Spring MVC框架

分享人:黄阳阳

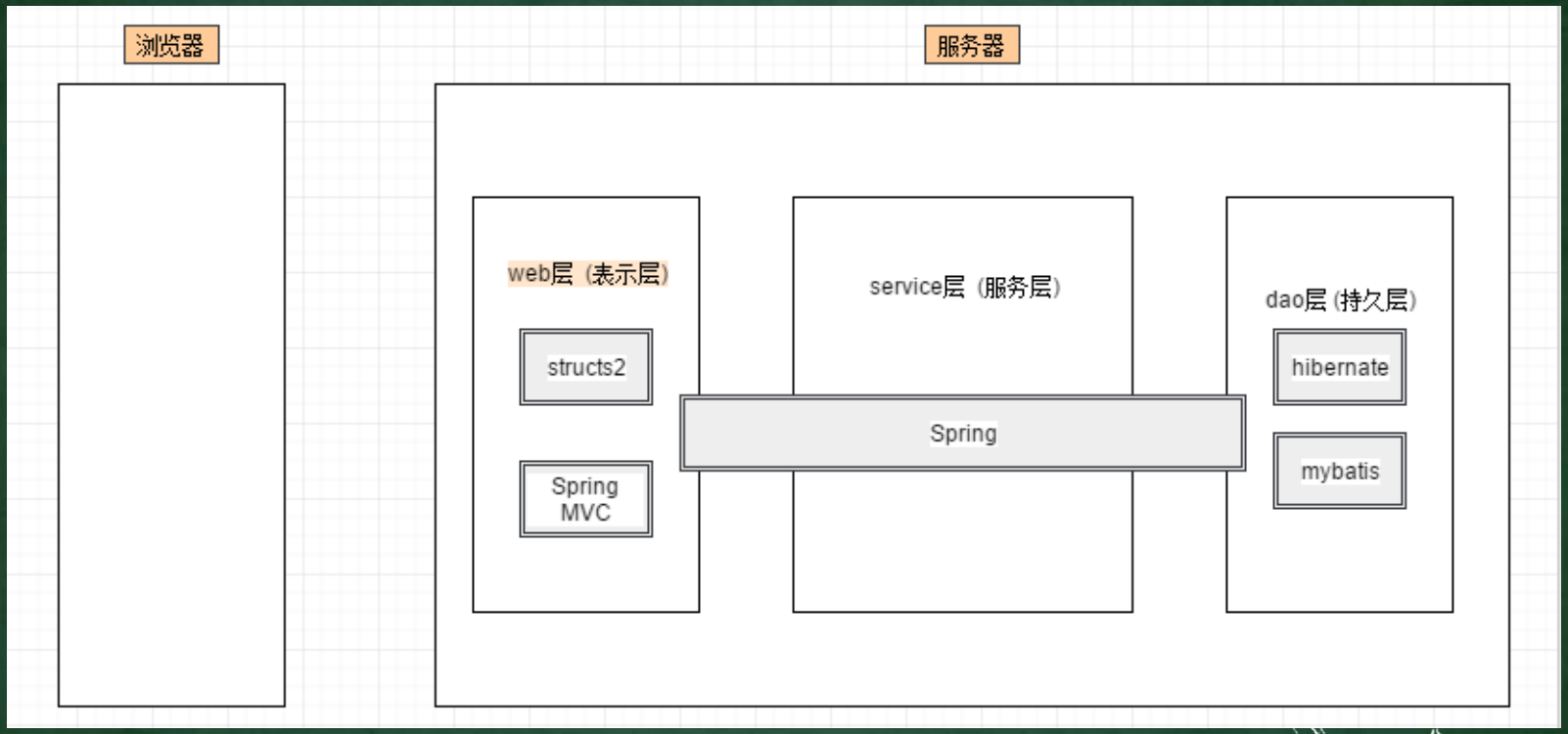
日期：2017/12/18

内容：

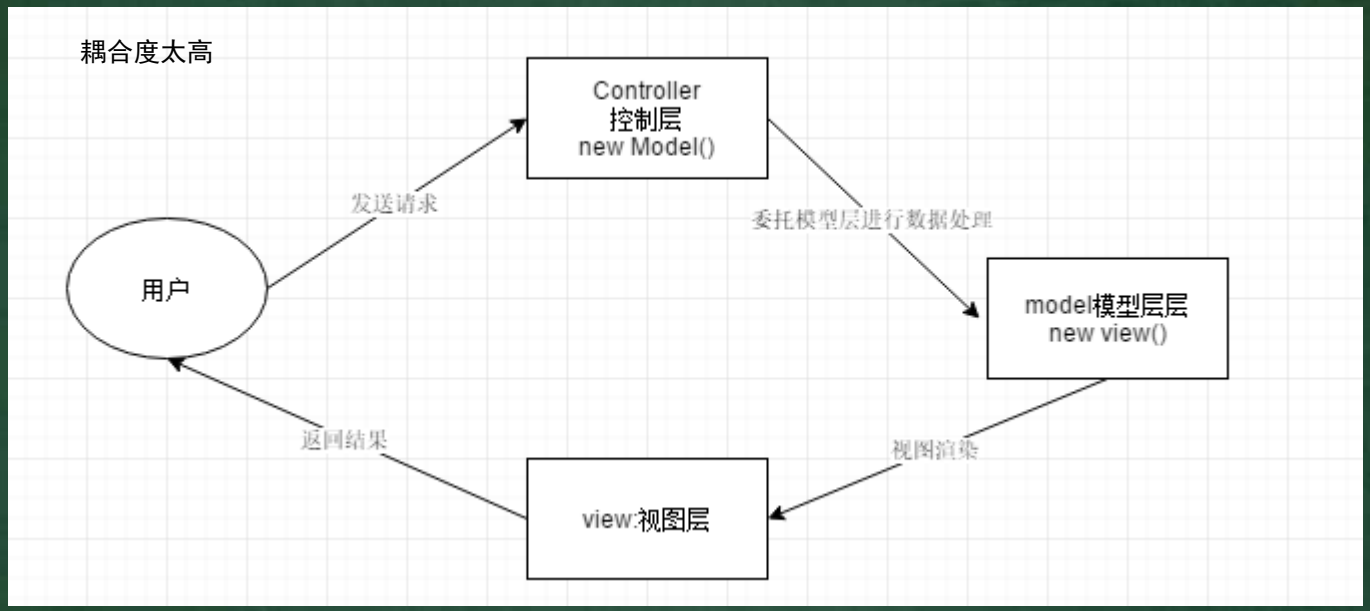
- 什么是Spring MVC
- 使用Spring MVC的优势
- Spring MVC运行原理
- Spring MVC的具体实现

什么是Spring MVC?

