# 课程回顾

CRUD基本要求: 完成两表的连接查询、单表的添加、修改和删除

```
1 1 多条件模糊查询
2 1-1 dao层sQL拼接方式
3 sql语句尾部if+append()完成sql条件
4 query(sql,,list.toArray())
5 1-2 多条件回显问题:
6 直询按钮,查询结果,还要将查询条件再一次转发jsp显示
7 <select>
9 </select>
10 </select>
2 分页
```

# 课程目标

- 1 分页
- 2 会话技术
- 3 Cookie
- 4 Session

# 课程实施

## 1 分页

## 1-1 Page类

理解page类作用, Page类SSM提供了

1 作用: Page类封装整个分页查询的结果,将结果转发给jsp显示

```
package cn.kgc.domain;
3 import java.util.List;
4
   /**
    * @Author: lc
6
   * @Date: 2022/5/21
    * @Description: 保存jsp需要显示分页信息
8
9
   * @version: 1.0
10
   public class Page<T> { //new Page<BookInfo>
11
12
13
       * 分页后需要显示的查询结果集合信息
14
      * 即jsp表格需要显式图书信息
```

```
15
16
        private List<T> list;
17
        //总页数 字段 数据库列名称为字段
18
        //private Integer totalPages;
19
        //查询结果总行数
20
        private Integer totalResults;
21
        //页大小
22
        private Integer pageSize;
23
        //页码,当前显示的页码
24
        private Integer pageNum;
25
        //是否有下一页
26
        //private boolean hasNextPage;
27
        //是否有上一页
        //private boolean hasPrevPage;
28
29
        //属性: getter 只读属性
30
31
        public List<T> getList() {
32
            return list;
33
34
        //属性: setter 只写属性
        public void setList(List<T> list) {
35
36
            this.list = list;
37
        }
        // el读取数据${totalPages}
38
39
        public Integer getTotalPages() {
            //举例 13/3==>总页码: 4+1 15/3==>总页码5页
40
41
            return getTotalResults()%getPageSize()!=0?
    getTotalResults()/getPageSize()+1:getTotalResults()/getPageSize();
42
43
        /*public void setTotalPages(Integer totalPages) {
44
45
            this.totalPages = totalPages;
        }*/
46
47
48
        public Integer getTotalResults() {
49
            return totalResults;
50
        }
51
        public void setTotalResults(Integer totalResults) {
52
53
            this.totalResults = totalResults;
54
        }
55
56
        public Integer getPageSize() {
57
            return pageSize;
58
        }
59
60
        public void setPageSize(Integer pageSize) {
61
            this.pageSize = pageSize;
        }
62
63
64
        public Integer getPageNum() {
65
            return pageNum;
66
        }
67
68
        public void setPageNum(Integer pageNum) {
69
            this.pageNum = pageNum;
70
        }
71
```

```
72
        public boolean isHasNextPage() {
73
             return getPageNum()!=getTotalPages();
74
        }
75
76
        //public void setHasNextPage(boolean hasNextPage) {
77
        // this.hasNextPage = hasNextPage;
78
        //}
79
        public boolean isHasPrevPage() {
80
81
             return getPageNum()!=1;
82
        }
83
84
        //public void setHasPrevPage(boolean hasPrevPage) {
        // this.hasPrevPage = hasPrevPage;
85
86
        //}
87
    }
```

