



FPT POLYTECHNIC



Conceive Design Implement Operate

JSP, EL & JSTL

J4 PROGRAMMING – SERVLET/JSP & JPA

www.poly.edu.vn

- ❑ JSP (Java Server Page), EL (Expression Language) và JSTL (Java Standard Tag Library) là bộ ba giúp lập trình render giao diện từ phía server.
- ❑ Trước JSP 2.0 trang JSP là sự trộn lẫn giữa mã java và HTML, làm cho JSP rất khó đọc, khó quản lý (không chuẩn markup).
- ❑ Từ 2.x trở đi, EL và JSTL được áp dụng làm giảm độ phức tạp của lập trình JSP đồng thời trang JSP rõ ràng hơn, dễ quản lý hơn, đúng chuẩn markup hơn.
- ❑ *Trong môn học này chúng ta áp dụng EL, JSTL và lược bỏ một số kỹ thuật lập trình JSP cũ kỹ trước đây (không viết mã java trong jsp).*

□ Phần 1: JSP & EL

- ❖ JSP – Java Server Page
- ❖ EL – Expression Language

□ Phần 2: JSTL

- ❖ Thư viện lõi (core)
- ❖ Thư viện định dạng (format)
- ❖ Thư viện hàm (functions)





① JSP VÀ EL

❑ Directives (chỉ thị)

- ❖ `<%@page pageEncoding="utf-8"%>`
- ❖ `<%@include file="sub-page.jsp"%>`
- ❖ `<%@taglib uri="" prefix=""%>`

❑ Standard Actions (hành động chuẩn)

- ❖ `<jsp:include page="sub-page.jsp"/>`
- ❖ `<jsp:forward page="sub-page.jsp"/>`
- ❖ `<jsp:param name="" value=""/>`
- ❖ `<jsp:useBean id="" class="" scope=""/>`
- ❖ `<jsp:setProperty id="" scope=""/>`

❑ <%**@page** *pageEncoding*="utf-8"%>

❖ Khai báo trang JSP hỗ trợ UTF-8 (Tiếng Việt)

❑ <%**@include** *file*="sub-page.jsp"%>

❖ Chỉ thị này giúp module hóa các thành phần giao diện, giúp chèn toàn bộ mã JSP của trang sub-page.jsp tại vị trí đặt chỉ thị @include. Công việc chèn này xảy ra tại thời điểm dịch. Vì vậy xem mã trong sub-page.jsp là một phần của trang JSP hiện tại.

❑ <%**@taglib** *uri*="" *prefix*=""%>

❖ Nhúng bộ thư viện thẻ vào trang JSP (trình bày kỹ ở phần JSTL)

main.jsp

```
<%@ page pageEncoding="utf-8"%>
<!DOCTYPE html>
<html>
<head>
    <meta charset="utf-8">
    <title>Insert title here</title>
</head>
<body>
    <h1>Directive Demo</h1>
    <%@include file="sub.jsp"%>
</body>
</html>
```

sub.jsp

```
<form action="/upload" method="post">
    <input name="fullname" placeholder="Fullname?">
    <br>
    <button>Submit</button>
</form>
```

CÁC THUỘC TÍNH CỦA <%@PAGE%>

❑ Chỉ thị @page ngoài thuộc tính pageEncoding còn có một số thuộc tính khác ít được sử dụng như hình bên, bạn có thể tham khảo thêm.

