

Expression Language (EL) trong JSP

Expression Language (EL) cho phép JSP developers truy cập các đối tượng java thông qua tag.

JSP EL cho phép chúng ta tạo các Expression, gồm số học và logic. Bên trong một JSP EL, bạn có thể sử dụng các integer, các số floating point, string, các hằng có sẵn true hoặc false cho các giá trị Boolean, và null. Đây là tính năng mới được thêm vào trong JSP công nghệ phiên bản 2.0.

JavaBean

Một JavaBean là một lớp Java được xây dựng một cách đặc biệt trong Java và được mã hóa theo JavaBeans API Specifications.

Sau đây là các đặc trưng duy nhất giúp phân biệt một JavaBean với các lớp khác trong Java:

- Nó cung cấp một constructor mặc định, không có tham số.
- Nó có thể xếp thứ tự và triển khai Serializable interface.
- Nó có thể có một số đặc tính mà có thể được đọc và được viết.
- Nó có thể có một số phương thức "getter" và "setter" cho các đặc tính.

Đọc tại

https://www.tutorialspoint.com/jsp/jsp_java_bean.htm

Ví dụ:

```
public class Student{
    private int rollnum;
    private String name;
    public Student() {
    }
    public Student(int rollnum, String name) {
        this.rollnum = rollnum;
        this.name = name;
    }
    public int getRollnum() {
        return rollnum;
    }
    public void setRollnum(int rollnum) {
        this.rollnum = rollnum;
    }
    public String getName() {
        return name;
    }
    public void setName(String name) {
        this.name = name;
    }
}
```

```
}
```

File JSP:

```
<form action="display.jsp" method="post">
    Enter ur roll number:
    <input type="number" name="rollnum"/><br/>
    Enter ur name:
    <input type="text" name="name"/><br/>
    <input type="submit" value="Submit">
</form>
```

Khai báo sử dụng EL

```
<%@page isELIgnored="false" %>
```

File display.jsp

<pre>//scriptlet -jsp <% Entity.Student s1=new Student(); int rollnum = Integer.parseInt(request.getParameter("rollnum")); String name= request.getParameter("name"); Student s=new Student(rollnum, name); %> Roll number: <%=s.getRollnum()%>
 Name: <%=s.getName()%>
</pre>	<pre><jsp:useBean id="s1" class="entity.Student" scope="request"> </jsp:useBean> <jsp:setProperty name="s1" property="*" /> Roll number:<jsp:getProperty name="s1" property="rollnum"/>
 Name:<jsp:getProperty name="s1" property="name"/>
 <jsp:useBean id="s2" class="entity.Student" scope="request"/> <jsp:setProperty name="s2" property="rollnum" value="\${param.rollnum}"/> <jsp:setProperty name="s2" property="name" value="\${param.name}"/> Roll number:<jsp:getProperty name="s2" property="rollnum"/>
 Name:<jsp:getProperty name="s2" property="name"/>
</pre>
--	--

