序号: 24

学号: 16430121

常州大学

实 验 报 告

课 程 名 称：Java EE 程序设计与应用开发

实 验 题 目：Servlet 基础编程

实 验 次 数：第 二 次实验　实验时间：2019 年 4 月 9 日

学 生 姓 名：林锦雄

学　　院：信息数理　专 业 班 级：计算机 162

指 导 教 师：陆洁茹　评 阅 成 绩：

一、实验目的：

1. 掌握如何创建 Servlet。
2. 掌握 Servlet 的生命周期。
3. 掌握如何在 Servlet 中使用 JSP 页面中常用的内置对象。

二、实验仪器、设备

1、硬件环境

PC 微机；2G 以上内存；VGA 显示格式。

2、软件环境

Windows XP 以上操作系统，JDK，Tomcat 服务器等。

三、实验内容与要求

在数据库中建立表格 T_BOOK(BOOKID, BOOKNAME, BOOKPRICE)，插入一些记录。如图 1 所示。

```
MariaDB [(none)]> use books;
Reading table information for completion of table and column names
You can turn off this feature to get a quicker startup with -A

Database changed
MariaDB [books]> select * from T_BOOK;
+--------+--------------------+-----------+
| BOOKID | BOOKNAME           | BOOKPRICE |
+--------+--------------------+-----------+
| 10001  | 童话故事           |      32.5 |
| 10002  | 安徒生童话         |      23.8 |
| 10003  | 格林童话           |      15.8 |
| 10004  | 希腊神话全集       |    159.99 |
| 10005  | 数据结构           |      43.5 |
| 10006  | 编译原理           |      57.9 |
| 10007  | BigBen             |      32.3 |
+--------+--------------------+-----------+
7 rows in set (0.001 sec)

MariaDB [books]>
```

图 1. books 数据库中的 T_BOOK 表

1. 编写一个模糊查询图书的应用，输入图书名称的模糊资料，显示查询的图书的 ID，名称和价格。要求提交给 Servlet 来处理。

searchBooksByName.jsp 页面只有一个查询表单，把需要查询的书名的关键词提交给 searchBooksByNameServlet 处理。搜索数据库中书名中带有‘话’的书如图 2 所示。

```
1. <!-- searchBooksByName.jsp -->
2. <form method="post" action="searchBooksByNameServlet">
3.     请输入需要查询的图书名称：<input type="text" name="bookName"><br><br>
4.     <input type="submit" value="查询"> ~
5.     <input type="reset" value="重置"><br>
6. </form>
```

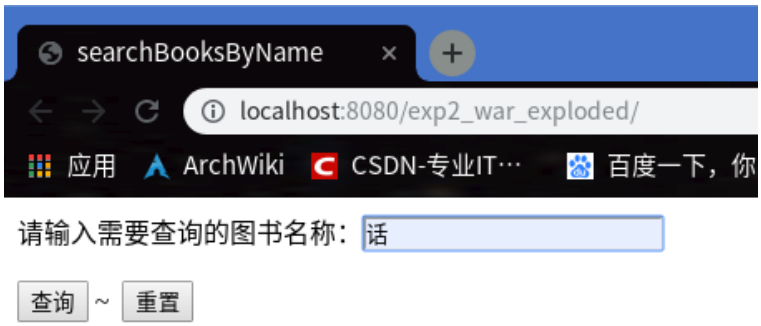

图 2. 搜索页面

searchBooksByNameServlet 接受搜索网页传来的关键词，连接 books 数据库，根据关键词查询 T_BOOK 表中全部符合要求的书的数据，并把这些查到的书的数据存入一个 ArrayList 对象中，最后存入 JSP 内置对象 session 中，通过 session 把数据返回给前端网页。

```
1. @WebServlet(name = "searchBooksByNameServlet")
2. protected void doPost(HttpServletRequest request, HttpServletResponse response) t
hrows ServletException, IOException {
3.     request.setCharacterEncoding("utf-8");
4.     String keyWord = request.getParameter("bookName");
5.     HttpSession session = request.getSession();
6.     try {
7.         Class.forName("com.mysql.jdbc.Driver");
8.         Connection conn = DriverManager.getConnection("jdbc:mysql://localhost:330
6/books?characterEncoding=UTF-8","root","bingjie123");
9.         String sql = "select BOOKID,BOOKNAME,BOOKPRICE from T_BOOK where BOOKNAME
 like ?";
10.          PreparedStatement ps = conn.prepareStatement(sql);
11.          ps.setString(1,"%" + keyWord + "%");
12.          ResultSet rs = ps.executeQuery();
13.          ArrayList<Book> books = new ArrayList<>();
14.          while(rs.next()) {
15.              Book book = new Book();
16.              book.setBOOKID(rs.getString(1));
17.              book.setBOOKNAME(rs.getString(2));
18.              book.setBOOKPRICE(rs.getDouble(3));
19.              books.add(book);
20.          }
21.          rs.close();
22.          ps.close();
23.          conn.close();
24.          session.setAttribute("books",books);
25.          RequestDispatcher rd = request.getRequestDispatcher("/WEB-INF/jsps/show
Books.jsp");
26.          rd.forward(request,response);
27.      } catch (ClassNotFoundException e) {
28.          e.printStackTrace();
29.      } catch (SQLException e) {
30.          e.printStackTrace();
31.      }
32. }
```