- Ⓐ autoFlush="true"
- Ⓐ buffer="8kb"
- Ⓐ contentType="text/html; charset=ISO-8859-1"
- Ⓐ deferredSyntaxAllowedAsLiteral="false"
- Ⓐ errorPage
- Ⓐ extends
- Ⓐ import
- Ⓐ info
- Ⓐ isELIgnored="false"
- Ⓐ isErrorPage="false"
- Ⓐ isThreadSafe="true"
- Ⓐ language="java"
- Ⓐ session="true"

❑ **<jsp:include page="sub-page.jsp"/>**

- ❖ Render giao diện của sub-page.jsp tại vị trí đặt thẻ (tương tự req.getRequestDispatcher().include() trong servlet)
- ❖ Chú ý sự khác biệt so với chỉ thị <%@include%>
 - <%@include%> chèn mã trong khi đó <jsp:include> render kết quả
 - <%@include%> xảy ra lúc dịch trong khi đó <jsp:include> xảy ra lúc chạy

❑ **<jsp:forward page="sub-page.jsp"/>**

- ❖ Chuyển tiếp sang sub-page.jsp (tương tự req.getRequestDispatcher().forward() trong servlet)

❑ Chú ý:

- ❖ <jsp:include/> sau khi render giao diện sẽ quay trở lại trang JSP hiện tại để thực hiện tiếp mã phía dưới trong khi đó <jsp:forward> không quay trở lại trang hiện tại
- ❖ Trước khi forward không nên render giao diện, nhiệm vụ render giao diện là của trang JSP cuối cùng trong chuỗi forward.

PHÂN BIỆT <JSP:INCLUDE> VÀ <%@INCLUDE%>

1...
2...
<%@include file="sub.jsp"%>
3...

