第5章
表达式语言

本章内容

- 5.1 理解表达式语言
- 5.2 使用EL运算符
- 5.3 使用EL访问数据

5.1 理解表达式语言

- EL并不是一种通用的编程语言，它仅仅是一种数据访问语言。网页作者通过它可以很方便地在JSP页面中访问应用程序数据，无需使用小脚本(<%和%>)或JSP请求时表达式(<%=和%>)，甚至不用学习Java语言就可以使用表达式语言。
- 作为一种数据访问语言，EL具有自己的运算符、语法和保留字。作为JSP开发员，我们的工作是创建EL表达式并将其添加到JSP的响应中。

5.1.1 表达式语言的调用

- 在JSP 2.0的页面中，表达式语言的使用形式如下。

  ${expression}

- 表达式语言是以$开头，后面是一对大括号，括号里面是合法的EL表达式。
- 该结构可以出现在JSP页面的模板文本中，也可以出现在JSP标签的属性值中，只要属性允许常规的JSP表达式即可。

5.1.1 表达式语言的调用

- 下面是在JSP模板文本中使用EL表达式。

```
<ul>
  <li>客户名：${customer.custName}
  <li>Email地址：${customer.email}
</ul>
```

- 下面是在JSP标准动作的属性中使用EL表达式。

```
<jsp:include page = "${expression1}" />
<c:out value = "${expression2} " />
```

5.1.2 表达式语言的功能

- 表达式语言的主要功能包括：

（1）提供了一组简单的运算符。

（2）对作用域变量的方便访问。

（3）对JavaBeans对象访问的简单表示。

（4）对集合元素的简单访问。

（5）对请求参数、Cookie和其他请求数据的简单访问。

（6）提供了在EL中使用Java函数的功能。

5.1.3 表达式语言与JSP表达式的区别

- JSP表达式的使用格式为：

  <%=expression %>

- 这里的expression为合法的Java表达式，它属于脚本语言的代码。在expression中可以使用由脚本声明的变量。

5.1.3 表达式语言与JSP表达式的区别

- EL表达式的格式为：

  ${expression}

- 这里的expression是符合EL规范的表达式，并且不需要包含在标签内。在EL表达式中不能使用脚本中声明的变量。

5.2 使用EL运算符

- EL作为一种简单的数据访问语言，提供了一套运算符。
- EL的运算符包括：算术运算符、关系运算符、逻辑运算符、条件运算符、empty运算符以及属性与集合访问运算符。
- 这些运算符与Java语言中使用的运算符类似，但在某些细节上仍有不同。

5.2 使用EL运算符

- 5.2.1 算术运算符
- 5.2.2 关系与逻辑运算符
- 5.2.3 条件运算符
- 5.2.4 empty运算符
- 5.2.5 属性与集合元素访问运算符

5.2.1 算术运算符

- 表5-1给出了在这些类型上的算术运算符。

算术运算符	说明	示 例	结果
+	加	${6.80 + -12}	-5.2
-	减	${15-5}	10
*	乘	${2 * 3.14159}	6.28318
/或div	除	${25 div 4} 与 ${25/4}	6.25
%或mod	取余	${24 mod 5} 与 ${24 % 5}	4

5.2.2 关系与逻辑运算符

- EL的关系运算符与一般的Java代码的关系运算符类似，如表5-2所示。

关系运算符	说明	示 例	结果
== 或 eq	相等	${3==5}或${3 eq 5}	false
!= 或ne	不相等	${3!=5}或${3 ne 5}	true
< 或lt	小于	${3<5}或${3 lt 5}	true
> 或gt	大于	${3>5}或${3 gt 5}	false
<= 或le	小于等于	${3<=5}或${3 le 5}	true
>= 或ge	大于等于	${3>=5}或${3 ge 5}	false

5.2.2 关系与逻辑运算符

- 关系表达式产生的boolean型值可以与EL的逻辑运算符结合运算，这些运算符如表5-3所示。

逻辑运算符	说明	示 例	结果
&& 或 and	逻辑与	${(9.2 >=4) && (1e2 <= 63)}	false
|| 或or	逻辑或	${(9.2>= 4) || (1e2 <= 63)}	true
! 或not	逻辑非	${ not 4 >= 9.2) }	true

