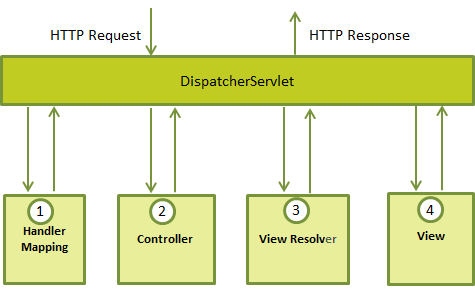
# Springmvc

## 简介



以下是对应于到DispatcherServlet的传入HTTP请求的事件顺序：

在接收到HTTP请求后，DispatcherServlet会查询HandlerMapping以调用相应的Controller。

Controller接受请求并根据使用的GET或POST方法调用相应的服务方法。 服务方法将基于定义的业务逻辑设置模型数据，并将视图名称返回给DispatcherServlet。

DispatcherServlet将从ViewResolver获取请求的定义视图。

当视图完成，DispatcherServlet将模型数据传递到最终的视图，并在浏览器上呈现。

所有上述组件，即: HandlerMapping，Controller和ViewResolver是WebApplicationContext的一部分，它是普通ApplicationContext的扩展，带有Web应用程序所需的一些额外功能。

## Jar包

* servlet-api-x.y.z.jar
* commons-logging-x.y.z.jar
* spring-aop-x.y.z.jar
* spring-beans-x.y.z.jar
* spring-context-x.y.z.jar
* spring-core-x.y.z.jar
* spring-expression-x.y.z.jar
* spring-webmvc-x.y.z.jar
* spring-web-x.y.z.jar

## 配置文件

### Web.xml

|  |
| --- |
| <!— 加入Springmvc框架 -->  <servlet>  <servlet-name>HelloWeb</servlet-name>  <servlet-class>  org.springframework.web.servlet.DispatcherServlet  </servlet-class>  <!— springmvc配置自身文件读取路径方法一 -->  <init-param>  <param-name>contextConfigLocation</param-name>  <!— springmvc配置自身文件读取路径和文件名方法一 -->  <param-value>/WEB-INF/classes/cn/itcast/javaee/springmvc/config/springmvc.xml</param-value>  <!— springmvc配置文件名方法，读取路径为src目录下二 -->  <param-value>classpath:spring.xml</param-value>  </init-param>  <load-on-startup>1</load-on-startup>  </servlet>  <servlet-mapping>  <servlet-name>HelloWeb</servlet-name>  <url-pattern>\*.jsp</url-pattern>  </servlet-mapping>  <!— springmvc配置自身文件读取路径和文件名方法三 -->  <context-param>  <param-name>contextConfigLocation</param-name>  <param-value>/WEB-INF/HelloWeb-servlet.xml</param-value>  </context-param>  <listener>  <listener-class>  org.springframework.web.context.ContextLoaderListener  </listener-class>  </listener> |

### Springmvc配置文件

Hello-servlet.xml

|  |
| --- |
| <beans xmlns="http://www.springframework.org/schema/beans"  xmlns:context="http://www.springframework.org/schema/context"  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-3.0.xsd  http://www.springframework.org/schema/context  http://www.springframework.org/schema/context/spring-context-3.0.xsd">  <!— 扫描springmvc注解 -->  <context:component-scan base-package="com.yiibai" />  <!— 扫描springmvc注解 -->  <!— 扫描springmvc注解 -->  <!— 扫描springmvc注解 -->  <!— 扫描springmvc注解 -->  <!— 扫描springmvc注解 -->  <bean class="org.springframework.web.servlet.view.InternalResourceViewResolver">  <property name="prefix" value="/WEB-INF/jsp/" />  <property name="suffix" value=".jsp" />  </bean>  </beans> |

## Api

### 映射器

#### Bean名字url映射器

方式是将表现层的url和bean name绑定在一起

|  |
| --- |
| <!-- BeanNameUrlHandlerMapping 默认配置就是,所以用的时候,就不用配置 -->  <bean class="org.springframework.web.servlet.handler.BeanNameUrlHandlerMapping"></bean> |

将bean的id当做url

#### 控制器类名映射器

|  |
| --- |
| <!-- 控制类的类名控制器，访问时类名首字母需要小写 -->  <bean class="org.springframework.web.servlet.mvc.support.ControllerClassNameHandlerMapping"> </bean> |

* 对于/helloWorld.html或 /hello{任何字母}.html，DispatcherServlet将请求转发到HelloController类。
* 对于/welcome.html，DispatcherServlet会将请求转发给WelcomeController类。
* 对于/welcome1.html，DispatcherServlet将不会找到任何控制器，服务器将抛出404状态错误。

#### 简单url映射器

|  |
| --- |
| <bean class="org.springframework.web.servlet.handler.SimpleUrlHandlerMapping">  <property name="mappings">  <props>  <prop key="/welcome.html">welcomeController</prop>  <prop key="/helloWorld.html">helloController</prop>  </props>  </property>  </bean> |

使用上面的配置，如果URI：

* 对于/helloWorld.html请求，DispatcherServlet将请求转发到HelloController类。
* 对于/welcome.html，DispatcherServlet会将请求转发给WelcomeController类。

### 控制器

#### 多行为控制器

|  |
| --- |
| public class UserController extends MultiActionController{} |

|  |
| --- |
| <bean class="org.springframework.web.servlet.handler.BeanNameUrlHandlerMapping"/>  <bean name="/home.html" class="com.yiibai.springmvc.UserController" />  <bean name="/user/\*.html" class="com.yiibai.springmvc.UserController" /> |