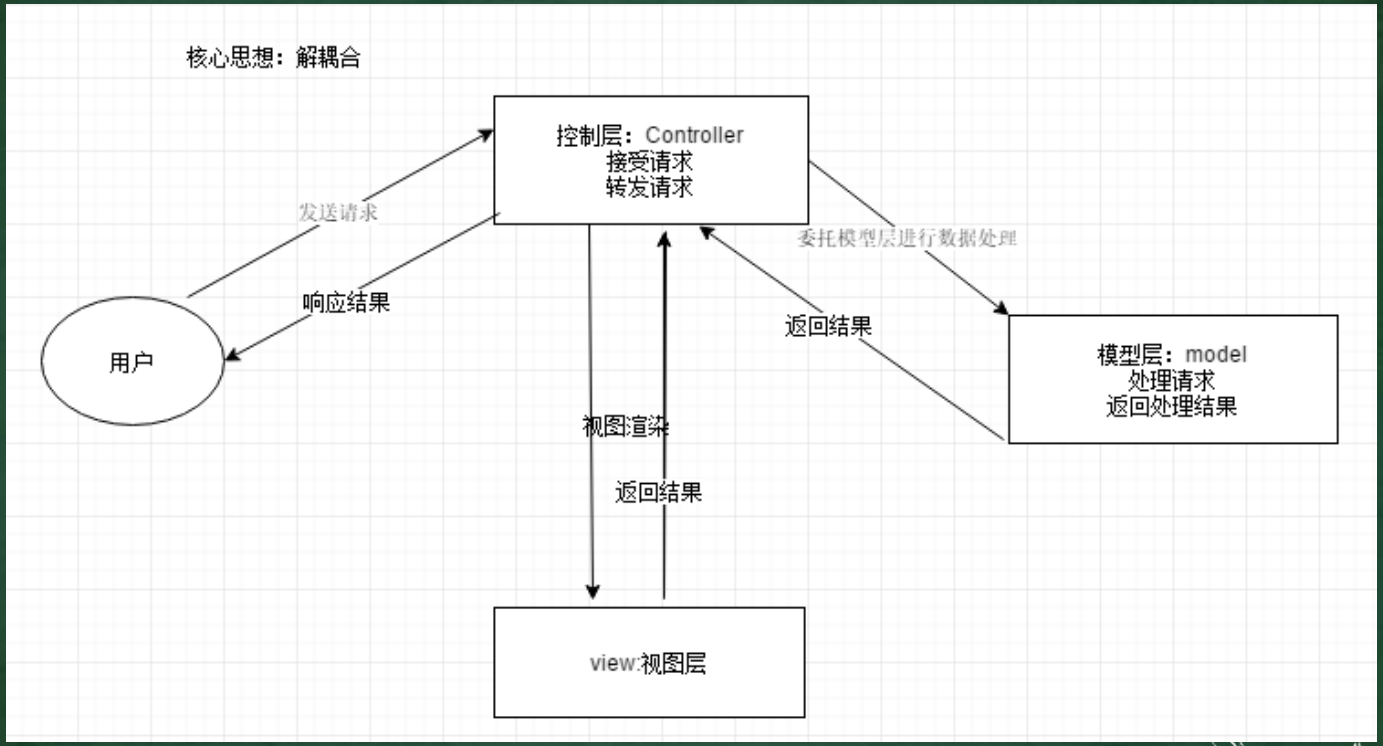
JAVAEE框架体系结构



早期的设计模式


耦合度太高
Controller
控制层
new Model()
发送请求
委托模型层进行数据处理
用户
model模型层层
new view()
返回结果
视图渲染
view:视图层

MVC设计模式


核心思想：解耦合
控制层：Controller
接受请求
转发请求
发送请求
响应结果
用户
委托模型层进行数据处理
返回结果
模型层：model
处理请求
返回处理结果
视图渲染
返回结果
view:视图层

Spring MVC框架定义

Spring MVC是一种基于Java的实现了MVC设计模式的请求驱动类型的轻量级Web框架，即使用了MVC架构模式的思想，将web层进行职责解耦，基于请求驱动指的就是使用请求-响应模型，框架的目的就是帮助我们简化开发，Spring MVC也是要简化我们日常Web开发的。

使用Spring MVC的优势？

springmvc优势

- 1、清晰的角色划分：前端控制器（DispatcherServlet）、处理器映设器（HandlerMapping）、处理器适配器（HandlerAdapter）、视图解析器（ViewResolver）、处理器或页面控制器（Controller）、验证器（ Validator ）、命令对象（Command 请求参数绑定到的对象就叫命令对象）、表单对象（Form Object 提供给表单展示和提交到的对象就叫表单对象）。

- 2、分工明确，而且扩展点相当灵活，可以很容易扩展，虽然几乎不需要；

- 3、由于命令对象就是一个POJO，无需继承框架特定API，可以使用命令对象直接作为业务对象；

- 4、和Spring 其他框架无缝集成，是其它Web框架所不具备的；

- 5、可适配，通过HandlerAdapter可以支持任意的类作为处理器
- 6、功能强大的数据验证、格式化、绑定机制；
- 7、利用Spring提供的Mock对象能够非常简单的进行Web层单元测试；
- 8、本地化、主题的解析的支持，使我们更容易进行国际化和主题的切换。
- 9、强大的JSP标签库，使JSP编写更容易。
- 10、等等……

有了Struts2，为什么还需要springmvc?

