链滴

SpringMVC 回顾笔记 (八)---- 视图和视图解析器

作者：pride

原文链接：https://ld246.com/article/1542509473657

来源网站：链滴

● 总述

  请求处理方法执行完成后，最终返回一个 ModelAndView 对象。对于那些回String，View 或 ModeMap 等类型的处理方法，Spring MVC 也会在内部将它们装配成一个ModeAndView 对象，它包含了逻辑名和模型对象的视图Spring MVC 借助视图解析器（ViewResolver）到最终的视图对象（View），最终的视图可以是 JSP ，也可能是Excel、JFreeChart 等各种表现形式视图。

● 视图:

视图的作用是渲染模型数据，将模型里的数据以某种形式呈现给客户。

视图对象由视图解析器负责实例化。由于视图是无状态的，所以他们不会有线程安全的问题常用的视实现类:

1.URL视图资源:

1).IntenalResourceView 将JSP 或其它资源封装成一个视图，是InternalResourceViewResolver 默使用的视图实现类

2).JstlView 如果JSP文件中使用了 JSTL 国际化标签的功能，则需要使用该视图类

2.文档视图:

AbstractExcelView Excel 文档视图的抽象类。该视图类基于POI构造Excel文档

● 视图解析器

一.具体描述

 SpringMVC 为逻辑视图名的解析提供了不同的策略，可以在 Spring WEB 上下文中配置一或多种解析策略，并指定他们之间的先后顺序。

每一种映射策略对应一个具体的视图解析器实现类。

视图解析器的作用比较单一:将逻辑视图解析为一个具体的视图对象。

所有的视图解析器都必须实现 ViewResolver 接口;

实现类:

1.解析为Bean的名字

1).BeanNameViewResolver 将逻辑视图名解析为一个 Bean，Bean 的 id 等于逻辑视图名

2).解析为URL文件

InternalResourceViewResolver 将视图名解析为一个URL文件，一般使用该解析器将视图名映射为个保存在WEB-INF目录下的程序文件（如JSP）

程序员可以选择一种视图解析器或混用多种视图解析器

• 每个视图解析器都实现了 Ordered 接口并开放出一个 order 属性，可以通过 order 属性指定解析的优先顺序，order 越小优先级越高。

• SpringMVC 会按视图解析器顺序的优先顺序对逻辑视图名进行解析，直到解析成功并返回视图对，否则将抛出 ServletException 异常

```
    <!-- 配置视图解析器：如何把 handler 方法返回值解析为实际的物理视图
      方法返回值会通过视图解析器解析为实际的物理视图，对于InternalResourceViewResolver 视解析器，
      会做出如下的解析: 通过
      prefix + returnVal + suffix 这样的方式得到实际的物理视图，然后做转发操作
    -->
    <bean class="org.springframework.web.servlet.view.InternalResourceViewResolver">
      <property name="prefix" value="/WEB-INF/views/"></property>
```

```
    <property name="suffix" value=".jsp"></property>
    <property name="order" value="2"></property>
  </bean>
```

二.JstlView

 若项目中使用了 JSTL，则 SpringMVC 会自动把视图由InternalResourceView 转为 JstlVie
若使用 JSTL 的 fmt 标签则需要在 SpringMVC 的配置文件中配置国际化资源文件

在SpringMVC配置:

```
    <!-- 配置国际化文件 -->
    <bean id="messageSource"
    class="org.springframework.context.support.ResourceBundleMessageSource">
      <property name="basename" value="i18n"></property>
    </bean>
```

在目标文件配置:

```
  <!--配置标签-->
   <%@ taglib prefix="fmt" uri="http://java.sun.com/jsp/jstl/fmt" %>

  <!--内容-->
  <fmt:message key="i18n.username"></fmt:message>
  <br><br>
  <fmt:message key="i18n.password"></fmt:message>
  <br><br>
```

● mvc:viewcontroller 标签

若希望直接响应通过 SpringMVC 渲染的页面，不经过 handler 方法 ,可以使用 mvc:viewcontroller 签实现

eg:

```
<mvc:view-controller path ="/success" view-name ="success"/>
   这里的path="success" 相当于 @RequestMapping("/success")
   这里的view-name="success" 相当于 returen "success"
```

 当然只配置这句话的话 会使通过控制器来映射的url无法访问到页面了所以在实际开发中通常需要配置<mvc:annotation-driven></mvc:annotation-driven>

 原因：如果没有 <mvc:annotation-driven/>，那么所有的@Controller注解可能就有解析，所有当有请求时候都没有匹配的处理请求类，就都去<mvc:default-servlet-handler/>即default servlet处理了。

 <mvc:annotation-driven /> 会自动注册DefaultAnnotationHandlerMapping与AnnotationMethodHandlerAdapter 两个bean,是spring MVC为@Controllers分发请求所必须的。

● 关于重定向

 如果返回的字符串中带 forward: 或 redirect: 前缀时，SpringMVC 会对他们进行特殊处理：将 forward: 和redirect: 当成指示符，其后的字符串作为 URL 来处理

– redirect:success.jsp：会完成一个到 success.jsp 的重定向的操作

– forward:success.jsp：会完成一个到 success.jsp 的转发操作