Jsp标签分类

1. 内置标签，不用导入
2. Jstl标签。需要导入
3. 自定义标签

动作标签

转发标签

<jsp:forward page=””>

//url参数

<jsp:param name=”” value=”” />

</jsp:forward>

包含标签，属于动态包含

<jsp:include page=”” />

静态包含不能传参数，动态可以

JSTL标签库

核心标签库

国际化标签库

EL函数库

使用JSTL标签库步骤

1. 导入jstl支持的jar包，

使用javaee项目自动导入jstl支持jar包

1. 使用tablib指令导入标签库

<%@tablib uri=”tld文件的uri名称” prefix=”简写” %>

核心标签库的重点标签

保存数据

<c:set></c:set>

Ep:

这个标签把变量保存到域对象中，默认保存到page域

<c:set var=”变量名” value=”变量值” scope=”变量作用域” ></c:set>

获取数据

<c:out value=”” ></c:out>

Ep：

<c:out value=”${变量名}” default=”变量值为null的时候显示的值” escapeXml=”false” ></c:out>

escapeXml : 默认为true， 是否对value进行转义，默认转义

单条件判断,没有elseif

<c:if test=””> </c:if>

ep：

<c:if test=”${empty msg}” ></c:if>

多条件判断

<c:choose></c:choose>

<c:when test=”” ></c:when>

<c:otherwise></c:otherwise>

Ep:

<c:choose> //开始多条件判断

<c:when test=”${age > 0 && age <= 10}” > 相当于 if elseif

</c:when>

<c:when test=”${age >1 0 && age <= 20}” >

</c:when>

<c:when test=”${age >2 0 && age <= 30}” >

</c:when>

<c:when test=”${age > 30 && age <= 40}” >

</c:when>

<c:otherwise> 相当于else

</c:otherwise>

</c:choose>

循环数据

<c:forEach></c:forEach>

Ep：

<%--

begin=”” 从哪个元素开始遍历

end=”” 到哪个元素结束

step=”” 步长

items=”” 需要遍历的数据

var=”” 每个元素的名称

varStatus=”” 当前正在遍历的元素的状态对象

--%>

<c:forTokens items=”” delims=”” ></c:forTokens> //循环特殊字符串，能分割字符串

Ep：

String str = “1-2-3-4-5-5”;

<c:forTokens items=”${str}” delims=”分隔符” var=”每个遍历的结果” ></c:forTokens>

重定向

<c:redirect url=”地址”></c:redirect>

自定义标签

自定义标签步骤

1. 创建一个类文件，继承SimpleTagSupport类
2. 覆盖doTag()方法
3. 在web项目的WEB-INF目录下创建tld文件
4. jsp页面导入标签库并使用

Ep

public class TestTag extends SimpleTagSupport {

pulic void doTag() throws JspException, IOException{

//向浏览器输出客户的ip地址

PageContext pageContext = (PageContext)this.getJspContext();

HttpServletRequest request = (HttpServletRequest)pageContext.getRequest();

String ip = request.getRemoteHost();

JspWriter out = pageContext.getOut();

out.write("使用自定义标签输出客户的IP地址："+ip);

}

}

自定义标签处理器类的生命周期

SimpleTag接口

void setJspContext(JspContext pc ) (必调用)

设置pageContext对象，传入pageContext，通过getJspContext（）方法获取到pageContext对象

void setParent( JapTag parent)

设置父标签对象，传入父标签对象，如果没有父标签，则不调用此方法，通过getParent（）方法获取父标签对象

void setJspBody(JspFragment jspBdoy) 设置标签题内容，如果没有标签体内容则不调用此方法

void doTag() 执行标签调用的方法

自定义标签的作用

1. 控制标签体内容是否输出
2. 控制标签体余下内容是否输出
3. 控制重复输出的标签体内容
4. 改变标签体内容
5. 带属性的标签

案例 ， 这个案例对应1,2,3,4作用

public void doTag() throws JspException , IOException{

//得到标签体内容

JspFragment jspBdoy = this.getJspBody();

1,执行invoke方法就能把标签体内容输出到浏览器中，不调用就不输出，执行多次就输出多次

JspWriter out = this.getJspContext().getOut();

jspBody.invoke(out输出流);

jspBosy.invoke(null); 等价于上面两行代码

如果是需要手动输出的

this.getJspContext().getOut().Write(要输出的内容);

2,如果不想输出内容就抛出异常

throw new SkipPageException();

直接输出内容用out.Write(); 上面那个是输出标签体内容的

}

如果标签要加属性

在tld文件里配置

<attribute>

<name>age</name> 标签名称

<required>true</required> 是否必须有内容

<rtexprvalue>false</rtexprvalue> 是否使用el表达式

</attribute>

然后在对应的标签类里设置属性的get set方法即可

tld文件的<body-content></body-content>是标签体的内容格式

jsp ： 可执行jsp和java代码

scriptless：不能写jsp和java代码

empty：必须是空标签

tagdependent：可以写jsp和java代码但是不执行