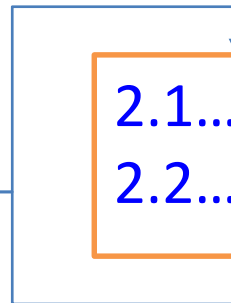
2.1...
2.2... sub.jsp



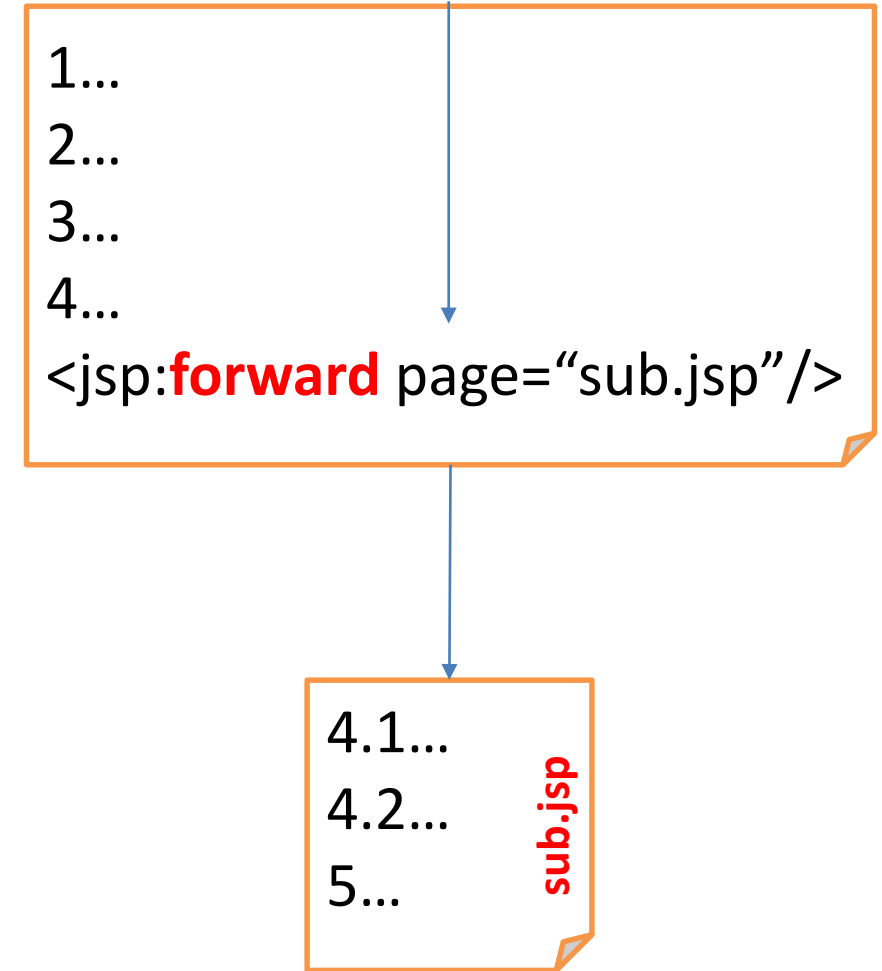
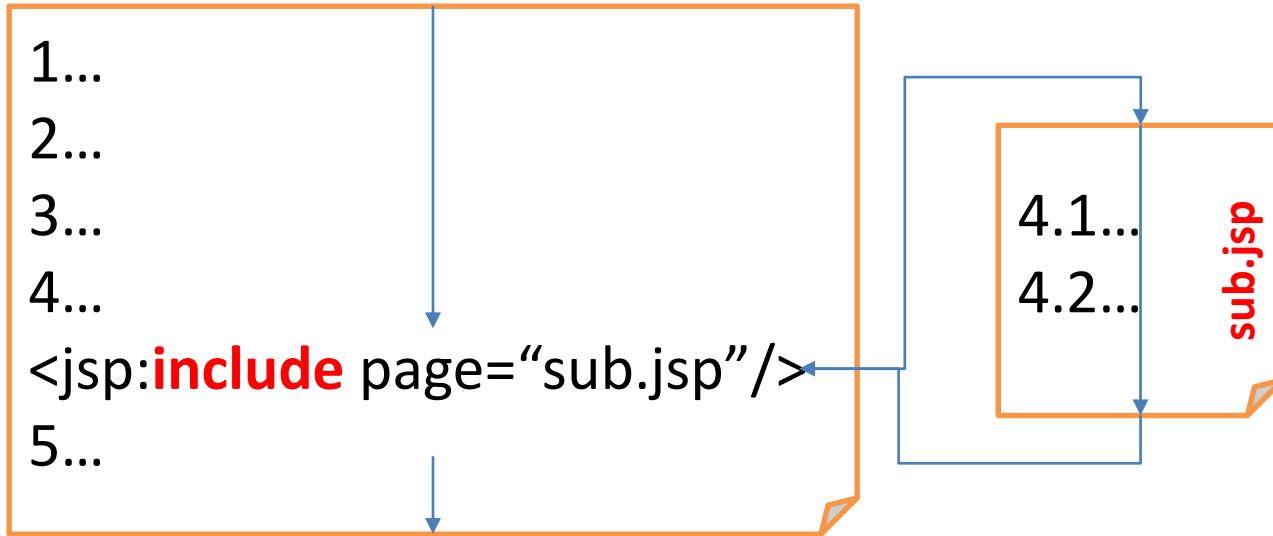
1...
2...
2.1...
2.2...
3...

1...
2...
<jsp:include page="sub.jsp"/>
3...

2.1...
2.2... sub.jsp



PHÂN BIỆT <JSP:INCLUDE> VÀ <JSP:FORWARD>



❑ `<jsp:param name="" value=""/>`

- ❖ Được sử dụng để tạo tham số truyền cho trang con khi sử dụng `<jsp:include>` và `<jsp:forward>`

```
<%@ page pageEncoding="utf-8"%>
<!DOCTYPE html>
<html>
<head>
    <meta charset="utf-8">
    <title>Insert title here</title>
</head>
<body>
    <h1>Main Page</h1>
    <jsp:include page="sub.jsp">
        <jsp:param value="Hello sub page" name="message"/>
    </jsp:include>
</body>
</html>
```

