struts2标签（转载）

1：输出action中属性值，在jsp页面中显示用：<s:property value="属性名" />

2：在页面中实现自动增加的序号用iterator的statuts的index属性 eg：

<s:iterator value="#request.inOutAccountList" id="data" status="listStat">

<s:property value="#listStat.index+1"/>

<!—输出当前迭代元素的索引à

<s:property value="#listStat.index"/>

<!—输出当前迭代了几个元素à

<s:property value="#listStat.count"/>

<!—返回当前迭代元素的索引是否为奇数à

<s:property value="#listStat.odd"/>

<!—返回当前迭代元素的索引是否为偶数à

<s:property value="#listStat.event"/>

<!—返回当前元素是否为第一个元素à

<s:property value="#listStat.first"/>

<!—返回当前元素是否为最后一个元素à

<s:property value="#listStat.last"/>

</s:iterator>

3:在action类中取得request和session对象的方法

Map session = ActionContext.getContext().getSession();

HttpServletRequest request = ServletActionContext.getRequest ();

设置它们的值的方法:

session.put("operation", "add");

request.setAttribute("name", name); 页面中取得它们的值:

<s:property value="#session.operation"/>

<s:property value="#session[‘operation’]"/>

${sessionScope.TEST} == ${session.TEST}

<s:if test="#session.operation =='add'">

<s:if test="#session[‘operation’]=='add'">

<s:if test="#session[‘operation’]==null">

<s:property value="#request.name"/>

${requestScope.name} == ${request.name}

4:页面中奇偶行样式不一样的控制方法：

<tr class="<s:if test='#listStat.odd==true'>tableStyle-tr1</s:if><s:else>tableStyle-tr2</s:else>" >

5:单选框和复选框的使用方法

1):可以设置默认选中值，注意list的值的设置，通过这种方式使key和value不一样，这种方法比较常用(checkboxlist or radio)

<s:radio name="uncarInsPolicy.policyStateCode"

list="#{'5':'通过' , '2':'不通过'}" listKey="key" listValue="value" value='5' />

2):这里的key和value的值是一样的(checkboxlist or radio)

<s:checkboxlist list="{'Red', 'Blue', 'Green'}" name="favoriteColor"/>

6:struts2 中的标签会生成类似由<tr><td></td></tr>构成的字串（具体什么标签生成什么，可以查 看生成后的页面的源代码）如果不限制这些多余代码的生成，页面将变得无法控制，所以一般我们是不希望它生成多余的代码的，具体的设置方法如果，在 struts.xml中统一配置

<constant name="struts.ui.theme" value="simple"/>加上该句即可

也可以通过在页面中将tag的theme属性设为"simple"取消其默认的表格布局

不过最好是：自定义一个theme，并将其设为默认应用到整个站点，如此一来就可以得到统一的站点风格

7：jsp页面中格式化日期的方法 <s:date name="unCarInsModificationInfo.createTime" format="yyyy-MM-dd" nice="false"/>这样就可以将日期格式化为yyyy－MM－dd的形式

8：默认情况下，当请求action发生时，Struts运行时（Runtime）根据struts.xml里的Action映射集(Mapping)，实例化action对应的类，并调用其execute方法。当然，我们可以通过以下两种方法改变这种默认调用 1）在classes/sturts.xml中新建Action，并指明其调用的方法

比如想调用action类中的

public String aliasAction() {

message ="自定义Action调用方法";

return SUCCESS;

}

则在classes/sturts.xml中加入下面代码：

<action name="AliasHelloWorld" class="tutorial.HelloWorld"method="aliasAction">

<result>/HelloWorld.jsp</result>

</action>

既可用action名调用该方法了

2）（比较常用）

访问Action时，在Action名后加上“!xxx”（xxx为方法名）。

9：Struts 2.0有两个配置文件，struts.xml和struts.properties都是放在WEB-INF/classes/下。

struts.xml用于应用程序相关的配置

struts.properties用于Struts 2.0的运行时（Runtime）的配置

10:在action类中取得web下某一文件夹物理路径（绝对路径）的方法

filePath = ServletActionContext.getServletContext().getRealPath("/upLoadFiles")

11：要想返回的页面不是一个直接JSP页面而是要先通过返回action中的方法读取相应的数据再返回到jsp页面，有两种方法

1）在struts.xml中这么设置

<result name="list" type="redirect-action">sysmanage/UserBaseInfoAction!findUserBaseInfo.action</result>

