**EL 全名为Expression Language**

EL 语法很简单，它最大的特点就是使用上很方便。接下来介绍EL主要的语法结构：

${sessionScope.user.sex}

所有EL都是以${为起始、以}为结尾的。上述EL范例的意思是：从Session的范围中，取得

用户的性别。假若依照之前JSP Scriptlet的写法如下：

User user =(User)session.getAttribute("user");

String sex =user.getSex( );

两者相比较之下，可以发现EL 的语法比传统JSP Scriptlet 更为方便、简洁。

**EL表达式是配合JSTL使用的**。

你需要引入JSTL标签库(<c:forEach items=""></c:forEach>,<c:if test=""></c:if>等属于jstl表达式)

**<%@taglib uri=" jsp标签的命名空间 " prefix="c"%>**

**.与 [ ] 运算符**

EL 提供 . 和 [ ] 两种运算符来导航数据。下列两者所代表的意思是一样的：

${sessionScope.user.sex}等于${sessionScope.user["sex"]}

. 和 [ ] 也可以同时混合使用，如下：

${sessionScope.shoppingCart[0].price}

回传结果为shoppingCart中第一项物品的价格。

**不过，以下两种情况，两者会有差异：**

(1) 当要存取的属性名称中包含一些特殊字符，如. 或 – 等并非字母或数字的符号，就一定要使用 [ ]，

例如：${user.My-Name }

上述是不正确的方式，应当改为：${user["My-Name"] }

(2) 我们来考虑下列情况：

${sessionScope.user[data]}

此时，data 是一个变量，假若data的值为"sex"时，那上述的例子等于${sessionScope.user.sex}；

假若data 的值为"name"时，它就等于${sessionScope.user.name}。

因此，如果要动态取值时，就可以用上述的方法来做，但无法做到动态取值。

**一般就是把要的数据放在对象中，把对象作为属性存在某个范围，然后取出属性再get数据**

**EL 变量**

EL 取变量数据的方法很简单，例如：${username}。它的意思是取出某一范围中名称为username的变量。

**因为我们并没有指定哪一个范围的username，所以它的默认值会先从Page 范围找，假如找不到，**

**再依序到Request、Session、Application范围。假如途中找到username，就直接回传，不再继续找下去**，（找这四个对象的属性）

但是假如全部的范围都没有找到时，就回传null，当然**EL表达式还会做出优化，页面上显示空白，而不是打印输出NULL。**

|  |  |  |
| --- | --- | --- |
| 属性范围（jstl名称） | EL中的名称 | JSP内置对象 |
| Page | PageScope | page |
| Request | RequestScope | request |
| Session | SessionScope | session |
| Application | ApplicationScope | application |

**这四个隐含对象只能用来取得范围属性值**，即JSP中的getAttribute(String name)，却不能取得其他相关信息，

例如：

**JSP中的request对象除可以存取属性之外，还可以取得用户的请求参数或表头信息等等。**

**但是在EL中，它就只能单纯用来取得对应范围的属性值，**

我们也可以指定要取出哪一个范围的变量：

|  |  |
| --- | --- |
| 范例 | 说明 |
| ${pageScope.username} | 取出Page范围的username变量 |
| ${requestScope.username} | 取出Request范围的username变量 |
| ${sessionScope.username} | 取出Session范围的username变量 |
| ${applicationScope.username} | 取出Application范围的username变量 |

其中，pageScope、requestScope、sessionScope和applicationScope都是EL 的隐含对象，

由它们的名称可以很容易猜出它们所代表的意思，

例如：${sessionScope.username}是取出Session范围的username 变量。这种写法是不是比之前JSP 的写法：

String username =(String) session.getAttribute("username");容易、简洁许多.

**自动转变类型**

EL 除了提供方便存取变量的语法之外，它另外一个方便的功能就是：**自动转变类型**，我们来看下面这个范例：

${param.count + 20}

假若**窗体传来count的值为10**时，那么上面的结果为30。之前没接触过JSP 的读者可能会认为上面的例子是理所当然的，

但是在**JSP 1.2 之中不能这样做**，原因是**从窗体所传来的值，它们的类型一律是String**，所以当你接收之后，必须再将它转为其他类型，

如：int、float 等等，然后才能执行一些数学运算，下面是之前的做法：

String str\_count =request.getParameter("count");

int count =Integer.parseInt(str\_count);

count = count + 20;  
所以，注意不要和java的语法（当字符串和数字用“+”链接时会把数字转换为字符串）搞混淆喽。

**如果需要在支持表达式语言的页面中正常输出“$”符号，则在“$”符号前加转义字符“\”，否则系统以为“$”是表达式语言的特殊标记。**