1、SpringMVC的概述

SpringMVC是一个Web层（控制层）的框架，主要用来负责与客户端（浏览器端）交互，以及业务逻辑的调用。

Spring框架是Service层（业务访问层）的框架。

SpringMVC的特点是简单、易用，性能佳。

SpringMVC替代了之前的Struts2框架，也是用于替代Servlet技术的功能。

2、SpringMVC相比Servlet技术的优势

（1）Servlet的开发配置相对麻烦，Servlet特别多的时候web.xml文件就会非常臃肿

（2）每个Servlet都只能处理一个功能，如果需要多个功能就需要开发多个Servlet，项目中存在大量的Servlet也会显得臃肿。

（3）获取请求参数进行类型转换，包括封装到bean的过程比较繁琐。

（4）其它开发中不方便的地方，例如：乱码问题、数据格式的处理、表单的验证等等。

3、SpringMVC的组件

（1）前端控制器（DispatcherServlet）

本质上前端控制器DispatcherServlet是一个Servlet，相当于一个中转站，所有的访问都会走到这个Servlet中，在根据配置进行中转到响应的Handler（处理器）中进行处理，获取到数据和视图后，在使用相应视图做出响应。

（2）处理器映射器（HandlerMapping）

本质上是一段映射关系，将访问路径和对应的Handler存储为映射关系，在需要时供前端控制器查阅。

（3）处理器适配器（HandlerAdapter）

本质上就是一个适配器，可以根据配置的要求找到对应的Handler来运行。前端控制器通过处理器映射器找到对应的Handler信息之后，将请求响应和对应Handler信息交由处理器适配器处理，处理器适配器找到真正的Handler来执行，Handler执行，会产生一个ModelAndView类型的结果返回给前端过滤器。

（4）视图解析器（ViewResolver）

本质上也是一个映射关系，可以将视图名称映射到真正的视图地址。

前端控制器调用处理器适配器完成后得到ModelAndView结果，将View信息传递给视图解析器得到真正的View（浏览器端的JSP页面）。

（5）视图（View）

本质上是将Handler处理器中返回的Model数据嵌入到视图解析器解析后得到的jsp页面中，向浏览器端做出响应。

4、SpringMVC的第一个入门案例

Step1：新建一个Web项目

Step2：导入开发需要的jar包

Step3：在web.xml文件中配置前端控制器DispatcherServlet

Step4：创建一个SpringMVC框架的核心配置文件

SpringMVC默认会自动在Web项目中的WEB-INF目录下去寻找（前端控制器

servlet-name）的名字-servlet.xml作为当前SpringMVC的核心配置文件。

一般习惯把核心配置文件放在src目录下，需要在web.xmml中配置一下核心配置文件的位置。

springmvc.xml核心配置文件需要引入Spring框架开发中的一些约束（beans,context,mvc...）

step5：在src下定义包，在包下开发Handler，在Handler中定义方法，通过@RequestMapping完成对方法的注解配置，并方法返回ModelAndView对象。

Step6：在springmvc.xml核心文件中配置视图解析器

Step7：前端控制器根据视图解析器的解析结果，跳转到真正的View视图页面，并在真

正的View视图页面中获取Model中的数据

**SpringMVC框架的细节1—@RequestMapping**

5、@RequestMapping注解

通过注解方法实现路径到处理器方法的映射。

可以用在类或方法上

用在方法上表示将该方法作为一个处理器，且和指定路径做映射；

用在类上则配置的路径会作为这个类中所有处理器中方法的父路径使用。

6、@RequestMapping注解中的属性

（1）value属性

指定要将当前处理器（类或者方法）绑定到哪个访问路径上

可以配置多个路径；

路径中也可以使用\*号作为通配符。

（2）method属性

指定当前处理器处理哪种提交方式的请求（理解为Servlet类中使用对应的doGet

或者doPost方法）；

如果不指定则接收任意请求方式的请求（理解为Servlet类中使用servce方法）

（3）params属性

用来限制请求中的参数

params属性有一些格式要求：

格式1：只指定名称，要求必须具有该名称的请求参数

格式2：以”!名称”的方式指定要求必须不包含指定名称的请求参数

格式3：以”名称=值”或着”名称!=值”的方式指定，要求必须具有指定的名称

的参数，且值必须等于或者不等于给定的值

**SpringMVC框架的细节2---处理器中的方法可以定义的参数**

7、处理器中的方法可以定义的参数

（1）和web开发相关的对象

可以在控制器方法中直接接受web开发相关的对象，来完成web开发相关的功能

HttpServletRequest

HttpServletResponse

HttpSession

WebRequest

WebRequest对象相当于是HttpServletRequest和HttpSession对象的合体,同时提供了HttpServletRequest和HttpSession对象中所具有的方法。

（2）和模型对象相关的

ModelAndView对象

Model对象

Map对象

ModelMap对象

（3）请求参数相关的

A 处理器方法可以通过定义HttpServletRequest对象来获取请求参数

B 处理器方法可以定义方法的参数名和请求参数名一致的方式，用来获取请求

参数。

C 处理器方法定义的参数名和请求参数名不一致的情况，可以使用

@RequestParam注解来获取请求参数传递过来的值。

D 请求参数中有多个同名的参数的情况，既可以使用字符串数组的方式接收，也

可以使用字符串的方式接收。

E 处理器的方法可以接收参数，并自动封装到javabean中。

F 处理器的方法接收参数，javabean包含其它javabean的情况

（4）请求参数中中文乱码问题

方法1：request.setCharacterEncoding(“utf-8”)

此方法必须用在获取任何请求参数之前

此方法只针对Post提交有效

方法2：修改tomcat服务器的server.xml中的配置

<Connection port=”8080”... URIEncoding=”UTF-8”/>

此方法只针对于Get提交有效

方法3：手动编码解决乱码问题

byte[] data=str.getByte(“iso8859-1”);

str=new String(data,”utf-8”)

此方法对Get和Post都有效

每个参数都需要手动编码处理太麻烦

方法4；Spring框架提供了过滤器处理乱码，只针对Post提交有效

@RequestMapping、@RequestParams

**SpringMVC框架的细节3 – 作用域属性**

8、@RequestAttribute

用在处理器方法参数中

用来从request域中获取指定名称属性的属性值

9、@SessionAttribute

用在处理器方法参数中

用来从session域中获取指定名称属性的属性值

**SpringMVC框架的细节4 – Cookie技术**

10、@CookieValue

用在处理器方法参数中

用来从Cookie中获取指定名称的Cookie值

**SpringMVC框架的细节5 - 处理器方法的返回值类型**

（1）ModelAndView返回值类型

返回一个ModelAndView对象，在其中封装Model和View信息

（2）字符串返回值类型

A 普通字符串类型：返回的字符串将作为视图名来使用

B forward组成的字符串类型：通过返回forward:/xxx格式的字符串表明要**转发**的指

定地址

C redirect组成的字符串类型：通过返回redirect:/xxx格式的字符串表示要**重定向**

的地址

（3）void返回值类型

如果返回值类型是void，则SpringMVC框架会采用默认视图名来作为当前请求视图名

称来使用。

默认路径：/WEB-INF/jsp/处理器上的@RequestMapping映射地址/处理器方法上的@RequestMapping的映射地址.jsp的页面地址

（4）其它返回值类型

除以上之外返回的任何内容都会被存入模型对象中供后续使用；

值为返回的数据，键为返回类型首字母转小写；

而返回的视图名是默认视图名，即等同于返回值为void时的视图名