

东软睿道内部公开

文件编号: D000-

SpringMVC框架技术

版本: 3.6.0

第3章 映射请求、方法返回值 和参数绑定

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

- ✓ 了解参数绑定的默认支持的类型；
- ✓ 理解参数绑定过程；
- ✓ 掌握@RequestMapping注解映射请求；
- ✓ 掌握方法的返回值类型；
- ✓ 掌握参数绑定，包括简单类型、pojo类型；
- ✓ 掌握自定义参数绑定；
- ✓ 掌握集合类型参数绑定

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
映射请求	@RequestMapping修饰类	掌握		线下	RequestMapping修饰类
	@RequestMapping修饰方法	掌握		线下	RequestMapping修饰方法
方法返回值	返回ModelAndView	掌握		线下	返回ModelAndView
	返回String	掌握	难	线下	返回String
	返回void	掌握		线下	返回void
参数绑定	参数绑定过程	理解		线下	参数绑定过程
	默认支持的类型	了解		线下	默认支持的类型
	简单类型绑定	掌握		线下	简单类型绑定
	简单pojo绑定	掌握		线下	简单pojo绑定
	包装pojo绑定	掌握	难	线下	包装pojo绑定
	自定义参数绑定	掌握	难	线下	自定义参数绑定
	数组绑定	掌握		线下	数组绑定
	集合类型绑定	掌握	难	线下	集合类型绑定
本章实战项目任务实现	实战项目任务实现	掌握		线下	实战项目任务实现

本章实战项目任务

- ❖ 通过本章内容的学习，完成《跨境电商系统》用户注册页面的显示，以及用户注册模块的业务处理
- ❖ 运行效果如图：



CONTENTS

目录

01

映射请求

02

方法返回值

03

参数绑定

04

本章实战项目任务实现

@RequestMapping映射请求

- ❖ 案例：使用springmvc完成商品信息修改
 - 首先查询出商品
 - 点击商品后面的修改链接，进入到商品修改页面
 - 点击修改页面的提交按钮，显示操作成功

← → ↻ ① localhost:8080/ch03-springmvc01/items/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
冰箱	5000.0	2018-02-28 09:04:19	冰箱质量好价格低	修改
洗衣机	2000.0	2018-02-28 09:04:19	洗衣机质量好价格低	修改
笔记本电脑	8000.0	2018-02-28 09:04:19	笔记本电脑质量好价格低	修改
空调	2000.0	2018-02-28 09:04:19	空调质量好价格低	修改
彩电	6000.0	2018-02-28 09:04:19	彩电质量好价格低	修改

← → ↻ ① localhost:8080/ch03-springmvc01/items/editItems.action?id=1 🔍 ☆ ⋮

修改商品信息：

商品名称	<input type="text" value="冰箱"/>
商品价格	<input type="text" value="5000.0"/>
商品生产日期	<input type="text" value="2018-02-28 09:46:05"/>
商品简介	<input type="text" value="冰箱质量好价格低"/>
<input type="button" value="提交"/>	

← → ↻ ① localhost:8080/ch03-springmvc01/items/editItemsSubmit.action

操作成功！

- 全部代码参见：ch03-springmvc01工程

@RequestMapping映射请求

- ❖ Spring MVC 使用 @RequestMapping 注解为控制器指定可以处理哪些 URL 请求
- ❖ 在控制器的类定义及方法定义处都可标注
 - ▶ 类定义处：提供初步的请求映射信息。相对于 WEB 应用的根目录
 - ▶ 方法处：提供进一步的细分映射信息。相对于类定义处的 URL。若类定义处未标注 @RequestMapping，则方法处标记的 URL 相对于WEB 应用的根目录
- ❖ DispatcherServlet 截获请求后，就通过控制器上@RequestMapping 提供的映射信息确定请求所对应的处理方法。

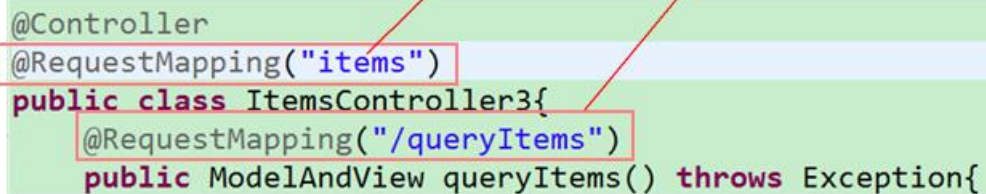
@RequestMapping修饰类

❖ 注解@RequestMapping修饰类

- ▶ 提供初步的请求映射信息，相对于 WEB 应用的根目录

```
<a href="${pageContext.request.contextPath }/items/queryItems.action">查询测试</a><br><br>
```

