实验09 Spring MVC框架：数据绑定及回写

**【实验目的及要求】**

1. 熟悉Spring MVC框架的数据绑定及回写方法；
2. 基本数据类型绑定；
3. POJO对象及包装POJO对象绑定；
4. 数组/集合绑定；
5. JSON数据；
6. 要求所有回答的文本格式：**五号，宋体、1.5倍行距，保留段单元格背景。**

**【实验步骤】**

**1、绑定基本数据类型**

（1）在IDEA中新建Maven Web项目javaee\_ex12。在pom.xml文件中加入spring-webmvc及Servlet等依赖，内容如下。

|  |
| --- |
|  |

（2）单击“Run→Edit Configurations”，弹出“Edit Configurations”对话框架，如图9.1所示。

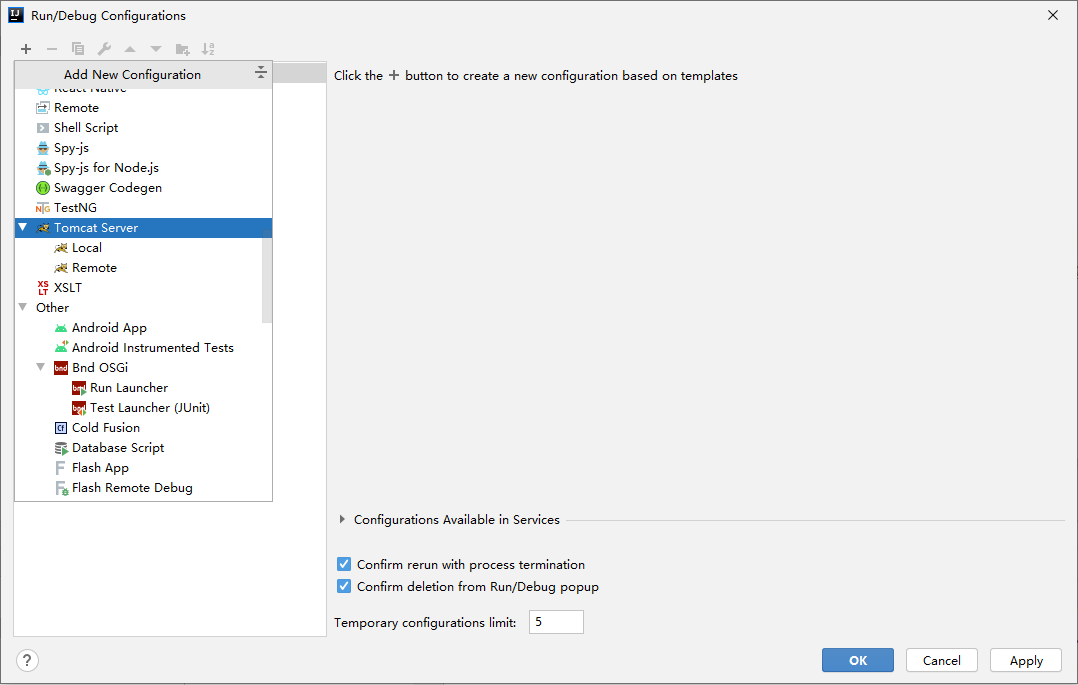


图9.1 “Edit Configurations”对话框

（3）单击“+”按钮，选择“Tomcat Server”项目下的“Local”，添加tomcat运行参数配置界面，如图9.2所示。单击“Configure”按钮，选择一个本地安装的Tomcat服务器（这里选择的是Tomcat 9.0.14版本），在“Named”文本框中输出一个可辨别的名称，如“Tomcat9”。若对话框下面有警告信息，可在“Deployment”选项卡中，单击“+”按钮，在“Select Artifacts to Deploy”对话框中（如图9.3所示）选择一个部署方案。

