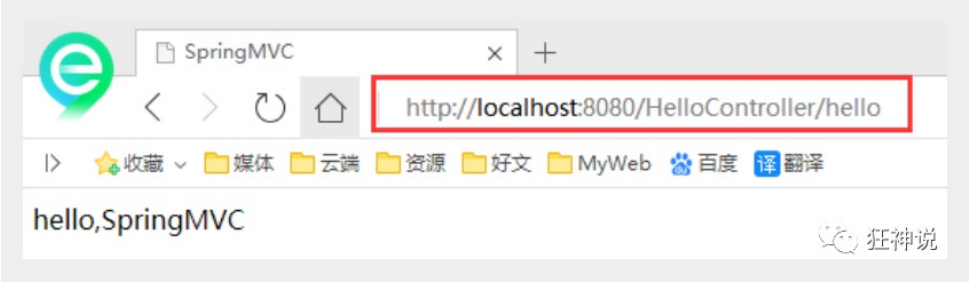
在WEB-INF/jsp目录中创建hello.jsp， 视图可以直接取出并展示从Controller带回的信息；

可以通过EL表示取出Model中存放的值，或者对象；

```
<%@ page contentType="text/html;charset=UTF-8" language
="java" %><html><head>   <title>SpringMVC</title></head
><body>${msg}</body></html>
```

8、配置Tomcat运行

配置Tomcat ， 开启服务器 ， 访问 对应的请求路径！



OK，运行成功！

小结

实现步骤其实非常的简单：

1. 新建一个web项目

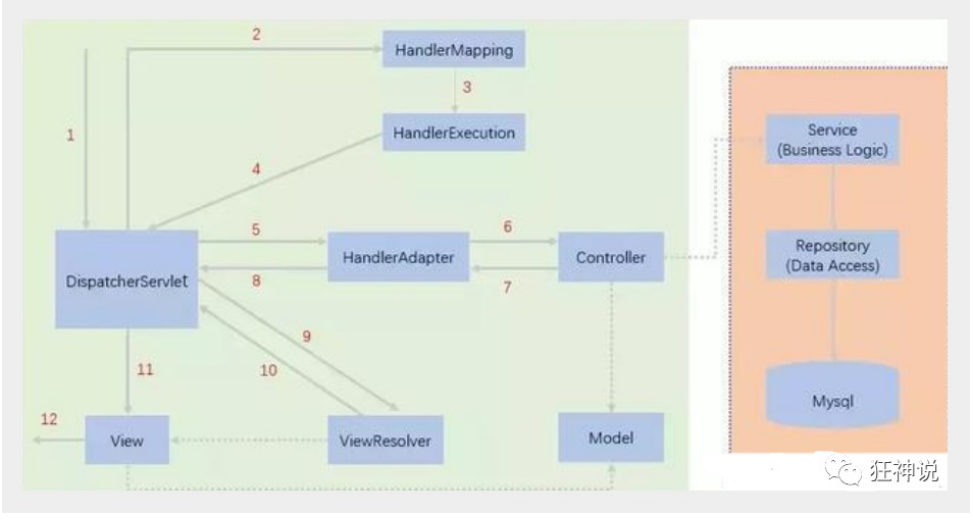
2. 导入相关jar包
3. 编写web.xml , 注册DispatcherServlet
4. 编写springmvc配置文件
5. 接下来就是去创建对应的控制类 , controller
6. 最后完善前端视图和controller之间的对应
7. 测试运行调试.

使用springMVC必须配置的三大件：

处理器映射器、处理器适配器、视图解析器

通常，我们只需要**手动配置视图解析器**，而**处理器映射器**和**处理器适配器**只需要开启**注解驱动**即可，而省去了大段的xml配置

再来回顾下原理吧~



下面我们准备研究下Controller及RestFul风格！

end

视频同步更新，这次一定！

“赠人玫瑰，手有余香”

狂神说 的赞赏码

狂神说