KaliberMind
ACADEMY

JSP

JSTL(JSP Standard Tag Library)

Table of Contents

- JSTL Core Tags
- JSTL Function Tags
- JSTL Formatting Tags
- JSTL XML Tags
- JSTL SQL Tags

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

Advantage of JSTL

Fast Developement JSTL provides many tags that simplifies the JSP.

Code Reusability We can use the JSTL tags in various pages.

No need to use scriptlet tag It avoids the use of scriptlet tag.

JSTL Tags

There JSTL mainly provides 5 types of tags:

Tag Name	**Description**
Core tags	The JSTL core tag provide variable support, URL management, flow control etc. The url for the core tag is **http://java.sun.com/jsp/jstl/core** . The prefix of core tag is **c**.
Function tags	The functions tags provide support for string manipulation and string length. The url for the functions tags is **http://java.sun.com/jsp/jstl/functions** and prefix is **fn**.
Formatting tags	The Formatting tags provide support for message formatting, number and date formatting etc. The url for the Formatting tags is **http://java.sun.com/jsp/jstl/fmt** and prefix is **fmt**.
XML tags	The xml sql tags provide flow control, transformation etc. The url for the xml tags is **http://java.sun.com/jsp/jstl/xml** and prefix is **x**.
SQL tags	The JSTL sql tags provide SQL support. The url for the sql tags is **http://java.sun.com/jsp/jstl/sql** and prefix is **sql**.

- For creating JSTL application, you need to load jstl.jar file.

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

JSTL Core Tags List

Tags	Description
c:out	It display the result of an expression, similar to the way <%=...%> tag work.
c:import	It Retrives relative or an absolute URL and display the contents to either a String in 'var',a Reader in 'varReader' or the page.
c:set	It sets the result of an expression under evaluation in a 'scope' variable.
c:remove	It is used for removing the specified scoped variable from a particular scope.

c:catch	It is used for Catches any Throwable exceptions that occurs in the body.
c:if	It is conditional tag used for testing the condition and display the body content only if the expression evaluates is true.
c:choose, c:when, c:otherwise	It is the simple conditional tag that includes its body content if the evaluated condition is true.
c:forEach	It is the basic iteration tag. It repeats the nested body content for fixed number of times or over collection.
c:forTokens	It iterates over tokens which is separated by the supplied delimeters.
c:param	It adds a parameter in a containing 'import' tag's URL.
c:redirect	It redirects the browser to a new URL and supports the context-relative URLs.
c:url	It creates a URL with optional query parameters.

index.jsp

```
<%@ page isELIgnored="false" language="java" contentType="text/html; charset=ISO-8859-1"
    pageEncoding="ISO-8859-1"%>
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
"http://www.w3.org/TR/html4/loose.dtd">
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=ISO-8859-1">
<title>Insert title here</title>
</head>
<body>
        <h1>JSTL Core Tags</h1>
        <form action="welcome.jsp">
                        User Name: <input type="text" name="userName">
                        <input type="submit" value="submit">
        </form>
        <a href="WelcomeServlet">Test c:forEach</a>
</body>
</html>
```

WelcomeServlet.jsp

```
package com.Biditvats.Servlet;

import java.io.IOException;
import java.util.ArrayList;
import java.util.List;

import javax.servlet.RequestDispatcher;
import javax.servlet.ServletException;
import javax.servlet.annotation.WebServlet;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;


@WebServlet("/WelcomeServlet")
public class WelcomeServlet extends HttpServlet {
	private static final long serialVersionUID = 1L;

	protected void doGet(HttpServletRequest request, HttpServletResponse response) throws
ServletException, IOException {
		List<String> list = new ArrayList<>();
		list.add("Java");
		list.add("Hibernate");
		list.add("Spring");

		request.setAttribute("courses", list);

		RequestDispatcher dispatcher = request.getRequestDispatcher("welcome.jsp");
		dispatcher.forward(request, response);
	}
}
```