5.2.3 条件运算符

- EL的条件运算符的语法是：

  expression ? expression1 : expression2

- 表达式的值是基于expression的值，它是一个boolean表达式。如果expression的值为true，则返回expression1结果；如果expression的值为false，则返回expression2的结果。

5.2.3 条件运算符

- ${(5 * 5) == 25 ? 1 : 0} 的结果为 1；
- ${(3 gt 2) && !(12 gt 6) ? "Right" : "Wrong"} 的结果为Wrong；
- ${("14" eq 14.0) && (14 le 16) ? "Yes" : "No"} 的结果为Yes；
- ${(4.0 ne 4) || (100 <= 10) ? 1 : 0} 的结果为 0。

5.2.4 empty运算符

- empty运算符的使用格式为：

  ${empty expression}

- 它判断expression的值是否为null、空字符串、空数组、空Map或空集合，若是则返回true，否则返回false。

5.2.5　属性与集合访问运算符

- 属性访问运算符用来访问对象的成员，集合访问运算符用来检索Map、List或数组对象的元素。这些运算符在处理隐含变量时特别有用。在EL中，这类运算符有下面两个。
- 点号（.）运算符。
- 方括号（[]）运算符。

1. 点号（.）运算符

- 点号运算符用来访问Map对象一个键的值或bean对象的属性值，例如：param是EL的一个隐含对象，它是一个Map对象，下面代码返回param对象username请求参数的值：

  ${param.username}

- 再比如，假设customer是Customer类的一个实例，下面代码访问该实例的custName属性值：

  ${customer.custName}

2. 方括号（[]）运算符

- 方括号运算符除了可以访问Map对象键值和bean的属性值外，还可以访问List对象和数组对象的元素。例如：

  ${param ["username"]}

  或 ${ param ['username']}

  ${customer["custName"]}

- 下面程序使用表格的形式输出了使用各种运算符的EL表达式的值。

2. 方括号（[]）运算符

- 程序5.1 eloperator.jsp
- 为了在JSP页面中输出文本${2+5}，需要在“$”符号前使用转义字符“\”，否则将输出EL表达式的值。

5.3 使用EL访问数据

- 5.3.1 访问作用域变量
- 5.3.2 访问JavaBeans属性
- 5.3.3 访问集合元素
- 5.3.4 访问EL的隐含变量

5.3.1 访问作用域变量

- 在JSP页面中，可以使用JSP表达式访问作用域变量。一般做法是：在Servlet中使用setAttribute()将一个变量存储到某个作用域对象上，如HttpServletRequest、HttpSession及ServletContext等。然后使用RequestDispatcher对象的forward()将请求转发到JSP页面，在JSP页面中调用隐含变量的getAttribute()返回作用域变量的值。

5.3.1 访问作用域变量

- 使用EL就可以更方便地访问这些作用域变量。要输出作用域变量的值，只需在EL中使用变量名即可，例如：

  ${variable_name}

- 对该表达式，容器将依次在页面作用域、请求作用域、会话作用域和应用作用域中查找名为variable_name的属性。如果找到该属性，则调用它的toString()并返回属性值。如果没有找到，则返回空字符串（不是null）。

5.3.1 访问作用域变量

- 下面通过一个例子说明如何访问作用域变量。
- 程序5.2 VariableServlet.java
- 程序5.3 variables.jsp

5.3.2 访问JavaBeans属性

- 如果知道JavaBeans的完整名称和它的作用域，也可以使用下面JSP标准动作访问JavaBeans的属性：

```
<jsp:useBean id="employee"
   class="com.demo.Employee"
scope="session" />
<jsp:setProperty name="employee"
   property="empName" value="Hacker" />
<jsp:getProperty name="employee"
   property="empName" />
```

5.3.2 访问JavaBeans属性

- 如果使用表达式语言，就可以通过点号表示法很方便地访问JavaBeans的属性，如下所示：

  ${employee.empName}

- 使用表达式语言，如果没有找到指定的属性不会抛出异常，而是返回空字符串。

5.3.2 访问JavaBeans属性

- 使用表达式语言还允许访问嵌套属性。例如，如果Employee有一个address属性，它的类型为Address，而Address又有zipCode属性，则可以使用下面简单形式访问zipCode属性。

