**springmvc基础**

**1、本章面试题**

           springmvc（m=model v=view c=control）的工作原理(流程)

springmvc 常用注解

**2、知识点**

**2.1、课程回顾**

       spring aop核心思想

spring aop通知类型及实现方式

**2.2、本章重点**

         springmvc 工作原理（流程）

springmvc 实现过程

springmvc 常用注解

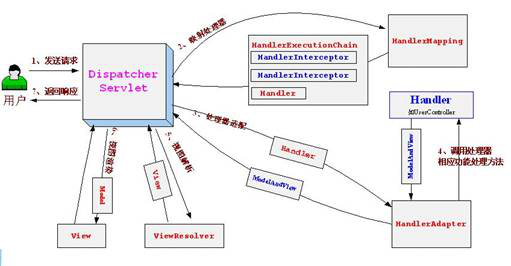
springmvc 参数传递和数据绑定

**3、具体内容**

**3.1 简介**

Spring mvc和Struts2都属于表现层（控制层）的框架,它是Spring框架的一部分.(表现层框架，学习核心，学习如何从页面（web端）取数据，如何与后台服务交互，交互后如何把数据绑定，在前端显示)

**3.2 工作原理**



1、客户端发出一个http请求给web服务器，web服务器对http请求进行解析，如果匹配 DispatcherServlet的请求映射路径（在web.xml中指定），web容器将请求转交给DispatcherServlet.

2、DipatcherServlet接收到这个请求之后将根据请求的信息以及HandlerMapping的配置找到处理请求的处理器（Handler）。

3-4、DispatcherServlet根据HandlerMapping找到对应的Handler,将处理权交给Handler（Handler将具体的处理进行封装），再由具体的HandlerAdapter对Handler进行具体的调用。

5、Handler对数据处理完成以后将返回一个ModelAndView()对象给DispatcherServlet。

6、Handler返回的ModelAndView()只是一个逻辑视图并不是一个正式的视图，DispatcherSevlet通过ViewResolver将逻辑视图转化为真正的视图View。

7、Dispatcher通过model解析出ModelAndView()中的参数进行解析最终展现出完整的view并返回给客户端。

**3.3 springmvc配置过程**

创建项目，添加支持

修改web.xml头部（低版本（2.3）的头部jsp默认不支持EL表达式）

<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\_3\_1.xsd"