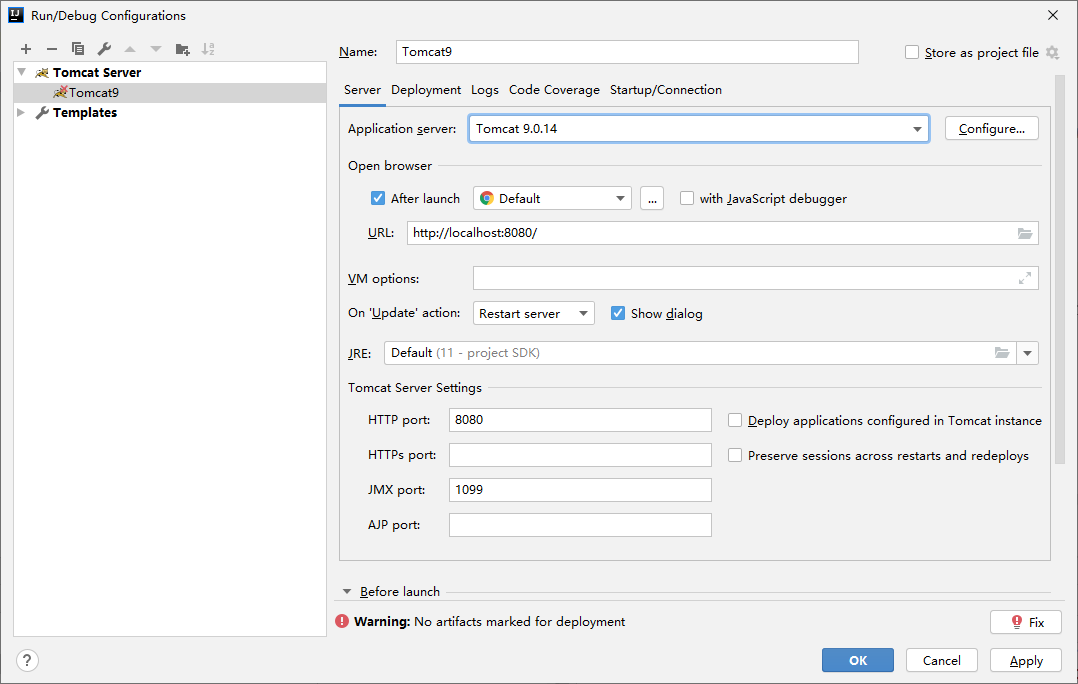


图9.2 Tomcat服务器配置（Server选项卡）

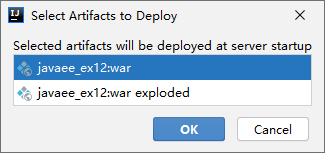


图9.3 “Select Artifacts to Deploy”对话框

**说明**：为了了解Tomcat服务器的更多配置方法，这里采用了与实验08不同的Tomcat服务器配置方式。值得注意的是，需要在电脑中预先安装好可用的Tomcat服务器。

（4）单击“OK”按钮，关闭对话框。此时若运行项目，IDEA将自动启动配置的Tomcat服务器，并在浏览器中显示index.jsp页面。

（5）依次添加Maven项目的默认目录main/java和main/resourses，并在main/java中创建包com.javaee.ex12.controller，在main/resourses中创建Spring MVC配置文件config/spring-mvc.xml。

|  |
| --- |
| *<?*xml version="1.0" encoding="UTF-8"*?>* <beans xmlns="http://www.springframework.org/schema/beans"  xmlns:context="http://www.springframework.org/schema/context"  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"  xsi:schemaLocation="http://www.springframework.org/schema/beans  http://www.springframework.org/schema/beans/spring-beans.xsd  http://www.springframework.org/schema/context  http://www.springframework.org/schema/context/spring-context.xsd">  *<!-- 配置 Spring MVC 要扫描的包 -->* <context:component-scan base-package="com.javaee.ex12.controller"/>  *<!-- 配置视图解析器 -->* <bean class="org.springframework.web.servlet.view.InternalResourceViewResolver">  <property name="prefix" value="/WEB-INF/pages/"/>  <property name="suffix" value=".jsp"/>  </bean> </beans> |