#### 1-2 Servlet针对查询结果实现分页

#### dao层BookInfoDaoImpl完成分页sql的实现

```
1
    @override
        public List<BookInfo> selectAll(Integer bookType, String bookName,
2
    Integer isBorrow,
3
                                        Integer pageNum,Integer pageSize) {
4
            //执行SQL语句
5
            StringBuilder sb=new StringBuilder();
6
            sb.append(" SELECT book_info.book_id id,book_info.book_code, ");
            sb.append(" book_info.book_name,book_info.book_type, ");
8
            sb.append(" book_info.book_author,book_info.publish_press, ");
9
            sb.append(" book_info.publish_date,book_info.is_borrow, ");
            sb.append(" book_type.type_name FROM book_info ");
10
11
            sb.append(" LEFT JOIN book_type ON book_info.book_type=book_type.id
    ");
12
            //where 1=1作用: 条件的拼接关键字and设置进去
            sb.append(" where 1=1 ");
13
14
            //String判断两个值: null ""
            //将所有的?对应的实参,使用集合保存
15
16
            List params=new ArrayList();
            if(!Objects.isNull(bookType)){
17
                sb.append(" AND book_info.book_type=? ");
18
19
                params.add(bookType);
20
21
            if(!Objects.isNull(bookName)&&!bookName.isEmpty()){
                sb.append(" AND book_info.book_name LIKE ? ");
22
23
                params.add("%"+bookName+"%");
24
25
            if(!Objects.isNull(isBorrow)){
26
                sb.append(" AND book_info.is_borrow=? ");
27
                params.add(isBorrow);
28
29
            //添加分页语句
            sb.append(" LIMIT ?,? ");
30
31
            //limit语句第一个? 对应的实参
32
            params.add((pageNum-1)*pageSize);
33
            //limit语句第二个?对应的实参
34
            params.add(pageSize);
```

```
35
           //qr.query(String 要执行的sql语句,
36
           // BeanHandler BeanListHandler ScalarHandler 转换的类型,
           // Object... sql语句?对应的实参)
37
38
           try {
39
               return qr.query(sb.toString(),
40
                       new BeanListHandler<>(BookInfo.class),
41
                       // Object... 底层就是数组,所以集合必须转换为数组,底层才能正常解
    析
                       params.toArray());
42
43
           } catch (SQLException e) {
               throw new RuntimeException(e);
44
45
           }
46
        }
```

#### dao层BookInfoDaoImpl处理count()获取查询结果总行数

```
@override
1
2
        public Object selectCount(Integer bookType, String bookName, Integer
    isBorrow) {
3
            //执行SQL语句
4
            StringBuilder sb=new StringBuilder();
5
            sb.append(" SELECT count(*) fROM book_info ");
            sb.append(" LEFT JOIN book_type ON book_info.book_type=book_type.id
6
    ");
7
            //where 1=1作用:条件的拼接关键字and设置进去
            sb.append(" where 1=1 ");
8
9
            //String判断两个值: null ""
            //将所有的?对应的实参,使用集合保存
10
11
            List params=new ArrayList();
            if(!Objects.isNull(bookType)){
12
                sb.append(" AND book_info.book_type=? ");
13
                params.add(bookType);
14
15
            if(!Objects.isNull(bookName)&&!bookName.isEmpty()){
16
                sb.append(" AND book_info.book_name LIKE ? ");
17
                params.add("%"+bookName+"%");
18
19
            if(!Objects.isNull(isBorrow)){
20
                sb.append(" AND book_info.is_borrow=? ");
21
22
                params.add(isBorrow);
23
            }
24
            //qr.query(String 要执行的sql语句,
25
26
            // BeanHandler BeanListHandler ScalarHandler 转换的类型,
            // Object... sql语句?对应的实参)
27
28
            try {
29
                return qr.query(sb.toString(),
30
                        new ScalarHandler<>(),
31
                        // Object... 底层就是数组,所以集合必须转换为数组,底层才能正常解
    析
32
                        params.toArray());
33
            } catch (SQLException e) {
                throw new RuntimeException(e);
34
35
           }
36
        }
```