version="3.1">

常量配置：<spring.version>4.2.4.RELEASE</spring.version>

<dependency>

<groupId>org.springframework</groupId>

<artifactId>spring-context</artifactId>

<version>${spring.version}</version>

</dependency>

<dependency>

<groupId>org.springframework</groupId>

<artifactId>spring-core</artifactId>

<version>${spring.version}</version>

</dependency>

<dependency>

<groupId>org.springframework</groupId>

<artifactId>spring-beans</artifactId>

<version>${spring.version}</version>

</dependency>

<!-- https://mvnrepository.com/artifact/org.springframework/spring-webmvc -->

<dependency>

<groupId>org.springframework</groupId>

<artifactId>spring-webmvc</artifactId>

<version>${spring.version}</version>

</dependency>

<!-- https://mvnrepository.com/artifact/javax.servlet/javax.servlet-api -->

<dependency>

<groupId>javax.servlet</groupId>

<artifactId>javax.servlet-api</artifactId>

<version>4.0.1</version>

</dependency>

<dependency>

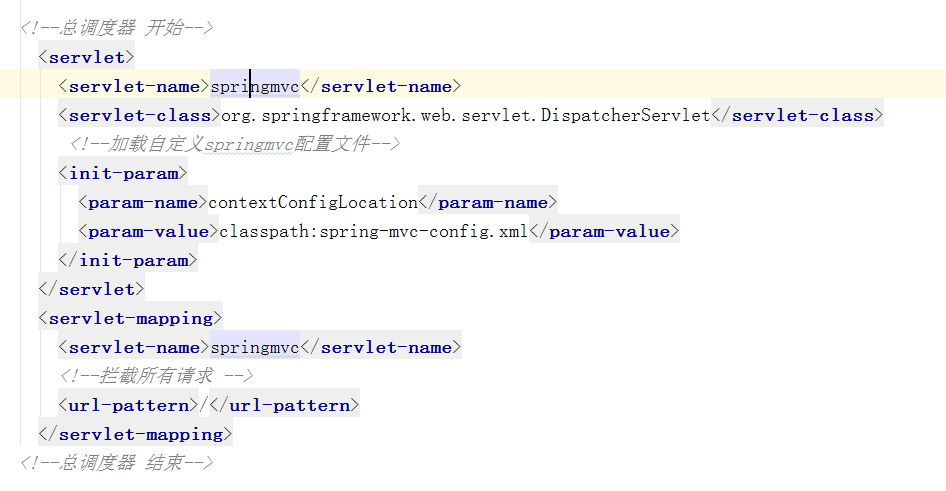
<groupId>jstl</groupId>

<artifactId>jstl</artifactId>

<version>1.2</version>

</dependency>

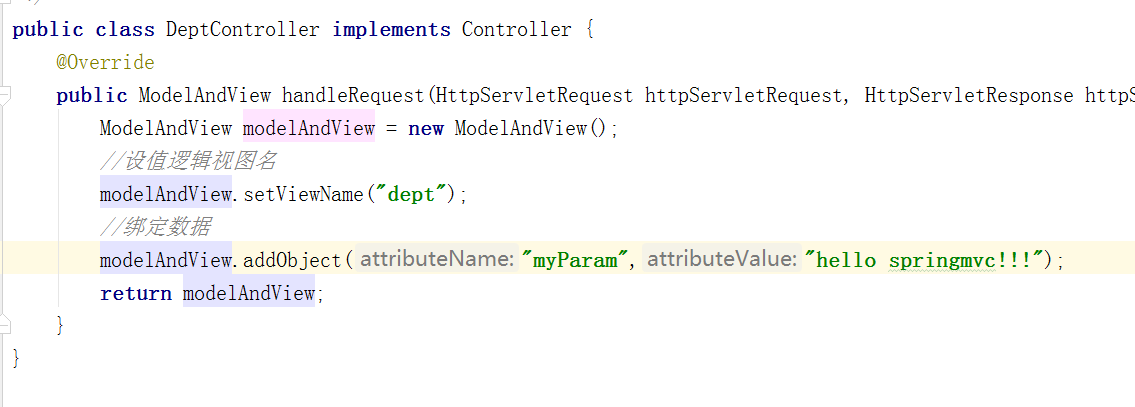
1，配置总调度器



2，创建springmvc配置文件,添加映射器，适配器，自定义处理和视图解析器



3，编写自定义处理器



4，编写页面



**3.4 常用注解（在ssm整合中讲解）**

**@Controller**

用于标记在一个类上，使用它标记的类就是一个SpringMVC Controller 对象。分发处理器将会扫描使用了该注解的类的方法，并检测该方法是否使用了@RequestMapping 注解。

**@Autowired**

注解是按照类型（byType）装配依赖对象，默认情况下它要求依赖对象必须存在，如果允许null值，可以设置它的required属性为false。如果我们想使用按照名称（byName）来装配，可以结合@Qualifier注解一起使用。

**@Resource**

默认按照ByName自动注入，由J2EE提供，需要导入包javax.annotation.Resource。@Resource有两个重要的属性：name和type，而Spring将@Resource注解的name属性解析为bean的名字，而type属性则解析为bean的类型。所以，如果使用name属性，则使用byName的自动注入策略，而使用type属性时则使用byType自动注入策略。如果既不制定name也不制定type属性，这时将通过反射机制使用byName自动注入策略。

**@RequestMapping**

用于处理请求地址映射，可以作用于类和方法上。用于类上，表示类中的所有响应请求的方法都是以该地址作为父路径。

value：定义request请求的映射地址

method：定义request址请求的方式，包括【GET, POST, HEAD, OPTIONS, PUT, PATCH, DELETE, TRACE.】默认接受get请求，如果请求方式和定义的方式不一样则请求无法成功。

params：定义request请求中必须包含的参数值。

**@RequestParam**

用于将请求参数区数据映射到功能处理方法的参数上

value：参数的名称

required： 为false可以null,true必须传递参数

defaultValue : 可以为null时的默认值

**@PathVariable**

用于将请求URL中的模板变量映射到功能处理方法的参数上，即取出uri模板中的变量作为参数。

value：参数的名称

required：定义传入参数是否为必须值

**@ResponseBody**

　　作用： 该注解用于将Controller的方法返回的对象，通过适当的HttpMessageConverter转换为指定格式后，写入到Response对象的body数据区。

使用时机：返回的数据不是html标签的页面，而是其他某种格式的数据时（如json、xml等）使用；

**3.5 参数传递，回传参数**

1，回传List<Map>到页面





**4、本章总结**

**4.1 总结本章知识点**

**4.2 面试题答案**

**4.3 预习下一章重点**