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



@RequestMapping修饰方法

❖ 注解@RequestMapping修饰方法

- ▶ 提供进一步的细分映射信息，相对于类定义处的 URL。
- ▶ @RequestMapping 除了可以使用请求 URL 映射请求外，还可以使用请求方法、请求参数及请求头映射请求
- ▶ @RequestMapping 的 value、method、params 及 headers 分别表示请求 URL、请求方法、请求参数及请求头的映射条件，他们之间是与的关系，联合使用多个条件可让请求映射更加精确化。

```
//限制http请求方法，可以post和get
@RequestMapping(value="/editItems",method={RequestMethod.GET,RequestMethod.POST} )

public ModelAndView editItems() throws Exception {
    ItemsCustom itemsCustom = itemsService.findItemsById(1);
    ModelAndView modelAndView = new ModelAndView();
    modelAndView.addObject("itemsCustom", itemsCustom);
    modelAndView.setViewName("/items/editItems");
    return modelAndView;
}
```

- ▶ 全部代码参见：ch03-springmvc01工程

@RequestMapping修饰方法

❖ params 和 headers支持简单的表达式:

- ▶ param1: 表示请求必须包含名为 param1 的请求参数
- ▶ !param1: 表示请求不能包含名为 param1 的请求参数
- ▶ param1 != value1: 表示请求包含名为 param1 的请求参数, 但其值不能为 value1
- ▶ { "param1=value1", "param2" }: 请求必须包含名为 param1 和 param2 的两个请求参数, 且 param1 参数的值必须为 value1

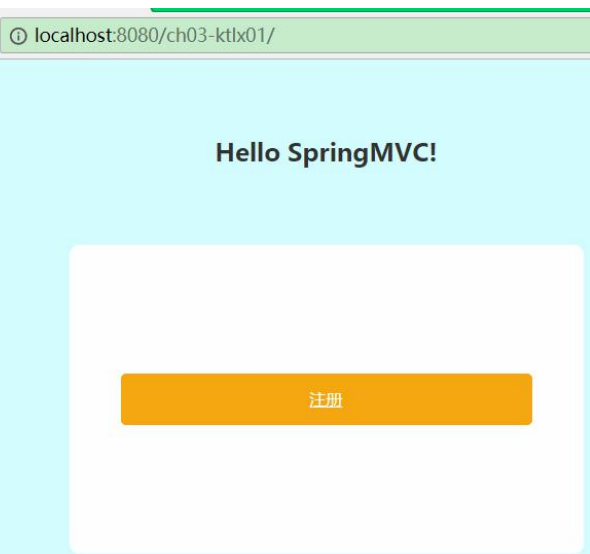
```
//@RequestMapping(value="/editItemsSubmit",params={"name","price!=5000.0"})//运行时出现异常
@RequestMapping(value="/editItemsSubmit",params={"name","price"})
public ModelAndView editItemsSubmit(int id,String name,float price) throws Exception {
    System.out.println("name="+name);
    System.out.println("price="+price);
    ModelAndView modelAndView = new ModelAndView();

    modelAndView.setViewName("success");
    return modelAndView;
}
```

- ▶ 全部代码参见: ch03-springmvc01工程

课堂练习（15分钟）

- ❖ 完成《跨境电商系统》注册页面register.jsp的显示、以及用户注册信息的提交（主要编写请求的映射代码，不用进行具体业务处理）
 - ▶ 使用@RequestMapping修饰controller类及方法
 - ▶ 使用@RequestMapping 的 value、method、params，限定请求方法和请求参数



CONTENTS

目录

01

映射请求

02

方法返回值

03

参数绑定

04

本章实战项目任务实现

controller方法的返回值

- ❖ controller方法的返回值
 - ▶ 返回ModelAndView
 - ▶ 返回字符串
 - ▶ 返回void

返回ModelAndView

- ❖ controller方法中定义ModelAndView对象并返回，对象中可添加model数据、指定view。

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

    //调用service查找 数据库，查询商品列表
    List<ItemsCustom> itemList = new ArrayList<ItemsCustom>();
    itemList = itemsService.findItemList(null);

    //返回ModelAndView
    ModelAndView modelAndView = new ModelAndView();
    //相当于request的setAttribute，在jsp页面中通过itemsList取数据
    modelAndView.addObject("itemsList", itemList);

