

课程回顾

1 Request的转发

```
1 代码实现:  
2 request.getRequestDispatcher().forward(requet,response)  
3 转发:  
4 服务器内部完成跳转, 浏览器地址栏的地址不会发生变化。  
5 浏览器在整个转发过程中, 始终只有一个请求和一个响应  
6  
7 适用场景:  
8 Request域对象存数据, 可以使用转发
```

2 Response的重定向

```
1 代码: response.sendRedirect("/web项目名称/url");  
2 //重定向实现原理  
3 response.setStatus(302);  
4 response.setHeader("location","/web项目名称/url")  
5 特点:  
6 服务器向浏览器返回302状态码, 浏览器看到302找location, 再一次向服务器发出请求, 浏览器地址  
  发生变化  
7 一次重定向会产生两个请求和两次响应。  
8  
9 适用场景:  
10 跨域访问其他项目  
11 浏览器地址发生变化的需求
```

3 Request实现数据共享, 域对象

```
1 域对象: 实现一定范围内, 数据存储, 便于其他资源进行数据获取并使用  
2 Request域对象范围: 一个请求内, 所有资源都可以访问  
3 所有的域对象, 方法统共三个:  
4 setAttribute(Object key,Object value)  
5 getAttribute(Object key):Object  
6 removeAttribute(Object key)
```

4 jsp使用

```
1 jsp是java服务器端的page, 所以jsp可以出现html css js jq Java代码  
2 html+java融合在一起, 控制起来阅读性差, 而且修改也不是很方便  
3  
4 <%%>  
5 <%=>
```

课程目标

1 图书管理系统

1-1 新增

1-2 修改

1-3 删除

4 MVC设计模式

5 EL+JSTL

课程实施

1 EL+JSTL

1-1 EL表达式

EL表达式作用：**获取**域对象中存入的数据

EL: Expression Language 表达式语言

1-2 语法

```
1  ${域对象存入值时使用key名称}
2  举例：
3  request.setAttribute("a","hello")
4  jsp:
5  ${a}
```

1-3 EL特点

```
1  语法更简洁，支持很多运算符，不再出现null
```

1-4 EL常用的运算符

运算符	说明	范例	结果
+	加	<code>\${17+5}</code>	22
-	减	<code>\${17-5}</code>	12
*	乘	<code>\${17*5}</code>	85
/或 div	除	<code>\${17/5}</code> 或 <code>\${17 div 5}</code>	3
%或 mod	取余	<code>\${17%5}</code> 或 <code>\${17 mod 5}</code>	2
==或 eq	等于	<code>\${5==5}</code> 或 <code>\${5 eq 5}</code>	true
!=或 ne	不等于	<code>\${5!=5}</code> 或 <code>\${5 ne 5}</code>	false
<或 lt	小于	<code>\${3<5}</code> 或 <code>\${3 lt 5}</code>	true
>或 gt	大于	<code>\${3>5}</code> 或 <code>\${3 gt 5}</code>	false
<=或 le	小于等于	<code>\${3<=5}</code> 或 <code>\${3 le 5}</code>	true
>=或 ge	大于等于	<code>\${3>=5}</code> 或 <code>\${3 ge 5}</code>	false
&&或 and	并且	<code>\${true&&false}</code> 或 <code>\${true and false}</code>	false
!或 not	非	<code>\${!true}</code> 或 <code>\${not true}</code>	false
或 or	或者	<code>\${true false}</code> 或 <code>\${true or false}</code>	true
empty	是否为空	<code>\${empty ""}</code> ，可以判断字符串、数据、集合的长度是否为 0，为 0 返回 true。empty 还可以与 not 或!一起使用。 <code>\${not empty ""}</code>	true

1-5 EL使用案例

使用Servlet在域对象中存入数据

```

1 package cn.kgc.servlet; /**
2  * @Author: lc
3  * @Date: 2022/5/17
4  * @Description: ${PACKAGE_NAME}
5  * @Version: 1.0
6  */
7
8 import cn.kgc.domain.BookInfo;
9 import cn.kgc.service.BookInfoService;
10 import cn.kgc.service.impl.BookInfoServiceImpl;
11
12 import javax.servlet.ServletException;
13 import javax.servlet.annotation.WebServlet;
14 import javax.servlet.http.HttpServlet;
15 import javax.servlet.http.HttpServletRequest;
16 import javax.servlet.http.HttpServletResponse;
17 import java.io.IOException;
18 import java.util.List;
19
20 @WebServlet("/BookInfoServlet")
21 public class BookInfoServlet extends HttpServlet {
22     @Override
23     protected void doGet(HttpServletRequest request, HttpServletResponse
response) throws ServletException, IOException {
24         //存 共享
25         request.setAttribute("test", "测试el表达式");
26         //转
27
28         request.getRequestDispatcher("/booklist.jsp").forward(request, response);