- 一.实现机制：

Struts2是基于过滤器实现的。Springmvc是基于servlet实现，servlet比过滤器快。

- 二.运行速度

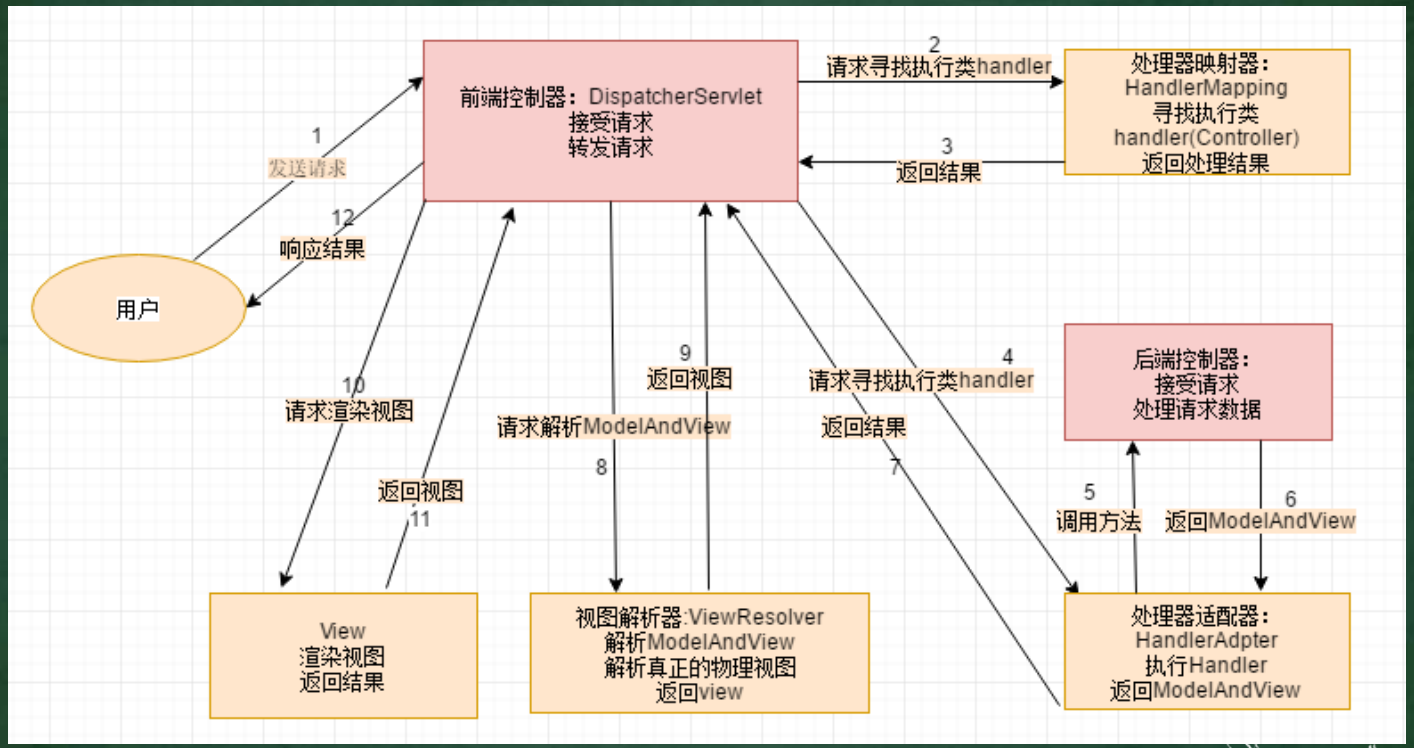
Struts2是多例，请求来了以后，struts2创建多少个对象：Springmvc是单例的。

- 三.参数封装来分析：

Struts基于属性进行封装，Springmvc基于方法封装。

Spring MVC的运行原理？

Spring MVC原理图（执行流程）

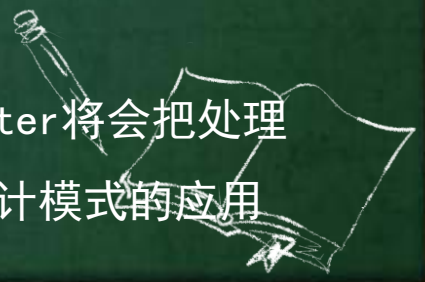


核心架构的具体流程步骤如下：

- 1、首先用户发送请求——>DispatcherServlet，前端控制器收到请求后自己不进行处理，而是委托给其他的解析器进行处理，作为统一访问点，进行全局的流程控制；