2）在action中返回时直接调用这个方法即可

return findList（）；

12：设置checkboxlist中默认值的方法

<s:checkboxlist name="skills1"="Skills 1" list="{ 'Java', '.Net', 'RoR', 'PHP' }" value="{ 'Java', '.Net' }" />

<s:checkboxlist name="skills2"

label="Skills 2" list="#{ 1:'Java', 2: '.Net', 3: 'RoR', 4: 'PHP' }" listKey="key" listValue="value" value="{ 1, 2, 3 }"/>

13：二级级连下拉框 <s:set name="foobar"

value="#{'Java': {'Spring', 'Hibernate', 'Struts 2'}, '.Net': {'Linq', ' ASP.NET 2.0'}, 'Database': {'Oracle', 'SQL Server', 'DB2', 'MySQL'}}" />

<s:doubleselect list="#foobar.keySet()" doubleName="technology" doubleList="#foobar[top]" label="Technology" />

14:在页面中取出list对象中的属性值 <s:if test="list!=null&&list.size()>0">

<s:iterator value="list" status=”stat”>

<!—如果list中存储形式为select id,name,title-- >

<s:property value="list[#stat.index][0]" />

<!—如果list中存储多对象如left join Role left join Group-- >

<s:property value="list[#stat.index][0].title" />

<s:property value="list[#stat.index][1].title" />

<!—如果为Map形式存储-- >

<s:iterator value="mapUsers.keySet()" id="id">

<s:property value="#id"/>

<s:iterator value="mapUsers.get(#id)">

<s:property value="nickname"/>||<s:property value="UName"/>

</s:iterator>

</s:iterator>

<!—判断是否为奇数-- >

<s:if test="stat.odd == true"></s:if>

<!—判断是否为第一条数据-- >

<s:if test="#stat.index==0">

<s:if test="#stat.first">

<!—判断属性是否为null-- >

<s:if test="list[#stat.index][0].title ==null”>

<!—将属性值声明为变量-- >

<s:set name="title" value="list[#stat.index][0].title " ></s:set>

<!—判断属性值-- >

<s:if test="list[#stat.index][0].title ==4">

<s:if test="list[#stat.index][0].title ==’admin’">

<s:if test="#title ==4">

<s:if test="list[#stat.index][0].title.length()>15">

<s:property value="list[#stat.index][0].title.substring(0,15)" />...

</s:if>

14:获取select值

<s:select list="listDept" name="selectDept" listKey="val" listValue="prntext" disabled = "true" value="deptSelected" />

<s:hidden name="selectDept" value="%{deptSelected}"/>

15:循环指定int型数据

其中first属性指定循环起始值，last指定循环终止值，其它相关属性可以查看org.apache.struts2.util.Counter类源码。在下面迭代器中输入循环的当前值，即：current

<s:bean name="org.apache.struts2.util.Counter">

<s:param name="first" value="1" />

<s:param name="last" value="pages" /><!—pages page为action中属性à

<s:iterator>

<s:if test="current-1==page">

<font color="red"><s:property /></font>

</s:if>

<s:else>

[<s:property />]

</s:else>

</s:iterator>

</s:bean>

16:输出国际化文件中的值

<s:text name=”key”>

${getText[“key”]}

17:输出Action中的消息

<s:actionmessage />

18:输出Action中的错误

<s:actionerror /> 引用(0) 推荐(0) 评论(0)阅读(10) 上一篇：IE下的js调试工具companion.js（转载）

A：

<s:a xhref=""></s:a>-----超链接，类似于html里的<a></a>

<s:action name=""></s:action>-----执行一个view里面的一个action

<s:actionerror/>-----如果action的errors有值那么显示出来

<s:actionmessage/>-----如果action的message有值那么显示出来

<s:append></s:append>-----添加一个值到list，类似于list.add();