* 对于/home.html请求，DispatcherServlet将请求转发到UserController类的 home()方法。
* 对于user/add.html，DispatcherServlet将请求转发到UserController类的 add()方法。
* 对于user/remove.html，DispatcherServlet将请求转发到UserController类的 remove()方法。

#### 属性方法名称控制器

|  |
| --- |
| public class UserController extends MultiActionController{} |

|  |
| --- |
| <bean class="com.yiibai.springmvc.UserController">  <property name="methodNameResolver">  <bean class="org.springframework.web.servlet.mvc.multiaction.PropertiesMethodNameResolver">  <property name="mappings">  <props>  <prop key="/user/home.html">home</prop>  <prop key="/user/add.html">add</prop>  <prop key="/user/update.html">update</prop>  </props>  </property>  </bean>  </property>  </bean> |

对于/user/home.html请求，DispatcherServlet将请求转发到UserController类的 home()方法。

对于user/add.html，DispatcherServlet将请求转发到UserController类的 add()方法。

对于user/remove.html，DispatcherServlet将请求转发到UserController类的 remove()方法。

#### 参数方法名称控制器

|  |
| --- |
| public class UserController extends MultiActionController{} |

|  |
| --- |
| <bean class="com.yiibai.springmvc.UserController">  <property name="methodNameResolver">  <bean class="org.springframework.web.servlet.mvc.multiaction.ParameterMethodNameResolver">  <property name="paramName" value="action"/>  </bean>  </property>  </bean> |

* 对于/user/\*.html?action=home请求，DispatcherServlet将请求转发到UserController类的 home()方法。
* 对于/user/\*.html?action=add，DispatcherServlet将请求转发到UserController类的 add()方法。
* 对于/user/\*.html?action=remove，DispatcherServlet将请求转发到UserController类的 remove()方法。

#### 可参数化的视图控制器

|  |
| --- |
| public class UserController extends MultiActionController{} |

|  |
| --- |
| <bean class="org.springframework.web.servlet.handler.SimpleUrlHandlerMapping">  <property name="mappings">  <value>  index.html=userController  </value>  </property>  </bean>  <bean id="userController" class="org.springframework.web.servlet.mvc.ParameterizableViewController">  <property name="viewName" value="index"/>  </bean> |

* 对于/index.html请求，DispatcherServlet会将请求转发到UserController控制器，并将viewName设置为index.jsp。

### 视图解析器

#### 内部资源视图解析器

|  |
| --- |
| <bean class="org.springframework.web.servlet.view.InternalResourceViewResolver">  <property name="prefix" value="/WEB-INF/jsp/"/>  <property name="suffix" value=".jsp"/>  </bean> |

DispatcherServlet会将响应转发到：前缀+返回字符串 + suffix = /WEB-INF/jsp/hello.jsp。

#### Xml视图解析器

|  |
| --- |
| <bean class="org.springframework.web.servlet.view.XmlViewResolver">  <property name="location">  <value>/WEB-INF/views.xml</value>  </property>  </bean> |

View.xml

|  |
| --- |
| <bean id="hello"  class="org.springframework.web.servlet.view.JstlView">  <property name="url" value="/WEB-INF/jsp/hello.jsp" />  </bean> |

DispatcherServlet会将请求转发到由view.xml中定义的hello对应的 hello.jsp 。

#### 资源绑定视图解析器

|  |
| --- |
| <bean class="org.springframework.web.servlet.view.ResourceBundleViewResolver">  <property name="basename" value="views" />  </bean> |

这里basename是指携带视图的资源束的名称。资源束的默认名称是views.properties，可以使用basename属性重写(也就是将views修改为其它名称，对应的views.properties文件名称也要修改)。

View.properties

|  |
| --- |
| hello.(class)=org.springframework.web.servlet.view.JstlView  hello.url=/WEB-INF/jsp/hello.jsp |

DispatcherServlet会将请求转发到由views.properties中定义的hello对应的 hello.jsp 。

这里“hello”是要匹配的视图名称。class指定视图类型，url是视图的位置。

#### 多解析器映射

如果想在spring mvc应用程序中使用多个视图解析器，那么可以使用order属性设置优先级顺序

|  |
| --- |
| <bean class="org.springframework.web.servlet.view.ResourceBundleViewResolver">  <property name="basename" value="views" />  <property name="order" value="0" />  </bean>  <bean class="org.springframework.web.servlet.view.InternalResourceViewResolver">  <property name="prefix" value="/WEB-INF/jsp/" />  <property name="suffix" value=".jsp" />  <property name="order" value="1" />  </bean> |

这里order属性定义了视图解析器的排序。0作为第一解析器，1作为下一解析器，等等。

View.properties

|  |
| --- |
| hello.(class)=org.springframework.web.servlet.view.JstlView  hello.url=/WEB-INF/jsp/hello.jsp |

对于/hello请求，DispatcherServlet会将请求转发到由views.properties中定义的hello对应的 hello.jsp 。

这里“hello”是要匹配的视图名称。class指定视图类型，url是视图的位置。

## 注解

1、@Controller注释将类定义为Spring MVC控制器。

2、@RequestMapping 表示url映射关系。value表示url映射路径；method 表示处理的请求类型，如RequestMethod.GET。该注解可以在类、方法、属性上同时定义，其路径相加得到的url即为该方法处理的url。

## 参考

Springmvc教程

<http://www.yiibai.com/spring_mvc/springmvc_beannameurlhandlermapping.html>