个人总结:%{}用于在标签的属性值被理解为字符串类型时，告诉执行环境%{}里的是OGNL表达式。注意是在Struts2的标签中,如果实在正常的jsp标签中获取某个变量的值,需要嵌套Struts2的标签或直接使用el表达式.

[**struts2中%{}与${}与#区别**](http://blog.csdn.net/dreamwbt/article/details/70313663)

2017-04-21 13:21 598人阅读 [评论](http://blog.csdn.net/dreamwbt/article/details/70313663#comments)(0) [收藏](javascript:void(0);) [举报](http://blog.csdn.net/dreamwbt/article/details/70313663#report)

http://static.blog.csdn.net/images/category_icon.jpg 分类：

Struts2（32） http://static.blog.csdn.net/images/arrow_triangle%20_down.jpg

版权声明：本文为博主原创文章，未经博主允许不得转载。

提示：有参考其他博客

在struts中符号的使用比较灵活，尤其是“%”、“#”、“$”这三个符号的使用。由于$广泛应用于EL中，这里重点写%和#符号的用法。

1、“#”符号有三种用途：     
(1)、访问非根对象（struts中值栈为根对象），OGNL在不添加任何符号的情况下，默认访问的是ValueStack中的值，也就是说是Action中的属性和方法；如OGNL上下文和Action上下文，#相当于ActionContext.getContext()；下表有几个ActionContext中有用的属性：     
名称             作用                                       例子        
parameters   包含当前HTTP请求参数的Map                      #parameters.id[0]作用相当于request.getParameter("id")      
      
request      包含当前HttpServletRequest的属性（attribute)的Map   #request.userName相当于request.getAttribute("userName")      
      
session      包含当前HttpSession的属性（attribute）的Map    #session.userName相当于session.getAttribute("userName")       
application  包含当前应用的ServletContext的属性（attribute）的Map   #application.userName相当于application.getAttribute("userName")      
      
注：attr 用于按request > session > application顺序访问其属性（attribute），#attr.userName相当于按顺序在以上三个范围（scope）内读取userName属性，直到找到为止。用于过滤和投影（projecting)集合，如books.{?#this.price<100}；构造Map，如#{'foo1':'bar1', 'foo2':'bar2'}。     
    
(2)、用于过滤和投影（projecting)集合，如: books.{?#this.price>35}       
books.{?#this.price>35}     
(3)、构造Map，如: #{'foo1':'bar1', 'foo2':'bar2'}       
#{'foo1':'bar1', 'foo2':'bar2'}这种方式常用在给radio或select、checkbox等标签赋值上。如果要在页面中取一个map的值可以这样写：      
<s:property value="#myMap['foo1']"/>       
<s:property value="#myMap['foo1']"/>   

2、“%”符号的用途是在标签的属性值被理解为字符串类型时，告诉执行环境%{}里的是OGNL表达式。让被理解为字符串的表达式，被真正当成ognl来执行。

举例：

<s:set name="myMap" value="#{'key1':'value1','key2':'value2'}"/>        
<s:property value="#myMap['key1']"/>        
<s:url value="#myMap['key1']" />      
<s:set name="myMap" value="#{'key1':'value1','key2':'value2'}"/>     
<s:property value="#myMap['key1']"/>     
<s:url value="#myMap['key1']"/>

上面的代码第2行会在页面上输出“value1”，而第3行则会输出"#myMap['key1']"这么一个字符串。原因：在property标签中，value之后的字符串会自动被解析为OGNL表达式，但是URL标签在被struts解析的时候默认当做字符串对待，所以这里输出的是原字符串 。

如果将第3行改写成这样：

<s:url value="%{#myMap['key1']}"/>       
<s:url value="%{#myMap['key1']}"/>     
则输出为“value1”。

由此说明struts2里不同的标签对ognl的表达式的理解是不一样的。当有些标签中的字符串对象不被理解成表达式的时候就可以通过%{}来显式指定它的属性为OGNL表达。

3、“$”有两种用途

(1)、在国际化资源文件中，引用OGNL表达式。     
(2)、在Struts 2配置文件中，引用OGNL表达式：      
<action name="saveUser" class="userAction" method="save">     
<result type="redirect">listUser.action?msg=${msg}（注意：这里的${}不是EL表达式，而是OGNL表达式，他只是在不同的地方定义的解析字符不一样而已）</result>     
</action>     
<action name="saveUser" class="userAction" method="save">         
<result type="redirect">listUser.action?msg=${msg}</result>     
</action>  
  
EL表达式:  
1.基本格式：${}  
2.4种取值范围：pageScope，requestScope，sessionScope，applicationScope  
   ${requestScope.user.name}  
3.基本运算：.和[]  .用于取属性[]用于取Array，List，Map，Set  
   ${requestScope.map['apple']},${requestScope.list[1]}  
4.{}里允许的运算：算术运算，关系运算，逻辑运算，empty/not empty空判断  
   ${1+1}  
   ${user.age<18}  
   ${(user.age>18)&&(user.sex=='男')}  
   ${empty user}判断user对象是否为空  
5.EL的11个隐含对象  
   pageContext,param和paramValues,header和headerValues，cookie,initParam和4种取值范围对象  
    \*pageContext可获取jsp的request,response,out,session,config,servletContext等对象  
       ${pageContext.session.uesr}  
    header  
       ${header.host}  
    \*cookie  
       ${cookie.key}