

JSTL

EL表达式 和 JSTL 结合，一起解决jsp页面中，大量Java代码(JSP片段)的问题

JSTL

JSTL标签库的使用是为类弥补html表的不足，规范自定义标签的使用而诞生的。在告别modle1模式开发应用程序后，人们开始注重软件的分层设计，不希望在jsp页面中出现java逻辑代码，同时也由于自定义标签的开发难度较大和不利于技术标准化产生了自定义标签库。

JSTL (JSP StandardTag Library, JSP标准标签库)是一个不断完善的开放源代码的JSP标签库，是由apache的jakarta小组来维护的。JSTL只能运行在支持JSP1.2和Servlet2.3规范的及其以上容器上，如tomcat 7.x。在JSP 2.0中也是作为标准支持的。可以应用于各种领域，如：基本输入输出、流程控制、循环、XML文件剖析、数据库查询及国际化和文字格式标准化的应用等

JSTL所提供的标签函数库主要分为五大类：

- (1) 核心标签库 (Core tag library)
- (2) I18N 格式标签库 (I18N-capable formatting tag library)

JSTL所提供的标签函数库主要分为五大类：

- (1) 核心标签库 (Core tag library)
- (2) I18N 格式标签库 (I18N-capable formatting tag library)
- (3) SQL 标签库 (SQL tag library)
- (4) XML 标签库 (XML tag library)
- (5) 函数标签库 (Functions tag library)

JSTL

JSTL	前置名称	URI	范例
核心标签库	c	http://java.sun.com/jsp/jstl/core	<c:out>
I18N 格式标签库	fmt	http://java.sun.com/jsp/jstl/xml	<fmt:formatDate>
SQL 标签库	sql	http://java.sun.com/jsp/jstl/sql	<sql:query>
XML 标签库	xml	http://java.sun.com/jsp/jstl/fmt	<x:forBach>
函数标签库	fn	http://java.sun.com/jsp/jstl/functions	<fn:split>

下面对JSTL的各个标签库进行简单的介绍：

- (1) 核心标签库中包含了实现WEB应用中的通用操作的标签。例如，用于输出一个变量内容的<c:out>标签、用于条件判断的<c:if>标签、用于迭代循环的<c:forEach>标签。
- (2) 国际化/格式化标签库中包含实现WEB应用程序的国际化的标签。例如，设置JSP页面的本地信息、

(2) 国际化/格式化标签库中包含实现WEB应用程序的国际化的标签。例如，设置JSP页面的本地信息、设置JSP页面的时区、绑定资源文件，使本地敏感的数据（例如数值、日期等）按照JSP页面中设置的本地格式显示。

(3) 数据库标签库中包含用于访问数据库和对数据库中的数据进行操作的标签。例如，从数据源中获得数据库连接、从数据库表中检索数据等。由于在软件分层的开发模型中，JSP页面仅用作表现层，我们一般不在JSP页面中直接操作数据库，而是在业务逻辑层或数据访问层操作数据库，所以，JSTL中提供的这套数据库标签库没有多大的实用价值。

(4) XML标签库中包含对XML文档中的数据进行操作的标签。例如，解析XML文档、输出XML文档中的内容，以及迭代处理XML文档中的元素。因为XML广泛应用于WEB开发，对XML文档的处理非常重要，XML标签库使处理XML文档变得简单方便，这也是JSTL的一个重要特征。

JSTL的使用

JSTL的使用

如果我们要在程序中使用这些标签，首先我们要引入jar，这里我们要引入两个jar，taglibs-standard-spec-1.2.5.jar（相当于之前的jstl.jar，属于接口定义类）和taglibs-standard-impl-1.2.5.jar jar（相当于之前的standard.jar，属于实现类）（jstl-1.2.jar和standard-1.1.2.jar，）然后将这两个jar放入到WEB-INF下的lib文件夹中。然后在我们的jsp页面上要引入

一 jar包准备：

1 官网下载

<http://tomcat.apache.org/download-taglibs.cgi>

Standard-1.2.5

JSTL的使用

apache-tomcat-8.0.30-windows-x64\apache-tomcat-8.0.30\webapps\examples\WEB-INF\lib

 taglibs-standard-impl-1.2.5.jar	2015-12-1 22:31	Executable Jar File	202 KB
 taglibs-standard-spec-1.2.5.jar	2015-12-1 22:31	Executable Jar File	40 KB