（6）在web.xml中添加编码过滤器及前端控制器配置信息（过滤器须放在前面），内容如下。

|  |
| --- |
|  |

（7）在com.javaee.ex12.controller包中添加控制器CourseController，代码如下。

|  |
| --- |
| package com.javaee.ex12.controller;  import org.springframework.stereotype.Controller; import org.springframework.ui.Model; import org.springframework.web.bind.annotation.RequestMapping; import org.springframework.web.bind.annotation.RequestMethod;  @Controller public class CourseController {  @RequestMapping(value = "/to\_course\_add\_basic", method = RequestMethod.*GET*)  public String toCourseAddBasic(){  return "**course/add\_course\_basic**";  }  @RequestMapping(value = "/course\_add\_basic", method = RequestMethod.*POST*)  public String courseAddBasic(int cid, String cname, int period,  String bname, String pub, float price, Model model){  model.addAttribute("cid", cid);  model.addAttribute("cname", cname);  model.addAttribute("period", period);  model.addAttribute("bname", bname);  model.addAttribute("pub", pub);  model.addAttribute("price", price);  return "**course/course\_info\_basic**";  } } |

（8）根据处理器方法toCourseAddBasic的映射路径，在index.jsp文件中添加的跳转链接，向toCourseAddBasic方法发送get请求。index.jsp内容如下。

|  |
| --- |
|  |

（9）在“WEB-INF/pages/course”目录中，创建add\_course\_basic.jsp文件，设计一表单，提交课程信息。注意，<form/>元素的action属性值要与处理器方法courseAddBasic的映射路径一致。add\_course\_basic.jsp文件内容如下。

|  |
| --- |
| <%@ page contentType="text/html;charset=UTF-8" language="java" %> <html> <head>  <meta http-equiv="Content-Type" content="text/html; charset=UTF-8">  <title>添加课程信息</title> </head> <body> <center>  <form id="form1" name="form1" method="post" action="**course\_add\_basic**">  <table width="500" height="95" border="0" cellpadding="0" cellspacing="1" style="background-color: #3A8ECD; color: #000;">  <tr>  <td height="30" colspan="2" align="center" valign="middle" bgcolor="#FFFFFF">课程信息</td>  </tr>  <tr>  <td width="114" height="30" align="right" valign="middle" bgcolor="#FFFFFF">课程编号：</td>  <td width="227" height="20" align="left" valign="middle" bgcolor="#FFFFFF">  <input name="cid" type="text" size="10" /></td>  </tr>  <tr>  <td height="30" align="right" valign="middle" bgcolor="#FFFFFF">课程名称：</td>  <td height="20" align="left" valign="middle" bgcolor="#FFFFFF">  <input type="text" name="cname" /></td>  </tr>  <tr>  <td height="30" align="right" valign="middle" bgcolor="#FFFFFF">学时数：</td>  <td height="20" align="left" valign="middle" bgcolor="#FFFFFF">  <input name="period" type="text" value="0" size="10" /></td>  </tr>  <tr>  <td height="30" align="right" valign="middle" bgcolor="#FFFFFF">教材名称：</td>  <td height="20" align="left" valign="middle" bgcolor="#FFFFFF">  <input type="text" name="bname" /></td>  </tr>  <tr>  <td height="30" align="right" valign="middle" bgcolor="#FFFFFF">出版社：</td>  <td height="20" align="left" valign="middle" bgcolor="#FFFFFF">  <input type="text" name="pub" /></td>  </tr>  <tr>  <td height="30" align="right" valign="middle" bgcolor="#FFFFFF">价格：</td>  <td height="20" align="left" valign="middle" bgcolor="#FFFFFF">  <input name="price" type="text" value="0." size="10" /></td>  </tr>  <tr>  <td height="30" colspan="2" align="center" valign="middle" bgcolor="#FFFFFF"><input type="submit" name="button" id="button" value="提交" />&nbsp;&nbsp;  <input type="reset" name="button2" id="button2" value="重置" /></td>  </tr>  </table>  </form> </center> </body> </html> |