`<h3>Sub Page</h3>`
`${param.message}`

- ❑ `<jsp:useBean id="" class="" scope=""/>`
 - ❖ Được sử dụng để tạo một đối tượng từ hoặc tham chiếu đến một đối tượng được tạo sẵn trong một scope (page, request, session, application)
- ❑ `<jsp:setProperty property="" name="" value=""/>`
 - ❖ Được sử dụng để thay đổi giá trị của thuộc tính bean
- ❑ `<jsp:getProperty property="" name=""/>`
 - ❖ Được sử dụng để xuất giá trị của thuộc tính bean
- ❑ Ví dụ:

```
<jsp:useBean id="now" class="java.util.Date" scope="page"/>  
<jsp:setProperty property="date" name="now" value="20"/>  
Month: <jsp:getProperty property="month" name="now"/>
```

- ❑ EL (ngôn ngữ biểu thức) được sử dụng để tạo các biểu thức làm việc trực tiếp với các attribute, parameter, cookie và kết xuất kết quả để tạo giao diện
- ❑ Ví dụ:
 - ❖ **`${xyz}`**
 - Xuất giá trị của attribute xyz trong một scope nào đó
 - ❖ **`${requestScope.xyz}`** hoặc **`${requestScope['xyz']}`**
 - Xuất giá trị của attribute xyz trong request scope
 - ❖ **`${param.xyz}`** hoặc **`${param['xyz']}`**
 - Xuất giá trị của tham số xyz
 - ❖ **`${cookie.xyz.value}`** hoặc **`${cookie['xyz'].value}`**
 - Xuất giá trị của cookie xyz
 - ❖ **`${requestScope.bean.xyz}`** hoặc **`${requestScope['bean'].xyz}`**
 - Xuất giá trị thuộc tính xyz của bean trong request scope

❑ Truy xuất attribute theo scope

- ❖ `${pageScope.xyz}` hoặc `${pageScope['xyz']}`
- ❖ `${requestScope.xyz}` hoặc `${requestScope['xyz']}`
- ❖ `${sessionScope.xyz}` hoặc `${sessionScope['xyz']}`
- ❖ `${applicationScope.xyz}` hoặc `${applicationScope['xyz']}`

❑ Tìm và truy xuất attribute

- ❖ `${xyz}`: Truy tìm attribute xyz theo thứ tự ưu tiên:
 - *Page -> Request -> Session -> Application*

- ❑ Truy xuất thuộc tính **bean**
 - ❖ `${bean.property}`
- ❑ Truy xuất phần tử thứ i của **list**
 - ❖ `${list[i]}`
- ❑ Truy xuất phần tử của **map**
 - ❖ `${map[key]}` hoặc `${map.key}`
- ❑ Truy xuất **tham số**
 - ❖ `${param[name]}` hoặc `${param.name}`
- ❑ Truy xuất **cookie**
 - ❖ `${cookie[name].value}` hoặc `${cookie.name.value}`

❑ Lớp JavaBean là lớp

- ❖ Phải là public
- ❖ Có Constructor mặc định (không tham số)
- ❖ Có getter và setter

❑ Cú pháp truy xuất:

- ❖ `${bean.property}`. Trong đó `bean.property` được hiểu là `bean.getProperty()`

❑ Ví dụ:

- ❖ `${cart.count} -> cart.getCount()`
- ❖ `${mail.from} -> mail.getFrom()`

Servlet

```
req.setAttribute("x", 1000);  
req.getSession().setAttribute("y", 2000);  
req.getServletContext().setAttribute("z", 3000);  
req.setAttribute("now", new Date());  
req.getServletContext().setAttribute("x", 5000);  
req.getRequestDispatcher("/views/el.jsp").forward(req, resp);
```

Expression Language

- requestScope.x: 1000 = 1000
- sessionScope.y: 2000 = 2000
- applicationScope.z: 3000 = 3000
- applicationScope.x: 5000 = 1000 ?
- Bean.month: 10
- Bean.year: 2020

<h1>Expression Language</h1>


```
<li>requestScope.x: ${requestScope.x} = ${x}</li>  
<li>sessionScope.y: ${sessionScope.y} = ${y}</li>  
<li>applicationScope.z: ${applicationScope.z} = ${z}</li>  
<li>applicationScope.x: ${applicationScope.x} = ${x}</li>  
<li>Bean.month: ${now.month + 1}</li>  
<li>Bean.year: ${now.year + 1900}</li>
```