二 在jsp页面引入jstl标签库

<%@ taglib uri="http://java.sun.com/jsp/jstl/core" prefix="c"%>

JSTL核心标签库

JSTL 核心标签库

JSTL的核心标签库标签共13个，从功能上可以分为4类：表达式控制标签、流程控制标签、循环标签、URL操作标签。使用这些标签能够完成JSP页面的基本功能，减少编码工作。

- (1) 表达式控制标签：`out`标签、`set`标签、`remove`标签、`catch`标签。
- (2) 流程控制标签：`if`标签、`choose`标签、`when`标签、`otherwise`标签。
- (3) 循环标签：`forEach`标签、`forTokens`标签。
- (4) URL操作标签：`import`标签、`url`标签、`redirect`标签。

标签	描述
<code><c:out></code>	用于在JSP中显示数据，就像 <code><%= ... %></code>
<code><c:set></code>	用于保存数据

JSTL 核心标签库

标签	描述
<code><c:out></code>	用于在JSP中显示数据，就像 <code><%= ... %></code>
<code><c:set></code>	用于保存数据
<code><c:remove></code>	用于删除数据
<code><c:catch></code>	用来处理产生错误的异常状况，并且将错误信息储存起来
<code><c:if></code>	与我们在一般程序中用的if一样
<code><c:choose></code>	本身只当做 <code><c:when></code> 和 <code><c:otherwise></code> 的父标签

JSTL 核心标签库

<code><c:import></code>	指令——把别的JSP文件引入到当前JSP文件中，通过此标签向其他JSP文件引入
<code><c:forEach></code>	基础迭代标签，接受多种集合类型
<code><c:forTokens></code>	根据指定的分隔符来分隔内容并迭代输出
<code><c:param></code>	用来给包含或重定向的页面传递参数
<code><c:redirect></code>	重定向至一个新的URL。
<code><c:url></code>	使用可选的查询参数来创造一个URL

```
<c:out value="HelloWorld"></c:out>
```

选择标签

选择标签

```
</body>
</html>
```

用于复杂判断的`<c:choose> <c:when> <c:otherwise>`类似于“if elseif else”

- ①`<c:choose>` 标签没有属性，类似于父标签 `<c:when>` `<c:otherwise>`类似于子标签
- ②`<c:when>` 标签等价于if语句，有一个test属性，该属性表示需要判断的条件
- ③`<c:otherwise>`标签等价于else，没有属性。

`<c:if>`没有对应的else 所以 c:choose 一列的可以完成 if else的控制系

例：

```
<%@ page language="java" import="java.util.*" pageEncoding="UTF-8"%>
```

循环标签

循环标签

<c:forEach>标签使用进行循环的控制标签

例：

<c:forEach item="\${list}" var="user" > \${user.name} </c:forEach>

<c:forEach>标签的属性和说明

属性	描述
items	进行循环的集合（可选）
begin	开始条件（可选）
end	结束条件（可选）

goodsServlet类

```
@WebServlet("/goodsServlet")
public class goodsServlet extends HttpServlet {
    protected void doPost(HttpServletRequest request, HttpServletResponse response)
        throws ServletException, IOException {
        doGet(request, response);
    }

    protected void doGet(HttpServletRequest request, HttpServletResponse response)
        throws ServletException, IOException {
        goodsService goodsService = new goodsServiceImpl();

        //获取商品列表
        List<Goods> goodsList = goodsService.findAllGoods();
        Goods goods = goodsService.findOneGoods(1); //获取一个商品信息
        //将商品列表存放到request中
        request.setAttribute("goodsList", goodsList);
        request.setAttribute("goods", goods); //将商品存放到request中
        request.getRequestDispatcher("/JSTL.jsp").forward(request, response);
    }
}
```

