1、MVC 概念

MVC 全称：Model 模型、 View 视图、 Controller 控制器。

MVC 最早出现在 JavaEE 三层中的 Web 层，它可以有效的指导 Web 层的代码如何有效分离，单独工作。

- View 视图：只负责数据和界面的显示，不接受任何与显示数据无关的代码，便于程序员和美工的分工合作——

JSP/HTML。

- Controller 控制器：只负责接收请求，调用业务层的代码处理请求，然后派发页面，是一个“调度者”的角色——Servlet。

转到某个页面。或者是重定向到某个页面。

- Model 模型：将与业务逻辑相关的数据封装为具体的 JavaBean 类，其中不掺杂任何与数据处理相关的代码——

JavaBean/domain/entity/pojo。

MVC 是一种思想

MVC 的理念是将软件代码拆分成为组件，单独开发，组合使用（**目的还是为了降低耦合度**）。

2. 书城第五阶段代码实现

2.1 图书模块

2.1.1 编写图书模块的数据库表

2.1.1.1 创建表

```sql
DROP TABLE IF EXISTS `t_book`;
CREATE TABLE `t_book` (
  `id` int NOT NULL AUTO_INCREMENT,
  `name` varchar(100) DEFAULT NULL,
  `price` decimal(11,2) DEFAULT NULL,
  `author` varchar(100) DEFAULT NULL,
  `sales` int DEFAULT NULL,
  `stock` int DEFAULT NULL,
  `img_path` varchar(200) DEFAULT NULL,
  PRIMARY KEY (`id`)
) ENGINE=InnoDB AUTO_INCREMENT=25 DEFAULT CHARSET=utf8mb4
COLLATE=utf8mb4_0900_ai_ci;
```

2.1.1.2 插入测试数据

```sql
INSERT INTO `t_book` VALUES (1, 'UNIX 高级环境编程', 99.15, '小红红', 210, 810,
'static/img/default.jpg');
INSERT INTO `t_book` VALUES (2, 'javaScript 高级编程', 69.15, '白哥', 210, 810,
'static/img/default.jpg');
INSERT INTO `t_book` VALUES (3, '大话设计模式', 89.15, '白哥', 20, 10,
'static/img/default.jpg');
INSERT INTO `t_book` VALUES (4, '人月神话', 88.15, '花哥', 20, 80,
'static/img/default.jpg');
INSERT INTO `t_book` VALUES (5, 'java 从入门到放弃', 80.00, '小绿绿', 9999, 9,
'static/img/default.jpg');
INSERT INTO `t_book` VALUES (6, '数据结构与算法', 78.50, '小红红', 6, 13,
'static/img/default.jpg');
INSERT INTO `t_book` VALUES (7, '怎样拐跑别人的媳妇', 68.00, '龙伍', 99999, 52,
'static/img/default.jpg');
INSERT INTO `t_book` VALUES (8, '木虚肉盖饭', 16.00, '小胖', 1000, 50,
'static/img/default.jpg');
INSERT INTO `t_book` VALUES (9, 'C++编程思想', 45.50, '花哥', 14, 95,
'static/img/default.jpg');
INSERT INTO `t_book` VALUES (10, '蛋炒饭', 9.90, '周星星', 12, 53,
'static/img/default.jpg');
INSERT INTO `t_book` VALUES (11, '赌神', 66.50, '龙伍', 125, 535,
'static/img/default.jpg');
INSERT INTO `t_book` VALUES (12, 'Java 编程思想', 99.50, '阳哥', 47, 36,
'static/img/default.jpg');
INSERT INTO `t_book` VALUES (13, 'JavaScript 从入门到精通', 9.90, '小绿绿', 85,
95, 'static/img/default.jpg');
INSERT INTO `t_book` VALUES (14, 'cocos2d-x 游戏编程入门', 49.00, '小绿绿', 52,
62, 'static/img/default.jpg');
INSERT INTO `t_book` VALUES (15, 'C 语言程序设计', 28.00, '谭浩强', 52, 74,
'static/img/default.jpg');
```

```
INSERT INTO `t_book` VALUES (16, 'Lua 语言程序设计', 51.50, '雷丰阳', 48, 82,
'static/img/default.jpg');
INSERT INTO `t_book` VALUES (17, '西游记', 12.00, '罗贯中', 19, 9999,
'static/img/default.jpg');
INSERT INTO `t_book` VALUES (18, '水浒传', 33.05, '华仔', 22, 88,
'static/img/default.jpg');
INSERT INTO `t_book` VALUES (19, '操作系统原理', 133.05, '刘优', 122, 188,
'static/img/default.jpg');
INSERT INTO `t_book` VALUES (20, '数据结构 java 版', 173.15, '小兰兰', 21, 81,
'static/img/default.jpg');
INSERT INTO `t_book` VALUES (21, 'UNIX 高级环境编程', 99.15, '乐天', 210, 810,
'static/img/default.jpg');
INSERT INTO `t_book` VALUES (22, 'javaScript 高级编程', 69.15, '白哥', 210, 810,
'static/img/default.jpg');
INSERT INTO `t_book` VALUES (23, '大话设计模式', 89.15, '白哥', 20, 10,
'static/img/default.jpg');
INSERT INTO `t_book` VALUES (24, '人月神话', 88.15, '花哥', 20, 80,
'static/img/default.jpg');
```