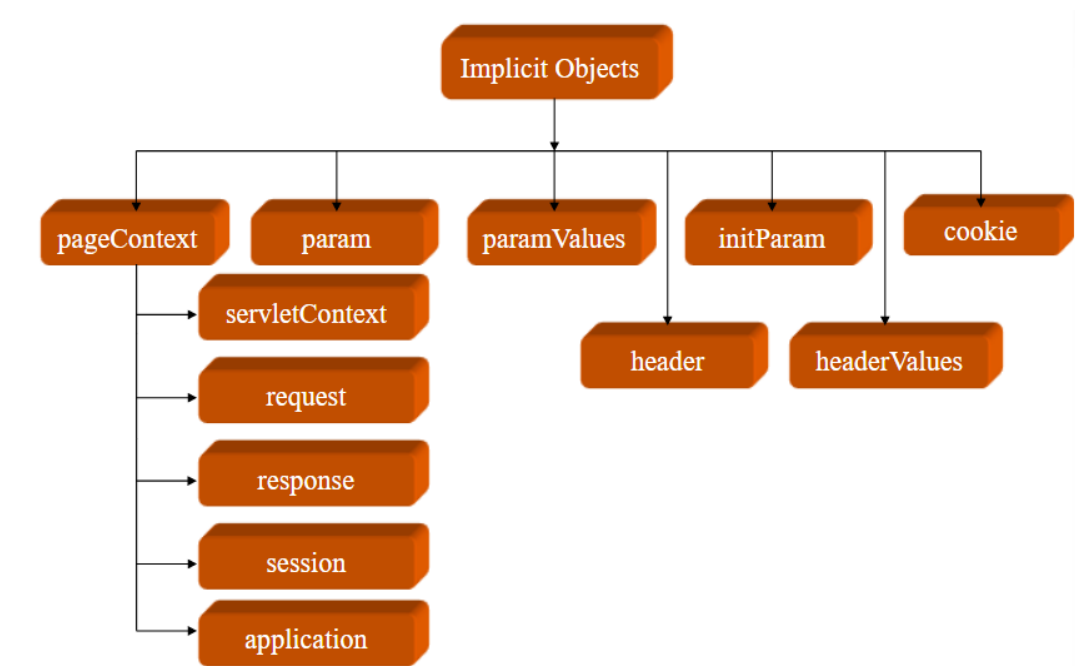
Expression Language

Roll number: \${param["rollnum"]} - \${param.rollnum} Name: \${param.name} <%=request.getParameter("name")%>

Các đối tượng ẩn trong JSP EL

JSP Expression Language hỗ trợ các đối tượng ẩn sau:

Đối tượng ẩn	Miêu tả
pageScope	Các biến scope từ phạm vi page
requestScope	Các biến scope từ phạm vi request
sessionScope	Các biến scope từ phạm vi session
applicationScope	Các biến scope từ phạm vi application
param	Các tham số request, dạng các chuỗi
paramValues	Các tham số request, dạng các tập hợp của các chuỗi
header	HTTP request header dạng các chuỗi
headerValues	HTTP request header dạng các tập hợp của các chuỗi
initParam	Các tham số khởi tạo context
cookie	Các giá trị Cookie
pageContext	Đối tượng JSP PageContext cho page hiện tại



Toán tử cơ bản trong EL

JSP Expression Language (EL) hỗ trợ hầu hết các toán tử số học và logic được hỗ trợ bởi Java. Dưới đây là danh sách các toán tử được sử dụng thường xuyên nhất.

Toán tử	Miêu tả
.	Truy cập một đặc tính của Bean hoặc Map entry
[]	Truy cập một phân tử mảng hoặc List
()	Nhóm một subexpression để thay đổi thứ tự ước lượng
+	Phép cộng
-	Phép trừ hoặc phủ định một giá trị
*	Phép nhân
/ hoặc div	Phép chia
% hoặc mod	Phép chia lấy phần dư
== hoặc eq	Kiểm tra có bằng hay không
!= hoặc ne	Kiểm tra tính không bằng

< hoặc lt	Kiểm tra tính nhỏ hơn
> hoặc gt	Kiểm tra tính lớn hơn
<= hoặc le	Kiểm tra tính nhỏ hơn hoặc bằng
>= hoặc ge	Kiểm tra tính lớn hơn hoặc bằng
&& hoặc and	Phép AND logic
hoặc or	Phép OR logic
! hoặc not	Phản bù Boolean một ngôi
empty	Kiểm tra các giá trị biến rỗng