    //指定视图
    modelAndView.setViewName("items/itemsList");

    return modelAndView;
}
```

- ▶ 全部代码参见：ch03-springmvc02工程

返回字符串

- ❖ 返回字符串时，有三种含义
 - ▶ 表示返回逻辑视图名
 - ▶ `redirect`重定向
 - ▶ `forward`页面转发
- ❖ 涉及到数据的传递时，处理器形参中添加如下类型的参数，处理适配器会默认识别并进行赋值
 - ▶ `HttpServletRequest`
 - ★ 通过`request`对象获取请求信息
 - ▶ `HttpSession`
 - ★ 通过`session`对象得到`session`中存放的对象
 - ▶ `Model/ModelMap`
 - ★ `ModelMap`是`Model`接口的实现类，将`model`数据填充到`request`域，向页面传递数据

返回字符串

❖ 返回逻辑视图名

- ▶ 真正视图(jsp路径)=前缀+逻辑视图名+后缀

```
@RequestMapping(value="/editItems",method={RequestMethod.GET,RequestMethod.POST} )  
public String editItems(HttpServletRequest request) throws Exception {  
    ItemsCustom itemsCustom = itemsService.findItemsById(1);  
    //将数据保存到request中, 在jsp页面中通过itemsCustom取数据  
    request.setAttribute("itemsCustom", itemsCustom);  
    return "/items/editItems";  
}
```

逻辑视图名

- ▶ 上面的代码也可以改为如下写法:

```
@RequestMapping(value="/editItems",method={RequestMethod.GET,RequestMethod.POST} )  
public String editItems(Model model) throws Exception {  
    ItemsCustom itemsCustom = itemsService.findItemsById(1);  
    //将数据保存到request中, 在jsp页面中通过itemsCustom取数据  
    model.addAttribute("itemsCustom", itemsCustom);  
    return "/items/editItems";  
}
```

- ▶ 全部代码参见: ch03-springmvc02工程

返回字符串

❖ redirect重定向

- ▶ redirect重定向特点：浏览器地址栏中的url会变化。修改提交的request数据无法传到重定向的地址。因为重定向后重新进行request（request无法共享）

```
//controller方法返回string,重定向
@RequestMapping("/editItemsSubmit")
public String editItemsSubmit() throws Exception {

    return "redirect:queryItems.action";
}
```

- ▶ 全部代码参见：ch03-springmvc02工程

返回字符串

❖ forward页面转发

- ▶ 通过forward进行页面转发，浏览器地址栏url不变，request可以共享。

```
@RequestMapping("/editItemsSubmit")  
//controller方法返回string,请求转发  
public String editItemsSubmit() throws Exception {  
  