2.1.2 编写图书模块的 JavaBean

```
package com.qidi.pojo;

import java.math.BigDecimal;

/**
 * @author 白世鑫
 * @title: Book
 * @projectName Tomcat
 * @description:
 * @date 2020/9/2  1:14 上午
 */
public class Book {

    private Integer id;
    private String name;
    private BigDecimal price;
    private String author;
    private Integer sales;
    private Integer stock;
    private String img_path = "static/img/default.jpg";

    public Book() {
    }

    public Book(Integer id, String name, BigDecimal price, String author,
Integer sales, Integer stock, String img_path) {
```

```
        this.id = id;
        this.name = name;
        this.price = price;
        this.author = author;
        this.sales = sales;
        this.stock = stock;
        if(img_path!=null || "".equals(img_path)){
            this.img_path = img_path;
        }
    }

    public Integer getId() {
        return id;
    }

    public void setId(Integer id) {
        this.id = id;
    }

    public String getName() {
        return name;
    }

    public void setName(String name) {
        this.name = name;
    }

    public BigDecimal getPrice() {
        return price;
    }

    public void setPrice(BigDecimal price) {
        this.price = price;
    }

    public String getAuthor() {
        return author;
    }

    public void setAuthor(String author) {
        this.author = author;
    }

    public Integer getSales() {
        return sales;
    }

    public void setSales(Integer sales) {
        this.sales = sales;
```

```
    }

    public Integer getStock() {
        return stock;
    }

    public void setStock(Integer stock) {
        this.stock = stock;
    }

    public String getImg_path() {
        return img_path;
    }

    public void setImg_path(String img_path) {
        if(img_path!=null || "".equals(img_path)){
            this.img_path = img_path;
        }
    }

    @Override
    public String toString() {
        return "Book{" +
                "id=" + id +
                ", name='" + name + '\'' +
                ", price=" + price +
                ", author='" + author + '\'' +
                ", sales=" + sales +
                ", stock=" + stock +
                ", img_path='" + img_path + '\'' +
                '}';
    }
}
```

2.1.3 编写图书模块的 Dao 和测试 Dao

BookDao.java

```
package com.qidi.dao;

import com.qidi.pojo.Book;

import java.util.List;
```

```java
/**
 * @author 白世鑫
 * @description
 * @date 2020/9/2
 */
public interface BookDao {

    public int addBook(Book book);

    public int updateBook(Book book);

    public int deleteBookById(Integer id);

    public Book queryBookById(Integer id);

    public List<Book> queryBooks();
}
```

BookDaoImpl.java

```java
package com.qidi.dao.impl;

import com.qidi.dao.BookDao;
import com.qidi.pojo.Book;

import java.util.List;

/**
 * @author 白世鑫
 * @title: BookDaoImpl
 * @projectName Tomcat
 * @description:
 * @date 2020/9/2  1:28 上午
 */
public class BookDaoImpl extends BaseDao implements BookDao {
    @Override
    public int addBook(Book book) {
        String sql = "insert into
t_book(name,price,author,sales,stock,img_path) values(?,?,?,?,?,?)";
        return
update(sql,book.getName(),book.getPrice(),book.getAuthor(),book.getSales(),boo
k.getStock(),book.getImg_path());
    }

    @Override
    public int updateBook(Book book) {
        String sql = "update t_book set
name=?,price=?,author=?,sales=?,stock=?,img_path=? where id=?";
```

```
        return
update(sql,book.getName(),book.getPrice(),book.getAuthor(),book.getSales(),boo
k.getStock(),book.getImg_path(),book.getId());
    }

    @Override
    public int deleteBookById(Integer id) {
        String sql = "delete from t_book where id=?";
        return update(sql,id);
    }

    @Override
    public Book queryBookById(Integer id) {
        String sql = "select id,name,price,author,sales,stock,img_path from
t_book where id=?";
        return queryForOne(Book.class,sql,id);
    }

    @Override
    public List<Book> queryBooks() {
        String sql = "select id,name,price,author,sales,stock,img_path from
t_book";
        return queryForList(Book.class,sql);
    }
}
```