#### servlet层BookInfoServlet完成将service返回的查询结果,封装成Page类对象

```
1
    package cn.kgc.controller; /**
 2
     * @Author: 1c
 3
     * @Date: 2022/5/17
 4
     * @Description: ${PACKAGE_NAME}
     * @version: 1.0
 5
 6
     */
 7
 8
    import cn.kgc.domain.BookInfo;
9
    import cn.kgc.domain.BookType;
10
    import cn.kgc.domain.Page;
    import cn.kgc.service.BookInfoService;
11
    import cn.kgc.service.impl.BookInfoServiceImpl;
12
13
    import cn.kgc.service.impl.BookTypeServiceImpl;
14
15
    import javax.servlet.ServletException;
    import javax.servlet.annotation.WebServlet;
16
17
    import javax.servlet.http.HttpServlet;
18
    import javax.servlet.http.HttpServletRequest;
19
    import javax.servlet.http.HttpServletResponse;
20
    import java.io.IOException;
    import java.util.List;
21
22
    @webServlet("/BookInfoServlet")
23
24
    public class BookInfoServlet extends HttpServlet {
25
        @Override
26
        protected void doGet(HttpServletRequest request, HttpServletResponse
    response) throws ServletException, IOException {
27
            //请求发送的搜索条件中,图书名称可能是中文,GET必须处理中文
28
            //tomcat8 9 10针对get没有乱码
29
            //取 请求体或url地址传输的表单数据?略
30
            String bookTypeStr = request.getParameter("bookType");//请求没有数据
    时, bookTypeStr null
31
            String bookName = request.getParameter("bookName");//bookName null
            String pageNumStr=request.getParameter("pageNum");
32
33
            Integer pageSize=3;
            //解决get请求乱码问题 trim():去除字符串首尾的无效空格
34
35
            if (bookName!=null && !bookName.trim().isEmpty()) {
                bookName=new String(bookName.getBytes("iso8859-1"),"utf-8");
36
37
            String isBorrowStr = request.getParameter("isBorrow");
38
39
40
            //类型转换
41
            Integer pageNum=1;
42
            try {
                //valueOf(null)不能正常转换
43
44
                pageNum=Integer.valueOf(pageNumStr);
45
            } catch (Exception e) { }
46
            Integer bookType=null;
47
            try {
                //valueOf(null)不能正常转换
48
49
                bookType=Integer.valueOf(bookTypeStr);
50
            } catch (Exception e) { }
51
52
            Integer isBorrow=null;
53
            try {
```

```
//valueOf(null)不能正常转换
54
55
               isBorrow=Integer.valueOf(isBorrowStr);
           } catch (Exception e) { }
56
57
           //调 service层
58
           BookInfoService service = new BookInfoServiceImpl();
59
           //思考:查询按钮提交请求,是不是可以使用这个Servlet来处理??
60
           List<BookInfo> books =
    service.getAll(bookType,bookName,isBorrow,pageNum,pageSize);
           //查询结果封装成Page对象
61
62
           Page<BookInfo> page=new Page<>();
           //设置属性,以便jsp显示
63
           page.setList(books);
64
65
           page.setPageNum(pageNum);//1
           page.setPageSize(pageSize);//3
66
67
           //books是分页查询结果集合,books.size作为总行数使用吗?不行
           //查询结果的总行数: dao+ select count(*) from 。。。。
68
           Integer count = new BookInfoServiceImpl().getCount(bookType,
69
    bookName, isBorrow);
70
           page.setTotalResults(count);//3
71
72
           //查询图书分类信息
73
           List<BookType> types = new BookTypeServiceImpl().getAllBookTypes();
           //存 共享
74
           //request.setAttribute("list",books);//显示图书信息
75
76
           request.setAttribute("page",page);
77
           request.setAttribute("types", types);//显示图书分类信息
78
           //为了实现查询条件的回显, 所以要将所有查询条件存入域
           //查询条件封装成一个类对象,一次性存入转发给jsp
79
80
           request.setAttribute("type",bookType);
81
           request.setAttribute("bookName",bookName);
82
           request.setAttribute("borrow",isBorrow);
83
           //转
84
           //jsp本质就是一个servlet
85
     request.getRequestDispatcher("/booklist.jsp").forward(request, response);
86
       }
87
88
       @override
89
       protected void doPost(HttpServletRequest request, HttpServletResponse
    response) throws ServletException, IOException {
90
           doGet(request, response);
91
        }
92
    }
```