```

```
28     }
29 }
```

jsp上使用EL获取数据

```
1  <%@ page contentType="text/html; charset=UTF-8" language="java" %>
2  <html>
3  <head>
4      <title>图书列表页</title>
5  </head>
6  <body>
7      <!--使用el获取request域存入的数据
8      el获取数据，没有null
9      el数据获取和显示，没有任何逻辑（if switch while for...）
10
11
12     <!-- 获取request域中test键对应的值 -->
13     ${test}
14     <!-- 获取request域中list键对应的值 -->
15     ${list}
16 </body>
17 </html>
18
```

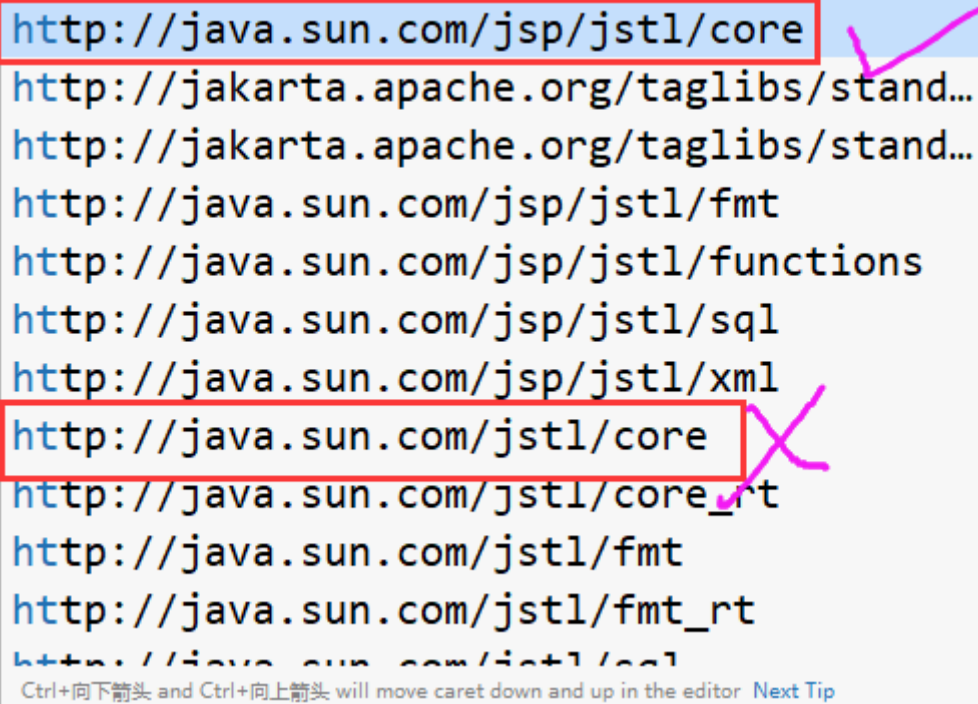
1-2 JSTL使用

JSTL:Java Standard Tag Lib java提供一套标准标签库

JSTL作用：提供jsp使用标签实现java代码的逻辑控制

1-2-1 JSTL使用步骤

1. 项目导入jstl jar包
2. jsp页面使用指令<%@taglib prefix="" uri="" %>
3. 在jsp里面使用定义的标签名称，控制代码即可



<http://java.sun.com/jsp/jstl/core>
<http://jakarta.apache.org/taglibs/stand...>
<http://jakarta.apache.org/taglibs/stand...>
<http://java.sun.com/jsp/jstl/fmt>
<http://java.sun.com/jsp/jstl/functions>
<http://java.sun.com/jsp/jstl/sql>
<http://java.sun.com/jsp/jstl/xml>
<http://java.sun.com/jstl/core>
http://java.sun.com/jstl/core_rt
<http://java.sun.com/jstl/fmt>
http://java.sun.com/jstl/fmt_rt
<http://java.sun.com/jstl/sql>