showBooks.jsp 通过 session 对象接收 searchBooksByNameServlet 处理好的数据，并把接收到的这些书的信息按顺序打印给用户。如图 3 所示。

```
<!-- showBooks.jsp -->
<table border="2">
    <tr>
        <th>图书编号</th>
        <th>图书名称</th>
        <th>图书价格</th>
    </tr>
    <%
        ArrayList<Book> books = (ArrayList<Book>) session.getAttribute("books");

        if(books != null) {
            for(int i = 0;i < books.size();i++) {
     %>
     <tr>
        <td><%=books.get(i).getBOOKID()%></td>
        <td><%=books.get(i).getBOOKNAME()%></td>
        <td><%=books.get(i).getBOOKPRICE()%></td>
     </tr>
     <%
            }
        }
     %>
</table>
```

图书编号	图书名称	图书价格
10001	童话故事	32.5
10002	安徒生童话	23.8
10003	格林童话	15.8
10004	希腊神话全集	159.99

图 3. 搜索结果

2. 在第 1 题中，图书信息后面增加一个“添加到购物车”链接，单击，可以将图书添加到购物车。页面底部有一个“查看购物车”链接，可以到另一个页面中查看购物车中的内容。购物车内容显示时，后面有一个“从购物车中删除”链接，单击，又能够将该图书从购物车中删除。要求使用 DAO 和 VO，所有的动作由 Servlet 完成。

添加了“添加到购物车”、“查看购物车”两个链接后的 showBooks.jsp 如下所示。每个图书信息后面都有一个“添加到购物车”链接，单击链接可以将对应的图书添加到购物车。页面底部有一个“查看购物车”链接，可以到另一个页面

中查看购物车中的内容。如图 4 所示。

```
<!-- 修改后的 showBooks.jsp -->
<table border="2">
    <tr>
        <th>图书编号</th>
        <th>图书名称</th>
        <th>图书价格</th>
    </tr>
    <%
        ArrayList<Book> books = (ArrayList<Book>) session.getAttribute("books");
         if(books != null) {
             for(int i = 0;i < books.size();i++) {
     %>
     <tr>
         <td><%=books.get(i).getBOOKID()%></td>
         <td><%=books.get(i).getBOOKNAME()%></td>
         <td><%=books.get(i).getBOOKPRICE()%></td>
         <td><a href="addCartServlet?BOOKID=<%=books.get(i).getBOOKID()%>">添加到购物车</a> </td>
     </tr>
     <%
             }
         }
     %>
</table>
<hr>
<a href="showCart.jsp">查看购物车</a>
```

