

东软睿道内部公开

文件编号: D000-

SpringMVC框架技术

版本: 3.6.0

第2章 处理器映射器和适配器

东软睿道教育信息技术有限公司
(版权所有, 翻版必究)

Copyright © Neusoft Educational Information Technology Co., Ltd
All Rights Reserved



本章教学目标

- ✓ 了解非注解的处理器映射器；
- ✓ 了解注解的处理器映射器和适配器；
- ✓ 理解非注解的处理器适配器；
- ✓ 掌握注解开发步骤；

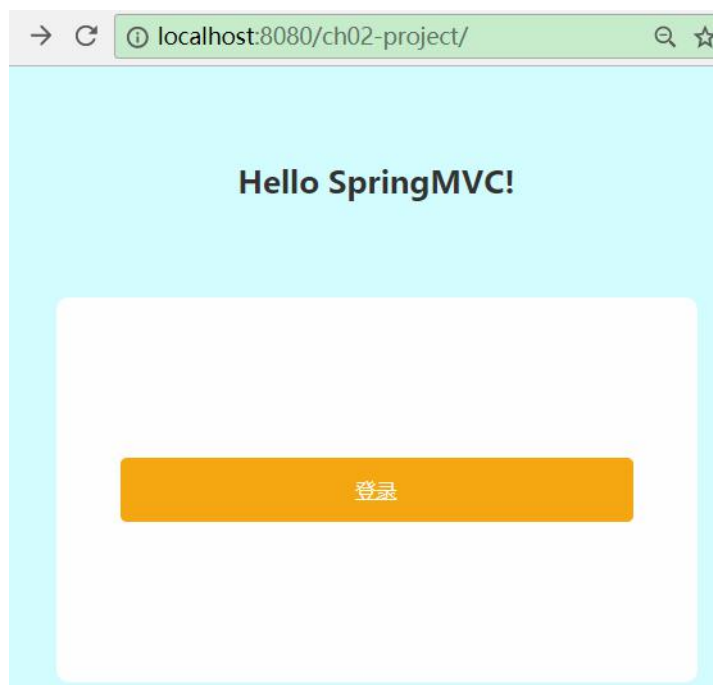


本章教学内容

节	知识点	掌握程度	难易程度	教学形式	对应在线微课
配置文件开发	非注解的处理器映射器	了解	普通	线下	非注解的处理器映射器
	非注解的处理器适配器	掌握	普通	线下	非注解的处理器适配器
注解开发	注解的处理器映射器	了解	普通	线下	注解的处理器映射器
	注解的处理器适配器	了解	普通	线下	注解的处理器适配器
	开发步骤	掌握	普通	线下	开发步骤
本章实战项目任务实现	实战项目任务实现	掌握	普通	线下	实战项目任务实现

本章实战项目任务

- ❖ 通过本章学习，使用注解开发方法，完成《跨境电商系统》登录页面内容的显示。
- ❖ 项目运行效果如下



CONTENTS

目录

01

配置文件开发

02

注解开发

03

本章实战项目任务实现

非注解的处理器映射器

- ❖ HandlerMapping处理器映射器：
 - ▶ HandlerMapping 负责根据request请求找到对应的Handler处理器及Interceptor拦截器，将它们封装在HandlerExecutionChain 对象中给前端控制器返回。
- ❖ 在classpath下的springmvc.xml中配置处理器映射器
 - ▶ org.springframework.web.servlet.handler.BeanNameUrlHandlerMapping
 - ▶ org.springframework.web.servlet.handler.SimpleUrlHandlerMapping
- ❖ 多个映射器可以并存，前端控制器判断url能让哪些映射器映射，就让正确的映射器处理。
- ❖ 映射器、适配器、视图解析器可以不声明，Spring自动分配。
- ❖ 如果声明的话，需要将使用到的全部声明出来。


非注解的处理器映射器

- ❖ BeanNameUrlHandlerMapping处理器映射器，根据请求的url与spring容器中定义的bean的name进行匹配，从而从spring容器中找到bean实例。

```
<bean class="org.springframework.web.servlet.handler.BeanNameUrlHandlerMapping"/>
```

```
<bean name="/hello.action" class="com.neuedu.controller.HelloController"/>
```

