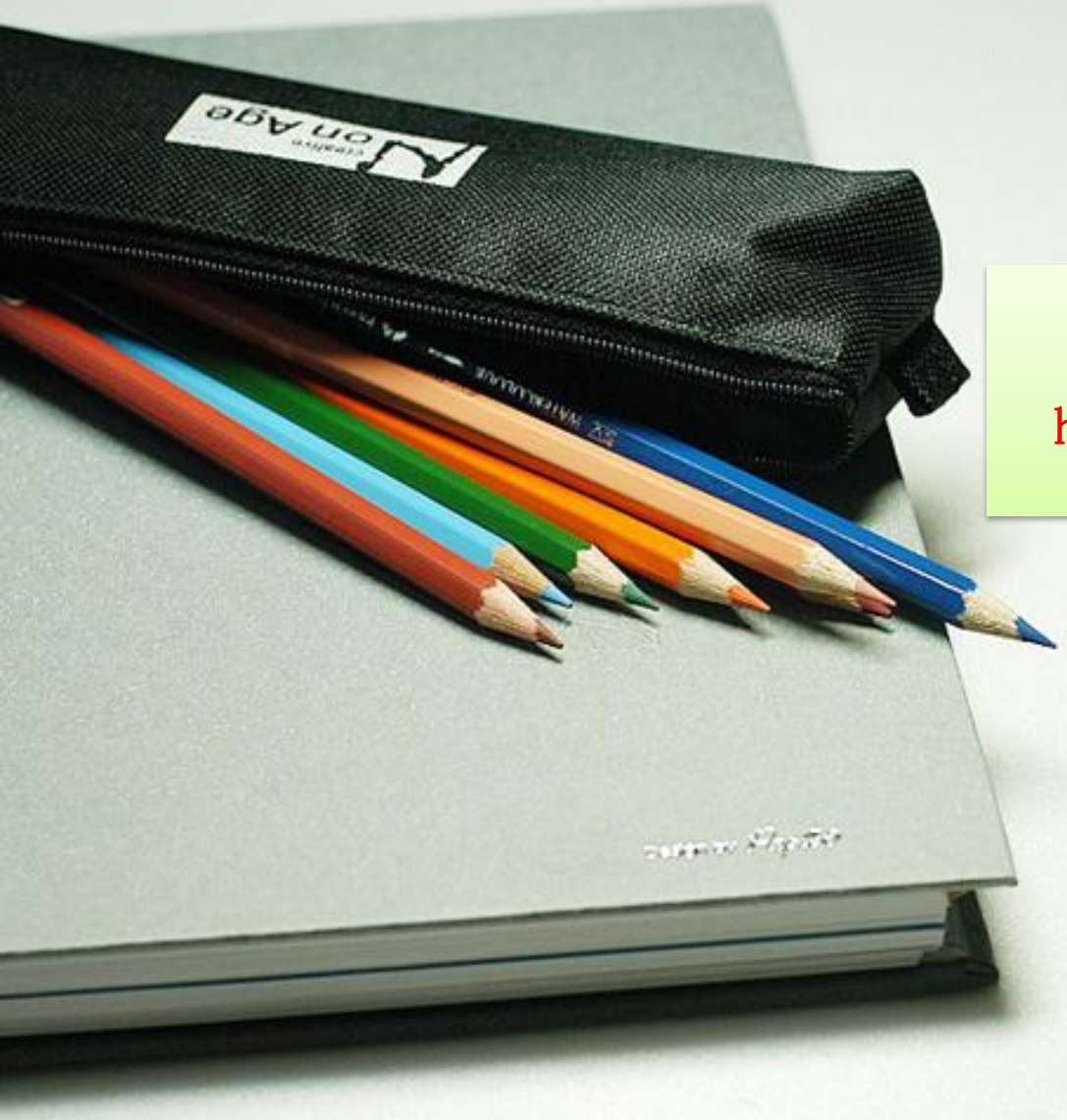

北京圣思园科技有限公司
http://www.shengsiyuan.com

主讲人：张龙

Expression Language

The purpose of EL is to aid in producing scriptless JSP pages.