    return "forward:queryItems.action";  
}
```

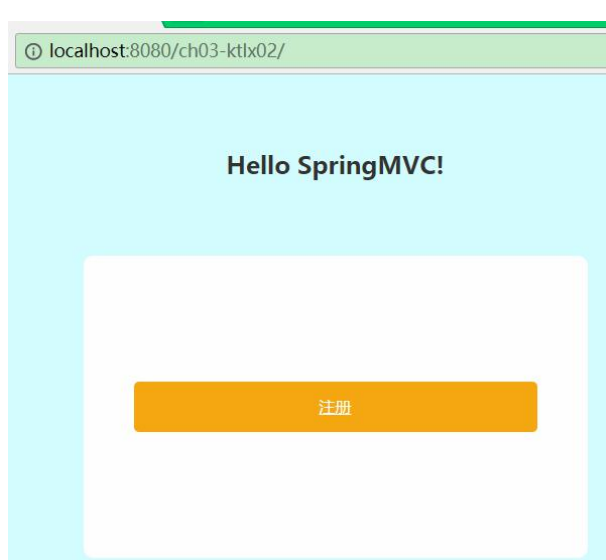
- ▶ 全部代码参见：ch03-springmvc02工程

返回void

- ❖ 当返回void时，在controller方法形参上可以定义request和response，使用request或response指定响应结果：
 - ▶ 使用request转向页面，如下：
 - ★ `request.getRequestDispatcher("页面路径").forward(request, response);`
 - ▶ 可以通过response页面重定向：
 - ★ `response.sendRedirect("url")`
 - ▶ 可以通过response指定响应结果，例如响应json数据如下：
 - ★ `response.setCharacterEncoding("utf-8");`
 - ★ `response.setContentType("application/json;charset=utf-8");`
 - ★ `response.getWriter().write("json串");`
- ▶ 第二章已经讲过，全部代码参见：ch02-springmvc02工程

课堂练习（10分钟）

- ❖ 完成《跨境电商系统》注册页面register.jsp的显示、以及用户注册信息的提交（主要编写方法及返回值代码，不用进行具体业务处理）
 - ▶ 在本章第一节练习的基础上，修改controller方法的返回值类型为String类型和void类型



CONTENTS

目录

01

映射请求

02

方法返回值

03

参数绑定

04

本章实战项目任务实现

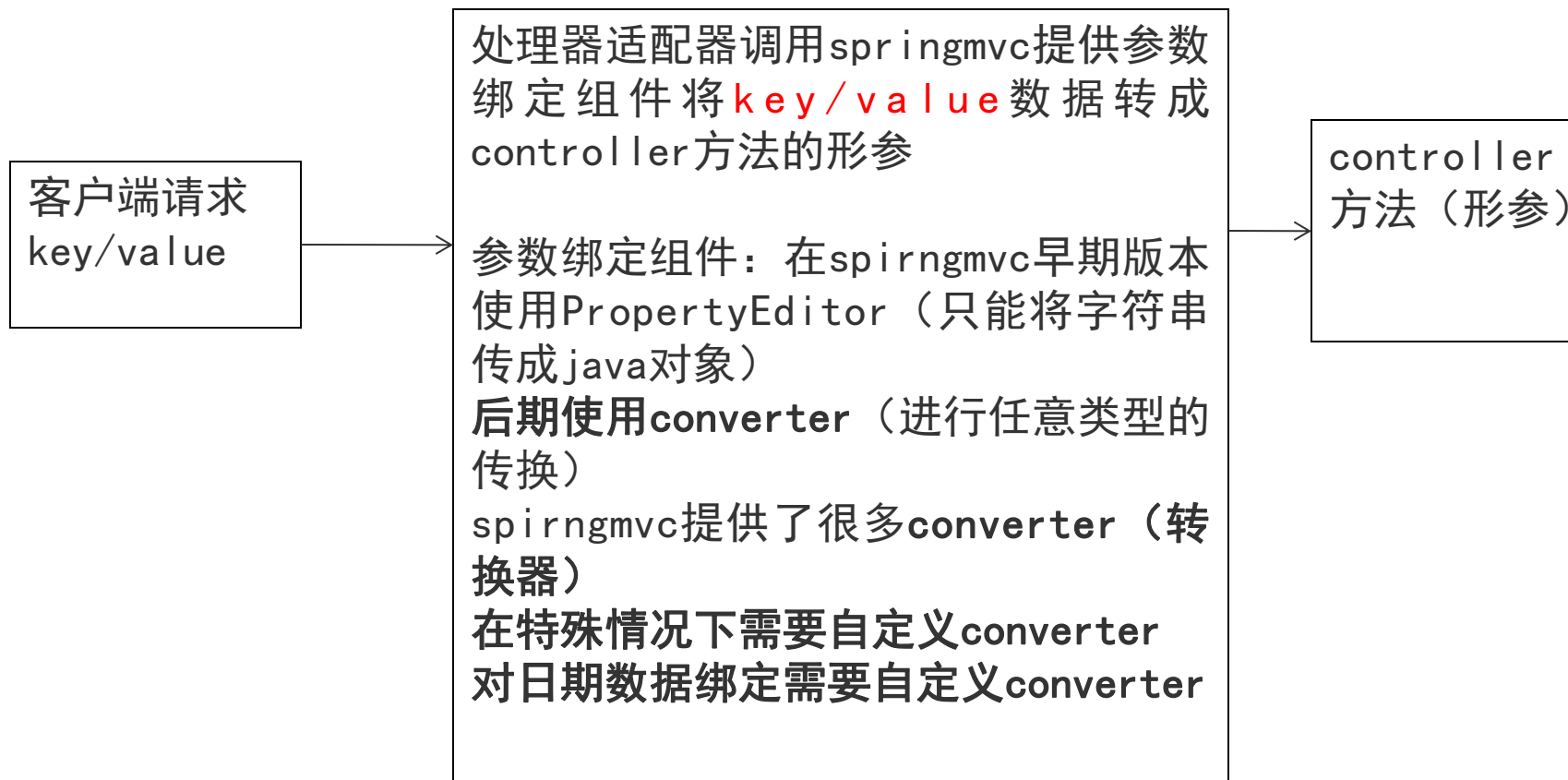
参数绑定过程

❖ springmvc参数绑定过程

- ▶ 从客户端请求key/value数据，经过参数绑定，将key/value数据绑定到controller方法的形参上。
- ▶ springmvc中，接收页面提交的数据是通过方法形参来接收，而不是在controller类定义成员变量接收。

参数绑定过程

❖ springmvc参数绑定过程



默认支持的类型

- ❖ 处理器形参中添加如下类型的参数，处理适配器会默认识别并进行赋值。
 - ▶ `HttpServletRequest`
 - ★ 通过request对象获取请求信息
 - ▶ `HttpServletResponse`
 - ★ 通过response处理响应信息
 - ▶ `HttpSession`
 - ★ 通过session对象得到session中存放的对象
 - ▶ `Model/ModelMap`
 - ★ `ModelMap`是`Model`接口的实现类，将model数据填充到request域，向页面传递数据