请求的url



- ❖ 全部代码参见：ch02-springmvc01工程

非注解的处理器映射器

- ❖ `simpleUrlHandlerMapping` 是 `BeanNameUrlHandlerMapping` 的增强版本，它可以将 url 和处理器 bean 的 id 进行统一映射配置。
- ❖ 全部代码参见：ch02-springmvc01工程

```
<bean id="ItemsController1" name="/queryItems.action" class="com.neuedu.controller.ItemsController1"/>

<bean class="org.springframework.web.servlet.handler.SimpleUrlHandlerMapping">
  <property name="mappings">
    <props>
      <prop key="/queryItems1.action">ItemsController1</prop>
      <prop key="/queryItems2.action">ItemsController1</prop>
    </props>
  </property>
</bean>
```

请求的url

非注解的处理器适配器

- ❖ HandlerAdapter 处理器适配器：
 - ▶ HandlerAdapter 会根据适配器接口对后端控制器进行包装（适配），包装后即可对处理器进行执行，通过扩展处理器适配器可以执行多种类型的处理器，这里使用了适配器设计模式。
- ❖ 在 classpath 下的 springmvc.xml 中配置处理器适配器
 - ▶ org.springframework.web.servlet.mvc.SimpleControllerHandlerAdapter
 - ▶ org.springframework.web.servlet.mvc.HttpRequestHandlerAdapter
- ❖ 多个适配器可以并存，前端控制器判断 url 能让哪些适配器适配，就让正确的适配器处理。

非注解的处理器适配器

- ❖ SimpleControllerHandlerAdapter 简单控制器处理器适配器，所有实现了 `org.springframework.web.servlet.mvc.Controller` 接口的 Bean 通过此适配器进行适配、执行。

```
public class ItemsController1 implements Controller {  
  
    @Override  
    public ModelAndView handleRequest(HttpServletRequest request,  
        HttpServletResponse response) throws Exception {
```

- ❖ 配置如下：

```
<bean class="org.springframework.web.servlet.mvc.SimpleControllerHandlerAdapter"/>
```

- ❖ 全部代码参见：ch02-springmvc02工程

非注解的处理器适配器

- ❖ `HttpRequestHandlerAdapter`，http请求处理器适配器，所有实现了 `org.springframework.web.HttpRequestHandler` 接口的Bean通过此适配器进行适配、执行。

```
public class ItemsController2 implements HttpRequestHandler {  
    @Override  
    public void handleRequest(HttpServletRequest request,  
        HttpServletResponse response) throws ServletException, IOException {
```

- ❖ 配置如下：

```
<bean class="org.springframework.web.servlet.mvc.HttpRequestHandlerAdapter"/>
```

- ❖ 全部代码参见：ch02-springmvc02工程

非注解的处理器适配器

- ❖ `HttpRequestHandlerAdapter` 的 `handleRequest` 方法没有返回值 `ModelAndView`，可以通过 `request` 或者 `response` 对象的请求转发和重定向进行页面跳转，也可通过 `response` 对象修改定义响应内容，比如返回 json 数据：

```
response.setCharacterEncoding("utf-8");  
response.setContentType("application/json;charset=utf-8");  
response.getWriter().write("json串");
```

- ❖ 全部代码参见：ch02-springmvc02工程

课堂练习(15分钟)

❖ 1、理解教学案例代码

❖ 2、填空

Springmvc框架中_____负责根据request请求找到对应的Handler处理器及Interceptor拦截器，将它们封装在HandlerExecutionChain 对象中给前端控制器返回。