<s:autocompleter></s:autocompleter>-----自动完成<s:combobox>标签的内容，这个是ajax

B：

<s:bean name=""></s:bean>-----类似于struts1.x中的，JavaBean的值

C：

<s:checkbox></s:checkbox>-----复选框

<s:checkboxlist list=""></s:checkboxlist>-----多选框

<s:combobox list=""></s:combobox>-----下拉框

<s:component></s:component>-----图像符号

D：

<s:date/>-----获取日期格式

<s:datetimepicker></s:datetimepicker>-----日期输入框

<s:debug></s:debug>-----显示错误信息

<s:div></s:div>-----表示一个块，类似于html的<div></div>

<s:doubleselect list="" doubleName="" doubleList=""></s:doubleselect>-----双下拉框

E：

<s:if test=""></s:if>

<s:elseif test=""></s:elseif>

<s:else></s:else>-----这3个标签一起使用，表示条件判断

F：

<s:fielderror></s:fielderror>-----显示文件错误信息

<s:file></s:file>-----文件上传

<s:form action=""></s:form>-----获取相应form的值

G：

<s:generator separator="" val=""></s:generator>----和<s:iterator>标签一起使用

H：

<s:head/>-----在<head></head>里使用，表示头文件结束

<s:hidden></s:hidden>-----隐藏值

I：

<s:i18n name=""></s:i18n>-----加载资源包到值堆栈

<s:include value=""></s:include>-----包含一个输出，servlet或jsp页面

<s:inputtransferselect list=""></s:inputtransferselect>-----获取form的一个输入

<s:iterator></s:iterator>-----用于遍历集合

L：

<s:label></s:label>-----只读的标签

M：

<s:merge></s:merge>-----合并遍历集合出来的值

O：

<s:optgroup></s:optgroup>-----获取标签组

<s:optiontransferselect doubleList="" list="" doubleName=""></s:optiontransferselect>-----左右选择框

P：

<s:param></s:param>-----为其他标签提供参数

<s:password></s:password>-----密码输入框

<s:property/>-----得到'value'的属性

<s:push value=""></s:push>-----value的值push到栈中,从而使property标签的能够获取value的属性

R：

<s:radio list=""></s:radio>-----单选按钮

<s:reset></s:reset>-----重置按钮

S：

<s:select list=""></s:select>-----单选框

<s:set name=""></s:set>-----赋予变量一个特定范围内的值

<s:sort comparator=""></s:sort>-----通过属性给list分类

<s:submit></s:submit>-----提交按钮

<s:subset></s:subset>-----为遍历集合输出子集

T：

<s:tabbedPanel id=""></s:tabbedPanel>-----表格框

<s:table></s:table>-----表格

<s:text name=""></s:text>-----I18n文本信息

<s:textarea></s:textarea>-----文本域输入框

<s:textfield></s:textfield>-----文本输入框

<s:token></s:token>-----拦截器

<s:tree></s:tree>-----树

<s:treenode label=""></s:treenode>-----树的结构

U：

<s:updownselect list=""></s:updownselect>-----多选择框

<s:url></s:url>-----创建url

文章出处：DIY部落(<http://www.diybl.com/course/3_program/java/javajs/20090426/165550.html>)

（1）：<s:textfield> ---- 文本输入框

使用：<s:textfield name=”实体Bean。属性”></s:textfield>

（2）：<s:textarea> ----- 文本域输入框

使用：<s:textarea name=”实体Bean。属性”></s:textarea>

（3）：<s:password> ----- 密码输入框

使用：<s:password name=”实体Bean。属性”></s:password>

前三个基本都是一样的。如果是要显示值的话可以这样：value = “实体Bean。Get\*\*\*()”。

（4）：<s:radio list=””> ---- 单选按钮

使用：<s:radio list=””>其中list 属性是必须要有的。

<1>:第一种方式：list = “#{‘male’:’男’,’female’:’女’}”

<2>:第二中方式：list = “#request/session.list”。 ---- 与action结合起来了。

如果要默认选中的话后面加上value = “”.也可以通过javascript的方式来默认选中。

（5）：<s:url/> --- url连接

<s:a/> --- 超链接

这两个标签一般结合起来来使用。

使用：<s:url id = “id” action = “userAction”/><s:a href=”%{id}”/>

（6）：<s:form/> --- 获取相应form的值

使用：

<1>：<s:form action = “userAction”>

<2>：<s:url id=”userId” action = “userAction”> -- 定义了一个userId的指向userActionde 路径

<s:form action=”%{userId}”>

（7）：<s:submit/> ---- 提交标签

（8）：<s:reset/> ---- 重置标签

（9）：<s:hidden/> ---- 隐藏域标签

使用：<s:hidden name=”实体Bean。属性” value=”%{#request/session.实体Bean。属性}”/>

（10）：<s:combobox/> ---- 下拉框配合输入框一起使用

使用：<s:combobox name=”age” list=””>

<1>:list = “{18,20,30}”

<2>:list = “#request/session.list”

（11）：<s:checkbox/> ---- 多选框

使用：<s:checkbox name=”自己随便起” value = “值” > 足球

（12）：<s:checkboxlist/> ---- 复选框

使用：<s:checkboxlist name=”自己随便起” label = “标签前面的名字” list=””>

<1>：list = “{‘hibernate’, ‘spring’, ‘strust2’}”

<2>：list = “#request/session.list”.

