jsp 页面中jstl 标签详解
分类： java 技术总结
jspjstlJavahtml
(?)[+]
fncontains 函数
fncontainsIgnoreCase 函数
fnstartsWith 函数
fnendsWith 函数
fnindexOf 函数
fnreplace 函数
fnsubstring 函数
fnsubstringBefore 函数
fnsubstringAfter 函数
fnsplit 函数
fnjoin 函数
fntoLowerCase 函数
fntoUpperCase 函数
fntrim 函数
fnescapeXml 函数
fnlength 函数
JSLT 标签库，是日常开发经常使用的，也是众多标签中性能最好的。把常用的内容，放在这里备份一份，随用随查。尽量做到不用查，就可以随手就可以写出来。这算是 Java 程序员的基本功吧，一定要扎实。
JSTL 全名为 JavaServer Pages Standard Tag Library，目前最新的版本为 1.1 版。JSTL 是由 JCP(Java Community Process)所制定的标准规范，它主要提供给 Java Web 开发人员一个标准通用的标签函数库。

Web 程序员能够利用 JSTL 和 EL 来开发 Web 程序，取代传统直接在页面上嵌入 Java 程序(Scripting)的做法，以提高程序的阅读性、维护性和方便性。

JSTL 1.1 必须在支持 Servlet 2.4 且 JSP 2.0 以上版本的 Container 才可使用
<%@ taglib %>引入标签库
==========
===
1、以 classPath 中，加入jar 包： standard-1.1.2.jar ， jstl-1.1.2.jar
2、在相目\WEB-INF\tld\文件夹中放入常用的 tld 文件：c.tld，fmt.tld
3、在 jsp 文件的顶部加入以下内容：
Java 代码
<%@ taglib uri="http://java.sun.com/jsp/jstl/core" prefix="c" %>
<%@ taglib uri="http://java.sun.com/jsp/jstl/fmt" prefix="fmt"%>
<%@ taglib uri="http://java.sun.com/jsp/jstl/functions" prefix="fn" %>
<c:> 核心标签库
==========

====
JSTL 核心标签库(C 标签)标签共有 13 个，功能上分为 4 类：
1.表达式控制标签：out、set、remove、catch
2.流程控制标签：if、choose、when、otherwise
3.循环标签：forEach、forTokens
4.URL 操作标签：import、url、redirect
<c:forEach> 标签

为循环控制，它可以将集合(Collection)中的成员循序浏览一遍。
<c:forEach> 标签的语法 说明 ：
语法 1：迭代一集合对象之所有成员
Html 代码
<c:forEach [var="varName"] items="collection" [varStatus="varStatusName"] [begin="begin"] [end="end"] [step="step"]>
本体内容
</c:forEach>
语法 2：迭代指定的次数
Html 代码
<c:forEach [var="varName"] [varStatus="varStatusName"] begin="begin" end="end" [step="step"]>
本体内容
</c:forEach>
<c:forEach> 标签的 属性说明 ：

名 称	说 明	EL	类型	必须	默认值
var	用来存放现在指到的成员	N	String	否	无
items	被迭代的集合对象	Y	Arrays Collection Iterator Enumeration Map String	否	无
varStatus	用来存放现在指到的相关成员信息	N	String	否	无
begin	开始的位置	Y	int	否	0
end	结束的位置	Y	int	否	最后一个成员
step	每次迭代的间隔数	Y	int	否	1

<c:forEach> 标签的 属性 ：varStatus 属性：它的提供另外四个属性:index,count,fist 和 last，它们个自的意义如下:
Java 代码
属性 类型 意义
index number 现在指到成员的索引
count number 总共指到成员的总和
first boolean 现在指到成员是否为第一个
last boolean 现在指到成员是否为最后一个
<c:forEach> 遍历 List 列表：

对于一个基本类型的数组，当前元素将作为相应包装类（Integer、Float 等等）的一个实例提供。

Html 代码

```
<c:forEach items="${domainList }" var="item">
<tr>
<td align="center" valign="middle">${item["domain"]==null?" ":item["domain"]}</td>
<td align="center" valign="middle"><fmt:formatDate value="${item['bind_date']}" pattern="yyyy-MM-dd HH:mm:ss"/></td>
<td align="center" valign="middle">
<c:if test="${item['domain']!=null}">
<a href="javascript:;" id="${item['domain']}" class="del"> </a>
</c:if>
</td>
</tr>
</c:forEach>
```