Ctrl+向下箭头 and Ctrl+向上箭头 will move caret down and up in the editor Next Tip

1-2-1 JSTL常用流程控制语句的案例演示

```
1 <!--使用el获取request域存入的数据
2 el获取数据，没有null
3 el数据获取和显示，没有任何逻辑（if switch while for...）
4 --%>
5 ${test}
6 <!--${list}--%>
7 <!--
8 JSTL：书写规范遵循html标签的书写规范
9 --%>
10 <!-- if语法
11 if(12>0){
12     HelloWorld
13 }
14 choose语句
15 if(){
16 }
17 else{
18 }
19 }
20
21 if(){
22
23 }else if(){
24
25 }else if(){
26
27 }
28 --%>
29 <c:if test="${12>0}">
30     HelloWorld
31 </c:if>
32
33 <c:choose>
34 <!-- if(1==1){
```

```

35
36 }else{
37     1!=1
38 }
39 --%>
40     <c:when test="${1==1}">
41         1==1
42     </c:when>
43 <!--     else if() --%>
44     <c:when test="${1>1}">
45         1>1
46     </c:when>
47 <!--     else{} --%>
48     <c:otherwise>
49         1!=1
50     </c:otherwise>
51 </c:choose>
52 <!--
53 for(int a=1;a<=10;i++){
54 循环体
55 }
56 --%>
57 <c:forEach var="a" begin="1" end="10" step="1">
58     ${a}
59 </c:forEach>
60 <!--
61 for(Object book:集合或数组)
62 --%>
63 <c:forEach var="book" items="${list}">
64     ${book}<br>
65 </c:forEach>

```

1-3 JSLT+EL完成图书信息的列表显示

编号	图书编号	图书名称	图书分类	图书作者	出版社	出版日期	借阅状态	查看详情	操作
1	GBK_001	格林童话	格林	少儿	人民教育出版社	2021-07-24 11:12:15.0	已借阅	查看详情	删除 修改
2	GBK_002	小王子	Jerry	少儿	人民教育出版社	2021-07-24 11:12:15.0	已借阅	查看详情	删除 修改
3	GBK_003	最好的朋友	crystal	少儿	人民教育出版社	2022-05-18 10:11:39.0	未借阅	查看详情	删除 修改
4	GBK_004	小猪佩奇	Peggy	少儿	人民教育出版社	2021-07-24 11:12:15.0	已借阅	查看详情	删除 修改
5	XS_001	飘a飘	了了	文学	清华大学出版社	2021-07-24 00:00:00.0	已借阅	查看详情	删除 修改
6	WX_001	曾有一个我爱我如生命	顾漫	玄幻	人民日报出版社	2022-05-18 10:11:44.0	未借阅	查看详情	删除 修改
7	XS_002	杉杉来了	顾漫	文学	人民日报出版社	2021-07-24 11:12:15.0	已借阅	查看详情	删除 修改
8	YS_001	地下铁	老铁	传记	人民日报出版社	2021-07-24 11:12:15.0	已借阅	查看详情	删除 修改
9	XS_003	钢铁侠	霍果仁	文学	武汉铁路出版社	2021-07-24 11:12:15.0	已借阅	查看详情	删除 修改
18	SE_001	小情绪大情感	hahaha	少儿	heiheihei	2021-07-24 11:12:15.0	已借阅	查看详情	删除 修改
19	SE_002	小情绪大情感23	fdsf	少儿	fdsf	2021-07-24 11:12:15.0	已借阅	查看详情	删除 修改
21	SE_002	借我一段光明	不知道	经济	北大出版社	2021-07-24 11:12:15.0	已借阅	查看详情	删除 修改

参考jsp代码

```

1 <%@ page contentType="text/html;charset=UTF-8" language="java" %>
2 <!--导入jstl依赖
3 prefix设置值 自定义，一般建议以导入库单词首字母
4 uri: 使用标签官方提供的一个访问路径
5 --%>
6 <%@ taglib prefix="c" uri="http://java.sun.com/jsp/jstl/core" %>
7 <html>
8 <head>
9     <title>Title</title>
10 </head>