```
model.addAttribute("itemsCustom", itemsCustom);  
return "/items/editItems";
```

- ▶ 全部代码参见：ch03-springmvc02工程

简单类型绑定

❖ 简单类型

- ▶ 支持整型、字符串、单精度/双精度、布尔型
- ▶ 当请求的参数名称和处理器形参名称一致时会将请求参数与形参进行绑定。

```
public String editItem(Model model, Integer id) throws Exception {  
}
```

处理器方法:

```
public String editItem(Model model, Integer id, Boolean status) throws Exception
```

请求 url:

```
http://localhost:8080/springmvc/item/editItem.action?id=2&status=false
```

说明: 对于布尔类型的参数, 请求的参数值为 true 或 false。

简单类型绑定

❖ 简单类型

- ▶ 通过@RequestParam对简单类型的参数进行绑定。
- ▶ 如果不使用@RequestParam，要求request传入参数名称和controller方法的形参名称一致，方可绑定成功。
- ▶ 如果使用@RequestParam，不用限制request传入参数名称和controller方法的形参名称一致。

简单类型绑定

❖ @RequestParam的参数

- ▶ value: 参数名字, 即入参的请求参数名字, 如value=“item_id”表示请求的参数中的名字为item_id的参数的值将传入;
- ▶ required: 是否必须, 默认是true, 表示请求中一定要有相应的参数, 否则将报
- ▶ defaultValue: 默认值, 表示如果请求中没有同名参数时的默认值

```
//@RequestParam里边指定request传入参数名称和形参进行绑定。  
//通过required属性指定参数是否必须要传入,默认为true  
//通过defaultValue可以设置默认值, 如果id参数没有传入, 将默认值和形参绑定。  
public String editItems(Model model,  
    @RequestParam(value="id",defaultValue="2") Integer idd) throws Exception {
```

- ▶ 全部代码参见: [ch03-springmvc03工程](#)

课堂练习（10分钟）

- ❖ 完成《跨境电商系统》注册页面register.jsp的显示及用户注册处理
 - ▶ 使用简单类型参数绑定请求参数（用户名和密码，其他参数先不用绑定）
 - ▶ 在控制台输出用户名和密码

POJO类型绑定

❖ 简单POJO绑定

- ▶ 将pojo对象中的属性名与传递进来的参数名对应
- ▶ 如果传进来的参数名称和对象中的属性名称一致，则将参数值设置在pojo对象中

```
<table width="100%" border=1>
<tr>
  <td>商品名称</td>
  <td><input type="text" name="name" value="${itemsCustom.name }"/></td>
</tr>
<tr>
  <td>商品价格</td>
  <td><input type="text" name="price" value="${itemsCustom.price }"/></td>
</tr>
```

```
public class Items {
  private Integer id;
  private String name;
  private Float price;
  private String pic;
```

POJO类型绑定

❖ 简单POJO绑定

```
@RequestMapping("/editItemsSubmit")  
public String editItemsSubmit(Integer id,ItemsCustom itemsCustom) throws Exception {  
    itemsService.updateItems(id, itemsCustom);  
}
```

▶ 全部代码参见：ch03-springmvc03工程

课堂练习（10分钟）

- ❖ 完成《跨境电商系统》注册页面register.jsp的显示及用户注册处理
 - ▶ 使用简单POJO类型参数绑定请求参数（用户名和密码，其他参数先不用绑定）
 - ▶ 在控制台输出用户名和密码

POJO类型绑定

❖ 包装POJO绑定

- ▶ 将pojo对象作为一个包装对象的属性，controller方法中以该包装对象作为形参。

```
public class ItemsQueryVo {  
  
    //商品信息  
    private Items items;  
  
    //为了系统 可扩展性，对原始生成的po进行扩展  
    private ItemsCustom itemsCustom;
```

```
<td>商品名称: <input type="text" name="itemsCustom.name"/></td>  
<td><input type="submit" value="查询"/></td>
```

- ▶ 全部代码参见：ch03-springmvc04工程

POJO类型绑定

❖ 包装POJO绑定