JSP

SỬ DỤNG EL TRUY XUẤT LIST VÀ MAP

```
Map<String, Double> diems = new HashMap<>();  
diems.put("toan", 5.0);  
diems.put("ly", 7.0);
```

```
List<String> tens = new ArrayList<>();  
tens.add("Phượng");  
tens.add("Hồng");
```

```
req.setAttribute("map", diems);  
req.setAttribute("list", tens);
```

```
req.getRequestDispatcher("/views/el.jsp").forward
```

Servlet

MAP:

- map.toan: 5.0 = 5.0
- map.ly: 7.0 = 7.0
- (map.ly + map.toan)/2: 3.5

LIST:

- list[0]: Phượng
- list[1]: Hồng

MAP:

```
<li>map.toan: ${map.toan} = ${map['toan']}</li>  
<li>map.ly: ${map.ly} = ${map['ly']}</li>  
<li>(map.ly + map.toan)/2: ${((map.ly + toan)/2)}</li>
```

LIST:

```
<li>list[0]: ${list[0]}</li>  
<li>list[1]: ${list[1]}</li>
```

JSP

❑ pageContext là đối tượng ngầm định của trang JSP nó chứa request và response. Vì vậy bạn có thể sử dụng EL để truy xuất các property (getProperty()) của chúng

❖ `${pageContext.request.requestURI}`

➤ Truy xuất URI hiện tại

❖ `${pageContext.request.requestURL}`

➤ Truy xuất URL hiện tại

❖ `${pageContext.request.method}`

➤ Truy xuất method (POST, GET) hiện tại

❖ `${pageContext.request.contextPath}`

➤ Truy xuất đường dẫn ngữ cảnh (đường dẫn ứng dụng)

❖ `${pageContext.response.locale.language}`

➤ Truy xuất ngôn ngữ hiện tại









② JSTL

- ❑ JSTL có 5 bộ thư viện thẻ chuẩn hỗ trợ lập trình render giao diện phía server, truy xuất CSDL, xử lý XML
 - ❖ `<%@ taglib uri="http://java.sun.com/jstl/core_rt" prefix="c" %>`
 - ❖ `<%@ taglib uri="http://java.sun.com/jstl/fmt_rt" prefix="fmt" %>`
 - ❖ `<%@ taglib uri="http://java.sun.com/jsp/jstl/functions" prefix="fn" %>`
 - ❖ `<%@ taglib uri="http://java.sun.com/jstl/xml_rt" prefix="xml" %>`
 - ❖ `<%@ taglib uri="http://java.sun.com/jstl/sql_rt" prefix="sql" %>`
- ❑ Hai thư viện xml và sql làm việc với CSDL và XML ít được sử dụng nên không được giới thiệu trong môn học này (các bạn tham khảo thêm)

KHAI BÁO THƯ VIỆN CẦN THIẾT – POM.XML

```
<dependency>  
  <groupId>javax.servlet</groupId>  
  <artifactId>jstl</artifactId>  
  <version>1.2</version>  
</dependency>
```

Libraries

- >  Apache Tomcat v8.5 [Apache Tomcat v8.5]
- >  JRE System Library [JavaSE-1.8]
- ▼  Maven Dependencies
 - >  mail-1.4.7.jar - C:\Users\Admin\.m2\rep
 - >  activation-1.1.jar - C:\Users\Admin\.m2
 - >  jstl-1.2.jar - C:\Users\Admin\.m2\reposit