❖ 3、填空

在classpath下的_____中配置处理器映射器和适配器及视图解析器。

❖ 4、如下描述，正确的是（ ）（多选）？

A. 映射器、适配器、视图解析器可以不声明，Spring自动分配

B. 如果声明映射器的话，需要将使用到的全部声明出来。

C. 如果不配置映射器，默认使用的映射器是

`org.springframework.web.servlet.handler.BeanNameUrlHandlerMapping`

D. 如果不配置适配器，默认使用的适配器是

`org.springframework.web.servlet.mvc.SimpleControllerHandlerAdapter`

CONTENTS

目录

01

配置文件开发

02

注解开发

03

本章实战项目任务实现

注解的处理器映射器

❖ 注解的处理器映射器

▶ 在spring3.1之前使用

`org.springframework.web.servlet.mvc.annotation.DefaultAnnotationHandlerMapping`注解映射器。

▶ 在spring3.1之后使用

`org.springframework.web.servlet.mvc.method.annotation.RequestMappingHandlerMapping`注解映射器。



注解的处理器映射器

❖ RequestMappingHandlerMapping

- ▶ 注解式处理器映射器，对类中标记@RequestMapping的方法进行映射，根据RequestMapping定义的url匹配RequestMapping标记的方法，匹配成功返回HandlerMethod对象给前端控制器，HandlerMethod对象中封装url对应的方法Method。
- ▶ 从spring3.1版本开始，废除了DefaultAnnotationHandlerMapping的使用，推荐使用RequestMappingHandlerMapping完成注解式处理器映射。

❖ 配置如下：

```
<bean  
class="org.springframework.web.servlet.mvc.method.annotation.RequestMappingHandlerMapping"/>
```


注解的处理器适配器

❖ 注解的处理器适配器

▶ 在spring3.1之前使用

`org.springframework.web.servlet.mvc.annotation.AnnotationMethod
HandlerAdapter`注解适配器。

▶ 在spring3.1之后使用

`org.springframework.web.servlet.mvc.method.annotation.RequestMa
ppingHandlerAdapter`注解适配器。



注解的处理器适配器

❖ RequestMappingHandlerAdapter

- ▶ 注解式处理器适配器，对标记@ResquestMapping的方法进行适配。
- ▶ 从spring3.1版本开始，废除了AnnotationMethodHandlerAdapter的使用，推荐使用RequestMappingHandlerAdapter完成注解式处理器适配。

❖ 配置如下：

```
<bean  
class="org.springframework.web.servlet.mvc.method.annotation.RequestMappingHandlerAdapter"/>
```

注解开发

- ❖ 示例：使用注解开发方式实现商品查询
- ❖ 开发步骤
 - ▶ 1. 创建maven web工程
 - ▶ 2. pom.xml导入jar 包
 - ▶ 3. 在 web.xml 中配置 DispatcherServlet
 - ▶ 4. 配置注解映射器和适配器
 - ▶ 5. 编写注解处理器
 - ▶ 6. 在spring容器中加载处理器
 - ▶ 7. 编写视图
 - ▶ 8. 配置视图解析器
 - ▶ 9. 部署调试
- ❖ 全部代码参见：ch02-springmvc03工程

开发步骤

- ❖ 1-3步同配置文件开发方法
- ❖ 4. 配置注解映射器和适配器
 - ▶ 如下写法，二选一

```
<!-- 注解 -->  
  
<bean  
class="org.springframework.web.servlet.mvc.method.annotation.RequestMappingHandlerMapping"/>  
<bean  
class="org.springframework.web.servlet.mvc.method.annotation.RequestMappingHandlerAdapter"/>
```

```
<mvc:annotation-driven></mvc:annotation-driven>
```

- ▶ 使用 `mvc:annotation-driven`代替上边注解映射器和注解适配器配置
- ▶ `mvc:annotation-driven`默认加载很多的参数绑定方法
- ▶ 使用`mvc:annotation-driven`，不用配置上边的
`RequestMappingHandlerMapping`和`RequestMappingHandlerAdapter`
- ▶ 实际开发时使用`mvc:annotation-driven`

开发步骤

❖ 5. 开发注解Handler

- ▶ 使用注解@Controller修饰处理器
- ▶ 使用注解@RequestMapping修饰处理器的方法，参数使用请求 URL ，映射请求

```
@Controller
public class ItemsController3{
    @RequestMapping("/queryItems")
    public ModelAndView queryItems() throws Exception{
```

