Mục lục

[Mục lục 1](#_Toc24898833)

[Chương 1. JSTL 2](#_Toc24898834)

[1.1 Tổng quan 2](#_Toc24898835)

[1.2 Core Tags 3](#_Toc24898836)

[1.3 Formatting Tags 6](#_Toc24898837)

# JSTL

## Tổng quan

JSTL là một thành phần của nền tảng phát triển ứng dụng Web Java EE. Nó mở rộng đặc tả của JSP bằng cách thêm một thư viện chứa các thẻ cho các tác vụ phổ biến, như xử lý dữ liệu XML, thực thi có điều kiện, truy cập cơ sở dữ liệu, vòng lặp và quốc tế hóa.

JSTL được phát triển theo Java Community Process (JCP) dưới dạng Java Specification Request (JSR). Vào ngày 8 tháng 5 năm 2006 JSTL 1.2 đã được phát hành, tiếp theo là JSTL 1.2.1 vào ngày 7 tháng 12 năm 2011.

Hiện tại chúng ta có 5 loại chính được trình bày dưới đây:

|  |  |
| --- | --- |
| **Chức năng** | **Mô tả - Khai báo** |
| Các thẻ cơ bản - Core Tags | Chứa các lệnh điểu khiển như if, choose, vòng lặp… Để sử dụng, chúng ta cần khai báo như sau:  <%@taglib prefix="c" uri = "http://java.sun.com/jsp/jstl/core" %> |
| Thẻ định dạng và quốc tế hóa (Formatting and Localization Tags) | Thông thường thẻ loại này dùng để định dạng và quốc tế hóa, thường sử dụng để định dạng số, ngày tháng năm… Cú pháp khai báo như sau:  <%@taglib prefix="fmt" uri = "http://java.sun.com/jsp/jstl/fmt" %> |
| Các thẻ SQL - SQL tags | Nó cung cấp các hỗ trợ cho việc tương tác với cơ sở dữ liệu quan hệ như MySql, SQL server, Oracle… Cú pháp của nó là:  <%@taglib prefix="sql" uri = "http://java.sun.com/jsp/jstl/sql" %> |
| Các thẻ XML - XML Tags | Chức năng chính của thẻ này là đọc và phân tích dữ liệu từ các file XML. Để sử dụng, khai báo như sau:  <%@taglib prefix="x" uri = "http://java.sun.com/jsp/jstl/xml" %> |
| Các thẻ hàm JSTL (JSTL Functions Tags) | Thẻ này cung cấp một số chức năng mà chúng ta có thể sử dụng để thực hiện các toán tử thông thường là String như nối chuỗi. Cú pháp khai báo như sau:  <%@taglib prefix="fn" uri="http://java.sun.com/jsp/jstl/functions" %> |

Table 1‑1 Các taglib trong jstl

Vì chúng ta sử dụng maven để quản lý các thư viện nên dependency sẽ khai báo như sau:

<dependency>

<groupId>javax.servlet</groupId>

<artifactId>jstl</artifactId>

<version>1.2</version>

</dependency>

Sau đây là các chức năng mà từng loại thẻ trên cung cấp kèm theo ví dụ.

## Core Tags

### <c:out />

Thẻ <c:out /> hiển thị kết quả của một expression, nó giống với cách mà scriptlet <%= %> hoạt động. Điểm khác biệt ở chỗ nó cho phép chúng ta truy cập đến các thuộc tính của một đối tượng dễ dàng hơn.

<c:out value=*"Xin chào tôi tên là* ${user.name}*"*/>

### <c:set />

Nó giúp chúng ta tạo một đối tượng trong 1 scope nào đó.

<c:set var=*"message"* value=*"hello SV-STU"* scope=*"session"*/>

Hoặc thiết lập giá trị cho 1 đối tượng được truyền tới View như sau.

<c:set target=*"*${user}*"* property=*"name"* value=*"Tran Van A"*/>

### <c:remove />

Để xóa một đối tượng thuộc một scope xác định, ta sử dụng tag <c:remove /> như sau:

<c:remove var=*"message"* scope=*"session"*/>

### <c:catch />

Bắt những exception bị quăng ra trong body.

<c:catch var=*"Exception"*>

<% **int** i = (10 / 0); %>

</c:catch>

Khi có exception xảy ra, một biến có tên Exception được tạo và lưu trữ exception này.

### <c:if />

Thi hành đoạn code khi thỏa mãn điều kiện.

<c:if test=*"*${Exception != null}*"*>

err is: ${Exception.message}

</c:if>

### <c:choose /> <c:when /> <c:otherwise />

Ta có thể hình dung nó giống như trường hợp if -> else if -> else hoặc trường hợp switch case.