```
@RequestMapping("/queryItems")  
public ModelAndView queryItems(HttpServletRequest request, ItemsQueryVo itemsQueryVo)  
    //调用service查找 数据库，查询商品列表  
    List<ItemsCustom> itemList = new ArrayList<ItemsCustom>();  
    itemList = itemsService.findItemsList(itemsQueryVo);
```

▶ 全部代码参见：ch03-springmvc04工程

课堂练习（10分钟）

❖ 理解并独立完成包装pojo的参数绑定教学案例



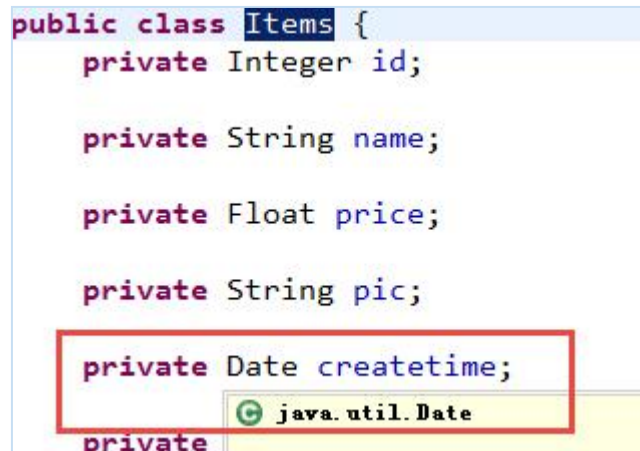
自定义参数绑定

- ❖ 根据业务需求使用自定义参数绑定。
 - ❖ 需要向处理器适配器中注入自定义的参数绑定组件。
 - ❖ 对于controller形参中pojo对象，如果属性中有日期类型，需要自定义参数绑定。
 - ▶ 将请求日期数据串转成日期类型，要转换的日期类型和pojo中日期属性的类型保持一致。
- ▶ 全部代码参见：ch03-springmvc04工程

自定义参数绑定

- ❖ 自定义参数绑定，将日期串转成`java.util.Date`类型。

```
public class Items {  
    private Integer id;  
  
    private String name;  
  
    private Float price;  
  
    private String pic;  
  
    private Date createtime;  
    private
```



自定义参数绑定

❖ 自定义Converter，实现Converter接口

```
public class CustomDateConverter implements Converter<String,Date>{  
    @Override  
    public Date convert(String source) {  
        //实现 将日期串转成日期类型(格式是yyyy-MM-dd HH:mm:ss)  
        SimpleDateFormat simpleDateFormat = new SimpleDateFormat("yyyy-MM-dd HH:mm:ss");  
        try {  
            //转成直接返回  
            return simpleDateFormat.parse(source);  
        } catch (ParseException e) {  
            // TODO Auto-generated catch block  
            e.printStackTrace();  
        }  
        //如果参数绑定失败返回null  
        return null;  
    }  
}
```

自定义参数绑定

❖ 配置方式

- ▶ 需要向处理器适配器中注入自定义的参数绑定组件。

```
<mvc:annotation-driven conversion-service="conversionService"></mvc:annotation-driven>
```

```
<!-- 自定义参数绑定 -->
<bean id="conversionService" class="org.springframework.format.support.FormattingConversionServiceFactoryBean">
  <!-- 转换器 -->
  <property name="converters">
    <list>
      <!-- 日期类型转换 -->
      <bean class="com.neuedu.controller.converter.CustomDateConverter"/>
    </list>
  </property>
```

集合类型绑定

❖ 集合类型绑定

- ▶ 支持字符串数组、List、Map

❖ 例如

- ▶ 商品的批量删除
- ▶ 页面选中多个checkbox向controller方法传递

```
<c:forEach items="${itemsList}" var="item">
<tr>
  <td><input type="checkbox" name="items_id" value="${item.id }"></td>
  <td>${item.name }</td>
  <td>${item.price }</td>
  <td><fmt:formatDate value="${item.createtime}" pattern="yyyy-MM-dd">
  <td>${item.detail }</td>
  <td><a href="${pageContext.request.contextPath }/items/editItems.ac
</tr>
</c:forEach>
```

```
//批量删除
```

```
@RequestMapping("/deleteItems")
```

```
public String deleteItems(Integer[] items_id) {
```

- ▶ 全部代码参见：ch03-springmvc05工程

集合类型绑定

❖ List

- ▶ List中存放对象，并将定义的List放在包装类中，controller方法中使用包装对象接收。

❖ 例如

- ▶ 商品的批量修改

```
public class ItemsQueryVo {  
  
    //商品信息  
    private Items items;  
  