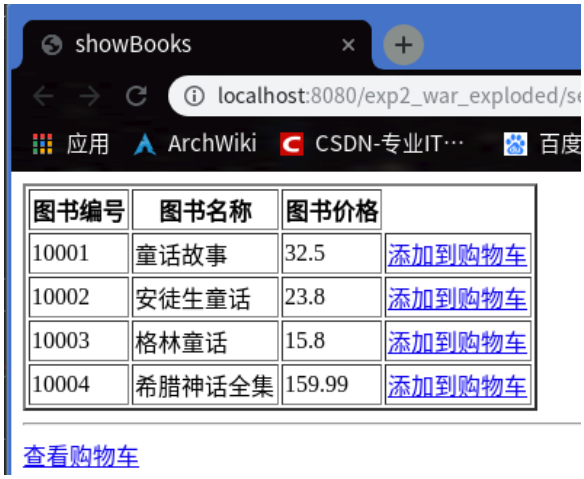

图 4. 修改后的搜索结果

addCartServlet 接收 showBooks.jsp 中传来的需要购买的书的 BOOKID，连接数据库，找到对应的 BOOKID 的书的信息，并把查到的书的信息整合到名为 carts 的 ArrayList 对象中，最后通过 session 对象传回给 showBooks.jsp 网页，至此，成功添加需要购买的书进入购物车。把安徒生童话、格林童话和希腊神话全集这三本书添加到购物车，然后查看购物车，如图 5 所示。

```
@WebServlet(name = "addCartServlet")
protected void doGet(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {
    request.setCharacterEncoding("utf-8");
```

```
4.      String BOOKID = request.getParameter("BOOKID");
5.      HttpSession session = request.getSession();
6.      ArrayList<Book> carts = (ArrayList<Book>) session.getAttribute("carts");
7.      if(carts == null) {
8.          carts = new ArrayList<>();
9.      }
10.      try {
11.          Class.forName("com.mysql.jdbc.Driver");
12.          Connection conn = DriverManager.getConnection("jdbc:mysql://localhost:3
306/books?characterEncoding=UTF-8","root","bingjie123");
13.          String sql = "select BOOKID,BOOKNAME,BOOKPRICE from T_BOOK where BOOKID
 = ?";
14.          PreparedStatement ps = conn.prepareStatement(sql);
15.          ps.setString(1,BOOKID);
16.          ResultSet rs = ps.executeQuery();
17.          while(rs.next()) {
18.              Book book = new Book();
19.              book.setBOOKID(rs.getString("BOOKID"));
20.              book.setBOOKNAME(rs.getString("BOOKNAME"));
21.              book.setBOOKPRICE(rs.getDouble("BOOKPRICE"));
22.              carts.add(book);
23.          }
24.          rs.close();
25.          ps.close();
26.          conn.close();
27.          session.setAttribute("carts",carts);
28.          RequestDispatcher rd = request.getRequestDispatcher("/WEB-INF/jsps/show
Books.jsp");
29.          rd.forward(request,response);
30.      } catch (ClassNotFoundException e) {
31.          e.printStackTrace();
32.      } catch (SQLException e) {
33.          e.printStackTrace();
34.      }
35.  }
```

showCart.jsp 是显示购物车内容的网页，该网页是通过查询 session 对象中的购物车信息来显示给用户，购物车内容显示时，每个书信息后面有一个“从购物车中删除”链接，单击链接，能够将该图书从购物车中删除。

```
1. <!-- showCart.jsp -->
2. <table border="2">
3.      <tr>
4.          <th colspan="3">购物车</th>
5.      </tr>
6.      <tr>
7.          <th>图书编号</th>
8.          <th>图书名称</th>
9.          <th>图书价格</th>
10.      </tr>
11.      <%
12.          ArrayList<Book> carts = (ArrayList<Book>) session.getAttribute("carts");
13.          if(carts != null) {
14.              for(int i = 0;i < carts.size();i++) {
15.      %>
16.      <tr>
17.          <td><%=carts.get(i).getBOOKID()%></td>
```

```
18.          <td><%=carts.get(i).getBOOKNAME()%></td>
19.          <td><%=carts.get(i).getBOOKPRICE()%></td>
20.          <td><a href="deleteCartServlet?BOOKID=<%=carts.get(i).getBOOKID()%>">从
购物车中删除</a> </td>
21.      </tr>
22.      <%
23.              }
24.          }
25.      %>
26.  </table>
```