  ${employee.address.zipCode}

- 上面的方法不能使用<jsp:useBean>和<jsp:getProperty>实现。

5.3.2 访问JavaBeans属性

- 下面通过一个示例来说明对JavaBeans属性的访问。该例中有两个JavaBeans，分别为Address，它有三个字符串类型的属性，city、street和zipCode；Employee是在前面的类的基础上增加了一个Address类型的属性address表示地址。

5.3.2 访问JavaBeans属性

- 在EmployeeServlet.java程序中创建了一个Employee对象并将其设置为请求作用域的一个属性，然后将请求转发到JSP页面，在JSP页面中使用下面的EL访问客户地址的三个属性。

```
<li>城市:${employee.address.city}
<li>街道:${employee.address.street}
<li>邮编:${employee.address.zipCode}
```

5.3.2 访问JavaBeans属性

- 程序5.4 Address.java
- 程序5.5 Employee.java
- 程序5.6 EmployeeServlet.java
- 程序5.7 beanDemo.jsp

5.3.3 访问集合元素

- 在EL中可以访问各种集合对象的元素，集合可以是数组、List对象或Map对象。这需要使用数组记法的运算符([])。
- 例如，假设有一个上述类型的对象attributeName，可以使用下面形式访问其元素。

  ${attributeName[entryName]}

5.3.3 访问集合元素

（1）如果attributeName对象是数组，则entryName为下标。上述表达式返回指定下标的元素值。下面代码演示了访问数组元素。

```
<%
  String[] fruit = {"apple","orange","banana"};
  request.setAttribute("myFruit", fruit);
%>
My favorite fruit is:${myFruit[2]}
```

- 上面一行还可以写成：

```
My favorite fruit is:${myFruit["2"]}
```

5.3.3 访问集合元素

（2）如果attributeName对象是实现了List接口的对象，则entryName为索引。下面代码演示了访问List元素。

```
<%@ page import="java.util.ArrayList" %>
<%
  ArrayList<String> fruit = new ArrayList<String>();
  fruit.add("apple");
  fruit.add("orange");
fruit.add("banana");
request.setAttribute("myFruit", fruit);
%>
My favorite fruit is:${myFruit[2]}
```