<c:choose>

<c:when test=*"Điều kiện ở đây"*>

<!-- Xử lý điều kiện if ở đây-->

</c:when>

<c:otherwise>

<!-- Xử lý điều kiện else ở đây -->

</c:otherwise>

</c:choose>

### <c:import />

Thẻ được dùng để nhập nội dung của 1 trang hoặc 1 file khác vào trang JSP hiện tại.

<c:import var = *"index"* url = *"/views/jsp/index.jsp"*/>

### <c:forEach />

Xử dụng thẻ này khi ta muốn duyệt qua một danh sách nào đó.

<c:forEach var=*"items"* items=*"*${listItem}*"*>

<c:out value=*"*${items.id} *+* ${items.name}*"*></c:out>

</c:forEach>

### <c:forTokens />

Để lặp qua một giá trị chuỗi được phân tách bằng các dấu phân tách, ta sử dụng thẻ <c:forTokens />.

<c:forTokens items=*"Tôi, là. sinh, STU"* delims=*",."* var=*"myTokens"*>

<c:out value=*"*${myTokens}*"*/><br/>

</c:forTokens>

### <c:param />

Thẻ này được lồng trong thẻ <c:url /> để xác định các param sẽ truyền theo url đó.

<c:url value=*"/product"* var=*"itemProductDetail"*>

<c:param name=*"itemId"* value=*"Shoe10"*/>

</c:url>

<c:import url=*"*${itemProductDetail}*"*/>

Khi đó, url có dạng /product?itemId=Shoe10 và sẽ được import vào trang qua tag <c:import />, thông thường <c:url /> <c:param /> và <c:import /> sẽ đi chung với nhau.

### <c:redirect />

Qua tên thì chúng ta cũng có thể nhận biết được rằng, thẻ này sẽ giúp chuyển hướng trình duyệt qua một URL mới thay thế, nếu được yêu cầu các tham số kèm theo, ta có thể sử dụng thẻ <c:url /> để truyền vào thuộc tính url của thẻ này.

<c:redirect url=*"http://www.ytbhoanghai.dx.am"*/>

### <c:url />

Thẻ này giúp ta tạo ra một URL và tùy ý lưu trữ URL này trong một biến, cũng như thiết lập scope cho biến này

<c:url value=*"/product"* var=*"itemProductDetail"* scope=*"session"*/>

## Formatting Tags

Từ đây cho đến hết phần [1.3] các mục trình bày sẽ không kèm theo ví dụ. Bản thân không phủ nhận sự tiện ích mà các loại thư viện thẻ này mang lại. Song, do các chức năng mà chúng cung cấp lại không nằm trong phạm vi của đồ án này, hoặc bản thân cảm thấy không phù hợp để áp dụng vào đồ án này. Một phần nào đó cũng muốn giới thiệu vì nó là một phần thiết yếu của JSTL vì thế nên sẽ cung cấp thông tin theo dạng tên thẻ và chức năng. Chúng ta cũng có thể tìm hiểu chi tiết về chúng qua sách hoặc các tài liệu public trên mạng.

|  |  |
| --- | --- |
| **Tên thẻ** | **Chức năng** |
| <fmt:bundle /> | Thẻ này chứa các cặp khóa–giá trị được sử dụng để quốc tế hóa hoặc bản địa hóa ứng dụng web. |
| <fmt:formatDate /> | Thẻ này sử dụng các style và pattern được cung cấp để định dạng thông tin ngày hoặc thời gian |
| <fmt:formatNumber /> | Được sử dụng để định dạng các giá trị là số, tiền tệ hoặc phần trăm. |
| <fmt:message /> | Được sử dụng để hiển thị các thông báo được bản địa hóa bằng cách thay thế các khóa được chỉ định, với thông điệp thực tế được tải từ gói tài nguyên. |
| <fmt:param /> | Được sử dụng để cung cấp các giá trị tham số cho các thông điệp văn bản. |
| <fmt:parseDate /> | Được sử dụng để phân tích cú pháp đại diện cho chuỗi ngày hoặc thời gian. |
| <fmt:parseNumber /> | Nó được sử dụng để phân tích chuỗi đại diện của số, tiền tệ hoặc tỷ lệ phần trăm. |
| <fmt:requestEncoding /> | Thẻ này được sử dụng để xác định ký tự mã hóa của request. Dựa vào nó để giải mã dữ liệu được nhập bởi người dùng. |
| <fmt:setBundle /> | Cũng như thẻ <fmt:Bundle /> nhưng nó được sử dụng trong toàn bộ trang JSP. |
| <fmt:setLocale /> | Sử dụng thẻ này để thiết lập thông tin tên miền địa phương cụ thể trong biến cấu hình. Được dùng để bản địa hóa trang web. |
| <fmt:setTimeZone /> | Dùng thẻ này để thiết lập giá trị múi giờ cần thiết. |
| <fmt:timeZone /> | Thẻ này được sử dụng để chỉ định thông tin múi giờ cho bất kỳ định dạng nào trong phần body. |