JSTL.jsp

```
<%@ page contentType="text/html; charset=UTF-8" language="java" %>
<%@taglib prefix="c" uri="http://java.sun.com/jsp/jstl/core" %>
<html>
<head>
    <title>商品展示</title>
</head>
<body>
```

```

<%--输出标签--%>
你好:<c:out value="Hello JSTL"></c:out><br>
<c:set var="test" value="test的值"></c:set>
${test}<br>
<c:out value="${test}"></c:out><br><br>
<%-----%>

<%
//商品信息列表
List<Goods> goodsList = (List<Goods>)request.getAttribute("goodsList");
String username = "李沛";
session.setAttribute("username",username);
String sex = "1";
request.setAttribute("sex",sex);
%>
<%--选择标签 test:为判断条件--%>
<%--if标签--%>
<c:if test="${sessionScope.username == '李沛'}" var="flag">
    欢欢迎您:<c:out value="${username}"></c:out>
</c:if><br><br>
<%--choose标签--%>
<c:choose>
    <c:when test="${requestScope.sex == 1}">
        男
    </c:when>
    <c:when test="${requestScope.sex == 2}">
        女
    </c:when>
    <c:otherwise>
        性别未知
    </c:otherwise>
</c:choose><br><br>
<h3>商品列表信息</h3>
<%--循环标签--%>
<c:forEach items="${requestScope.goodsList}" begin="0" step="2" end="4"
var="str">
    ${str}
</c:forEach><br><br>
<%--格式化标签--%>
<%
    Date date = new Date();
    request.setAttribute("date",date);
%>
<fmt:formatDate value="${date}" pattern="yyyy-MM-dd hh:mm:ss"></fmt:formatDate>
<br>
<%--数字格式化--%>
<fmt:formatNumber value="1233" type="currency" pattern=".00">
</fmt:formatNumber><br>
<fmt:formatNumber value="1233.9999" pattern="#,#00.0#"></fmt:formatNumber><br>
<br>
<%--函数标签库--%>
${fn:toLowerCase("AAAAAAAAAAAAA")}>

```

```
</body>  
</html>
```

格式化标签

格式化标签

<fmt:formatDate>

<%--引入JSTL格式化标签库 --%>

<%@taglib uri="http://java.sun.com/jsp/jstl/fmt" prefix="fmt" %>

标签用于将日期格式化

I

例:

<%

pageContext.setAttribute("date", new Date());

%>

\${date }

| <fmt:formatDate value="\${date }" pattern="yyyy-MM-dd HH:mm:ss"/>

格式化标签

<fmt:formatDate>

<%--引入JSTL格式化标签库 --%>

<%@taglib uri="http://java.sun.com/jsp/jstl/fmt" prefix="fmt" %>

I

标签用于将日期格式化

例:

<%

pageContext.setAttribute("date", new Date());

%>

\${date }

| <fmt:formatDate value="\${date }" pattern="yyyy-MM-dd HH:mm:ss"/>

格式化标签

标签用于将日期格式化

例:

<%

pageContext.setAttribute("date", new Date());

%>

\${date }

| <fmt:formatDate value="\${date }" pattern="yyyy-MM-dd HH:mm:ss"/>

数字格式化

<fmt:formatNumber value="1234" type="currency" pattern="\$.00"/>

<fmt:formatNumber value="123456.7891" pattern="#,##0.0#"/>

格式化标签

0	代表一位数字
E	使用指数格式
#	代表一位数字，若没有则显示0
.	小数点
,	数字分组分隔符
:	分隔格式

Functions标签库

Functions标签库

由于在JSP页面中显示数据时，经常需要对显示的字符串进行处理，SUN公司针对于一些常见处理定义了一套EL函数库供开发者使用。这些EL函数在JSTL开发包中进行描述，因此在JSP页面中使用SUN公司的EL函数库，需要导入JSTL开发包，并在页面中导入EL函数库，如下所示：

```
<%@ taglib uri="http://java.sun.com/jsp/jstl/functions" prefix="fn" %>
```

Function标签库主要被用于EL表达式语句中，Functions为EL表达式语句提供了许多有用的功能。

Functions标签库，共16个函数

函数名	函数说明	使用举例
fn:contains	判断字符串是否包含另外一个字符串	<c:if test="\${fn:contains(name, searchString)}">
fn:containsIgnoreCase	判断字符串是否包含另外一个字符串(大小写无关)	<c:if test="\${fn:containsIgnoreCase(name, searchString)}">
fn:endsWith	判断字符串是否以某外字符串结束	<c:if test="\${fn:endsWith(filename, 'txt')}>

当当