BookDaoImplTest.java

```
package test.com.qidi.dao.impl;

import com.qidi.dao.BookDao;
import com.qidi.dao.impl.BookDaoImpl;
import com.qidi.pojo.Book;
import org.junit.Test;
import org.junit.Before;
import org.junit.After;

import java.math.BigDecimal;
import java.util.List;

/**
 * BookDaoImpl Tester.
 *
 * @author <Authors name>
 * @version 1.0
 * @since <pre>9月 2, 2020</pre>
 */
public class BookDaoImplTest {
```

```
    private BookDao bookDao = new BookDaoImpl();

    /**
     * Method: addBook(Book book)
     */
    @Test
    public void testAddBook() throws Exception {
        bookDao.addBook(new Book(null, "玉蒲团", new BigDecimal(99.99), "小白
哥", 1234, 22, null));
    }

    /**
     * Method: updateBook(Book book)
     */
    @Test
    public void testUpdateBook() throws Exception {
        bookDao.updateBook(new Book(25, "金瓶梅插图版", new BigDecimal(999.99),
"Admiral", 1234, 12, null));
    }

    /**
     * Method: deleteBookById(Integer id)
     */
    @Test
    public void testDeleteBookById() throws Exception {
        bookDao.deleteBookById(25);
    }

    /**
     * Method: queryBookById(Integer id)
     */
    @Test
    public void testQueryBookById() throws Exception {
        System.out.println(bookDao.queryBookById(25));
    }

    /**
     * Method: queryBooks()
     */
    @Test
    public void testQueryBooks() throws Exception {
        List<Book> books = bookDao.queryBooks();
        for (Book book : books) {
            System.out.println(book);
        }
    }
}
```

2.1.4 编写图书模块的 Service 和测试 Service

BookService.java

```
package com.qidi.service;

import com.qidi.pojo.Book;

import java.util.List;

/**
 * @author 白世鑫
 * @description
 * @date 2020/9/2
 */
public interface BookService {

    public int addBook(Book book);

    public int updateBook(Book book);

    public int deleteBookById(Integer id);

    public Book queryBookById(Integer id);

    public List<Book> queryBooks();

}
```

BookServiceImpl.java

```
package com.qidi.service.impl;

import com.qidi.dao.BookDao;
import com.qidi.dao.impl.BookDaoImpl;
import com.qidi.pojo.Book;
import com.qidi.service.BookService;

import java.util.List;

/**
 * @author 白世鑫
 * @title: BookServiceImpl
 * @projectName Tomcat
 * @description:
 * @date 2020/9/2  1:46 上午
 */
```

```
public class BookServiceImpl implements BookService {

    private BookDao bookDao = new BookDaoImpl();

    @Override
    public int addBook(Book book) {
        return bookDao.addBook(book);
    }

    @Override
    public int updateBook(Book book) {
        return bookDao.updateBook(book);
    }

    @Override
    public int deleteBookById(Integer id) {
        return bookDao.deleteBookById(id);
    }

    @Override
    public Book queryBookById(Integer id) {
        return bookDao.queryBookById(id);
    }

    @Override
    public List<Book> queryBooks() {
        return bookDao.queryBooks();
    }
}
```