```

```

11 <body>
12 <!-- dreamweaver --%>
13 <table width="100%" border="1" cellspacing="0" cellpadding="0">
14     <caption>
15         图书信息一览表
16     </caption>
17     <tr>
18         <th>编号</th>
19         <th>图书编号</th>
20         <th>图书名称</th>
21         <th>图书分类</th>
22         <th>图书作者</th>
23         <th>出版社</th>
24         <th>出版日期</th>
25         <th>借阅状态</th>
26         <th>查看详情</th>
27         <th>操作</th>
28     </tr>
29     <c:forEach items="${list}" var="book">
30         <tr>
31             <!-- e1 使用属性名必须与类中定义属性名大小写一样 --%>
32             <td>${book.id}</td>
33             <td>${book.book_code}</td>
34             <td>${book.book_name}</td>
35             <td>${book.book_author}</td>
36             <td>${book.type_name}</td>
37             <td>${book.publish_press}</td>
38             <td>${book.publish_date}</td>
39             <td>${book.is_borrow?"已借阅":"未借阅"}</td>
40             <td><a href="">查看详情</a></td>
41             <td><a href="">删除</a><a href="">修改</a></td>
42         </tr>
43     </c:forEach>
44 </table>
45 </body>
46 </html>

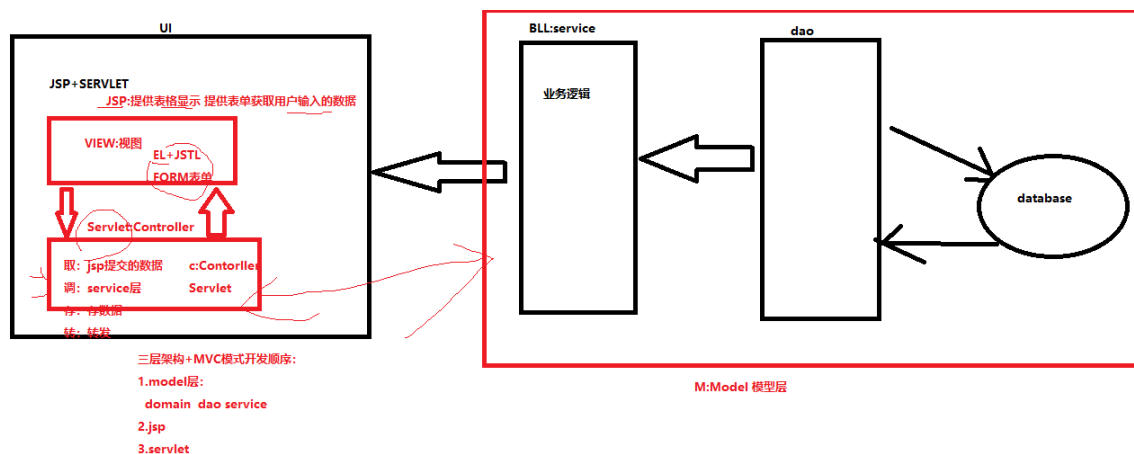
```

2 MVC设计模式

用在JavaWeb项目中，不是三层架构。

2-1 MVC设计模式优点

JSP Model2适合多人合作开发大型的Web项目，各司其职，互不干涉，有利于开发中的分工，有利于组件的重用。但是，Web项目的开发难度加大，同时对开发人员的技术要求也提高了。



3 基于MVC改造图书关系的页面显示方式

3-1 修改web.xml设置默认欢迎页

```

1 <?xml version="1.0" encoding="UTF-8"?>
2 <web-app xmlns="http://java.sun.com/xml/ns/javaee"
3         xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
4         xsi:schemaLocation="http://java.sun.com/xml/ns/javaee
5         http://java.sun.com/xml/ns/javaee/web-app_2_5.xsd"
6         version="2.5">
7     <!-- 配置当前项目，默认显示页面-->
8     <welcome-file-list>
9 <!--
10     浏览器输入网址: http://localhost:8080/, 访问资源会冲下面的welcome-file自动匹配
11 -->
12     <welcome-file>index.jsp</welcome-file>
13     <welcome-file>main.html</welcome-file>
14     <welcome-file>index.html</welcome-file>
15 </welcome-file-list>
16 </web-app>