（10）在“WEB-INF/pages/course”目录中，创建course\_info\_basic.jsp文件，以EL方式显示请求域中的对象。course\_info\_basic.jsp文件内容如下。

|  |
| --- |
| <%@ page contentType="text/html;charset=UTF-8" language="java" %> <%@ page isELIgnored ="false" %> <html> <head>  <title>课程信息</title> </head> <body> <p>课程编号：${cid}</p> <p>课程名称：${cname}</p> <p>学时数：${period}</p> <p>教材名称：${bname}</p> <p>出版社：${pub}</p> <p>价格：${price}</p> </body> </html> |

（11）运行项目，给出表单提交及数据显示的截图。

|  |
| --- |
|  |

（12）请详细阐述从index.jsp到course\_info\_basic.jsp的请求跳转过程。对于每一步跳转，要求给出请求方式、映射路径衔接情况、浏览器地址栏变化及数据传递情况等。

|  |
| --- |
|  |

**2、绑定POJO对象**

（1）接第1个实验，在com.javaee.ex12.po包中创建实体类Course（Java Bean，其属性与表单元素的name属性一致），内容如下。

|  |
| --- |
| package com.javaee.ex12.po;  public class Course {  private int cno;  private String cname;  private int period;  private String bname;  private String pub;  private float price;   // 省略了getter和setter方法 } |

（2）在控制器类CourseController中，添加处理方法toCourseAddPojo和CourseAddPojo。在CourseAddPojo方法中，以POJO方式绑定数据，代码如下。

|  |
| --- |
| @GetMapping("/to\_course\_add\_pojo")  public String toCourseAddPojo(){  return "course/add\_course\_pojo";  }  @PostMapping("/course\_add\_pojo")  public String courseAddPojo(Course course, Model model){  model.addAttribute("course", course);  return "course/course\_info\_pojo";  } |

（3）在“WEB-INF/pages/course”目录中，创建add\_course\_pojo.jsp文件，将add\_course\_basic.jsp文件中的内容复制过来，注意，<form/>元素的action属性值要与处理器方法courseAddPojo的映射路径一致。add\_course\_pojo.jsp文件内容如下。

|  |
| --- |
|  |

（4）在“WEB-INF/pages/course”目录中，创建course\_info\_pojo.jsp文件，以EL方式显示请求域中的对象。course\_info\_pojo.jsp文件内容如下。

|  |
| --- |
|  |

（5）在index.jsp文件中，添加跳转至处理器方法toCourseAddPojo的链接，内容如下。

|  |
| --- |
|  |

（6）运行项目，分别给出POJO绑定下的表单提交及数据显示的截图。

|  |
| --- |
|  |

（7）详细分析和比较POJO绑定与基本数据类型绑定，在信息提交、绑定及回写等方面的区别，总结POJO绑定的优势。

|  |
| --- |
|  |

**3、绑定包装POJO**

（1）接上一个实验，在com.javaee.ex12.po包中创建实体类Book，内容如下。

|  |
| --- |
| package com.javaee.ex12.po;  public class Book {  private String bname;  private String pub;  private float price;   // 省略了getter和setter方法 } |

（2）在com.javaee.ex12.po包创建实体类CourseEx，内容如下。

|  |
| --- |
| package com.javaee.ex12.po;  public class CourseEx {  private int cno;  private String cname;  private int period;  private Book book;   // 省略了getter和setter方法 } |