开发步骤

❖ 6. 在spring容器中加载Handler

```
<!-- 对于注解的Handler可以单个配置实际开发中建议使用组件扫描 -->  
<context:component-scan base-package="com.neuedu.controller"></context:component-scan>  
  
<!-- 如果使用组件扫描，必须省略下面代码 -->  
<!-- bean class="com.neuedu.controller.ItemsController3"/-->
```

开发步骤

❖ 7. 编写视图



The screenshot shows an IDE with the Package Explorer on the left and the editor on the right. The Package Explorer shows the project structure for 'ch02-springmvc03', with 'itemsList.jsp' highlighted under 'src/main/webapp/WEB-INF/jsp/items'. The editor displays the content of 'itemsList.jsp', which is a JSP page for displaying a list of items. The code includes a table structure with columns for item name, price, creation date, description, and an edit link. A red box highlights the JSTL loop tag on line 29.

```
20 商品列表:
21 <table width="100%" border=1>
22 <tr>
23     <td>商品名称</td>
24     <td>商品价格</td>
25     <td>生产日期</td>
26     <td>商品描述</td>
27     <td>操作</td>
28 </tr>
29 <c:forEach items="${itemsList}" var="item">
30 <tr>
31     <td>${item.name }</td>
32     <td>${item.price }</td>
33     <td><fmt:formatDate value="${item.createtime}" pattern="yyyy-MM-dd HH:mm:ss"/></td>
34     <td>${item.detail }</td>
35
36     <td><a href="${pageContext.request.contextPath }/item/editItem.action?id=${item.id}">修改</a>
37
38 </tr>
39 </c:forEach>
40
41 </table>
```


开发步骤

❖ 8. 配置视图解析器

```
<!-- 视图解析器，添加前后缀 -->
<bean class="org.springframework.web.servlet.view.InternalResourceViewResolver">
  <property name="prefix" value="/WEB-INF/jsp/" />
  <property name="suffix" value=".jsp"/>
</bean>
```

```
//指定视图
// modelAndView.setViewName("/WEB-INF/jsp/items/itemsList.jsp");

//配置视图解析器的前后缀后，写法修改如下：
modelAndView.setViewName("items/itemsList");

return modelAndView;
```


开发步骤

❖ 9. 部署测试

← → ↻ localhost:8080/ch02-springmvc03/queryItems.action 🔍 ☆ ⋮

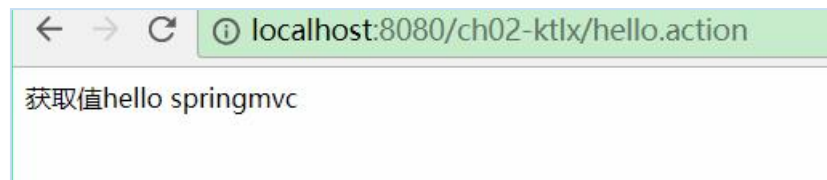
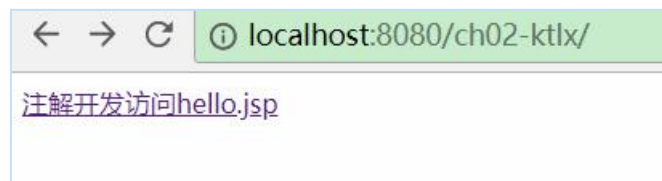
查询条件：

商品列表：

商品名称	商品价格	生产日期	商品描述	操作
联想笔记本	6000.0		ThinkPad T430 联想笔记本电脑！	修改
苹果手机	5000.0		iphone6苹果手机！	修改