```

3-2 添加index.jsp, 使用jsp指令实现页面请求的转发

```

1 <%--
2     Created by IntelliJ IDEA.
3     User: Administrator
4     Date: 2022/5/18
5     Time: 11:08
6     To change this template use File | Settings | File Templates.
7 --%>
8 <%@ page contentType="text/html; charset=UTF-8" language="java" %>
9 <html>
10 <head>
11     <title>图书管理系统</title>
12 </head>
13 <body>
14 <%-- 想看所有的数据的请求，偷偷转发给控制器--%>
15 <%-- JSP指令
16     request.getReq(Url地址).forward()
17     jsp:forward page="servlet的url">
18 --%>
19 <jsp:forward page="/BookInfoServlet"></jsp:forward>

```



```
20 </body>
21 </html>
```

3-3 启动服务器，在浏览器地址输入项目网址即可看到图书信息

```
1 http://localhost:8080/回车即可看到图书列表
```

图书信息一览表									
编号	图书编号	图书名称	图书分类	图书作者	出版社	出版日期	借阅状态	查看详情	操作
1	GBK_001	格林童话	格林	少儿	人民教育出版社	2021-07-24 11:12:15.0	已借阅	查看详情	删除修改
2	GBK_002	小王子	Jerry	少儿	人民教育出版社	2021-07-24 11:12:15.0	已借阅	查看详情	删除修改
3	GBK_003	最好的朋友	crystal	少儿	人民教育出版社	2022-05-18 10:11:39.0	未借阅	查看详情	删除修改
4	GBK_004	小猪佩奇	Peggy	少儿	人民教育出版社	2021-07-24 11:12:15.0	已借阅	查看详情	删除修改
5	XS_001	飘a飘	了了	文学	清华大学出版社	2021-07-24 00:00:00.0	已借阅	查看详情	删除修改
6	WX_001	曾有一个爱我如生命	顾漫	玄幻	人民日报出版社	2022-05-18 10:11:44.0	未借阅	查看详情	删除修改
7	XS_002	杉杉来了	顾漫	文学	人民日报出版社	2021-07-24 11:12:15.0	已借阅	查看详情	删除修改
8	YS_001	地下铁	老铁	传记	人民日报出版社	2021-07-24 11:12:15.0	已借阅	查看详情	删除修改
9	XS_003	钢铁侠	霍果仁	文学	武汉铁路出版社	2021-07-24 11:12:15.0	已借阅	查看详情	删除修改
18	SE_001	小情绪大情感	hahaha	少儿	heiheihei	2021-07-24 11:12:15.0	已借阅	查看详情	删除修改
19	SE_002	小情绪大情感23	fdsf	少儿	fdsf	2021-07-24 11:12:15.0	已借阅	查看详情	删除修改
21	SE_002	借我一段光明	不知道	经济	北大出版社	2021-07-24 11:12:15.0	已借阅	查看详情	删除修改

4 图书管理系统

4-1 删除

domain层のBookInfo

```
1 package cn.kgc.domain;
2
3 import java.util.Date;
4
5 /**
6  * @Author: lc
7  * @Date: 2022/5/17
8  * @Description: 实体类列名和列类型与select查询结果集类型、列名一致
9  * @Version: 1.0
10 */
11 public class BookInfo {
12     private Integer id;//数据库sql起别名
13     private String book_code;
14     private String book_name;
15     private String book_author;
16     private Integer book_type;
17     private String publish_press;
18     private Date publish_date;
19     private boolean is_borrow;
20
21     //连接查询，需要补充主表的信息
22     private String type_name;
23
24     public Integer getId() {
25         return id;
26     }
27
28     public void setId(Integer id) {
29         this.id = id;
30     }
31
32     public String getBook_code() {
33         return book_code;
```

```
34     }
35
36     public void setBook_code(String book_code) {
37         this.book_code = book_code;
38     }
39
40     public String getBook_name() {
41         return book_name;
42     }
43
44     public void setBook_name(String book_name) {
45         this.book_name = book_name;
46     }
47
48     public String getBook_author() {
49         return book_author;
50     }
51
52     public void setBook_author(String book_author) {
53         this.book_author = book_author;
54     }
55
56     public Integer getBook_type() {
57         return book_type;
58     }
59
60     public void setBook_type(Integer book_type) {
61         this.book_type = book_type;
62     }
63
64     public String getPublish_press() {
65         return publish_press;
66     }
67
68     public void setPublish_press(String publish_press) {
69         this.publish_press = publish_press;
70     }
71
72     public Date getPublish_date() {
73         return publish_date;
74     }
75
76     public void setPublish_date(Date publish_date) {
77         this.publish_date = publish_date;
78     }
79
80     public boolean isIs_borrow() {
81         return is_borrow;
82     }
83
84     public void setIs_borrow(boolean is_borrow) {
85         this.is_borrow = is_borrow;
86     }
87
88     public String getType_name() {
89         return type_name;
90     }
91
```

```

92     public void setType_name(String type_name) {
93         this.type_name = type_name;
94     }
95
96     @Override
97     public String toString() {
98         final StringBuilder sb = new StringBuilder("BookInfo{");
99         sb.append("id=").append(id);
100        sb.append(", book_code=").append(book_code).append('\n');
101        sb.append(", book_name=").append(book_name).append('\n');
102        sb.append(", book_author=").append(book_author).append('\n');
103        sb.append(", book_type=").append(book_type);
104        sb.append(", publish_press=").append(publish_press).append('\n');
105        sb.append(", publish_date=").append(publish_date);
106        sb.append(", is_borrow=").append(is_borrow);
107        sb.append(", type_name=").append(type_name).append('\n');
108        sb.append('}');
109        return sb.toString();
110    }
111 }