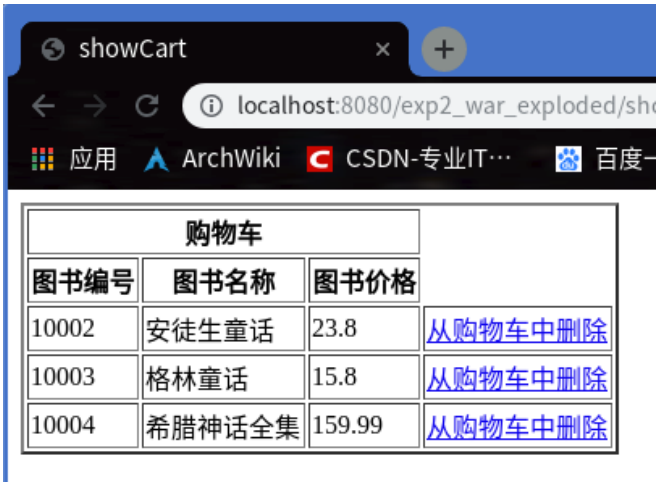

图 5. 添加一些书后的购物车

deleteCartServlet 接收 showCart.jsp 中传来的需要删除的书的 BOOKID，直接访问 JSP 内置对象 session 中 carts 对象里存储的购物车内容，找到需要删除的 BOOKID 对应的书，删除它，然后返回到 showCart.jsp 中显示删除后的购物车内容，如图 6 所示。

```
1. @WebServlet(name = "deleteCartServlet")
2. protected void doGet(HttpServletRequest request, HttpServletResponse response) th
rows ServletException, IOException {
3.      request.setCharacterEncoding("utf-8");
4.      HttpSession session = request.getSession();
5.      ArrayList<Book> carts = (ArrayList<Book>) session.getAttribute("carts");
6.      if(carts == null) {
7.          carts = new ArrayList<>();
8.      }
9.      String BOOKID = request.getParameter("BOOKID");
10.      for(int i = 0;i < carts.size();i++) {
11.          if(carts.get(i).getBOOKID().equals(BOOKID)) {
12.              carts.remove(i);
13.              break;
14.          }
15.      }
16.      session.setAttribute("carts",carts);
17.      RequestDispatcher rd = request.getRequestDispatcher("/WEB-INF/jsps/showCart
.jsp");
18.      rd.forward(request,response);
19.  }
```

图 6. 删除一些书后的购物车

3. 为网站配置欢迎页面 index.html，如果找不到，则为 index.jsp。并进行测试。

在 web.xml 中设置两个欢迎页面，第一个为 index.html，第二个为 index.jsp，新建一个 index.html 文件，内容为“welcome”，存放在/WEB-INF/htmls 目录下，重新部署网页并直接访问虚拟路径，会访问到 index.html 页面，如图 7 所示。

```
<!-- web.xml -->
<welcome-file-list>
    <welcome-file>/WEB-INF/htmls/index.html</welcome-file>
    <welcome-file>/index.jsp</welcome-file>
</welcome-file-list>
```

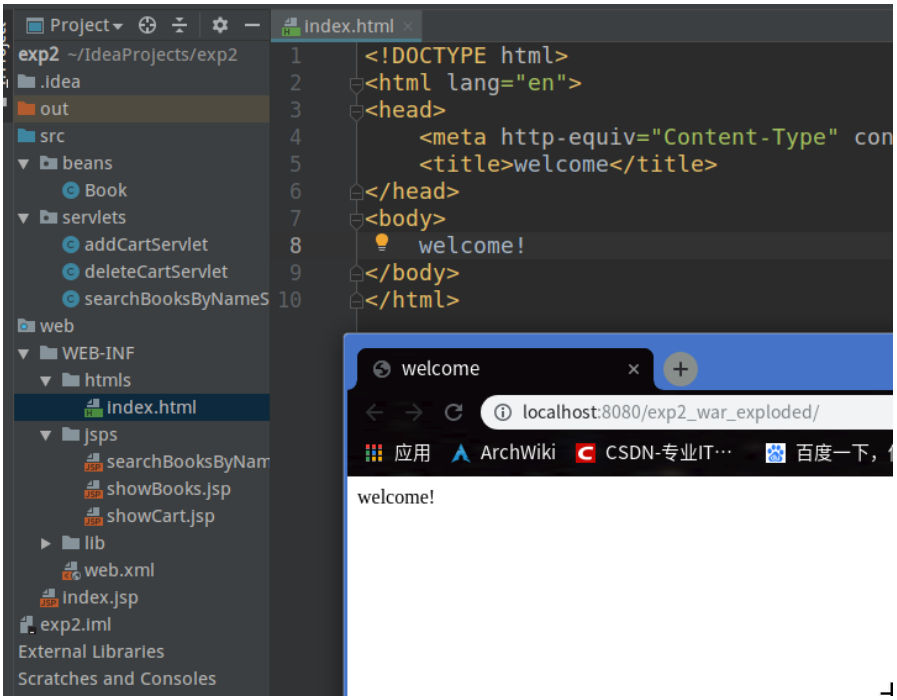

图 7. 找到 index.html 的结果

删除 htmls 目录以及其全部子目录和文件（即删除 index.html），重新部署网页，并直接访问虚拟路径，由于找不到第一个欢迎页面 index.html，所以会按顺

序往下寻找第二个欢迎页面 index.jsp，找到，访问该网页，如图 8 所示。

```
<!-- index.jsp -->
<jsp:forward page="WEB-INF/jsps/searchBooksByName.jsp"></jsp:forward>
```