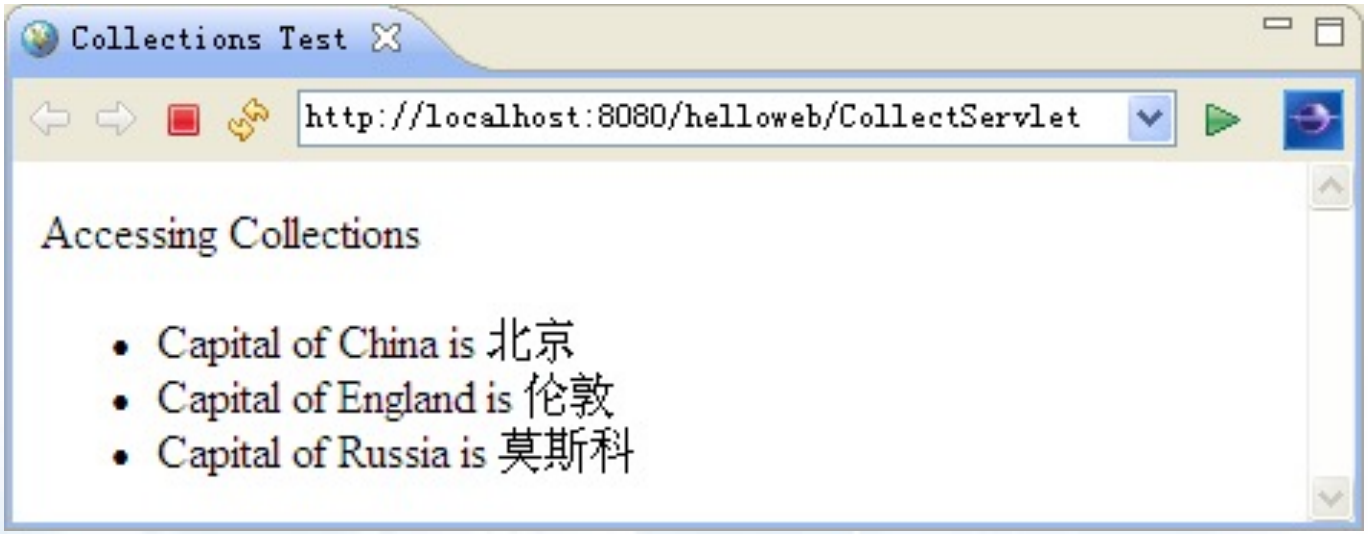
5.3.3 访问集合元素

（3）如果attributeName对象是实现了Map接口的对象，则entryName为键，相应的值通过Map对象的get(key)获得的，例如：

```
Map<String,String> capital =
                    new HashMap<String,String>();
capital.put("England","伦敦");
capital.put ("China","北京");
capital.put ("Russia","莫斯科");
request.setAttribute("capital", capital);
The capital of China is:${capital["China"]}<br>
The capital of Russia is:${capital.Russia}
```

5.3.3 访问集合元素

- 程序5.8 CollectServlet.java
- 程序5.9 collections.jsp



5.3.4 访问EL隐含变量

- 在JSP页面的脚本中可以访问JSP隐含变量，如request、session、application等。
- EL表达式中也定义了一套自己的隐含变量。使用EL可以直接访问这些隐含变量。
- 表5-4给出了EL中可以使用的11个隐含变量及其说明。

1. pageContext变量

- pageContext是PageContext类型的变量。PageContext类依次拥有request、response、session、out和servletContext属性，使用pageContext变量可以访问这些属性的属性。下面是一些例子。

  ${pageContext.request.method}
  ${pageContext.request.remoteAddr}

2. param和paramValues变量

- param和paramValues变量用来从ServletRequest中检索请求参数值。param变量是调用给定参数名的getParameter(String name)的结果，使用EL表示如下。

  ${param.name}

- 类似地，paramValues是使用getParameterValues(String name)返回给定名称的参数值的数组。要访问参数值数组的第一个元素，可使用下面代码。

${paramValues.name[0]}

3. header和headerValues变量

- header和headerValues变量是从HTTP请求头中检索值，它们的运行机制与param和paramValues类似。下面代码使用EL显示了请求头host的值。

${header.host}或${header["host"]}

- 类似地，headerValues.host是一个数组，它的第一个元素可使用下列表达式之一显示。

${headerValues.host[0]}

${headerValues.host["0"]}

${headerValues.host['0']}

4. cookie变量

- 使用EL的cookie隐含变量得到客户向服务器发回的Cookie数组，即调用request对象的getCookies()的返回结果。如果要访问cookie的值，则需要使用Cookie类的属性value（即getValue方法）。因此，下面一行可以输出名为userName的Cookie的值。如果没有找到这个cookie对象，则输出空字符串。

  ${cookie.userName.value}

- 使用cookie变量还可以访问会话Cookie的ID值，例如：

  ${cookie.JSESSIONID.value}

5. initParam变量

- initParam变量存储了Servlet上下文的参数名和参数值。例如，假设在DD中定义了如下初始化参数。

```
<context-param>
    <param-name>email</param-name>
    <param-value>hacker@163.com</param-value>
</context-param>
```

5. initParam变量

- 则可以使用下面的EL表达式得到参数email的值。

  ${initParam.email}

- 如果通过JSP脚本元素访问该Servlet上下文参数，应该使用下面表达式。

```
<%= application.getInitParameter("email")
%>
```

5. pageScope、requestScope、sessionScope和applicationScope变量

- 这几个隐含变量很容易理解，它们用来访问不同作用域的属性。例如，下面代码在会话作用域中添加一个表示商品价格的totalPrice属性，然后使用EL访问该属性值。

```
<%
    session.setAttribute("totalPrice",1000);
%>
${sessionScope.totalPrice}
```

- 注意，访问应用作用域的属性应使用applicationScope变量而不是使用pageContext变量。

5.4 小结

- 表达式语言（EL）是JSP 2.0中增加的新特征，EL仅仅是一种数据访问语言，通过它可以很方便地在JSP页面中访问应用程序数据，无需使用小脚本和请求时表达式。
- 表达式语言最重要的目的是创建无脚本的JSP页面。为了实现这个目的，EL定义了自己的运算符、语法等，它完全能够替代传统的JSP中的声明、表达式和小脚本。