```

dao层のBookInfoDao

```

1  package cn.kgc.dao;
2
3  import cn.kgc.domain.BookInfo;
4
5  import java.util.List;
6
7  /**
8   * @Author: lc
9   * @Date: 2022/5/17
10  * @Description: 查询、修改、删除和新增
11  * @Version: 1.0
12  */
13  public interface BookInfoDao {
14      /**
15       * 查询所有的图书信息
16       * @return
17       */
18      List<BookInfo> selectAll();
19
20      /**
21       * 新增图书信息
22       * @param book
23       * @return
24       */
25      int insert(BookInfo book);
26      int update(BookInfo book);
27
28      /**
29       * 删除一条或多条数据
30       * @param pkIds
31       * @return
32       */
33      int delete(Integer pkIds);
34  }

```

dao层のBookInfoDaoImpl

```
1 package cn.kgc.dao.impl;
2
3 import cn.kgc.dao.BookInfoDao;
4 import cn.kgc.domain.BookInfo;
5 import cn.kgc.util.JDBCUtil;
6 import org.apache.commons.dbutils.QueryRunner;
7 import org.apache.commons.dbutils.handlers.BeanListHandler;
8
9 import java.sql.SQLException;
10 import java.util.List;
11
12 /**
13  * @Author: lc
14  * @Date: 2022/5/17
15  * @Description: cn.kgc.dao.impl
16  * @Version: 1.0
17  */
18 public class BookInfoDaoImpl implements BookInfoDao {
19     private QueryRunner qr=new QueryRunner(JDBCUtil.datasource);
20     @Override
21     public List<BookInfo> selectAll() {
22         StringBuilder sb=new StringBuilder();
23         sb.append("SELECT book_info.book_id id,book_info.book_code,");
24         sb.append("book_info.book_name,book_info.book_type,");
25         sb.append("book_info.book_author,book_info.publish_press,");
26         sb.append("book_info.publish_date,book_info.is_borrow,");
27         sb.append("book_type.type_name FROM book_info ");
28         sb.append("LEFT JOIN book_type ON
29 book_info.book_type=book_type.id");
30         //拼接查询条件
31
32         //拼接排序
33
34         //拼接limit 分页
35
36         try {
37             return qr.query(sb.toString(),new BeanListHandler<>
38 (BookInfo.class));
39         } catch (SQLException e) {
40             throw new RuntimeException(e);
41         }
42     }
43     @Override
44     public int insert(BookInfo book) {
45         return 0;
46     }
47     @Override
48     public int update(BookInfo book) {
49         return 0;
50     }
51
52     @Override
```

```

53     public int delete(Integer pkIds) {
54         /*StringBuilder sb=new StringBuilder();
55         sb.append("delete from book_info where book_id in (");
56         for(Integer id:pkIds) {
57             sb.append("?,");
58         }
59         //最后一个? 不需要,, 去除
60         sb.deleteCharAt(sb.length()-1);
61         sb.append(")");*/
62         try {
63             return qr.update("delete from book_info where book_id=?",pkIds);
64         } catch (SQLException e) {
65             throw new RuntimeException(e);
66         }
67     }
68 }