    //为了系统 可扩展性，对原始生成的po进行扩展  
    private ItemsCustom itemsCustom;  
  
    private List<ItemsCustom> itemsList;  
}
```

- ▶ 全部代码参见：ch03-springmvc05工程

集合类型绑定

❖ List

```
<c:forEach items="${itemsList}" var="item" varStatus="status">
<tr>
  <td><input name="itemsList[${status.index}].name" value="${item.name}"/></td>
  <td><input name="itemsList[${status.index}].price" value="${item.price}"/></td>
  <td><input name="itemsList[${status.index}].createtime" value="<fmt:formatDate val
  <td><input name="itemsList[${status.index}].detail" value="${item.detail}"/></td>
</tr>
</c:forEach>
```

```
//批量修改提交
@RequestMapping("/editItemsAllSubmit")
public String editItemsAllSubmit(ItemsQueryVo itemsQueryVo) {
    //调用service方法
    System.out.println("-----editItemsAllSubmit-----");
    return "success";
}
```

▶ 全部代码参见：ch03-springmvc05工程

课堂练习（20分钟）

- ❖ 简述如何实现自定义参数绑定？
- ❖ 理解并独立完成集合类型的参数绑定教学案例



CONTENTS

目录

01

映射请求

02

方法返回值

03

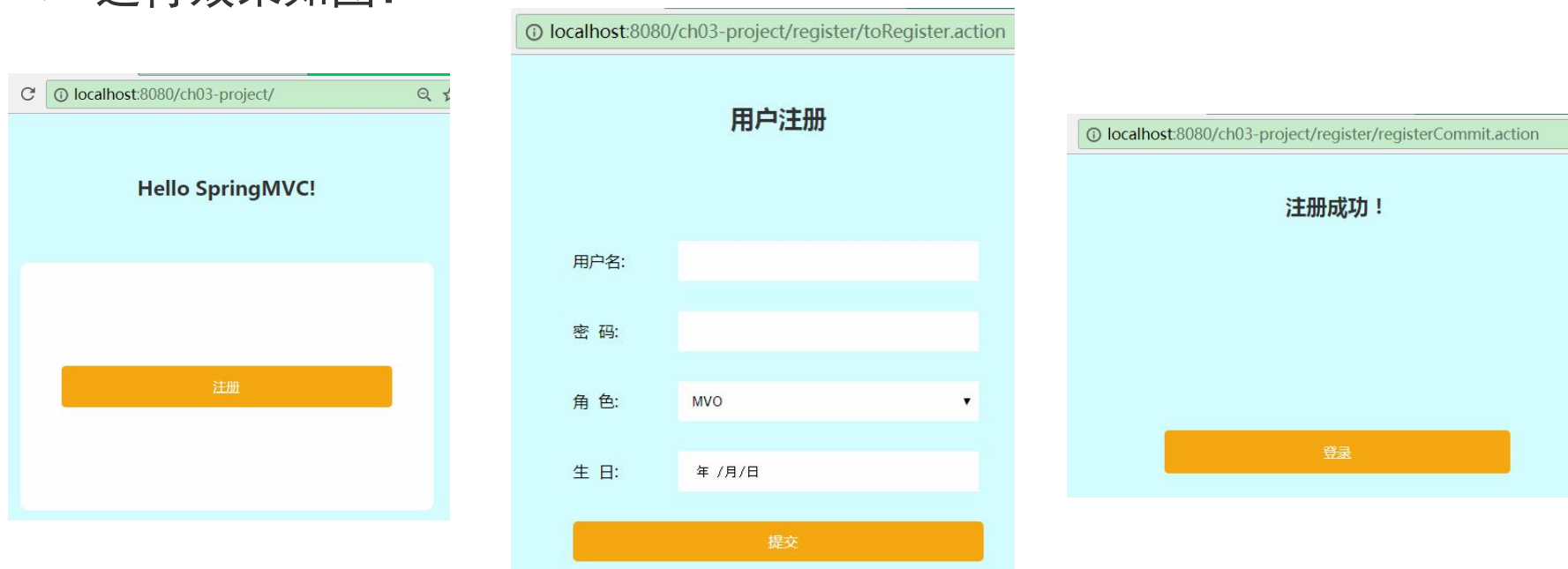
参数绑定

04

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，完成《跨境电商系统》用户注册页面的显示，以及用户注册模块的业务处理
 - ▶ 创建工程代码ch03-project，搭建开发环境
 - ▶ 编写controller、service、dao、pojo类