如果想默认选中的话，那么加上value=””

（13）：<s:if test=""></s:if>

<s:elseif test=""></s:elseif>

<s:else></s:else> ---- -这3个标签一起使用，表示条件判断

使用：

<s:if test="%{false}">

<div>Will Not Be Executed</div>

</s:if>

<s:elseif test="%{true}">

<div>Will Be Executed</div>

</s:elseif>

<s:else>

<div>Will Not Be Executed</div>

</s:else>

（14）：<s:div/> --- 表示一个块，类似与html中的div

（15）：<s:generator/> ---- 一般和<s:iterator/>一起使用。

使用：

<s:generator separator = “,” val=”%{aaa, bbb, ccc, ddd, eee}”>

<s:iterator>

<s:property/><br />

</s:iterator>

</s:generator>

<s:iterator value=”days” status=”d”>

<s:property name=”d.\*\*\*”/>

</s:iterator>

Days表示：在action中存的值，status ：表示起的别名。<s:property name=”d.\*\*\*”>循环的一个个属性。

（16）：<s:select/> ---- 下拉框的使用

使用：

<s:select label=”请选择” list=”{‘book’, ‘pen’, ‘moon’}” value=”%{‘pen’}”>

Value : 表示默认值。

（17）：<s:bean/> ----- Bean标签，当然需要一个JavaBean。它的属性值的操作是经由Bean标签中的参数属性来进行赋值。当然，它还有一个id属性可以进行赋值，这样就可以在上下文中使用这个Bean.

使用：<s:bean name=”le.tks.Books” id=”book”>

<s:param name=”bookName”>jsf</s:param>

<s:property name=”%{bookname}”/>

</s:bean>

页面输出的结果：jsf。

（18）：<s:date/> ---- 方便在页面中进行格式化的输出。

使用：

<s:date name=”currentDate” format=”dd/MM/yyyy”>

页面中显示的结果：24/09/2008. 格式还有很多：MM/dd/yyyy、MM/dd/yyyy hh:mm:ss等。

（19）：<s:include/> ---- 包含标签,是把这个页面中的所有的内容都包含进来。

使用：<s:include name=”/\*\*/\*\*/\*\*.jsp”/>

（20）：<s:param/> --- param标签用于传递参数，如给<s:bean>标签传递参数，它有两个属性：

<1>:name(String)：参数名。

<2>：value(Object)：参数值。

使用：

<a href="

<s:url action="book">

<s:param name="gId" value="20"></s:param>

<s:param name="operate" value="50"></s:param>

<s:param name="projectName" value="30"></s:param>

</s:url>">

insert</a>

（21）：<s:set/> --- Set标签用户将某一值赋给某一变量，因此，任何对该项值的引用都可以通过该变量来得到该值.

使用：

<s:set name="teacher" value="%{'数学高级教师'}"/>

教师职称：<s:property value="#teacher"/>

页面显示：教师职称：数学高级教师。

（22）：<s:token/> ---- 防止重复提交

使用：在页面加载时，<s:token/>会产生一个GUID（Globally Unique Identifier，

全局唯一标识符）值的隐藏输入框如：

<input type="hidden" name="struts.token.name" value="struts.token"/>

<input type="hidden" name="struts.token" value="BXPNNDG6BB11ZXHPI4E106CZ5K7VNMHR"/>

放在页面中随便的一个地方。<s:token/>

发表于 @ 2009年03月16日　10:03:00 | 评论( 0 ) | 编辑| 举报| 收藏

新一篇:Java学习之Iterator(迭代器)的一般用法 （转）

本文来自CSDN博客，转载请标明出处：http://blog.csdn.net/nerain/archive/2009/03/16/3993576.aspx