Example 1: Calculate

```
<h1>Calculating with EL</h1>
<form method="post">
  Enter number 1:
  <input type="text" name="num1" value="${param.num1}"/><br/>
  Enter number 2:
  <input type="text" name="num2" value="${param.num2}"/><br/>

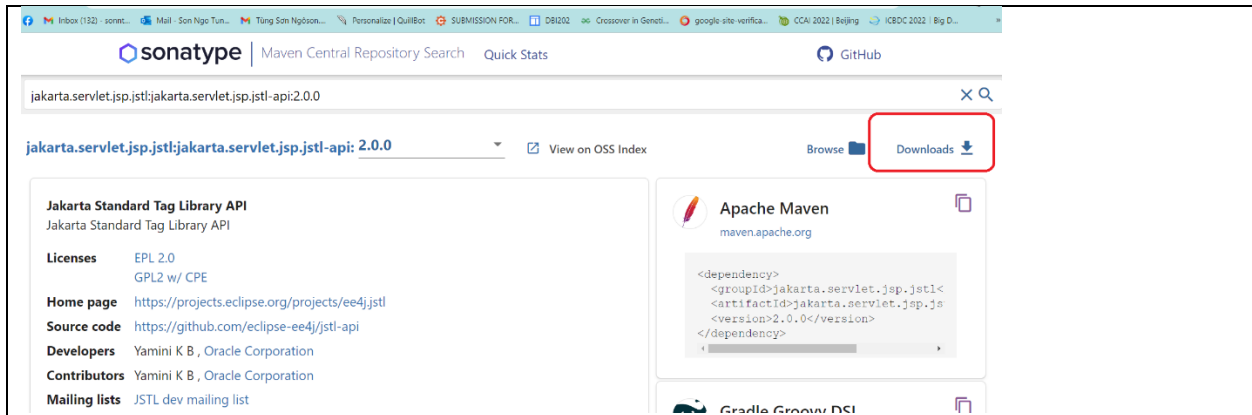
  <h3 style="color: blue">Sum: ${param.num1 + param.num2}</h3>
  <h3 style="color: blue">Avg: ${ (param["num1"] + param["num2"])/2}</h3>
  <input type="submit" value="Calculate"/>
</form>
```

JSTL (JSP Standard Tag Library)

Latest version of JSTL build on J2EE was 1.2x. Since the Tomcat 10 is using Jakartified version (with jakarta.* package instead of javax.* package), use JSTL 2.0 to solve this dependency:

Download: Jakarta Standard Tag Library API

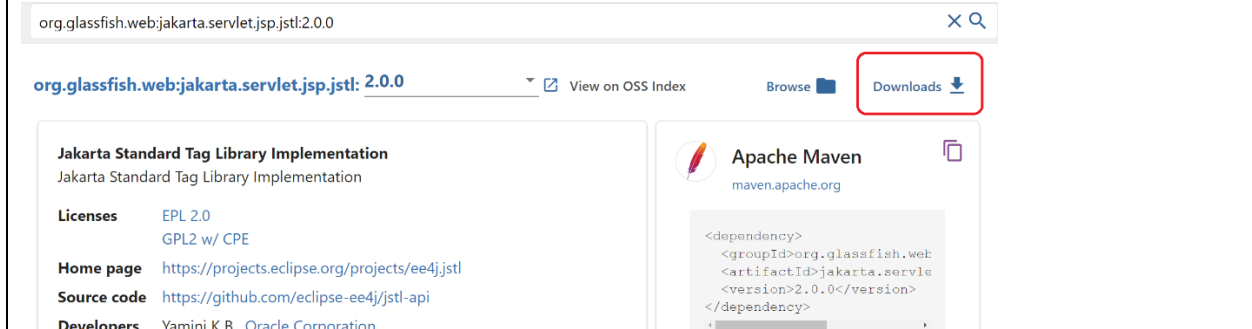
Link: <https://search.maven.org/artifact/jakarta.servlet.jsp.jstl/jakarta.servlet.jsp.jstl-api/2.0.0/jar>



The screenshot shows the Sonatype Maven Central Repository search results for the artifact `jakarta.servlet.jsp.jstl:jakarta.servlet.jsp.jstl-api:2.0.0`. The page displays the artifact's details, including its name, version, and a list of licenses (EPL 2.0, GPL2 w/ CPE). The 'Downloads' button is highlighted with a red box.

Download: Jakarta Standard Tag Library Implementation

Link: <https://search.maven.org/artifact/org.glassfish.web/jakarta.servlet.jsp.jstl/2.0.0/jar>



The screenshot shows the Maven Central search results for the artifact `org.glassfish.web:jakarta.servlet.jsp.jstl:2.0.0`. The page displays the artifact's details, including its name, version, and a list of licenses (EPL 2.0, GPL2 w/ CPE). The 'Downloads' button is highlighted with a red box.