 - ▶ 在springmvc.xml中原有配置的基础上，增加类型转换器定义和配置
 - ▶ 工程部署和运行<http://localhost:8080/ch03-project/>
- ❖ 运行效果如图：

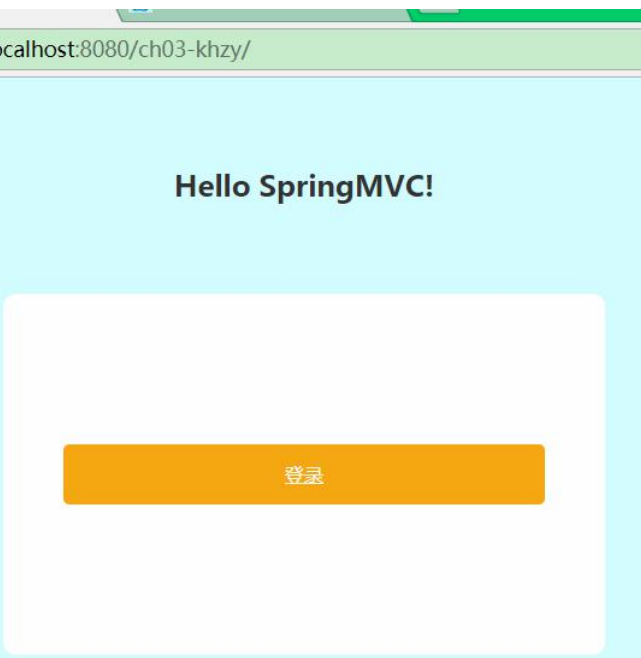


本章重点总结

- ❖ 了解参数绑定的默认支持的类型；
- ❖ 理解参数绑定过程；
- ❖ 掌握@RequestMapping注解映射请求；
- ❖ 掌握方法的返回值类型；
- ❖ 掌握参数绑定，包括简单类型、pojo类型；
- ❖ 掌握自定义参数绑定；

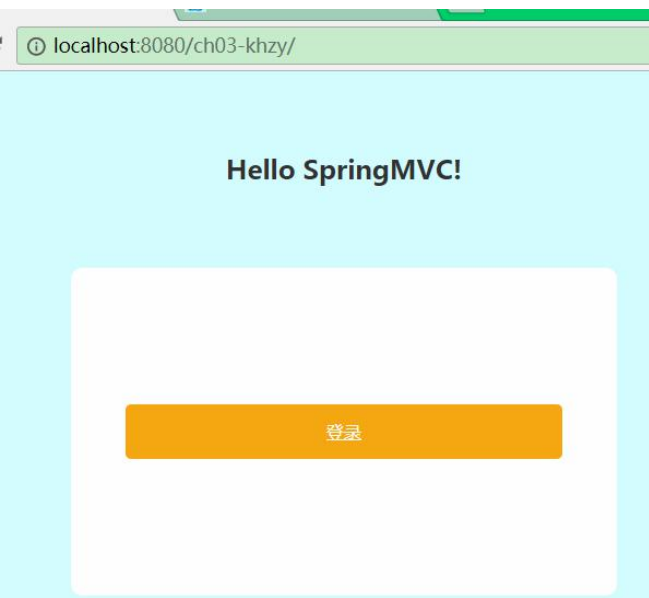
课后作业【必做任务】

- ❖ 1、完成《跨境电商系统》登录页面login.jsp的显示及登录处理（在controller中输出用户名和密码即可）
 - ▶ 使用简单类型参数绑定请求参数
 - ▶ 工程部署和运行<http://localhost:8080/ch03-khzy/>
- ❖ 运行效果如图：



课后作业【必做任务】

- ❖ 2、完成《跨境电商系统》登录页面login.jsp的显示及登录处理（在controller中输出用户名和密码即可）
 - ▶ 使用简单POJO类型参数绑定请求参数
 - ▶ 工程部署和运行<http://localhost:8080/ch03-khzy/>
- ❖ 运行效果如图：



课后作业【选做任务】

- ❖ 完成《跨境电商系统》商品基本信息录入功能。（录入的商品信息在controller中输出到控制台即可）
 - ▶ 商品可录入多种价格方案，前端发起一次请求将商品信息与关联的价格信息提交给后台处理。
 - ▶ 商品的价格信息使用List类型
 - ▶ 工程部署和运行<http://localhost:8080/ch03-khzy02/>

- ❖ 运行效果如图：

商品录入

skuCd:

GM00100

商品名称:

iphone8手机

借卖价:

7000

促销价:

6600

专享价:

6000

保存

```
Console Progress Problems
Tomcat v9.0 Server at localhost [Apache Tor
saveProduct start...
skuCd=GM00100
typeCd=PUB,price=7000
typeCd=PRO,price=6600
typeCd=VIP,price=6000
```


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）

