

SpringMVC

张海宁

贵州大学

hnzhang1@gzu.edu.cn

May 3, 2018

1 EL

2 JSTL

3 数据库分页

4 Appendix

- JSP 9 个内置对象
- JSP 4 个作用域
- 相关资源

EL

什么是 EL

EL 是指 jsp **E**xpression **L**anguage。EL 是 jsp2.0 引入的，其目的就是为了简化 jsp 页面中的的一些参数获取操作（using html like tags）。使用 EL 可以方便的读取存储在以下对象中的数据：

- JavaBean
- jsp 的一些内置对象
 - request
 - session
 - application
 - ...

语法结构

```
${ expression }
```

EL 的隐含对象

与 jsp 的内置对象 (参考本 ppt 第39,40页) 类似的概念, 可以直接使用。
EL 的隐含对象可以分为以下三类：

- 页面上下文对象
- 访问作用域范围的隐含对象
- 访问环境信息的隐含对象

页面上下文对象

通过页面上下文对象 `pageContext` 来获取 `jsp` 内置对象后，可以获取其属性，即 `getXXX()` 方法中的 `XXX` 部分。使用的时候去掉 `get` 并将 `XXX` 的首字母改为小写。比如下面的 `request.getMethod()`。

对应这些对象的 `getXXX()` 方法

```
1  ${pageContext.request }
2  <% request.getMethod(); %>
3  ${pageContext.request.method }
4
5  ${pageContext.response }
6  ${pageContext.out }
7  ${pageContext.session }
8  ${pageContext.page }
```

访问作用域范围的隐含对象

```
1 <%
2   pageContext.setAttribute("university", "pgGzu", 1);
3   request.setAttribute("university", "reqGzu");
4   session.setAttribute("university", "sessionGzu");
5   application.setAttribute("university", "appGzu");
6 %>
7 ${pageScope.university }
8 ${requestScope.university }
9 ${sessionScope.university }
10 ${applicationScope.university }
11 ${university }
```

结合 \$university，来看作用域的查找顺序：

page>request>session>application

访问环境信息的隐含对象

```
1  ${param.course }
2  <%request.getParameter("course"); %>
3  <!-- 获取复选框的值 -->
4  ${paramValues.multiCheckBox[0] }
5  ${paramValues.multiCheckBox[1] }
6
7  ${header.connection }
8  ${header["connection"] }
9  ${header["User-Agent"] }
10
11 ${cookie.JSESSIONID }
12 ${cookie.JSESSIONID.value }
```


param 示例

the form in test.jsp

```
1 <form action="showCookie.jsp" method="post">
2 课程:<select name="course">
3 <option value="AnalogCircuit">模电</option>
4 ...
5 </select>
6 备注:<input name="note" type="text">
7 <button type="submit">submit</button>
8 </form>
```

Question: 怎么在 showCookie.jsp 页面获取数据？

get the form parameter of test.jsp in showCookie.jsp

```
1 The course is: ${param.course}.
```

Question: 怎么在 showCookie.jsp 页面获取 cookie 相关的数据？

EL 的操作符

与 jsp 的内置对象类似的概念，可以直接使用。

Operator	Description	Operator	Description
.	访问 bean 里的属性 或 map 里的记录		
[]	访问 array 或 list 里 的元素	()	改变优先级
+ - * /	+ - * /	%	取余
==(eq)	判断是否相等	!=(neq)	判断是否不相等
<(lt)	判断是否小于	>(gt)	判断是否大于
<=(le)	判断是否小于或相等	>=(ge)	判断是否大于或相等
&&(and)	与	(or)	或
!(not)	非	empty	判断是否为空

Table: Basic Operators in EL

JSTL

什么是 JSTL

JSTL 是 **J**ava **S**erver **P**ages **S**tandard **T**ag **L**ibrary (JSTL) 的简称。JSTL 提供了一系列封装好了的 jsp 标签，可以取代在 jsp 页面中嵌入 java 代码的做法。

JSTL 实际上由 5 个功能不同的标签库组成：

- ① 核心标签库
- ② 格式标签库
- ③ SQL 标签库
- ④ XML 标签库
- ⑤ 函数标签库

核心标签库主要用于完成 JSP 页面的常用功能，包括：

- 表达式标签
 - `< c : out >`, `< c : set >`, `< c : remove >`, `< c : catch >`
- URL 标签
 - `< c : import >`, `< c : redirect >`, `< c : url >`, `< c : param >`
- 流程控制标签
 - `< c : if >`, `< c : choose >`, `< c : when >`, `< c : otherwise >`
- 循环控制标签
 - `< c : forEach >`, `< c : forTokens >`

核心标签库中各标签的作用

tag	Description
<code>< c : out ></code>	将表达式的值输出到 jsp 页面
<code>< c : set ></code>	在指定的作用域内定义变量或为变量赋值
<code>< c : remove ></code>	从指定的作用域中移除指定的变量
<code>< c : catch ></code>	捕获程序的异常
<code>< c : import ></code>	导入文件到 web 页面中
<code>< c : redirect ></code>	将 request 请求重定向到其他 URL
<code>< c : url ></code>	构造一个 URL
<code>< c : param ></code>	为其他标签提供参数信息
<code>< c : if ></code>	简单的条件判断
<code>< c : choose ></code> , <code>< c : when ></code> , <code>< c : otherwise ></code>	选择, 相当于 switch
<code>< c : forEach ></code>	遍历数组或集合类中的成员
<code>< c : forTokens ></code>	迭代字符串中由分隔符分隔的成员