（3）在控制器类CourseController中，添加处理方法toCourseAddPojoNest和CourseAddPojoNest。在CourseAddPojoNest方法中，以POJO方式绑定数据，代码如下。

|  |
| --- |
| @GetMapping("/to\_course\_add\_pojo\_nest") public String toCourseAddNestPojo(){  return "course/add\_course\_pojo\_nest"; } @PostMapping("/course\_add\_pojo\_nest") public String courseAddNestPojo(CourseEx courseEx, Model model){  model.addAttribute("course", courseEx);  return "course/course\_info\_pojo\_nest"; } |

（4）在“WEB-INF/pages/course”目录中，创建add\_course\_pojo\_nest.jsp文件，将add\_course\_pojo.jsp文件中的内容复制过来，注意，<form/>元素的action属性值要与处理器方法courseAddPojoNest的映射路径一致。add\_course\_pojo\_nest.jsp文件内容如下。

|  |
| --- |
|  |

（5）在“WEB-INF/pages/course”目录中，创建course\_info\_pojo\_nest.jsp文件，以EL方式显示请求域中的对象。course\_info\_pojo.jsp文件内容如下。

|  |
| --- |
|  |

（6）在index.jsp文件中，添加跳转至处理器方法toCourseAddPojoNest的链接，内容如下。

|  |
| --- |
|  |

（7）运行项目，分别给出嵌套POJO类绑定下的表单提交及数据显示的截图。

|  |
| --- |
|  |

（8）详细分析和比较POJO绑定与嵌套POJO类绑定绑定，在信息提交、绑定及回写等方面的区别，总结嵌套POJO类绑定的优势。

|  |
| --- |
|  |

**4、列表和数组绑定**

（1）接上一个实验，创建实体类com.javaee.ex12.po.Student，内容如下。

|  |
| --- |
| package com.javaee.ex12.po;  import java.util.List;  public class Student {  private String sno;  private String sname;  private int age;  private List<String> courseList;   // 省略了getter和setter方法 } |

（2）创建控制器类com.javaee.ex12.controller.StudentController，并在其中添加两个方法，分别用于页面跳转和添加学生信息，内容如下。

|  |
| --- |
| package com.javaee.ex12.controller;  import com.javaee.ex12.po.Student; import org.springframework.stereotype.Controller; import org.springframework.ui.Model; import org.springframework.ui.ModelMap; import org.springframework.web.bind.annotation.GetMapping; import org.springframework.web.bind.annotation.PostMapping;  @Controller public class StudentController {  @GetMapping("/to\_add\_student")  public String toAddStudent(){  return "student/add\_student";  }  @PostMapping("/add\_student")  public String addStudent(Student student, Model model){  model.addAttribute("student",student);  return "student/student\_info";  } } |

（3）在“WEB-INF/pages/student”目录中，创建add\_student.jsp文件，注意，<form/>元素的action属性值要与处理器方法addStudent的映射路径一致。add\_student.jsp文件内容如下。

|  |
| --- |
| <%@ page language="java" contentType="text/html; charset=UTF-8" pageEncoding="UTF-8"%> <!DOCTYPE html> <html> <head>  <meta charset="UTF-8">  <title>选修课程</title> </head> <body>  <form id="form1" name="form1" method="post" action="add\_student"><center>  <table width="500" border="0" cellspacing="1" cellpadding="0" bgcolor="#0099FF">  <tr>  <td height="30" colspan="2" align="center" valign="middle"><span style="color:#FFF">选课信息</span></td>  </tr>  <tr>  <td width="153" height="30" align="right" valign="middle" bgcolor="#FFFFFF">学号：</td>  <td width="344" align="left" valign="middle" bgcolor="#FFFFFF"><label for="textfield"></label>  <input type="text" name="sno" id="textfield" /></td>  </tr>  <tr>  <td height="30" align="right" valign="middle" bgcolor="#FFFFFF">姓名：</td>  <td align="left" valign="middle" bgcolor="#FFFFFF">  <input type="text" name="sname" id="textfield2" /></td>  </tr>  <tr>  <td height="30" align="right" valign="middle" bgcolor="#FFFFFF">年龄：</td>  <td align="left" valign="middle" bgcolor="#FFFFFF">  <input type="text" name="age" id="textfield3" /></td>  </tr>  <tr>  <td height="30" align="right" valign="middle" bgcolor="#FFFFFF">课程列表：</td>  <td align="left" valign="middle" bgcolor="#FFFFFF"><p>  <label>  <input type="checkbox" name="courseList" value="大学英语" id="courseName\_0" />大学英语</label>  <br />  <label>  <input type="checkbox" name="courseList" value="高等数学" id="courseName\_1" />高等数学</label>  <br />  <label>  <input type="checkbox" name="courseList" value="程序设计基础" id="courseName\_2" />程序设计基础</label>  <br />  <label>  <input type="checkbox" name="courseList" value="计算机网络" id="courseName\_3" />计算机网络</label>  <br />  </p></td>  </tr>  <tr>  <td height="30" colspan="2" align="center" valign="middle" bgcolor="#FFFFFF"><input type="submit" name="button" id="button" value="提交" />  &nbsp;  <input type="reset" name="button2" id="button2" value="重置" /></td>  </tr>  </table></center> </form> </body> </html> |