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

Thẻ	Thuộc tính quan trọng	Mô tả
<c:if test>	@test là biểu thức boolean	Tương tự lệnh if
<c:choose><c:when test><c:otherwise><c:choose>	@test là biểu thức boolean	Tương tự lệnh if...else if...else
<c:forEach var items>	@var phần tử hiện tại, @items là tập hợp	Duyệt tập hợp, tương tự for(:)
<c:forEachTokens var items delim>	@var phần tử hiện tại, @items chuỗi chứa các thành phần, @delim là chuỗi phân cách	Duyệt các phần của chuỗi
<c:set var value scope>	@var là tên attribute, @value là giá trị của attribute, @scope là phạm vi chia sẻ	Tương tự scope.setAttribute()
<c:remove var scope>	@var là tên attribute, @scope là phạm vi chia sẻ	Tương tự scope.removeAttribute()
<c:url value var scope>	@var là tên attribute, @value là mapping uri, @scope là phạm vi chia sẻ	Tạo chuỗi uri
<c:import url>	@url là mapping uri	Include một servlet khác


```
<%@ page pageEncoding="utf-8"%>
<%@ taglib uri="http://java.sun.com/jstl/core_rt" prefix="c" %>
<!DOCTYPE html>
<html>
<head>
    <meta charset="utf-8">
    <title>JSTL Example</title>
</head>
<body>
    <c:if test="${!empty sessionScope.user}">
        <h1>Welcome, ${sessionScope.user.fullname}</h1>
    </c:if>
</body>
</html>
```

Nếu trong session có attribute user thì xuất họ và tên của user ra màn hình

THẺ <C:CHOOSE>..<<C:WHEN>..<<C:OTHERWISE>

```
<jsp:useBean id="now" class="java.util.Date"/>
```

```
<c:choose>
```

```
  <c:when test="{now.day == 0}">Chủ nhật</c:when>
```

```
  <c:when test="{now.day == 6}">Thứ bảy</c:when>
```

```
  <c:otherwise>Ngày trong tuần</c:otherwise>
```

```
</c:choose>
```

Lệnh này tương tự if(ĐK1)... else if(ĐKN)...else

❑ <c:forEach var="name" items="array, collection hoặc map">

```
Map<String, Double> diems = new HashMap<>();  
diems.put("toan", 5.0);  
diems.put("ly", 7.0);  
req.setAttribute("map", diems);
```

Servlet

```
List<String> tens = new ArrayList<>();  
tens.add("Phượng");  
tens.add("Hồng");  
req.setAttribute("list", tens);
```

```
<c:forEach var="item" items="${list}">  
    <li>${item}</li>  
</c:forEach>
```

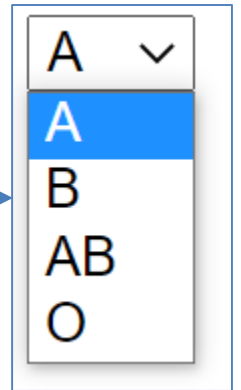
JSP

```
<c:forEach var="entry" items="${map}">  
    <li>${entry.key} = ${entry.value}</li>  
</c:forEach>
```

- Phượng
- Hồng
- ly = 7.0
- toan = 5.0

❑ <c:forTokens var="" items="" delims="">

```
<select>
  <c:forTokens var="blood" items="A B AB O" delims=" ">
    <option>${blood}</option>
  </c:forTokens>
</select>
```




Blood group:

```
<c:forTokens var="blood" items="A B AB O" delims=" ">
  <label><input type="radio"> ${blood}</label>
</c:forTokens>
```

Blood group: ☐ A ☐ B ☐ AB ☐ O

```
<c:set var="message" value="Hello JSTL" scope="page"/>  
<c:set var="message" value="Chào JSTL" scope="request"/>  
<c:remove var="message" scope="page"/>  
<h2>${message}</h2>
```



Chào JSTL


```
<c:url var="url" value="/hello.php"/>  
<h2>${url}</h2>
```



/ContextPath/hello.php

```
<c:import url="/hello.php"/>
```

```
<%@ page pageEncoding="utf-8"%>
<%@ taglib uri="http://java.sun.com/jstl/fmt_rt" prefix="fmt" %>
<!DOCTYPE html>
<html>
<head><meta charset="utf-8"></head>
<body>
    <jsp:useBean id="now" class="java.util.Date"/>
    <h3><fmt:formatDate value="{now}" /></h3>
    <h3><fmt:formatDate value="{now}" pattern="EEE, dd-MM-yyyy" /></h3>
    <hr>
    <h3><fmt:formatNumber value="{now.time}" pattern="#,###.00" /></h3>
</body>
</html>
```