<c:forEach> 遍历 Map:

对于一个 java.util.Map，当前元素则作为一个 java.util.Map.Entry 提供。

Html 代码

```
<c:if test="${!empty permissionMap}">
<c:forEach items="${permissionMap}" var="item">
<tr>
<td>${item.value.id}</td>
<td>${item.value.urlOnClass}</td>
<td>${item.value.urlOnMethod}</td>
</tr>
</c:forEach>
</c:if>
```

<c:forTokens> 标签

用来浏览一字符串中所有的成员，其成员是由定义符号(delimiters)所分隔的。

<c:forTokens> 标签的语法 说明 ：

Html 代码

```
<c:forTokens items="stringOfTokens" delims="delimiters" [var="varName"]
[varStatus="varStatusName"] [begin="begin"] [end="end"] [step="step"]>
本体内容
</c:forTokens>
```

<c:forTokens> 标签的 属性说明 ：

名　称	说　明	EL	类　型	必 须	默认值
var	用来存放现在指到的成员	N	String	否	无
items	被迭代的字符串	Y	String	是	无
delims	定义用来分割字符串的字符	N	String	是	无
varStatus	用来存放现在指到的相关成员信息	N	String	否	无
begin	开始的位置	Y	int	否	0
end	结束的位置	Y	int	否	最后一个成员
step	每次迭代的间隔数	Y	int	否	1

<c:out> 标签

主要用来显示数据的内容

<c:out> 标签的语法 说明 ：

语法 1：没有本体(body)内容

Html 代码

```
<c:out value="value" [escapeXml="{true|false}"] [default="defaultValue"] />
```

语法 2：有本体内容

Html 代码

```
<c:out value="value" [escapeXml="{true|false}"]>
default value
</c:out>
```

<c:forEach> 标签的 属性说明 ： 略

一般来说，<c:out>默认会将<、>、’、” 和 & 转换为 <、>、'、" 和&。

假若不想转换时，只需要设定<c:out>的 escapeXml 属性为 fasle 就可以了。

<c:set> 标签

主要用来将变量储存至 JSP 范围中或是 JavaBean 的属性中。

<c:set> 标签的语法 说明 ：

语法 1：将value 的值储存至范围为 scope 的 varName 变量之中

Html 代码

```
<c:set value="value" var="varName" [scope="{ page|request|session|application }"]/>
```

语法 2：将本体内容的数据储存至范围为 scope 的 varName 变量之中

Html 代码

```
<c:set var="varName" [scope="{ page|request|session|application }"]>
… 本体内容
</c:set>
```

语法 3：将 value 的值储存至 target 对象的属性中

Html 代码

```
<c:set value="value" target="target" property="propertyName" />
```

语法 4：将本体内容的数据储存至 target 对象的属性中

Html 代码

```
<c:set target="target" property="propertyName">
… 本体内容
</c:set>
```

<c:set> 标签的 属性说明 ：

名　称	说　明	EL	类型	必须	默认值
value	要被储存的值	Y	Object	否	无
var	欲存入的变量名称	N	String	否	无
scope	var 变量的 JSP 范围	N	String	否	page
target	为一 JavaBean 或 java.util.Map 对象	Y	Object	否	无
property	指定 target 对象的属性	Y	String	否	无

<c:remove> 标签

主要用来移除变量。

<c:remove> 标签的语法 说明 ：

Html 代码

```
<c:remove var="varName" [scope="{ age|request|session|application }"] />
```

<c:catch> 标签

主要用来处理产生错误的异常状况，并且将错误信息储存起来。

<c:catch> 标签的语法 说明 ：

Html 代码

```
<c:catch [var="varName"] >
… 欲抓取错误的部分
</c:catch>
```

<c:if> 标签

的用途就和我们一般在程序中用的 if 一样。

<c:if> 标签的语法 说明 ：

语法 1：没有本体内容(body)

Html 代码

```
<c:if test="testCondition" var="varName" [scope="{page|request|session|application}"]/>
```

语法 2：有本体内容

Html 代码

```
<c:if test="testCondition" [var="varName"] [scope="{page|request|session|application}"]>
本体内容
</c:if>
```

示例：

Html 代码

```
<c:if test="${not empty item.publish_time}">
内容
</c:if>

<c:if test="${item['domain']!=null}">
内容
```

以上内容仅为本文档的试下载部分，为可阅读页数的一半内容。如要下载或阅读全文，请访问：https://d.book118.com/898024034007006041