#### booklist.jsp实现分页

```
1
   <%@ page contentType="text/html;charset=UTF-8" language="java" %>
2
   <%--导入jst1依赖
   prefix设置值 自定义,一般建议以导入库单词首字母
   uri: 使用标签官方提供的一个访问路径
   --%>
   <%@ taglib prefix="c" uri="http://java.sun.com/jsp/jstl/core" %>
 6
7
   <html>
8
   <head>
9
       <title>Title</title>
   </head>
10
11
    <script type="text/javascript">
```

```
12
       /**
13
        * 实现分页+多条件查询
14
        * @param pn表示查询条件下查询页码
        */
15
16
       function doSearch(pn){
17
           //单击查询按钮,向BookInfoServlet发出请求
18
           //发出请求时,需要带着查询条件过去
           //form走POST 地址栏地址能变,就是发请求统一qet
19
20
           // window.open();
21
           //js支持EL
           //定义字符串,保存搜索条件
22
23
           var query="bookType="+document.getElementById("type").value;//?应该
    是用户实际输入的数据
24
           query+="&bookName="+document.getElementById("book_name").value;
25
           query+="&isBorrow="+document.getElementById("borrow").value;
           //分页需要的参数pageNum
26
27
           if(!pn){//go按钮,进行分页,页码是读取的文本框用户输入的值
28
              pn=document.getElementById('txtPageNo').value
29
           }
           //对于用户输入的非法值,做一下合法性校验
30
           if(pn<1 || pn>${page.totalPages}){
31
32
              pn=1;
33
           }
34
           query+="%pageNum="+pn;
35
    location.href="${pageContext.servletContext.contextPath}/BookInfoServlet?"
   +query;
36
           // location.assign();
37
       }
38
   </script>
39
   <body>
40
    41
       42
43
              <h2 align="center">图书借阅系统</h2>
              <!--<p align="center">-->
44
45
              图书分类:
              <select name="book_type" id="type">
46
47
                  <option value="">全部</option>
48
                         从数据库查询的图书分类 --%>
                  <c:forEach items="${types}" var="t">
49
                      <%-- 要求: value指定主键 innerText:列值 汉字 --%>
50
51
                      <option value="${t.id}"</pre>
52
                             <c:if test="${type==t.id}">
53
                             selected
                          </c:if>
54
55
                      >${t.type_name}</option>
56
                  </c:forEach>
57
              </select>
58
              图书名称: <input type="text" id="book_name"
   value="${bookName}"/>
59
              是否借阅:
              <select name="is_borrow" id="borrow">
60
                  <option value="">===请选择====</option>
61
62
                  <option value="1"</pre>
                      <c:if test="${borrow==1}">
63
64
                         selected
65
                      </c:if>
```

```
66
                >已借阅
67
                </option>
                <option value="0"</pre>
68
                       <c:if test="${borrow==0}">
69
70
                          selected
                       </c:if>
71
72
                >未借阅
73
                </option>
74
             </select>
75
             submit和Image 提交表单按钮
76
77
             reset 重置按钮
78
             button 普通按钮
79
             --%>
             <input type="button" value="查询" id="search"</pre>
80
    onclick="doSearch(1)"/>
81
             <!--</p>-->
82
             83
                <%--
84
    href="${pageContext.servletContext.contextPath}/servlet或jsp"></a>
85
                --%>
86
    href="${pageContext.servletContext.contextPath}/GetBookTypesServlet">新增图
    书</a>
87
             <%--
88
89
             写地址如何区分是否带web项目名称
90
             看浏览器地址是否会发生变化
91
             带web路径
             转发不用带项目名
92
93
             --%>
94
95
             <%-- dreamwearver --%>
             96
97
                <caption>
98
                    图书信息一览表
                </caption>