- Syntax of EL in a JSP page:

  `${expr}`
- You can escape the expression:

  `\${expr}`
- Expressions can be used in two ways:
  - Attribute values in custom and standard actions
  - Within template text

Expression Language

Beans within the namespace available to the JSP can be accessed easily using EL.

- Beans can be accessed by way of dot notation:

  `${bean.attribute}`
- Beans can be located by searching through the scopes: page, request, session and application
- Bean scope can be specified by preceding the bean name with the scope

  `${sessionScope.cust.firstName}`

EL 默认对象

Implicit Object	Description
pageContext	The PageContext object
pageScope	A Map containing page-scoped attributes and their values
requestScope	A Map containing request-scoped attributes and their values
sessionScope	A Map containing session-scoped attributes and their values
applicationScope	A Map containing application-scoped attributes and their values
param	A Map containing request parameters and single string values

EL 默认对象

Implicit Object	Description
paramValues	A Map containing request parameters and their corresponding string arrays
header	A Map containing header names and single string values
headerValues	A Map containing header names and their corresponding string arrays
cookie	A Map containing cookie names and their values

Expression Language

For example,

```
${param.username}
```

If the bean returns an array, and element can specify its index using [] notation:

```
${paramValues.fruit[2]}
```

运算符

Five arithmetic operators defined:

Arithmetic Operation	**Operator**
Addition	+
Subtraction	-
Multiplication	*
Division	/ and div
Remainder	% and mod

运算符

Example operations:

EL Expression	Result
`${3 div 4}`	0.75
`${1 + 2 * 4}`	9
`${(1 + 2) * 4}`	12
`${32 mod 10}`	2

运算符

EL has six comparison operators:

Comparison	Operator
Equals	== and eq
Not equals	!= and ne
Less than	< and lt
Greater than	> and gt
Less than or equal	<= and le
Greater than or equal	>= and ge

运算符

- EL has three logical operators

Logical Operation	Operator
and	&& and and
or	|| and or
not	! and not

- Comparison and logical operations return a boolean
- Typically used as value for custom tag attribute
- Inserts `true` or `false` in output stream if used within template text

Expression Language

- EL 语法很简单，它最大的特点就是使用上很方便。
- ${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( );

.与[ ]运算符

- EL 提供．和 [ ] 两种运算符来存取数据。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。

两者差异

- (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"] }

两者差异

- 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。

EL 变量

- EL 存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，但是假如全部的范围都没有找到时，就回传null

EL 变量

属性范围	在 EL 中的名称
Page	PageScope
Request	RequestScope
Session	SessionScope
Application	ApplicationScope

EL 变量

- 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

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
- ${param.count + 20}
- 会将传来的count自动转化为数值

EL 保留字

And	eq	gt	true
Or	ne	le	false
No	lt	ge	null
instanceof	empty	div	mod

EL 隐含对象

隐含对象	类　　型	说　　明
PageContext	javax.servlet.ServletContext	表示此 JSP 的 PageContext
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

EL 隐含对象

隐含对象	类　　型	说　　明
paramValues	java.util.Map	如同

隐含对象	类　　型	说　　明
		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

EL 隐含对象

- 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，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

EL 隐含对象(见程序)

- ${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

pageContext

- 我们可以使用 ${pageContext}来取得其他有关用户要求或页面的详细信息

Expression	说　明
${pageContext.request.queryString}	取得请求的参数字符串
${pageContext.request.requestURL}	取得请求的 URL，但不包括请求之参数字符串
${pageContext.request.contextPath}	服务的 web application 的名称

pageContext(见程序)

${pageContext.request.method}	取得 HTTP 的方法(GET、POST)
${pageContext.request.protocol}	取得使用的协议(HTTP/1.1、HTTP/1.0)
${pageContext.request.remoteUser}	取得用户名称
${pageContext.request.remoteAddr }	取得用户的 IP 地址
${pageContext.session.new}	判断 session 是否为新的，所谓新的 session，表示刚由 server 产生而 client 尚未使用
${pageContext.session.id}	取得 session 的 ID
${pageContext.servletContext.serverInfo}	取得主机端的服务信息

EL Functions

- 参见Tomcat自带的例子