SpringMVC

1. It’s a framework
2. Servlet 升级版

Spring是容器 通过ioc能过管理对象 使用<bean>标签 @controller

Springmvc能够创建对象 放入到容器中，springmvc容器中放的是控制器对象

使用@controller 创建控制器对象，把对象放入到springmvc容器中，把创建对象作为控制器使用这个控制器对象能接受用户的请求，显示处理结果，就当做是一个servlet使用

使用@controller注解创建的是一个普通类对象，不是servlet, 因为servlet要继承父类httpsrverlt。Springmvc赋予了控制器对象一些额外的功能。

Web 开发底层是servlet， springmvc中有个对象是servlet – dispatchservlet类，中文叫中央调度器。Dispatchservlet：负责接收用户所有的请求，它把请求转发给controller对象，最后controller对象处理请求

Index.jsp -> dispatchservler -> 转发分配给controller对象（@controller注解创建的对象） -> 处理后结果返回dispatchservlet -> 再返回给index.jsp

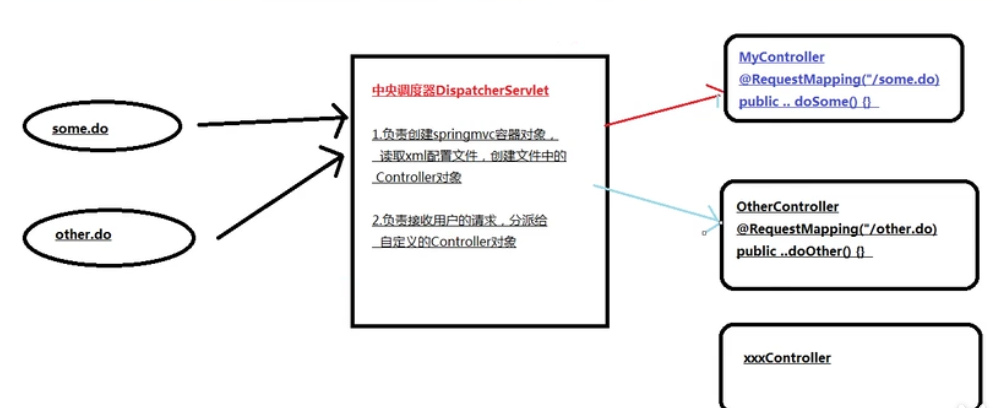
\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Springmvc 请求的处理流程

1. 在index.jsp 有个Somde.do的请求
2. tomcat(web.xml, 根据url-pattern的\*.d请求给dspatchservlet)
3. dispatcherservlet（根据springmvc.xml配置知道some.do -> MyController.doSome()）
4. 框架执行dosome()把得到modelandview进行处理，转发到show.jsp

上面的过程简化方式

Some.do – Dispatchservlet – Mycontroller



Springmvc执行过程源代码分析

1. tomcat启动，创建容器的过程

通过web.xml中的load-on-start标签指定的1，创建dispaltcherservlet对象，dispaltcherservlet的父类是继承httpservelt的，它是一个servlet，在被创建时，会执行init（）方法。

在init（）方法里

//创建容器，读取配置文件

WebapplicationContext ctx = new ClassPathXmlApplicationContext("springmvc.xml")

//把容器对象放入到servletcontext中

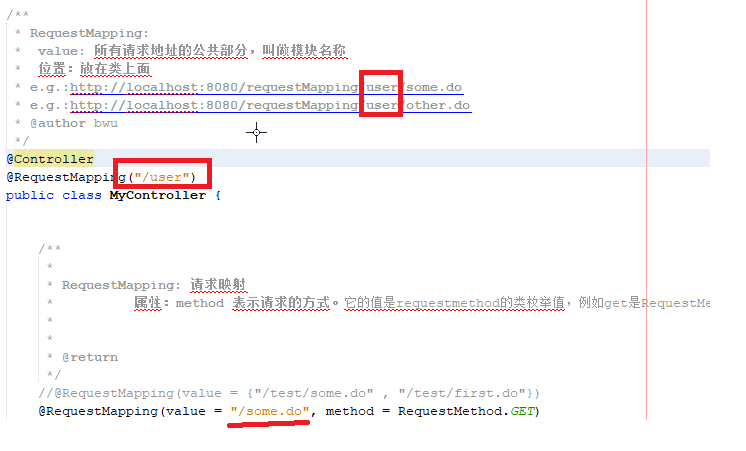
getServletContextx.setAttribute(key, ctx);