99
100
                101
                    编号
102
                    图书编号
103
                    图书名称
104
                    图书分类
105
                    图书作者
106
                    出版社
107
                    出版日期
108
                    借阅状态
109
                    查看详情
110
                    操作
111
                112
                <%
113
                    //java
                    //for
114
115
116
                <c:forEach items="${page.list}" var="book">
117
                    118
                          <%-- el 使用属性名必须与类中定义属性名大小写一样 --%>
                       ${book.id}
119
```

```
120
                         ${book_code}
121
                         ${book_name}
122
                         ${book_book_author}
123
                         ${book.type_name}
124
                         ${book.publish_press}
125
                         ${book.publish_date}
126
                         ${book.is_borrow?"已借阅":"未借阅"}
                         <a href="">查看详情</a>
127
128
                         129
                                <%--
                                 浏览器的地址会发生变化:地址里面就必须出现/项目
130
131
                                 浏览器地址不变化:地址就一定不要/项目
132
                                 ${pageContext.servletContext.contextPath}
133
                                 自动获取发布时配置的项目访问名称
134
135
                                 jsp或html中,数据提交的方式: GET协议走url地址
     POST协议,走的请求体
136
                                 GET请求提交参数格式:
137
                                 url?
    name=value&name2=value2&....&namen=valuen
138
139
                                --%>
140
                            <a
    href="${pageContext.servletContext.contextPath}/RemoveServlet?
    id=${book.id}"
                               onclick="return confirm('确认删除么?');">删除
141
    </a>
142
                               
143
    href="${pageContext.servletContext.contextPath}/GetModifyBookInfoServlet?
    id=${book.id}">修改</a>
144
                         145
                     146
                  </c:forEach>
147
                  148
149
                         <a href="javascript:doSearch(1);">首页</a>
                         <%-- 控制jsp翻页问题 --%>
150
151
                         <c:choose>
152
                            <c:when test="${!page.hasPrevPage}">
153
                                上一页
                            </c:when>
154
155
                            <c:otherwise>
156
                                <%-- 超链接外观: 鼠标停留变成小手形状 href不能删
    除,超链接的外观就不好看--%>
157
                                <a
    href="javascript:doSearch(${page.pageNum-1})">上一页</a>
158
                            </c:otherwise>
159
                         </c:choose>
160
                         <c:choose>
161
                            <c:when test="${!page.hasNextPage}">
162
                                下一页
163
                            </c:when>
                            <c:otherwise>
164
165
    href="javascript:doSearch(${page.pageNum+1})">下一页</a>
166
                            </c:otherwise>
167
                         </c:choose>
```

```
168
                           <a href="javascript:doSearch(${page.totalPages})">
    尾页</a>
169
                          ${page.pageNum}/${page.totalPages}页
                          <input type="text" id="txtPageNo" size="2"</pre>
170
    value="${page.pageNum}" />
                          <input type="button" value="Go" name="go"</pre>
171
    onclick="doSearch()"/>
172
                       173
                   174
               175
176
        177
    178 </body>
179 </html>
```