Table 1‑2 Formatting Tags trong jstl

### SQL Tags

|  |  |
| --- | --- |
| **Tên thẻ** | **Chức năng** |
| <sql:dateParam /> | Thẻ được sử dụng thể thiết lập thông tin thời gian, dùng làm tham số trong các câu truy vấn. |
| <sql:param /> | Sử dụng thẻ này để thiết lập các tham số cho câu truy vấn SQL. |
| <sql:query /> | Thẻ thực thi câu truy vấn SQL và lưu trữ kết quả trong 1 biến có phạm vi. |
| <sql:setDataSource /> | Thiết lập datasource để giao tiếp với cơ sở dữ liệu. |
| <sql:transaction /> | Thực thi các câu truy vấn sử dụng <sql:query /> hoặc <sql:update /> trong ngữ cảnh transaction. |
| <sql:update /> | Thẻ này không trả về giá trị, nó thực hiện các thao tác thêm, xóa và cập nhật các giá trị. |

Table 1‑3 SQL Tags trong jstl

### XML tags

|  |  |
| --- | --- |
| **Tên thẻ** | **Chức năng** |
| <x:forEach /> | Thẻ này được sử dụng để lặp qua các node trong tài liệu XML. |
| <x:if /> | Là thẻ điều kiện, thực thi phần body nếu biểu thức điều kiện là true. |
| <x:out /> | Sử dụng để hiển thị kết quả của một biểu thức (expression). |
| <x:param /> | Thẻ này được sử dụng để gửi tham số đến thẻ <x:transform /> |
| <x:parse /> | Được sử dụng để phân tích nội dung XML được cung cấp trong thuộc tính source hoặc trong thẻ body. |
| <x:set /> | Được sử dụng để lưu kết quả của một expression vào một biến scoped. |
| <x:transform /> | Thẻ này sẽ chuyển đổi tài liệu XML sử dụng mẫu XSLT. |
| <x:choose />  <x:when />  <x:otherwise /> | Thực hiện các việc có điều kiện khi phân tích tài liệu XML.  <x:choose /> là thẻ bao quanh, <x:when /> thực thi điều kiện if và <x:otherwise /> là else. |

Table 1‑4 XML tags trong jstl

### JSTL Functions

|  |  |
| --- | --- |
| **Tên thẻ** | **Chức năng** |
| <fn:contains /> | Kiểm tra xem một chuỗi đã cho có được chứa trong một chuỗi khác không. Nếu có hàm sẽ trả về true, ngược lại là false. |
| <fn:containsIgnoreCase /> | Giống như thẻ <fn:contains /> nhưng không phân biệt chữ hoa và chữ thường. |
| <fn:endsWith /> | Kiểm tra xem 1 chuỗi đã cho có kết thúc bằng hậu tố đã cho hay không. |
| <fn:escapeXml /> | Hiển thị các thẻ html hoặc xml trong chuỗi dưới dạng văn bản thuần. |
| <fn:indexOf /> | Trả về index của lần xuất hiện đầu tiên của chuỗi con trong chuỗi đã cho. |
| <fn:join /> | Kết hợp các phần tử của 1 mảng thành 1 chuỗi với dấu phân cách được chỉ định. |
| <fn:length /> | Trả về số lượng ký tự trong 1 chuỗi hoặc số mục trong bất kỳ một collection nào. |
| <fn:replace /> | Thay thế tất cả các lần xuất hiện của chuỗi đã cho bằng chuỗi mới và trả về chuỗi kết quả. |
| <fn:split /> | Trả về một mảng chứa các chuỗi được phân tách từ một chuỗi đã cho dựa trên dấu phân tách. |
| <fn:startsWith /> | Trả về giá trị boolean, kiểm tra xem một chuỗi đã cho có bắt đầu bằng một tiền tố đã cho hay không. |
| <fn:substring /> | Trả về một chuỗi con từ một chuỗi được cung cấp index bắt đầu và kết thúc. |
| <fn:substringAfter /> | Trả về một chuỗi sau chuỗi con đã được chỉ định. |
| <fn:substringBefore /> | Trả về một chuỗi trước chuỗi con đã được chỉ định. |
| <fn:toLowerCase /> | Chuyển đổi tất cả các ký tự của chuỗi đã cho thành ký tự thường. |
| <fn:toUpperCase /> | Chuyển đổi tất cả các ký tự của chuỗi đã cho thành ký tự in. |
| <fn:trim /> | Loại bỏ khoảng trắng từ cả 2 đầu của 1 chuỗi. |

Table 1‑5 JSTL functions