（4）在“WEB-INF/pages/student”目录中，创建student\_info.jsp文件，用于显示请求区域中的数据，内容如下。

|  |
| --- |
| <%@ page contentType="text/html;charset=UTF-8" language="java" %> **<%@ taglib prefix="c" uri="http://java.sun.com/jsp/jstl/core" %>** <%@ page isELIgnored ="false" %> <!DOCTYPE html> <html> <head>  <meta charset="UTF-8">  <title>选课信息</title> </head> <body> <center>选课信息</center> <table align="center" width="500" border="1" cellspacing="0" cellpadding="0" style="border-collapse:collapse" bordercolor="#0099FF">  <tr>  <td>学号</td>  <td>姓名</td>  <td>性别</td>  <td>课程列表</td>  </tr>  <tr>  <td>${student.sno}</td>  <td>${student.sname}</td>  <td>${student.age}</td>  <td>  <c:forEach items="${student.courseList}" var="cname">  ${cname}<br/>  </c:forEach>  </td> </table> </body> </html> |

（5）为了确保student\_info.jsp中，可以正确载入<%@ taglib prefix="c" uri="http://java.sun.com/jsp/jstl/core" %>，在pom.xml中，添加jstl依赖，内容如下。

|  |
| --- |
| <dependency>  <groupId>jstl</groupId>  <artifactId>jstl</artifactId>  <version>1.2</version> </dependency> <dependency>  <groupId>taglibs</groupId>  <artifactId>standard</artifactId>  <version>1.1.2</version> </dependency> |

（6）在index.jsp文件中，添加相应请求链接，内容如下。

|  |
| --- |
| <%@ page contentType="text/html;charset=UTF-8" language="java" %> <html> <body> <h2>Hello World!</h2> <a href="to\_course\_add\_basic">添加课程（基本数据类型绑定）</a><br/> <a href="to\_course\_add\_pojo">添加课程（POJO绑定）</a><br/> <a href="to\_course\_add\_pojo\_nest">添加课程（包装POJO类绑定）</a><br/> <a href="to\_add\_student">学生选课</a><br/> </body> </html> |

（7）运行项目，分别给出表单提交及数据显示的截图。

|  |
| --- |
|  |

（8）在实体类中增加一个数组属性private float[] scores;，用于存放所选课程的成绩。请修改add\_student.jsp、StudentController.java以及student\_info.jsp等文件，确保可以输入所选课程成绩，并在提交后正确显示。被修改的文件内容如下。

|  |
| --- |
| // add\_student.jsp |
| // StudentController.java |
| // student\_info.jsp |