```

service层のBookInfoService

```

1  package cn.kgc.service;
2
3  import cn.kgc.domain.BookInfo;
4
5  import java.util.List;
6
7  /**
8   * @Author: lc
9   * @Date: 2022/5/17
10  * @Description: cn.kgc.service
11  * @Version: 1.0
12  */
13  public interface BookInfoService {
14      /**
15       * 获取所有的图书信息
16       * @return
17       */
18      List<BookInfo> getAll();
19
20      /**
21       * 添加图书信息
22       * @param book
23       * @return
24       */
25      int add(BookInfo book);
26      int modify(BookInfo book);
27
28      /**
29       * 删除一条或多条数据
30       * @param pkIds
31       * @return
32       */
33      int remove(Integer pkIds);
34  }

```

service层のBookInfoServiceImpl

```
1 package cn.kgc.service.impl;
2
3 import cn.kgc.dao.BookInfoDao;
4 import cn.kgc.dao.impl.BookInfoDaoImpl;
5 import cn.kgc.domain.BookInfo;
6 import cn.kgc.service.BookInfoService;
7
8 import java.util.List;
9
10 /**
11  * @Author: lc
12  * @Date: 2022/5/17
13  * @Description: cn.kgc.service.impl
14  * @Version: 1.0
15  */
16 public class BookInfoServiceImpl implements BookInfoService {
17     private BookInfoDao dao=new BookInfoDaoImpl();
18     @Override
19     public List<BookInfo> getAll() {
20         return dao.selectAll();
21     }
22
23     @Override
24     public int add(BookInfo book) {
25         return dao.insert(book);
26     }
27
28     @Override
29     public int modify(BookInfo book) {
30         return dao.update(book);
31     }
32
33     @Override
34     public int remove(Integer pkIds) {
35         return dao.delete(pkIds);
36     }
37 }
```

view层のJSP页面设计

```
1 <%@ page contentType="text/html; charset=UTF-8" language="java" %>
2 <!-- 导入jstl依赖
3 prefix设置值 自定义，一般建议以导入库单词首字母
4 uri: 使用标签官方提供的一个访问路径
5 --%>
6 <%@ taglib prefix="c" uri="http://java.sun.com/jsp/jstl/core" %>
7 <html>
8 <head>
9     <title>Title</title>
10 </head>
11 <body>
12 <!-- dreamweaver --%>
13 <table width="100%" border="1" cellspacing="0" cellpadding="0">
14     <caption>
```

[illegible]

controller层のBookInfoController

```
1 package cn.kgc.controller; /**
2  * @Author: lc
3  * @Date: 2022/5/18
4  * @Description: ${PACKAGE_NAME}
5  * @Version: 1.0
6  */
7
8 import cn.kgc.service.BookInfoService;
9 import cn.kgc.service.impl.BookInfoServiceImpl;
10
11 import javax.servlet.*;
12 import javax.servlet.http.*;
13 import javax.servlet.annotation.*;
14 import java.io.IOException;
15 import java.io.PrintWriter;
16
17 @WebServlet("/RemoveServlet")
18 public class RemoveServlet extends HttpServlet {
19     @Override
20     protected void doGet(HttpServletRequest request, HttpServletResponse
response) throws ServletException, IOException {
21         //取 提交数据Servlet?? 需要!! 删除图书的主键ID
22         //需要吗? 就看调用service有没有参数, service提供方法有形参
23         String idStr = request.getParameter("id");
24         //请求协议发送来的数据, 获取默认类型都是string
25         //String==>int/Integer
26         Integer id = Integer.valueOf(idStr);
27         //调
28         BookInfoService service =new BookInfoServiceImpl();
29         //row是delete执行成功之后受影响行数
30         int row = service.remove(id);
31         //row==>删除成功或删除失败
32         response.setContentType("text/html;charset=utf-8");
33         PrintWriter writer = response.getWriter();
34         //封装一段响应体: js
35         writer.print("<script type='text/javascript'>alert('"+(row==1?"删除成
功":"删除失败")+"'");location.href='/index.jsp';</script>");
36     }
37
38     @Override
39     protected void doPost(HttpServletRequest request, HttpServletResponse
response) throws ServletException, IOException {
40         doGet(request, response);
41     }
42 }
```

课程总结

1 EL+JSTL

2 MVC处理请求的流程:

浏览器发请求给jsp，jsp将数据收集之后发送控制器eServlet,Servlet调用service处理结果，结果转发个
ijsp显示

3 删除图书信息
