1. Spring MVC总体设计

要用Spring MVC，只需要在web.xml中配置一个DispatherServlet，再定义一个dispatcherServlet-servlet.xml配置文件，这样就可以用了

其中web.xml中配置：

<servlet>

<servlet-name>dispatcherServlet</servlet-name>

<servlet-class>org.spingframework.web.servlet.DispatcherServlet</servlet-class>

</servlet>

<servlet-mapping>

<servlet-name>dispatcherServlet</servlet-name>

<url-pattern>/\*<url-pattern>

</servlet-mapping>

dispatcherServlet-servlet.xml配置（这里是配置spring各种web相关bean的地方，基本组成标签：urlMapping，intercepter，view，具体功能bean）：

<beans>

<!--定义Mapping-->

<bean id = “urlMapping”

class=”org.springframework.web.servlet.handler.SimpleUrlHandlerMapping”>

<property name = “mappings”>

<props>

<prop key=”demo.htm”>demo</prop>

</props>

</property>

</bean>

<bean id=”interceptor”

class=”org.springframework.web.servlet.theme.ThemeChangeInterceptor”/>

<!--定义View-->

<bean id=”viewResolver”

Class=”org.springframework.web.servlet.view.InternalResourceViewResolver”>

<property name=”viewClass”>

<value>org.springframework.web.servlet.view.InternalResourceView</value>

</property>

</bean>

<!--定义control-->

<bean id=”demo” class=”xxx.mvc.Demo”>

<property name=”viewPage”>

<value>/demo.htm</value>

</property>

</bean>

<beans>

如果用基于注解可以不写urlMapping和POJO bean，不过要写

<context:component-scan base-package="" />

在xml配置了这个标签后，spring可以自动去扫描base-pack下面或者子包下面的java文件，如果扫描到有@Component @Controller@Service等这些注解的类，则把这些类注册为bean

DispatcherServlet类继承了HttpServlet，在Servlet的init方法调用时DispatcherServlet执行Spring MVC的初始化，具体如下（应该都能在dispatcherServlet-servlet.xml配置）

protected void initStrategies(ApplicationContext context) {

initMultipartResolver(context);//处理文件上传

initLocalResolver(context);//处理国际化问题

initThemeResolver(context);//定义主题，根据用户喜好设置访问页面样式

initHandlerMappings(context);//定义请求映射关系，找到映射类

initHandlerAdapters(context);//定义处理Controller的对象，找到映射类的具体方法

initHandlerExceptionResolvers(context);//处理Handler出错时的情况

initRequestToViewNameTranslator(context);//把指定ViewName替换成想要格式

initViewResolvers(context);//把View解析成页面，可设置多个策略，默认按照JSP解析

}

1. Control设计

Spring MVC的Control主要由HandlerMapping和HandlerAdapters两个组件提供，HandlerMapping负责映射用户的URL和对应的处理类，HandlerMapping并没有规定这个URL与应用的处理类如何映射，HandlerMapping的初始化工作完成的两个最重要的工作：将URL与Handler的对应关系保存在handlerMap集合中，并将所有的interceptors对象保存在adaptedInterceptors数组中，等请求时执行所有的adaptedInterceptors数组中的interceptor对象，所有的interceptor必须实现HandlerInterceptor接口，

HandlerMapping接口中只定义了根据一个URL返回一个由HandlerExecutionChain代表的处理链，可以在这个处理链中添加任意的HandlerAdapters实例来处理这个URL对应的请求。

当Spring MVC将某个URL对应到某个Handler时，会在DispatcherServlet的handlerAdapters集合中找HandlerAdapter，如果这个handlerAdapter对象是SimpleControllerHandlerAdapter，则会调用Controller接口的public ModelAndView handle(HttpServletRequest request, HttpServeltResponse response, Object handler)方法，即处理具体逻辑的方法（如果用注解则不用这么麻烦，可以任意取名）

整个Spring MVC的调用是从DispatcherServlet的doService()开始的，在doService中将会把ApplicationContext，localeResolver，themeResolver等对象添加到request中以便后面使用，接着调用doDispatch方法来处理用户请求。

（下面的Handler可以理解为处理具体逻辑的Controller函数）

Control逻辑的关键：DispatcherServlet的handlerMappings集合中根据请求的URL匹配每个HandlerMapping对象中的某个Handler，匹配成功后将会返回这个Handler的处理链HandlerExecutionChain对象，而这个HandlerExecutionChain对象中将会包含用户自定义的多个HandlerInterceptor对象。在HandlerInterceptor接口中定义的3个方法中，preHandle和postHandle分别在Handler执行前和执行后执行，afterCompletion在View渲染完成，DispatcherServlet返回之前执行，当preHandle返回false，则直接执行afterCompletion后返回。接下来执行Handler对象的方法，Handler对象方法返回ModelAndView对象，则执行View渲染。

1. Model设计

ModelAndView对象是连接业务逻辑和View展现层的桥梁，持有一个ModelMap对象和一个View对象或View名称，ModelMap对象是渲染需要的变量，Handler可以直接返回，ModelMap只是在需要存储key value变量时才会用到

MVC中M就是model，连接业务逻辑和View的桥梁。具体到Spring MVC里就是ModelMap

1. View设计

Spring MVC里的RequestToViewNameTranslator和ViewResolver。主要是ViewResolver，用于指定解析View的class，比如指定为

<bean class=*"org.springframework.web.servlet.view.velocity.VelocityLayoutViewResolver"* id=*"viewResolver"*>