课堂练习（10分钟）

- ❖ 使用注解开发方式编写HelloController程序
 - ▶ 运行效果如下图：



CONTENTS

目录

01

配置文件开发

02

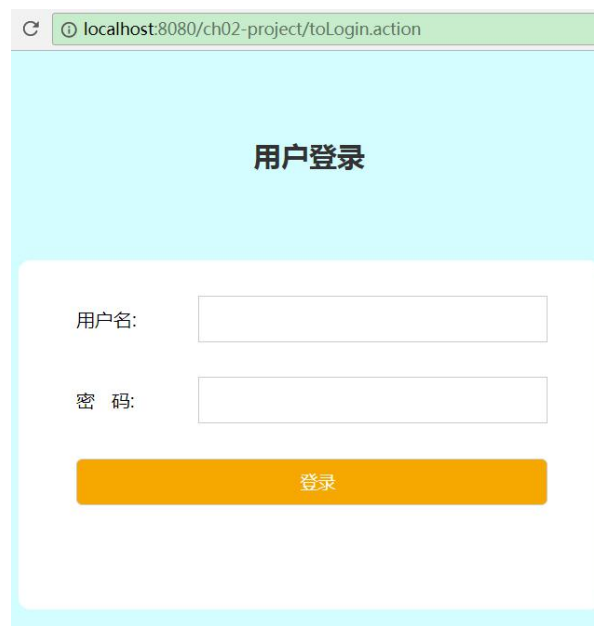
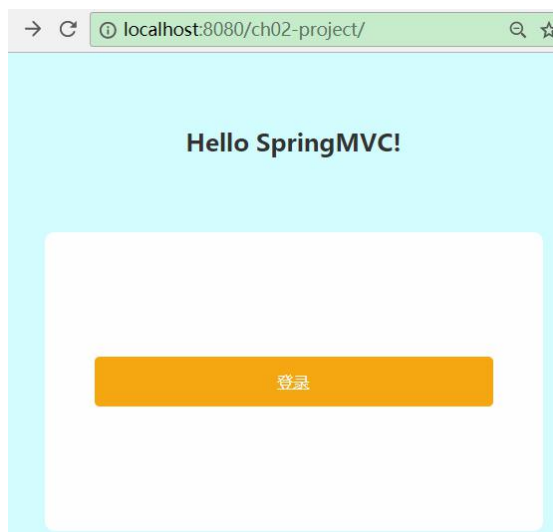
注解开发

03

本章实战项目任务实现

本章实战项目任务实现

- ❖ 通过本章内容的学习，使用注解开发方法，完成《跨境电商系统》登录页面内容的显示。
 - ▶ 创建工程代码ch02-project，搭建开发环境
 - ▶ 编写注解处理器
 - ▶ 在springmvc.xml中配置注解映射器和适配器、在spring容器中加载处理器
 - ▶ 工程部署和运行<http://localhost:8080/ch02-project/>
- ❖ 运行效果如图：