## 2 会话跟踪技术

### 2-1 会话概念

```
    会话即:浏览器和服务器之间的一次会晤。
    浏览器打开,会话产生,浏览器关闭,会话结束
    会话跟踪技术:用于浏览器从打开到关闭期间多个请求需要共享的数据解决方案
```

#### 2-2 如何实现会话跟踪

## 2-3 Cookie技术实现

```
1 服务器通过IE将cookie数据存入客户端,服务器也只能通过ie获取存储的数据
2 意味着:
3 IE存数据
4 火狐取数据
```

## 2-4 Cookie实现

特点:浏览器关闭,cookie保存的数据就丢失了。

保证cookie的数据不丢失的话,默认情况下,浏览器不能关闭

```
1
存入数据步骤:

2
1. servlet中创建Cookie对象

4 Cookie c=new Cookie(key,value);

5 2.将cookie发送给浏览器保存

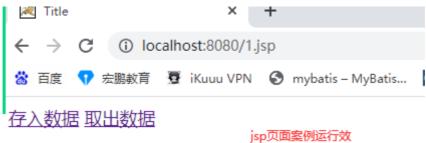
6 response.addCookie(保存cookie对象)
```

```
setMaxAge()设置Cookie有效期,Cookie数据存在时间与浏览器的开始和关闭没有关系,只要时间到
   了,Cookie的数据就被清理了。
10
   注意事项:
11
   1.setMaxAge(-1):默认配置,浏览器关闭,数据丢失
12
   2.setMaxAge(0):设置cookie作废
   3.cookie的获取有浏览器之分
13
14
   取出数据:
15
16 Cookie的数据统一由请求协议请求头提交服务器
17

    reqeust.getCookies();

18 2. 循环那key与要找的数据key进行比较,获取key对应值即可
```

#### 2-5 课堂案例



```
      1
      1.ServletA 存入数据: 随机数

      2
      2.ServletB 获取数据: 显示获取的随机数
```

#### ServletA存入数据

```
package cn.kgc.controller; /**
1.
    * @Author: lc
2
 3
    * @Date: 2022/5/23
 4
    * @Description: ${PACKAGE_NAME}
 5
    * @version: 1.0
    */
 6
7
8
   import javax.servlet.*;
9
    import javax.servlet.http.*;
10
    import javax.servlet.annotation.*;
11
    import java.io.IOException;
12
    import java.util.UUID;
13
    @webServlet("/ServletA")
14
15
    public class ServletA extends HttpServlet {
16
        @override
17
        protected void doGet(HttpServletRequest request, HttpServletResponse
    response) throws ServletException, IOException {
           //1.使用Cookie保存一个随机数
18
19
            //1-1 创建Cookie对象
20
           //随机数实现: Math.random():[0,1) new Random().nextInt(max):[0,Max)
21
            //UUID: java提供一个生成全球唯一标识符工具类
22
            UUID uuid = UUID.randomUUID();
            Cookie myCookie=new Cookie("num", uuid.toString());
23
            //干预cookie的存活时间
24
```

```
25
            myCookie.setMaxAge(2*60);//单位: 秒
26
            //1-2 cookie发送给浏览器保存
            response.addCookie(myCookie);
27
28
29
            //写响应体
30
            response.setContentType("text/html;charset=utf-8");
31
            response.getWriter().print("cookie保存成功!");
32
        }
33
34
        @override
35
        protected void doPost(HttpServletRequest request, HttpServletResponse
    response) throws ServletException, IOException {
36
            doGet(request, response);
        }
37
38
    }
```

#### ServletB获取数据

```
package cn.kgc.controller; /**
2
    * @Author: 1c
3
     * @Date: 2022/5/23
4
    * @Description: ${PACKAGE_NAME}
5
     * @version: 1.0
6
    */
7
8
   import javax.servlet.*;
9
    import javax.servlet.http.*;
10
    import javax.servlet.annotation.*;
11
    import java.io.IOException;
12
    @webServlet("/ServletB")
13
    public class ServletB extends HttpServlet {
14
15
        @override
        protected void doGet(HttpServletRequest request, HttpServletResponse
16
    response) throws ServletException, IOException {
            //1.获取这个浏览器提交的所有的cookie
17
18
            //细节:浏览器上没有保存任何跟该服务器相关的cookie, getCookie()返回值: null
            Cookie[] cookies = request.getCookies();
19
20
            //2.循环
21
            if(cookies==null | cookies.length==0){
                //响应:没有cookie数据
22
23
            }else{
                for(Cookie c:cookies){
24
25
                    //getName():获取cookie的key getValue():获取key对应的value
                    if(c.getName().equals("num")){
26
27
                        //获取该key对应的值
28
                        String value = c.getValue();
29
                        response.getWriter().print("uuid:"+value);
30
                    }
                }
31
32
            }
33
        }
34
35
        @override
36
        protected void doPost(HttpServletRequest request, HttpServletResponse
    response) throws ServletException, IOException {
            doGet(request, response);
```

```
38 | }
39 }
```