- 2、DispatcherServlet——>HandlerMapping,HandlerMapping将会把请求映射为HandlerExecutionChain对象（包含一个Handler处理器（页面控制器）对象、多个HandlerInterceptor拦截器）对象，通过这种策略模式，很容易添加新的映射策略；

- 3、DispatcherServlet——>HandlerAdapter，HandlerAdapter将会把处理器包装为适配器，从而支持多种类型的处理器，即适配器设计模式的应用，从而很容易支持很多类型的处理器；



核心架构的具体流程步骤如下：

- 4、HandlerAdapter——>处理器功能处理方法的调用，HandlerAdapter将会根据适配的结果调用真正的处理器的功能处理方法，完成功能处理；并返回一个ModelAndView对象（包含模型数据、逻辑视图名）；

- 5、ModelAndView的逻辑视图名——> ViewResolver， ViewResolver将把逻辑视图名解析为具体的View，通过这种策略模式，很容易更换其他视图技术；

- 6、View——>渲染，View会根据传进来的Model模型数据进行渲染，此处的Model实际是一个Map数据结构，因此很容易支持其他视图技术；

- 7、返回控制权给DispatcherServlet，由DispatcherServlet返回响应给用户，到此一个流程结束。

Spring MVC如何实现？

一.普通方式实现

- 1.创建一个web工程
- 2.导入相应的jar包
- 3.配置DispatcherServlet前端控制器（web.xml）

```
<servlet>
<servlet-name>springmvc</servlet-name>
<servlet-class>org.springframework.web.servlet.DispatcherServlet</servlet-class>
<!-- 默认加载方式
    默认加载必须规范:
       * 文件命名: servlet-name-servlet.xml====springmvc-servlet.xml
       * 路径规范: 必须在WEB-INF目录下面
 -->
 <init-param>
 <param-name>contextConfigLocation</param-name>
 <param-value>classpath:springmvc.xml</param-value>
 </init-param>
</servlet>

<servlet-mapping>
<servlet-name>springmvc</servlet-name>
<url-pattern>*.do</url-pattern>
</servlet-mapping>
```

- 4.配置处理器映射器（handlerMapping）
- 5.配置处理器适配器（handlerAdapter）
- 6.配置视图解析器（ResourceViewResolver）

配置springmvc.xml文件

```
<!-- 配置处理器映射器,springmvc默认的处理器映射器
BeanNameUrlHandlerMapping: 根据bean(自定义Controler)的name属性的url去寻找hanler(Action:Controller)
 -->
<bean class="org.springframework.web.servlet.handler.BeanNameUrlHandlerMapping"></bean>

<!-- 配置处理器适配器执行Controlelr ,springmvc默认的
SimpleControllerHandlerAdapter:执行Controller
-->
<bean class="org.springframework.web.servlet.mvc.SimpleControllerHandlerAdapter"></bean>

<!-- 配置自定义Controler -->
<bean name="/hello.do" class="cn.itcast.controller.MyController"></bean>

<!-- 配置sprigmvc视图解析器: 解析逻辑试图
     后台返回逻辑试图: index
     视图解析器解析出真正物理视图: 前缀+逻辑试图+后缀====/WEB-INF/jsps/index.jsp
-->
<bean class="org.springframework.web.servlet.view.InternalResourceViewResolver">
<property name="prefix" value="/WEB-INF/jsps/"></property>
<property name="suffix" value=".jsp"></property>
</bean>
</beans>
```

- 6.编写一个Controller类

```
public class MyController implements Controller{

    public ModelAndView handleRequest(HttpServletRequest arg0,
            HttpServletResponse arg1) throws Exception {
        //  接受请求，接受参数，验证参数
        //封装参数，调用业务方法
        //返回视图
        ModelAndView mv = new ModelAndView();
        //设置页面回显数据
        mv.addObject("hello", "欢迎学习springmvc! ");

        //指定跳转的视图
        //返回物理视图
        //mv.setViewName("/WEB-INF/jsps/index.jsp");
        //返回逻辑视图
        mv.setViewName("index");

        return mv;
    }

}
```