BookServiceImplTest.java

```
package test.com.qidi.service.impl;

import com.qidi.pojo.Book;
import com.qidi.service.BookService;
import com.qidi.service.impl.BookServiceImpl;
import org.junit.Test;
import org.junit.Before;
import org.junit.After;

import java.math.BigDecimal;
import java.util.List;

/**
 * BookServiceImpl Tester.
 *
 * @author <Authors name>
 * @version 1.0
```

```
 * @since <pre>9月 2, 2020</pre>
 */
public class BookServiceImplTest {

    private BookService bookService = new BookServiceImpl();

    /**
     * Method: addBook(Book book)
     */
    @Test
    public void testAddBook() throws Exception {
        bookService.addBook(new Book(null, "那小子真帅", new BigDecimal(89), "张
碧池", 3434, 231, null));
    }

    /**
     * Method: updateBook(Book book)
     */
    @Test
    public void testUpdateBook() throws Exception {
        bookService.updateBook(new Book(26, "那姑娘真美", new BigDecimal(899),
"张碧池", 343344, 0, null));
    }

    /**
     * Method: deleteBookById(Integer id)
     */
    @Test
    public void testDeleteBookById() throws Exception {
        bookService.deleteBookById(26);
    }

    /**
     * Method: queryBookById(Integer id)
     */
    @Test
    public void testQueryBookById() throws Exception {
        System.out.println(bookService.queryBookById(26));
    }

    /**
     * Method: queryBooks()
     */
    @Test
    public void testQueryBooks() throws Exception {
        List<Book> books = bookService.queryBooks();
        for (Book book : books) {
            System.out.println(book);
        }
```

```
    }
}
```

2.1.5 编写图书模块的web层和页面连调

2.1.5.1 图书列表功能实现

2.1.5.1.1 图解流程

2.1.5.1.2 创建 BookServlet

BookServlet.java

```
package com.qidi.web;

import com.qidi.pojo.Book;
import com.qidi.service.BookService;
import com.qidi.service.impl.BookServiceImpl;

import javax.servlet.ServletException;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
import java.io.IOException;
import java.util.List;

/**
 * @author 白世鑫
 * @title: BookServlet
 * @projectName Tomcat
 * @description:
 * @date 2020/9/2  2:05 上午
 */
public class BookServlet extends BaseServlet{

    private BookService bookService = new BookServiceImpl();

    protected void add(HttpServletRequest request, HttpServletResponse
response) throws ServletException, IOException {
        super.doPost(request, response);
```

```
    }
    protected void update(HttpServletRequest request, HttpServletResponse
response) throws ServletException, IOException {
        super.doPost(request, response);
    }
    protected void delete(HttpServletRequest request, HttpServletResponse
response) throws ServletException, IOException {
        super.doPost(request, response);
    }
    protected void list(HttpServletRequest request, HttpServletResponse
response) throws ServletException, IOException {
        //1.查询数据
        List<Book> books = bookService.queryBooks();
        //2.将数据保存到 request 域中
        request.setAttribute("books",books);
        //3.请求转发到 /pages/manager/book_manager.jsp 页面

 request.getRequestDispatcher("/pages/manager/book_manager.jsp").forward(reque
st,response);
    }
}
```

web.xml中配置请求地址 /manager/bookServlet 这里添加manager为了后面的权限管理

```
<?xml version="1.0" encoding="UTF-8"?>
<web-app xmlns="http://xmlns.jcp.org/xml/ns/javaee"
         xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
         xsi:schemaLocation="http://xmlns.jcp.org/xml/ns/javaee
http://xmlns.jcp.org/xml/ns/javaee/web-app_4_0.xsd"
         version="4.0">

    <servlet>
        <servlet-name>UserServlet</servlet-name>
        <servlet-class>com.qidi.web.UserServlet</servlet-class>
    </servlet>
    <servlet-mapping>
        <servlet-name>UserServlet</servlet-name>
        <url-pattern>/userServlet</url-pattern>
    </servlet-mapping>


    <servlet>
        <servlet-name>BookServlet</servlet-name>
        <servlet-class>com.qidi.web.BookServlet</servlet-class>
    </servlet>
    <servlet-mapping>
        <servlet-name>BookServlet</servlet-name>
        <url-pattern>/manager/bookServlet</url-pattern>
```

```
    </servlet-mapping>


</web-app>
```

2.1.5.1.3 修改图书管理请求地址

修改提取出的 manager_menu.jsp 中的请求地址:

```
<a href="manager/bookServlet?action=list">图书管理</a>
```

完整代码

```
<%--
  Created by IntelliJ IDEA.
  User: baishixin
  Date: 2020/9/1
  Time: 3:22 下午
  To change this template use File | Settings | File Templates.
--%>
<%@ page contentType="text/html;charset=UTF-8" language="java" %>
<div>
    <a href="manager/bookServlet?action=list">图书管理</a>
    <a href="order_manager.jsp">订单管理</a>
    <a href="../../index.jsp">返回商城</a>
</div>
```