### 2-6 学生练习

```
      1
      案例:显示上次访问时间

      2
      1.ServletA存入当前时间

      3
      2.ServletB获取当前时间

      4
      扩展:

      6
      ServletC获取时间是上一次访问时间?
```

#### 参考代码

```
package cn.kgc.controller; /**
 1
 2
     * @Author: 1c
 3
     * @Date: 2022/5/23
    * @Description:显示"上一次访问时间"
 4
 5
     * @version: 1.0
 6
     */
 7
 8
    import javax.servlet.ServletException;
    import javax.servlet.annotation.WebServlet;
9
10
    import javax.servlet.http.Cookie;
    import javax.servlet.http.HttpServlet;
11
    import javax.servlet.http.HttpServletRequest;
12
    import javax.servlet.http.HttpServletResponse;
13
14
    import java.io.IOException;
15
    import java.net.URLDecoder;
16
    import java.net.URLEncoder;
17
    import java.text.SimpleDateFormat;
    import java.util.Date;
18
19
20
    @webservlet("/servlet1")
21
    public class Servlet1 extends HttpServlet {
22
        @override
23
        protected void doGet(HttpServletRequest request, HttpServletResponse
    response) throws ServletException, IOException {
24
            response.setContentType("text/html;charset=utf-8");
            //1.尝试获取上一次访问时间
25
26
            Cookie[] cookies = request.getCookies();
            if(cookies==null || cookies.length==0) {
27
28
                //2.如果上次访问时间的key不存在,提示:首次访问该网站
29
                response.getWriter().print("首次访问该网站");
30
            }else {
31
                //3.如果上次访问时间的key存在,提示:上一次访问该网站时间是:
32
                for(Cookie c:cookies){
                    if(c.getName().equals("lastTime")){
33
34
                        String value = c.getValue();
35
                        //解码
36
                        value= URLDecoder.decode(value, "utf-8");
                        //May 23 Mon 2022
37
38
                        response.getWriter().print("上一次访问该网站时间
    是: "+value);
39
                    }
40
                }
```

```
41
42
            //cookie保存本次访问的时间,以便于下一次访问时获取
43
            Date now = new Date();
44
            //SimpleDateFormat
45
            SimpleDateFormat sdf = new SimpleDateFormat("yyyy年MM月dd日 HH时mm分ss
    秒SSS");
46
            String dateStr = sdf.format(now);
47
            //解决cookie存储中文的问题
            dateStr= URLEncoder.encode(dateStr, "utf-8");
48
49
            Cookie myCookie=new Cookie("lastTime",dateStr);
50
            myCookie.setMaxAge(2*60);
51
            response.addCookie(myCookie);
52
        }
53
54
        @override
        protected void doPost(HttpServletRequest request, HttpServletResponse
55
    response) throws ServletException, IOException {
56
            doGet(request, response);
57
        }
58
    }
```

## Cookie保存中文的问题以及解决方案

#### HTTP Status 500 - Control character in cookie value or attribute.

type Exception report

message Control character in cookie value or attribute.

description The server encountered an internal error that prevented it from fulfilling this request.

exception

```
java.lang.IllegalArgumentException: Control character in cookie value or attribute. Cookie女是不支持中文!!!!
org.apache.tomcat.util.http.CookieSupport.isHttpSeparator(CookieSupport.java:193)
org.apache.tomcat.util.http.CookieSupport.isHttpToken(CookieSupport.java:217)
org.apache.tomcat.util.http.ServerCookie.appendCookieValue(ServerCookie.java:186)
org.apache.catalina.connector.Response.generateCookieString(Response.java:1032)
org.apache.catalina.connector.Response.addCookie(Response.java:974)
org.apache.catalina.connector.ResponseFacade.addCookie(ResponseFacade.java:381)
cn.kgc.controller.Servletl.doGet(Servletl.java:43)
javax.servlet.http.HttpServlet.service(HttpServlet.java:728)
```