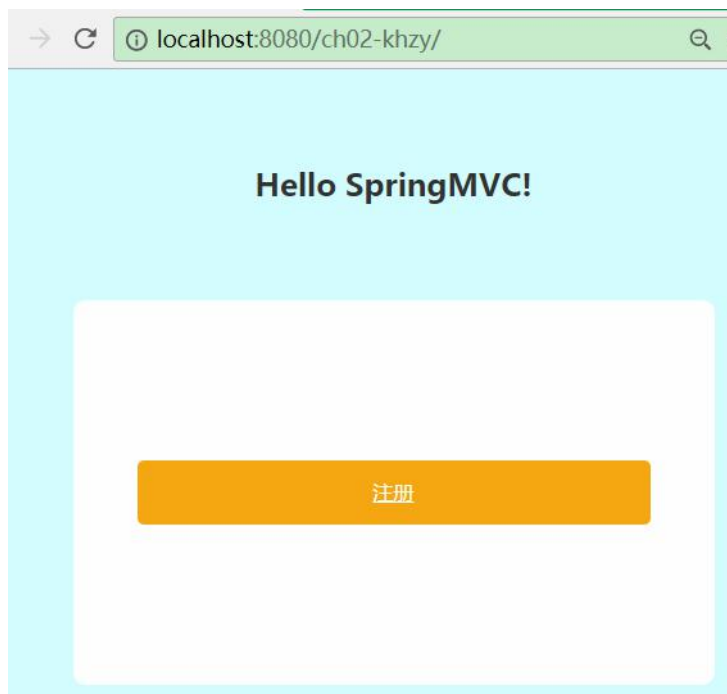
本章重点总结

- ❖ 了解非注解的处理器映射器；
- ❖ 了解注解的处理器映射器和适配器；
- ❖ 理解非注解的处理器适配器；
- ❖ 掌握注解开发步骤；



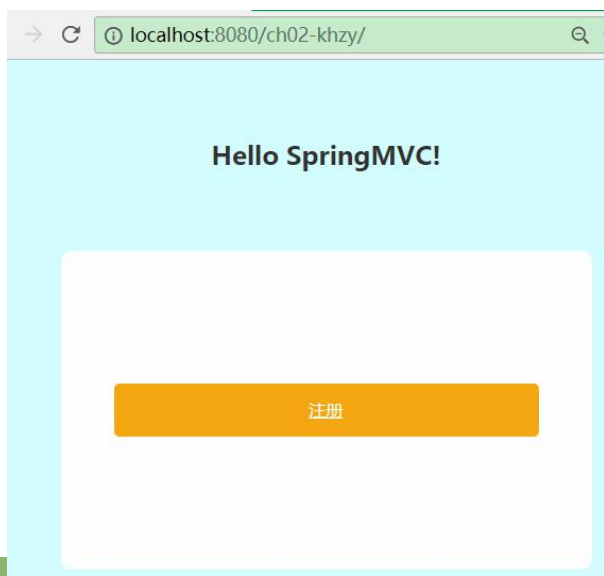
课后作业【必做任务】

- ❖ 1、使用注解开发方法，完成《跨境电商系统》注册页面内容的显示。
 - ▶ 创建工程代码ch02-khzy，搭建开发环境
 - ▶ 编写注解处理器
 - ▶ 在springmvc.xml中配置注解映射器和适配器、在spring容器中加载处理器
 - ▶ 工程部署和运行<http://localhost:8080/ch02-khzy/>
- ❖ 运行效果如图：



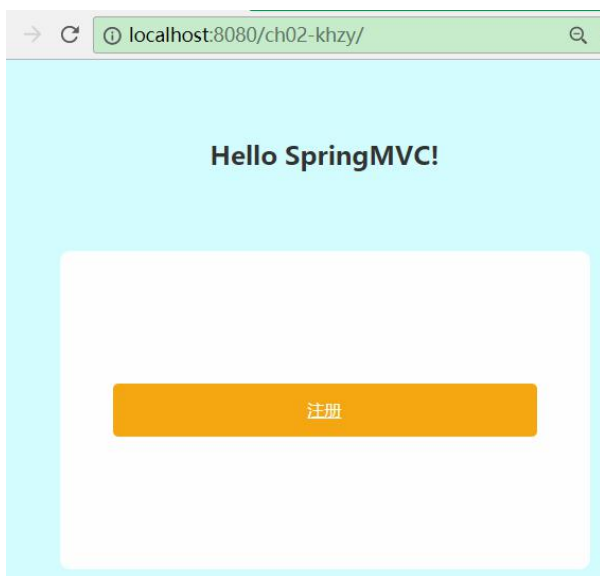
课后作业【必做任务】

- ❖ 2、使用配置文件开发方法，完成《跨境电商系统》注册页面内容的显示。
 - ▶ 搭建springmvc开发的环境
 - ▶ 编写工程代码ch02-khzy
 - ▶ 配置非注解的处理器映射器和适配器（ BeanNameUrlHandlerMapping处理器映射器和http请求处理器适配器）
 - ▶ 编写处理器
 - ▶ 工程部署和运行， <http://localhost:8080/ch02-khzy/>



课后作业【选做任务】

- ❖ 使用配置文件开发方法，完成《跨境电商系统》注册页面内容的显示。
 - ▶ 搭建springmvc开发的环境
 - ▶ 编写工程代码ch02-khzy
 - ▶ 配置非注解的处理器映射器和适配器（simpleUrlHandlerMapping处理器映射器和简单的控制器处理器适配器）
 - ▶ 编写处理器
 - ▶ 工程部署和运行，<http://localhost:8080/ch02-khzy/>



课后作业【线上任务】

❖ 线上任务

- ▶ 安排学员线上学习任务（安排学员到睿道实训平台进行复习和预习的任务，主要是进行微课的学习）