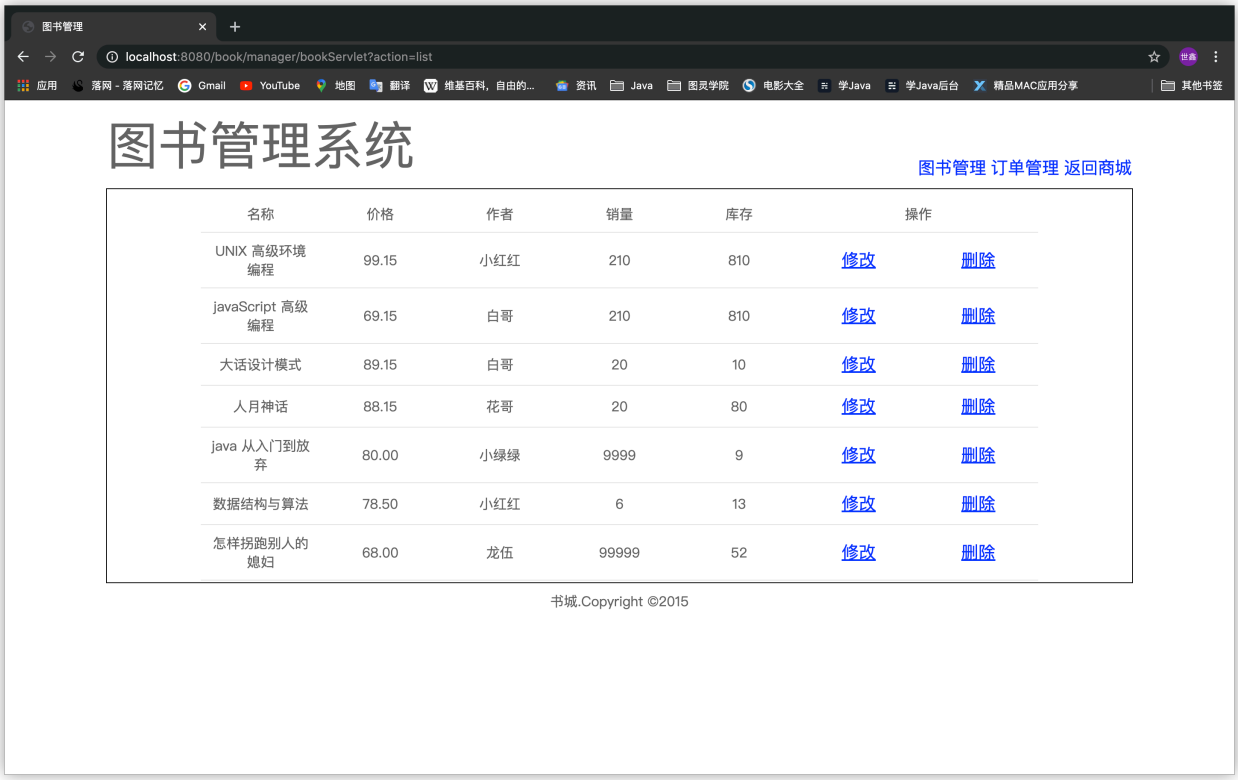
2.1.5.1.4 修改页面 遍历输出数据

book_manager.jsp

```
<%@ taglib prefix="c" uri="http://java.sun.com/jsp/jstl/core" %>
<%@ page contentType="text/html;charset=UTF-8" language="java" %>
<!DOCTYPE html>
<html>
<head>
<meta charset="UTF-8">
<title>图书管理</title>
```

```
  <%--  静态包含 base css样式 JQuery  --%>
  <%@ include file="/pages/common/header.jsp"%>
</head>
<body>

  <div id="header">
      <img class="logo_img" alt="" src="../../static/img/logo.gif" >
      <span class="wel_word">图书管理系统</span>
    <%-- 静态包含图书管理菜单 --%>
    <%@ include file="/pages/common/manager_menu.jsp"%>
  </div>

  <div id="main">
    <table>
      <tr>
        <td>名称</td>
        <td>价格</td>
        <td>作者</td>
        <td>销量</td>
        <td>库存</td>
        <td colspan="2">操作</td>
      </tr>

      <c:forEach items="${requestScope.books}" var="book">
        <tr>
          <td>${book.name}</td>
          <td>${book.price}</td>
          <td>${book.author}</td>
          <td>${book.sales}</td>
          <td>${book.stock}</td>
          <td><a href="book_edit.jsp">修改</a></td>
          <td><a href="#">删除</a></td>
        </tr>
      </c:forEach>


      <tr>
        <td></td>
        <td></td>
        <td></td>
        <td></td>
        <td></td>
        <td></td>
        <td><a href="book_edit.jsp">添加图书</a></td>
      </tr>
    </table>
  </div>

  <%--  静态包含页脚信息  --%>
```

```
  <%@ include file="/pages/common/footer.jsp"%>
</body>
</html>
```