7.定义视图页面（index.jsp）

根据视图解析路径：WEB-INF/jsps/index.jsp


```
</head>
<body>
${hello }
</body>
```

二. 注解方式实现

- 1. 创建一个web工程
- 2. 导入相应的jar包
- 3. 配置前端控制器（略）
- 4. 配置注解扫描
- 5. 配置注解映射器
- 6. 配置注解适配器
- 7. 配置视图解析器

8. 配置springmvc.xml文件

```
    <!-- 添加注解扫描!!! -->
<context:component-scan base-package="cn.whucs.controller"></context:component-scan>
<!-- 添加注解映射器 -->
<bean class="org.springframework.web.servlet.mvc.method.annotation.RequestMappingHandlerMapping"></bean>
<!-- 注解适配器 -->
<bean class="org.springframework.web.servlet.mvc.method.annotation.RequestMappingHandlerAdapter"></bean>
<!-- 视图解析器 -->
<bean class="org.springframework.web.servlet.view.InternalResourceViewResolver">
<property name="prefix" value="/WEB-INF/jsps/"></property>
<property name="suffix" value=".jsp"></property>
</bean>
```

注意：<mvc:annotation-driven /> 是一种简写形式，会自动注册DefaultAnnotationHandlerMapping与AnnotationMethodHandlerAdapter 两个bean。

- 9.编写一个Controller类

```
 */
@Controller
@RequestMapping("/user")
public class MyOneController {
    @RequestMapping("/fun")
    public String fun() {
        return "hello";
    }
}
```

- 10.编写hello的JSP页面返回结果视图（略）

Spring MVC常用注解

- 1.@Controller

在SpringMVC 中，控制器Controller 负责处理由DispatcherServlet 分发的请求，它把用户请求的数据经过业务处理层处理之后封装成一个Model ，然后再把该Model 返回给对应的View 进行展示。在SpringMVC 中提供了一个非常简便的定义Controller 的方法，无需继承特定的类或实现特定的接口，只需使用@Controller标记一个类是Controller ，然后使用@RequestMapping 和@RequestParam 等一些注解用以定义URL 请求和Controller 方法之间的映射，这样的Controller 就能被外界访问到。此外Controller 不会直接依赖于HttpServletRequest 和HttpServletResponse 等HttpServlet 对象，它们可以通过Controller 的方法参数灵活的获取到。

- 2.@RequestMapping

RequestMapping是一个用来处理请求地址映射的注解，可用于类或方法上。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。RequestMapping注解有六个属性，下面我们把它分成三类进行说明。

1、 value， method;

value：指定请求的实际地址，指定的地址可以是URI Template 模式;

method：指定请求的method类型， GET、POST、PUT、DELETE等;

2、consumes，produces

consumes：指定处理请求的提交内容类型（Content-Type），例如 application/json, text/html;

produces: 指定返回的内容类型，仅当request请求头中的(Accept)类型中包含该指定类型才返回;

3、params，headers

params：指定request中必须包含某些参数值是，才让该方法处理。

headers：指定request中必须包含某些指定的header值，才能让该方法处理请求。

- 3.@RequestBody/@ResponseBody

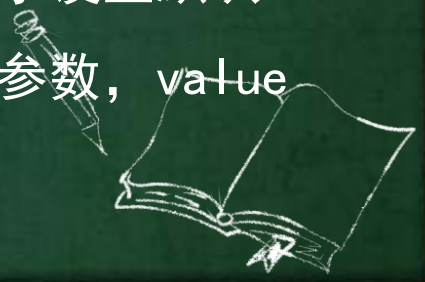
两种都是用来处理json数据，@RequestBody注解用于接收前台json数据，把json数据自动封装成javaBean,@ResponseBody是把后台javaBean转换成json格式的数据返回给前端页面。

- 4.@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。

- 5.@requestParam

主要用于在SpringMVC后台控制层获取参数，它有三个常用参数：defaultValue ,required ,value ；defaultValue 表示设置默认值，required 通过布尔类型设置是否是必须要传入的参数，value值表示接受的传入的参数类型。



谢谢观看