**EL 隐含对象**

JSP有9个隐含对象，而EL也有自己的隐含对象。EL隐含对象总共有11 个

**其中前五个和JSP隐式对象对应**

**但是那四个范围对象只能取属性值！比如：RequestScope不能setHeader之类的**

|  |  |  |
| --- | --- | --- |
| 隐含对象 | 类型 | 说明 |
| PageContext | javax.servlet.ServletContext | **表示此JSP的PageContext隐式对象**  **而它可以调用JSP所有的隐式对象** |
| PageScope | java.util.Map | 取得Page范围的属性名称所对应的值 |
| RequestScope | java.util.Map | 取得Request范围的属性名称所对应的值 |
| sessionScope | java.util.Map | 取得Session范围的属性名称所对应的值 |
| applicationScope | java.util.Map | 取得Application范围的属性名称所对应的值 |
| param | java.util.Map | 如同ServletRequest.getParameter(String name)。回传String类型的值 |
| paramValues | java.util.Map | 如同ServletRequest.getParameterValues(String name)。回传String[]类型的值 |
| header | java.util.Map | 如同ServletRequest.getHeader(String name)。回传String类型的值 |
| headerValues | java.util.Map | 如同ServletRequest.getHeaders(String name)。回传String[]类型的值 |
| cookie | java.util.Map | 如同HttpServletRequest.getCookies() |
| initParam | java.util.Map | 如同ServletContext.getInitParameter(String name)。回传String类型的值 |

**不过有一点要注意的是如果你要用EL输出一个常量的话，字符串要加双引号，不然的话EL会默认把你认为的常量当做一个变量来处理，**

**这时如果这个变量在4个声明范围不存在的话会输出空，如果存在则输出该变量的值。**

属性(Attribute)与范围(Scope)

**与范围有关的EL 隐含对象包含以下四个：pageScope、requestScope、sessionScope 和**

**applicationScope，它们基本上就和JSP的pageContext、request、session和application一样**，所以笔者在这里只稍略说明。

不过必须注意的是，**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，**

例如：我们要在session 中储存一个属性，它的名称为username，在JSP 中使用session.getAttribute("username")来取得username 的值，

但是在EL中，则是使用${sessionScope.username}来取得其值的。

**cookie**

所谓的cookie是一个小小的文本文件，它是以key、value的方式将Session Tracking的内容记录在这个文本文件内，这个文本文件通常存在于浏览器的暂存区内。

JSTL并没有提供设定cookie的动作，因为这个动作通常都是后端开发者必须去做的事情，而不是交给前端的开发者。

假若我们在cookie 中设定一个名称为userCountry的值，那么可以使用${cookie.userCountry}来取得它。

**header 和headerValues**

header 储存用户浏览器和服务端用来沟通的数据，当用户要求服务端的网页时，会送出一个记载要求信息的标头文件，例如：用户浏览器的版本、用户计算机所设定的区域等其他相关数据。假若要取得用户浏览器的版本，即${header["User-Agent"]}。另外在鲜少机会下，有可能同一标头名称拥有不同的值，此时必须改为使用headerValues 来取得这些值。

注意:因为User-Agent 中包含“-”这个特殊字符，所以必须使用“[]”，而不能写成

$(header.User-Agent)。

**initParam**

就像其他属性一样，我们可以自行设定web 站台的环境参数(Context)，当我们想取得这些参数initParam就像其他属性一样，

我们可以自行设定web 站台的环境参数(Context)，当我们想取得这些参数

[?](http://www.jb51.net/article/105314.htm)

|  |  |
| --- | --- |
| 1  2  3  4  5  6  7  8  9  10  11  12  13  14  15  16  17  18  19 | <?xml version="1.0"encoding="ISO-8859-1"?>    <web-appxmlns="<http://java.sun.com/xml/ns/j2ee>"    xmlns:xsi="<http://www.w3.org/2001/XMLSchema-instance>"    xsi:schemaLocation="<http://java.sun.com/xml/ns/j2ee/web-app_2_4.xsd>"    version="2.4">：    <context-param>    <param-name>userid</param-name>    <param-value>mike</param-value>    </context-param>：    </web-app> |

那么我们就可以直接使用${initParam.userid}来取得名称为userid，其值为mike 的参数。

下面是之前的做法：String userid =(String)application.getInitParameter("userid");

**param和paramValues**

在取得用户参数时通常使用一下方法：

request.getParameter(String name)  
request.getParameterValues(String name)

在 EL中则可以使用param和paramValues两者来取得数据。  
${param.name}  
${paramValues.name}

这里param 的功能和request.getParameter(String name)相同，而paramValues和  
request.getParameterValues(String name)相同。如果用户填了一个表格，表格名称为username，则我们就可以使用${param.username}来取得用户填入的值。

**看到这里，大家应该很明确EL表达式只能通过内置对象取值，也就是只读操作，如果想进行写操作的话就让后台代码去完成，毕竟EL表达式仅仅是视图上的输出标签罢了。**