2.1.5.2 添加图书功能实现

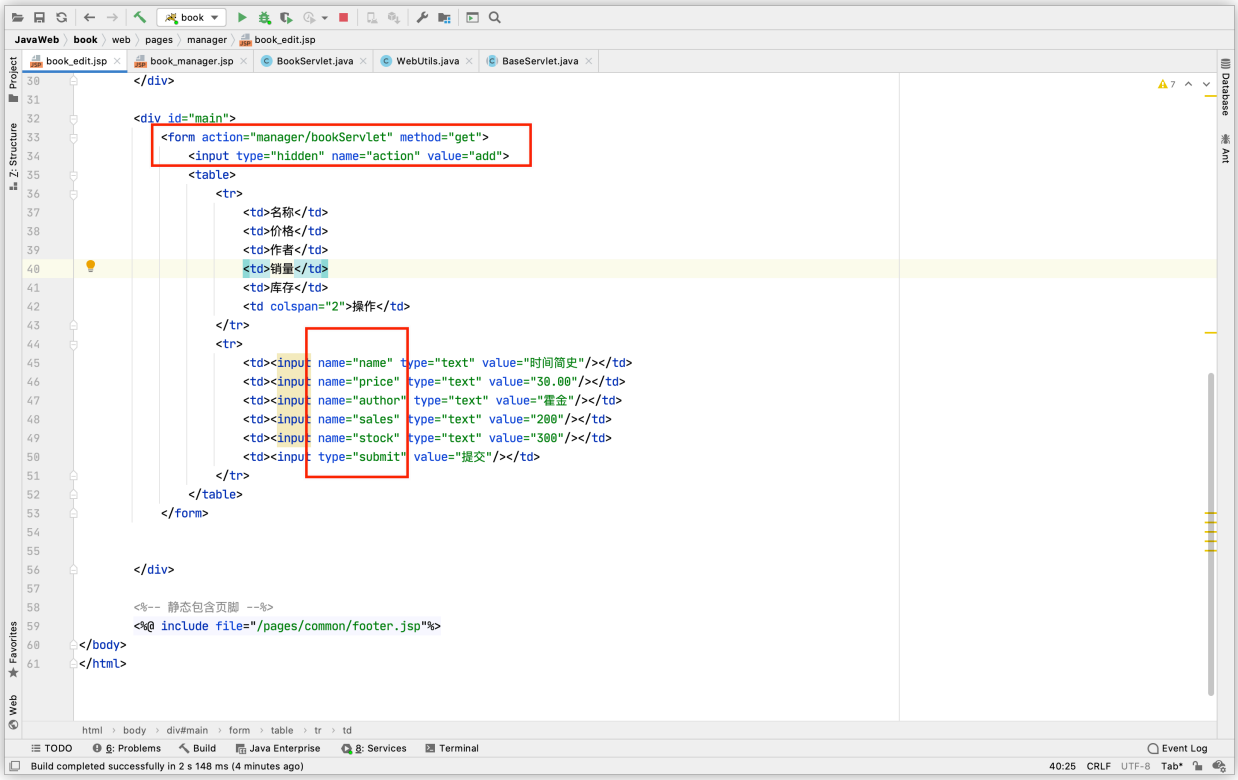
修改 添加图书的跳转路径

```
<td><a href="pages/manager/book_edit.jsp">添加图书</a></td>
```

1 BookServlet 程序中添加 add 方法

```
protected void add(HttpServletRequest request, HttpServletResponse response)
throws ServletException, IOException {
        //1.获取请求参数,封装成 Book 对象
        Book book = WebUtils.copyParamToBean(request, new Book());
        //2.调用 service 保存book对象
        bookService.addBook(book);
        //3.跳到图书列表页面
        request.getRequestDispatcher("/manager/bookServlet?
action=list").forward(request,response);
}
```