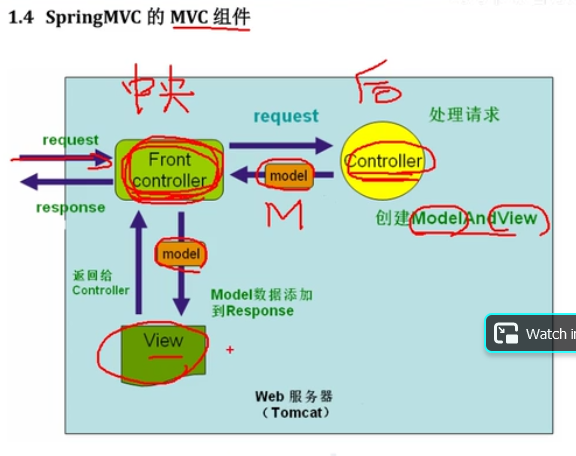
上面创建容器的租用：创建@controller注解所在的类的对象（mycontroller对象），把这个对象放入到springmvc容器，容器是个map，类似map.put（“mycontroller”， mycontroller对象）

1. 请求的处理过程

执行dispaltcherservlet中的service() -> dispaltcherservlet.doserivce() -> dispaltcherservlet.Dodispatch() (这个方法会调用mycontroller.dosome())

1. 公用用户模块







发起的请求是由哪里服务器程序处理的.

<http://localhost:8080/05-url-pattern/index.jsp>: 由tomcat完成请求处理工作 (jsp转为servlet)

<http://localhost:8080/05-url-pattern/js/jquery-1.11.1.min.js>: 由tomcat完成请求处理工作

05-url-pattern/images/stash.png: 由tomcat完成请求处理工作

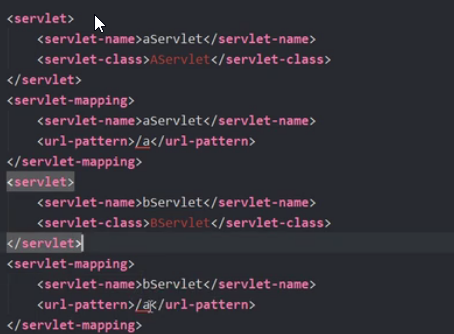
<http://localhost:8080/05-url-pattern/html/test.html> 由tomcat完成请求处理工作

<http://localhost:8080/05-url-pattern/some.do> 由DispatchServlet(sprintmvc 框架)完成请求处理工作

Tomcat本身能处理静态资源 (.jsp, .js, .png…) 的访问

Tomcat的web.xml中有一个servlet – default, 在服务器启动时创建

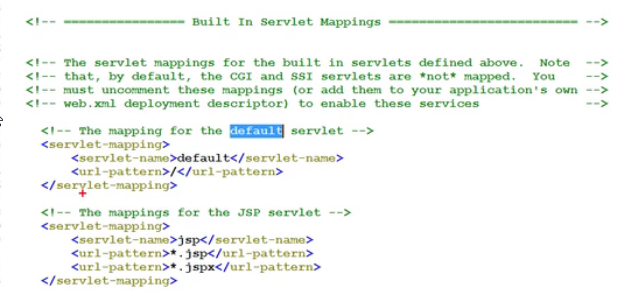




访问localhost:8080/myweb/a, 由aServlet处理

访问localhost:8080/myweb/b. 由bServlet处理

访问localhost:8080/myweb/c. 因为再web.xml找不到对应的servelet，没有映射到任何一个servlet





在jsp，html使用的地址都是前端页面的地址，都是相对地址

地址分类

1. 绝对地址，带有协议名称的绝对地址，<http://www.baidu.com>, <ftp://202.122.23.1>
2. 相对地址，没有协议开头的，例如 user/some.do, /user/some.do’

相对地址不能独立使用，必须又一个参考地址，通过参考地址+相对地址本身才能指定访问资源。

1. 参考地址

1）在页面中，访问地址不加“/”

访问的是 localhost:8080/ch06\_path/index.jsp

路径: localhost:8080/ch06\_path/

资源: index.jsp

在index.jsp发起user/some.do的请求，访问地址变为localhost:8080/ch06\_path/user/some.do

当你的地址没有斜杠开头，例如user/some.do，当你点击链接时，访问地址是当前页面的地址加上链接的地址。localhost:8080/ch06\_path/ + user/some.do

在springmvc.xml去掉了视图解析器后, 在这个处理器中视图为mv.setViewName(“/index.jsp”) 。此时在index.jsp 访问user/some.do 返回的地址是localhost:8080/ch06\_path/user/some.do，网页看到的内容是index.jsp 此时再点击some.do 返回的地址为变为：localhost:8080/ch06\_path/user/user/some.do (404)

解决方法

1. 再index.jsp中加入$(pageContext.request.getContextPath)
2. 加入base标签，是html标签，表示当前页面访问地址的基地址。

在页面中的没有“/”开头的地址，都是以base标签中的地址为基地址

使用base中的地址 + user/some.do 组成访问地址

2) 在页面中，访问地址加“/”

访问的是 localhost:8080/ch06\_path/index.jsp

路径: localhost:8080/ch06\_path/

资源: index.jsp

点击/user/some.do 访问地址变更为localhost:8080/user/some.do

参考地址的是：你的服务器地址，也就是localhost:8080/

如果你的资源不能访问，加入$(pageContext.request.getContextPath)