May 27, 2020
Wed, 27-05-2020
1,590,579,805,671.00

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

❑ Thư viện này cung cấp các hàm hỗ trợ xử lý chuỗi và tập hợp trong biểu thức EL

❑ Ví dụ: Giả sử có attribute s là một chuỗi

❖ `${fn:toUpperCase(s)}`

❖ `${fn:length(s)}`

❖ `${fn:substring(s, 10, 20)}`

❖ `${fn:substringAfter(s, 'VN')}`

❖ `<c:if test="${fn:contains(s, 'VN')}"> </c:if>`

❖ `<c:if test="${fn:startsWith(s, 'VN')}"> </c:if>`

❖ `<c:forEach var="item" items=${fn:split(s, '~')}>`

Tên hàm	Đối số	Trả về	Mô tả mục đích
fn:contains	String, String	boolean	Chuỗi (1) có chứa chuỗi (2) hay không
fn:containsIgnoreCase	String, String	boolean	Chuỗi (1) có chứa chuỗi (2) hay không (không phân biệt hoa thường)
fn:endsWith	String, String	boolean	Chuỗi (1) có kết thúc bởi (2) hay không
fn:escapeXML	String	String	Mã hóa thành thực thể các ký tự phạm cú pháp XML
fn:indexOf	String, String	int	Tìm vị trí xuất hiện đầu tiên của chuỗi (2) trong chuỗi (1)
fn:join	String[], String	String	Gia nhập các phần tử trong mảng (1) thành chuỗi sử dụng chuỗi(2) như là chuỗi phân cách.
fn:length	Map; array; Collection; Iterator; Enumeration; or String	int	Tìm độ dài của chuỗi hay số lượng các phần tử trong tập hợp.

Tên hàm	Đối số	Trả về	Mô tả mục đích
fn:replace	String, String, String	String	Thay thế chuỗi (1) bởi chuỗi (3) trong chuỗi (1)
fn:split	String, String	String[]	Tách chuỗi (1) thành mảng sử dụng chuỗi (2) như chuỗi phân cách
fn:startsWith	String, String	boolean	Chuỗi đối số thứ nhất có bắt đầu bởi chuỗi đối số thứ hai hay không
fn:substring	String, int, int	String	Lấy chuỗi trong chuỗi (1) tính từ vị trí (1) cho đến vị trí (3)
fn:substringAfter	String, String	String	Lấy chuỗi con trong chuỗi (1) đứng sau chuỗi (2)
fn:substringBefore	String, String	String	Lấy chuỗi con trong chuỗi (1) đứng trước chuỗi (2)
fn:toLowerCase	String	String	Đổi chuỗi sang chữ thường
fn:toUpperCase	String	String	Đổi chuỗi sang chữ HOA
fn:trim	String	String	Cắt bỏ khoảng trắng 2 đầu chuỗi

✓ JSP:

- ✓ `<%@ page%>`, `<%@ include%>`, `<%@ taglib%>`
- ✓ `<jsp:include>` `<jsp:param>`

✓ EL:

- ✓ `${attr}`, `${xyzScope.attr}`
- ✓ `${bean.property}`, `${map.key}`, `${map[key]}`
- ✓ `${param.name}`, `${cookie.name.value}`

✓ JSTL:

- ✓ Core: `<c:if>`, `<c:choose>`, `<c:forEach>`...
- ✓ Format: `<fmt:formatNumber>`, `<fmt:formatDate>`
- ✓ Functions: `${fn:toUpperCase(attr)}`, `${fn:length(attr)}`





Cảm ơn