2 修改 book_edit.jsp 页面



3.解决表单重复提交

修改add方法为重定向

```
protected void add(HttpServletRequest request, HttpServletResponse response)
throws ServletException,IOException {
        //1.获取请求参数,封装成 Book 对象
        Book book = WebUtils.copyParamToBean(request, new Book());
        //2.调用 service 保存book对象
        bookService.addBook(book);
        //3.跳到图书列表页面
//          request.getRequestDispatcher("/manager/bookServlet?
action=list").forward(request, response);
        response.sendRedirect(request.getContextPath() +
"/manager/bookServlet?action=list");
}
```

2.1.5.3 删除图书功能的实现

1. BookServlet 程序中的 delete 方法：

```
protected void delete(HttpServletRequest request, HttpServletResponse
response) throws ServletException, IOException {
        //1.获取请求的参数  id
        String strId = request.getParameter("id");
        //2.调用service删除图书
        bookService.deleteBookById(WebUtils.parseInt(strId,0));
        //3.重定向到 /book/manager/bookServlet?action=list
        response.sendRedirect(request.getContextPath() +
"/manager/bookServlet?action=list");
}
```

2. 给 WebUtils 工具类添加转换 int 类型的工具方法

```
    public static int parseInt(String strId,int defaultValue){
        try {
            return Integer.parseInt(strId);
        }catch (Exception e){
            e.printStackTrace();
        }
        return defaultValue;
    }
```

3. 修改删除的连接地址：

```
<td><a href="manager/bookServlet?action=delete">删除</a></td>
```

4. 给删除添加确认提示操作：

```
<script type="text/javascript">
        $(function () {
            // 给删除的 a 标签绑定单击事件，用于删除的确认提示操作
            $("a.deleteClass").click(function () {
                // 在事件的 function 函数中，有一个 this 对象。这个 this 对象，是当前
正在响应事件的 dom 对象。
                /**
                 * confirm 是确认提示框函数
                 * 参数是它的提示内容
                 * 它有两个按钮，一个确认，一个是取消。
                 * 返回 true 表示点击了，确认，返回 false 表示点击取消。
                 */
                return confirm("你确定要删除【" +
$(this).parent().parent().find("td:first").text() + "】?");
                // return false// 阻止元素的默认行为===不提交请求
            });
```

```
        });
</script>
```

5.删除添加事件

```
<td><a class="deleteClass" href="manager/bookServlet?
action=delete&id=${book.id}">删除</a></td>
```

2.1.5.4 修改图书功能的实现

1. 更新【修改】的请求地址

```
<td><a href="manager/bookServlet?action=getBook&id=${book.id}">修改</a></td>
```

2. BookServlet 程序中添加 getBook 方法

```
protected void getBook(HttpServletRequest request, HttpServletResponse
response) throws ServletException, IOException {
        //1.获取请求参数 id
        String id = request.getParameter("id");
        //2.根据id调用service查询图书信息
        Book book = bookService.queryBookById(WebUtils.parseInt(id, 0));
        //3.将查询到的图书信息保存到 request 域中
        request.setAttribute("book",book);
        //4.跳转到 book_edit.jsp

 request.getRequestDispatcher("/pages/manager/book_edit.jsp").forward(request,
response);
}
```

3.在 book_edit.jsp 页面中显示修改的数据

```
<tr>
            <td><input name="name" type="text"
value="${requestScope.book.name}"/></td>
            <td><input name="price" type="text"
value="${requestScope.book.price}"/></td>
            <td><input name="author" type="text"
value="${requestScope.book.author}"/></td>
            <td><input name="sales" type="text"
value="${requestScope.book.sales}"/></td>
            <td><input name="stock" type="text"
value="${requestScope.book.stock}"/></td>
            <td><input type="submit" value="提交"/></td>
</tr>
```

4.在 BookServlet 程序中添加 update 方法

```
protected void update(HttpServletRequest request, HttpServletResponse
response) throws ServletException, IOException {
        //1.获取请求参数 封装为 JavaBean
        Book book = WebUtils.copyParamToBean(request, new Book());
        //2.调用 service 更新图书
        bookService.updateBook(book);
        //3.重定向到 图书列表
        response.sendRedirect(request.getContextPath()+"/manager/bookServlet?
action=list");
}
```

5.解决 book_edit.jsp 页面，即要实现添加，又要实现修改操作。