JSTL 的配置

- ① 下载 JSTL 的 jar 文件

http:

[//tomcat.apache.org/download-taglibs.cgi#Standard-1.2.5](http://tomcat.apache.org/download-taglibs.cgi#Standard-1.2.5)

- ② 将下载好的 4 个 jar 文件，放到项目下的 WEB-INF 文件夹下的 lib 文件夹中
- ③ 在想使用 JSTL 标签的 jsp 页面中，定义引用的标签库和访问前缀

```
1 <%@ taglib prefix = "c"
```

```
2     uri = "http://java.sun.com/jsp/jstl/core" %>
```


表达式标签 <c:out>

<c:out> 标签用于将表达式的值输出到 jsp 页面中去。类似于 jsp 的 <%=exp%> 或 el 的 \${exp}。

<c:out value="" [escapeXml="true|false"] [default="defaultValue"] />

- value
用于指定变量或表达式，可以使用 el
- escapeXml
可选属性，默认为 true，转换。用于指定是否转换 value 中的特殊字符。比如将 < 转换为 <
- default
可选属性，默认为空字符串。用于指定当 value 的值为 null 时，将要显示的默认值。

<c:out value="
newline" /> 这里
 原样输出

<c:out value="
newline" escapeXml="false"/> 这里
 会被解析为 html 标记

变量设置标签 <c:set> |

<c:set> 标签用于在指定范围（page, request, session, application）中定义保存某个值的变量，或为指定的对象设置属性。

- ① 在 scope 范围内将值保存到变量中

```
<c:set var="name" value="value" [scope=" 范围"]/>
```

- ② 在 scope 范围内将标签体保存到变量中

```
<c:set var="name" [scope=" 范围"]/> 标签体 </c:set>
```

- ③ 将值保存在对象的属性中

```
<c:set value="value" target="object" property="propertyName"/>
```

value 可以使用 *el* 表达式；*target* 可以是 *javabean* 或 *map* 对象。

变量设置标签 <c:set> II

设置一个 request 变量，给一个 bean 赋值

```
1 <%@ taglib prefix="c"
2     uri="http://java.sun.com/jsp/jstl/core" %>
3 <c:set var="myName" value="gzu" scope="request"/>
4 <c:out value="myName is ${myName}"/>
5 <jsp:useBean id="st" class="bean.Student"/>
6 <c:set target="${st}" property="name" value="BD"/>
7 <c:out value="st.name is ${st.name}"/>
```

<c:remove> 变量移除标签 <c:catch> 捕获异常标签

- `<c:remove var="name" [scope="范围"]>`
最好指定范围，否则移除所有范围内的值。
- `<c:catch [var="exception"]> code may have exception...</c:catch>`
var 可以存储异常信息，供 catch 标签外的代码使用。

文件导入、重定向和 url 地址生成是 web 应用中常用的功能。JSTL 中提供的与 URL 相关的标签分别是：`<c:import>`、`<c:url>`、`<c:redirect>` 和一个通常与其他标签配合使用的 `<c:param>`。

<c:import> 导入标签

此标签可以导入站内或站外的静态或动态文件到 web 页面。

❶ 直接引用地址

```
<c:import url="url" >
```

❷ 将 url 地址的内容保存到 Reader 类型的对象中

```
<c:import url="url" varReader="name">
```

varReader: 用于指定一个变量名, 该变量用于以 Reader 类型存储被包含对象。

教材 264 页例 12.5

<c:url> 动态生成 URL 标签 I

<c:url> 标签可用于生成一个 URL 路径的字符串，这个字符串可以赋予 HTML 的 <a> 标记实现超链接，或利用这个生成的 URL 字符串实现网页转发与重定向。还可以与 <c:param> 标签结合动态添加 URL 的参数信息。

- ① 输出产生的 URL 字符串信息，如果指定了 var 和 scope 属性，则 URL 信息就不再输出，而是存储在变量中。

```
<c:url value="url" [var="name"] [scope="范围"] />
```

- ② 搭配 <c:param> 标签给 URL 附带参数。

```
<c:url value="url" var="name"> <c:param name="" value="">  
</c:url>
```

<c:url> 动态生成 URL 标签 II

示例代码

```
1 <c:url value="index.jsp"/>
2 <c:url value="index.jsp" var="backtoindex"/>
3 <a href=${backtoindex }>back to index</a>
4
5 pageScope:${pageScope.backtoindex }
6 requestScope:${requestScope.backtoindex }
7 sessionScope:${sessionScope.backtoindex }
8 applicationScope:${applicationScope.backtoindex }
```


<c:redirect> 重定向标签

- 1 简单指定重定向地址。

```
<c:redirect url="">
```

- 2 给重定向地址加上参数。

```
<c:redirect url="url"> <c:param name="" value=""/>  
</c:redirect>
```