（9）运行项目，分别给出表单提交及数据显示的截图。

|  |
| --- |
|  |

**5、JSON数据绑定及回写**

（1）接上一个实验，在pom.xml文件中添加JSON依赖，内容如下。

|  |
| --- |
| <dependency>  <groupId>com.fasterxml.jackson.core</groupId>  <artifactId>jackson-core</artifactId>  <version>2.13.1</version> </dependency> <dependency>  <groupId>com.fasterxml.jackson.core</groupId>  <artifactId>jackson-databind</artifactId>  <version>2.13.1</version> </dependency> <dependency>  <groupId>com.fasterxml.jackson.core</groupId>  <artifactId>jackson-annotations</artifactId>  <version>2.13.1</version> </dependency> |

（2）在webapp目录中创建js文件夹，将jquery-1.11.3.min.js文件复制到js中。

（3）修改spring-mvc.xml文件，内容如下。

|  |
| --- |
| *<?*xml version="1.0" encoding="UTF-8"*?>* <beans xmlns="http://www.springframework.org/schema/beans"  **xmlns:mvc="http://www.springframework.org/schema/mvc"**  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"  xmlns:context="http://www.springframework.org/schema/context"  xsi:schemaLocation="http://www.springframework.org/schema/beans  http://www.springframework.org/schema/beans/spring-beans-4.3.xsd  **http://www.springframework.org/schema/mvc  http://www.springframework.org/schema/mvc/spring-mvc-4.3.xsd**  http://www.springframework.org/schema/context  http://www.springframework.org/schema/context/spring-context-4.3.xsd">  *<!-- 配置 Spring MVC 要扫描的包 -->* <context:component-scan base-package="com.javaee.ex12.controller"/>  *<!-- 配置视图解析器 -->* <bean class="org.springframework.web.servlet.view.InternalResourceViewResolver">  <property name="prefix" value="/WEB-INF/pages/"/>  <property name="suffix" value=".jsp"/>  </bean>   **<mvc:annotation-driven/>**  *<!--<mvc:default-servlet-handler/>-->*   **<mvc:resources mapping="/js/\*\*" location="/js/"/>** </beans> |

说明：注意标有加粗字体的信息。

（4）在“WEB-INF/pages/book”目录中，创建book\_list.jsp文件，用于显示和添加book信息，内容如下。

|  |
| --- |
| <%@ page contentType="text/html;charset=UTF-8" language="java" %> <!DOCTYPE html> <html> <head>  <meta charset="UTF-8">  <title>测试Json</title>  </head> <body> <table align="center" width="500" border="1" cellspacing="0" cellpadding="0" style="border-collapse:collapse" bordercolor="#0099FF">  <tr id="table-title">  <td>书名</td>  <td>出版社</td>  <td>价格</td>  </tr> </table>  <br/>  <table width="300" border="0" cellspacing="0" cellpadding="0" align="center">  <tbody>  <tr>  <td>书名</td>  <td><input type="text" id="bname"></td>  </tr>  <tr>  <td>出版社</td>  <td><input type="text" id="pub"></td>  </tr>  <tr>  <td>价格</td>  <td><input type="text" id="price"></td>  </tr>  <tr>  <td>&nbsp;</td>  <td><input id="btn-add-book" type="button" value="添加"></td>  </tr>  </tbody> </table>  <script type="text/javascript" src="js/jquery-1.11.3.min.js"></script> <script type="text/javascript">  $("#btn-add-book").click(function(){  var bname = $("#bname").val(),  pub = $("#pub").val(),  price = $("#price").val();   $.post("add\_book",{"bname":bname,"pub":pub,"price":price},  function(data){  if (data == "OK"){  var newBook = "<tr><td>" + bname + "</td><td>" + pub + "</td><td>" + price + "</td></tr>";  $("#table-title").after(newBook);  }  })  }) </script> </body> </html> |

（5）在index.jsp中添加请求链接，内容如下。

|  |
| --- |
|  |

（6）运行项目，给出添加book信息的页面截图。

|  |
| --- |
|  |

（7）试分析JSON方式与前面的绑定及回写方式的差异。

|  |
| --- |
|  |