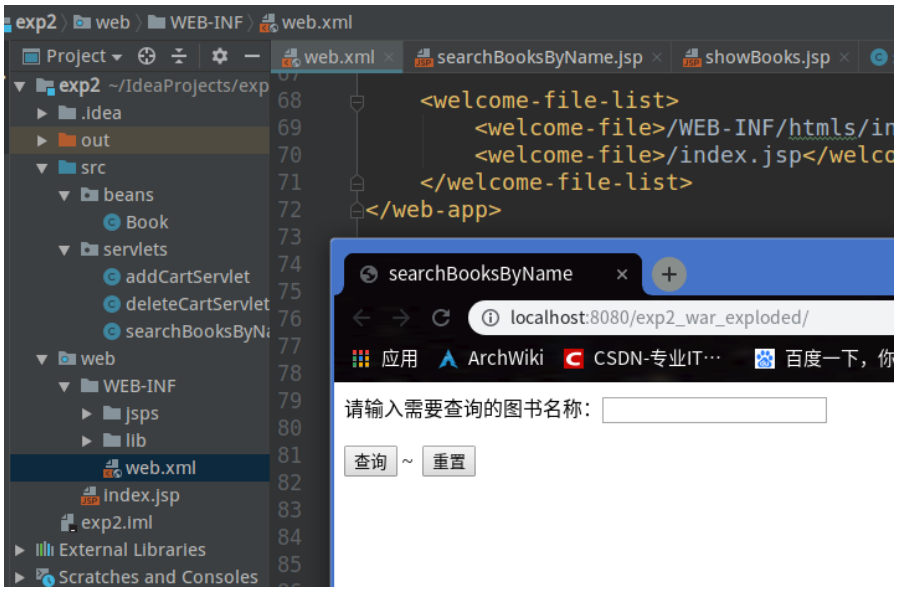

图 8. 找到 index.html 的结果

4. 图书查询过程中，需要连接数据库，将 driverClassName、url、username、password 保存在 web.xml 内，作为参数。并在 Servlet 的 init 函数中载入。

在 web.xml 中为 searchBooksByNameServlet 设置局部变量 DriverClassName、URL、UserName、Password。

```
<!-- web.xml -->
<servlet>
    <servlet-name>searchBooksByNameServlet</servlet-name>
    <servlet-class>servlets.searchBooksByNameServlet</servlet-class>
    <init-param>
        <param-name>DriverClassName</param-name>
        <param-value>com.mysql.jdbc.Driver</param-value>
    </init-param>
    <init-param>
        <param-name>URL</param-name>
        <param-value>jdbc:mysql://localhost:3306/books?characterEncoding=UTF-8</param-value>
    </init-param>
    <init-param>
        <param-name>UserName</param-name>
        <param-value>root</param-value>
    </init-param>
    <init-param>
        <param-name>Password</param-name>
        <param-value>bingjie123</param-value>
    </init-param>
</servlet>
```

在 searchBooksByNameServlet 中定义连接数据库需要的 DriverClassName、URL、UserName、Password 变量，在该 servlet 的 init()方法中，从 web.xml 中载入这四个变量的数据，载入到该 servlet 中，然后在 doPost()方法中使用载入的变量来连接 mysql 数据库。

```
@WebServlet(name = "searchBooksByNameServlet")
private String Driver;
private String URL;
private String UserName;
private String Password;
public void init() throws ServletException {
    Driver = this.getInitParameter("DriverClassName");
    URL = this.getInitParameter("URL");
    UserName = this.getInitParameter("UserName");
     Password = this.getInitParameter("Password");
     super.init();
}
protected void doPost(HttpServletRequest request, HttpServletResponse response)
throws ServletException, IOException {
    try {
        Class.forName(Driver);
        Connection conn = DriverManager.getConnection(URL,UserName,Password);
    } catch (ClassNotFoundException e) {
        e.printStackTrace();
    }
}
```

四、实验心得

本次实验，使我对 Servlet 基础编程有了一定的的理解，掌握了 Servlet 的创建和使用，以及其生命周期，并在 Servlet 中使用各种 JSP 内置对象。熟悉使用 DAO 和 VO 编程，实现层次的分开，降低耦合度，让 JSP 只作页面显示，所有操作由 servlet 完成。通过这次实验，不仅掌握了在 Java Web 项目中设置欢迎页面，还学会了在 web.xml 中为 servlet 设置局部变量和全局变量。