示例代码

```
1 <c:redirect url="Reg"/>  
2 <c:redirect url="index.jsp"/>
```

类似于一般的编程语言，JSTL 也有判断和循环相关功能的标签。

- 判断选择

`<c:if>` `<c:choose>` `<c:when>` `<c:otherwise>`

- 循环

`<c:forEach>` `<c:forTokens>`

<c:if> 条件判断标签

- 1 将判断结果保存在变量中

```
<c:if test="condition" var="name" />
```

- 2 将判断结果保存在变量中，并且可以根据判断结果去执行标签体。

```
<c:if test="condition" var="name" /> code...</c:if>
```

test 可使用 *el* 表达式。

```
1 <c:if test="${!backtoindex}">
2   <a href="${backtoindex}">back to index</a>
3 </c:if>
```

<c:choose> <c:when> <c:otherwise>

<c:choose> 标签只能做为 <c:when> <c:otherwise> 的父标签。

```
1 <c:choose>
2 <c:when test="${hour}>8 && hour<18 }">work time
3   </c:when>
4 <c:when test="${hour}>18 && hour<20}">meal time
5   </c:when>
6 <c:otherwise>do what you want</c:otherwise>
7
8 </c:choose>
```

<c:forEach> 循环标签 I

① 集合成员迭代

`<c:forEach items="data" [var="name"] [begin="st"] [end="finish"] [step="step"] [varStatus="statusName"]> 标签体 </c:forEach>`

② 数字索引迭代

`<c:forEach begin="start" end="finish" [step="step"] [var="name"] [varStatus="statusName"]> 标签体 </c:forEach>`

属性	说明
items	指定被循环的对象，一般为数组或集合类。
var	存储 items 指定对象的成员。
varStatus	指定循环的状态。（其拥有一个 index 属性，可以用于获取当前循环的索引值）

<c:forEach> 循环标签 II

获取集合中的元素

```
1 <%
2 List <String> lst = new ArrayList<String>();
3 lst.add("gzu");
4 lst.add("sdu");
5 lst.add("cqu");
6 pageContext.setAttribute("mylist", lst);
7 %>
8 <c:forEach items="${pageScope.mylist}" var="school"
9     varStatus="st">
10     ${st.index }:${school}<br>
11 </c:forEach>
```

<c:forEach> 循环标签 III

进行数字循环

```
1 <c:set var="sum" value="0"/>
2 <c:forEach var="i" begin="1" end="100">
3   ${sum = sum+i }
4 </c:forEach>
5   sum is:${sum }
```

<c:forTokens> 迭代标签 I

<c:forTokens> 迭代标签可以用指定的分隔符将一个字符串分割开，并获取分割后的值。

```
<c:forTokens items="String" delims="char" [var="name"] [begin="st"]  
[end="finish"] [step="step"] [varStatus="statusName"]> 标签体  
</c:forTokens>
```

属性	说明
items	指定要迭代的 String 对象。
delims	指定要使用的分割符。
var	保存了分割后的对象。
varStatus	指定循环的状态。(其拥有一个 index 属性，可以用于获取当前循环的索引值)

<c:forTokens> 迭代标签 II

使用迭代标签分隔字符串

```
1 <c:set var="mySchool" value="tzyz#zzti#cqut#gzu"/>
2 <c:forTokens items="${mySchool}" delims="#"
3   var="sc">
4   ${sc }<br>
5 </c:forTokens>
```

数据库分页

分页使用场景

分页一般指的是将从数据库查询来的数据，分成多个页面来显示。一般情况下有两种方法来达到数据分页的目的：

- ① 获取到所有的查询结果，在页面显示时进行处理分页
- ② 使用数据库的查询机制来分页

```
select * from teacher limit 2,3;
```

LIMIT 接受一个或两个数字参数。参数必须是一个整数常量。如果给定两个参数，第一个参数指定第一个返回记录行的偏移量，第二个参数指定返回记录行的最大数目。初始记录行的偏移量是 0(而不是 1)

① 实现小组项目的数据库分页功能¹

¹teacher.jsp 里面的数据库分页比较复杂，请参考教材 p222 页 10.4.2 实现。

The End

Appendix

内置对象

Object	Description	Scope
request	封装了客户端请求	一次会话 系统的全局变量 ServletConfig
response	封装了服务器端的回应信息	
out	向 response 中写数据	
session	保存一次会话的信息	
application	保存应用程序中的公有数据	
config		ServletConfig
pageContext	获取 jsp 页面上下文, 进而获取内置对象	this 的同意词
page	调用当前 jsp 页面中的变量或方法	
exception	用来处理异常信息的	

Table: implicit objects of jsp

返回5

Scope	Description
page	当前的页面
request	当前的请求
session	当前的会话
application	当前网站

Table: scopes of jsp

返回5

ppt、项目源代码及实验指导书的地址

- ① ppt
<https://github.com/gmsft/ppt/tree/master/javaweb>
- ② lab-java 项目
<https://github.com/gmsft/javaweb>
- ③ 实验指导书
放在 ppt 的仓库里