Check the document here:

<https://jakarta.ee/specifications/tags/2.0/jakarta-tags-spec-2.0.html>

The JSP Standard Tag Library (JSTL) represents a set of tags to simplify the JSP development.

Classification of The JSTL Tags

The JSTL tags can be classified, according to their functions, into the following JSTL tag library groups that can be used when creating a JSP page –

- **Core Tags**
- **Formatting tags**
- **SQL tags**
- **XML tags**
- **JSTL Functions**

<https://www.javatpoint.com/jstl>

JSTL Core Tags

The JSTL core tag provides variable support, URL management, flow control etc. The syntax used for including JSTL core library in your JSP is:

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

Example : <c:set>

```
<c:set var="n" value="10"/>  
<c:set var="id" value="${requestScope.id}"/>
```

Example 2: <c:forEach>

- Servlet

```
List<Student> data=new ArrayList<>();  
data.add(new Student(1, "Do thu Ha"));  
data.add(new Student(2, "Pham Luan"));  
data.add(new Student(3, "Vu Hai"));  
data.add(new Student(4, "Tran thi Teo"));  
request.setAttribute("data", data);  
request.getRequestDispatcher("list.jsp").forward(request, response);
```

- JSP

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

<pre><c:forEach items="\${requestScope.data}" var="s"> \${s.rollnum} \${s.name}
 </c:forEach></pre>	<pre>\${requestScope["data"]}</pre>
---	-------------------------------------

```
<c:forEach begin="${1}" end="${10}" var="i">  
    ${i}<br/>  
</c:forEach>
```

Servlet

```
HttpSession session =request.getSession(true);  
session.setAttribute("data", data);
```

<pre><c:forEach items="\${sessionScope.data}" var="s"> \${s.rollnum} \${s.name}
 </c:forEach></pre>	<pre>\${sessionScope["data"]}</pre>
---	-------------------------------------

Example 3: <c:if>

<pre><form method="post"> Enter a salary: <input type="text" name="salary"/>
 <c:set var="salary" scope="request" value="\${param.salary}"/> <c:if test="\${salary<200}"> <h3 style="color: blue">Tax: \${0}</h3> </c:if> <c:if test="\${(salary>=200)&&(salary<400)}"> <h3 style="color: blue">Tax: \${((salary-200)*0.1)}</h3> </c:if> <c:if test="\${salary>=400}"> <h3 style="color: blue">Tax: \${((200*0.1)+((salary-400)*0.15))}</h3> </c:if> <input type="submit" value="Calculate tax"/> </form></pre>

Example 4: <c:choose>

<pre><c:choose> <c:when test="\${salary<200}"> <h3 style="color: blue">Tax: \${0}</h3> </c:when> <c:when test="\${(salary>=200)&&(salary<400)}"> <h3 style="color: blue">Tax: \${((salary-200)*0.1)}</h3> </c:when> <c:when test="\${salary>=400}"> <h3 style="color: blue">Tax: \${((200*0.1)+((salary-400)*0.15))}</h3> </c:when> <c:otherwise> salary is undetermined... </c:otherwise> </c:choose></pre>
--

JSTL Formatting Tags

<https://www.javatpoint.com/jstl-formatting-tags>


```
<fmt:formatNumber pattern="##.##" value="${Math.PI}"/>
<fmt:formatDate pattern="dd/MM/yyyy" value="%=new Date()%"/>
```

JSTL Function Tags

The JSTL function provides a number of standard functions, most of these functions are **common string manipulation functions**. The syntax used for including JSTL function library in your JSP is:

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

<https://www.javatpoint.com/jstl-function-tags>

Example

```
<form method="post">
    Enter a paragraph:
    <input type="text" name="txt"/><br/>
    <c:set var="ftxt" value="${param.txt}" scope="request"/>
    <h3 style="color: blue">
        the number: ${fn:length(ftxt)}</h3>
    <br/>
    <input type="submit" value="Submit"/>
</form>
```

Viết lại bài đã học dùng EL và JSTL