note The full stack trace of the root cause is available in the Apache Tomcat/7.0.42 logs.

```
Apache Tomcat/7.0.42
```

```
1
           //中文存入Cookie前使用URL编码
2
           String name = URLEncoder.encode("姓名", "UTF-8");
           String value = URLEncoder.encode("张三", "UTF-8");
3
4
           //编码后的字符串保存到Cookie中
           Cookie c = new Cookie(name, value);
5
6
           c.setMaxAge(3600);
7
            response.addCookie(c);
8
9
           //从Cookie中获取中文后解码后再打印。
           String name = URLDecoder.decode(c.getName(), "UTF-8");
10
11
           String value = URLDecoder.decode(c.getValue(), "UTF-8");
           String s = name + ": " + value + "<br/>";
12
            response.getWriter().print(s);
```

## 3 Session

特点: 所有的数据存在服务器上

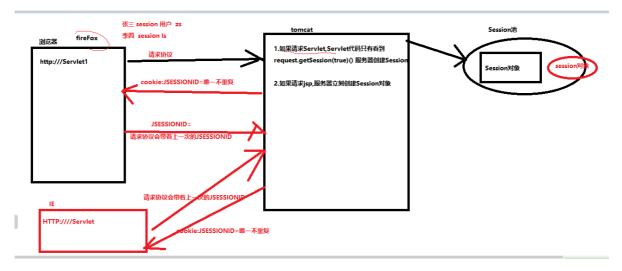
四大域对象: Session request

```
1 setAttribute(key,value)
2 getAttribute(key)
3 removeAttribute(key)
```

### 3-2 如何创建Session

```
1 request.getSession()和request.getSession(true):默认传入true,有Session直接返回session对象,如果没有session,创建session
2 request.getSession(boolean bool):false,有session获取session,如果服务器没有session,返回NULL
```

## 3-3 服务器保存Session的原理



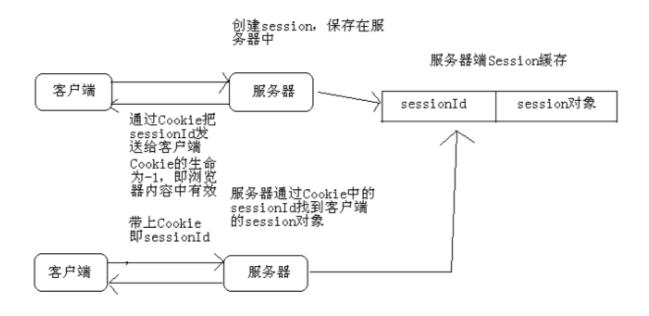
#### 课堂案例

```
package cn.kgc.controller; /**
 2
    * @Author: lc
    * @Date: 2022/5/23
 3
    * @Description: ${PACKAGE_NAME}
 5
    * @Version: 1.0
    */
 6
 7
8
    import javax.servlet.*;
9
    import javax.servlet.http.*;
    import javax.servlet.annotation.*;
10
11
    import java.io.IOException;
12
13
    @webServlet("/Servlet11")
    public class Servlet11 extends HttpServlet {
14
15
        protected void doGet(HttpServletRequest request, HttpServletResponse
16
    response) throws ServletException, IOException {
17
           //服务器帮我生成一个Session对象(我是指实际访问Servlet11那个浏览器)
            HttpSession session = request.getSession();
18
19
            //getId():获取Session对象分配给每个浏览器的唯一标识
            response.getWriter().print("给您生成的sessionid是: "+session.getId());
21
        }
```

```
22
23  @Override
24  protected void doPost(HttpServletRequest request, HttpServletResponse response) throws ServletException, IOException {
25     doGet(request, response);
26  }
27 }
```

#### 使用不同的浏览器访问Servlet11

每个浏览器上显示的id都不一样。



# 作业

CRUD+分页

## 预习

Session

JSP内置对象

文件下载和 上传