welcome.jsp

```
<%@ page isELIgnored="false" language="java" contentType="text/html; charset=ISO-8859-1"
    pageEncoding="ISO-8859-1"%>
<%@ taglib uri="http://java.sun.com/jsp/jstl/core" prefix="c" %>
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
"http://www.w3.org/TR/html4/loose.dtd">
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=ISO-8859-1">
<title>Insert title here</title>
</head>
<body>
	<h1>Example of c:out</h1>
	<h4>Welcome, <c:out value="${param.userName}" /> </h4>

	<h1>Example of c:set</h1>
	<c:set var="user" value="${param.userName}" />
	<h4>User, <c:out value="${user}" /></h4>

	<h1>Example of C:choose , c:when and c:otherwise(as if-else)</h1>
	<c:choose>
			<c:when test="${param.userName == 'admin' }">
				<h4>Admin Panel: ${param.userName }</h4>
			</c:when>
			<c:otherwise>
				<c:out value="Invalid user Name"></c:out>
			</c:otherwise>
	</c:choose>
```

```
        <h1>Example of C:choose , c:when and c:otherwise(as switch)</h1>
        <c:choose>
                        <c:when test="${param.userName == 'admin' }">
                            <h4>Admin Panel: ${param.userName }</h4>
                        </c:when>
                        <c:when test="${param.userName == 'manager' }">
                            <h4>Admin Panel: ${param.userName }</h4>
                        </c:when>
                        <c:when test="${param.userName == 'employee' }">
                            <h4>Admin Panel: ${param.userName }</h4>
                        </c:when>
                        <c:otherwise>
                    <%-- <c:redirect url="index.jsp" /> --%>
                        </c:otherwise>
        </c:choose>

        <h1>Example of c:forEach</h1>
        Courses:
        <select>
                <c:forEach items="${requestScope.courses}" var="course">
                        <option><c:out value="${course}" /></option>
                </c:forEach>
        </select>



        <h1>Example of c:import</h1>
        <c:import url="footer.jsp"></c:import>
</body>
</html>
```

footer.jsp

```
<%@ page isELIgnored="false" language="java" contentType="text/html; charset=ISO-8859-1"
   pageEncoding="ISO-8859-1"%>
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
"http://www.w3.org/TR/html4/loose.dtd">
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=ISO-8859-1">
<title>Insert title here</title>
</head>
<body>
 <p style="colo:#F00;">
      &copy; 2016 by WWW.BiditVats.com
 </p>
</body>
</html>
```

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:

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

JSTL Function Tags List

JSTL Functions	Description
fn:contains()	It is used to test if an input string containing the specified substring in a program.
fn:containsIgnoreCase()	It is used to test if an input string contains the specified substring as a case insensitive way.
fn:endsWith()	It is used to test if an input string ends with the specified suffix.
fn:escapeXml()	It escapes the characters that would be interpreted as XML markup.
fn:indexOf()	It returns an index within a string of first occurrence of a specified substring.
fn:trim()	It removes the blank spaces from both the ends of a string.
fn:startsWith()	It is used for checking whether the given string is started with a particular string value.
fn:split()	It splits the string into an array of substrings.
fn:toLowerCase()	It converts all the characters of a string to lower case.
fn:toUpperCase()	It converts all the characters of a string to upper case.
fn:substring()	It returns the subset of a string according to the given start and end position.
fn:substringAfter()	It returns the subset of string after a specific substring.
fn:substringBefore()	It returns the subset of string before a specific substring.
fn:length()	It returns the number of characters inside a string, or the number of items in a collection.
fn:replace()	It replaces all the occurrence of a string with another string sequence.

JSTL fn:contains() Function

The fn:contains() is used for testing if the string containing the specified substring. If the specified substring is found in the string, it returns true otherwise false.

The syntax used for including the fn:contains() function is:

```
boolean contains(java.lang.String, java.lang.String)
```

Let's see the simple example to understand the functionality of fn:contains() function:

```
<%@ taglib uri="http://java.sun.com/jsp/jstl/core" prefix="c" %>
<%@ taglib uri="http://java.sun.com/jsp/jstl/functions" prefix="fn" %>
<html>
<head>
<title>Using JSTL Functions</title>
</head>
<body>

<c:set var="String" value="Welcome to kalibermind"/>

<c:if test="${fn:contains(String, 'kalibermind')}">
   <p>Found kalibermind string<p>
</c:if>

<c:if test="${fn:contains(String, 'KALIBERMIND')}">
   <p>Found KALIBERMIND string<p>
</c:if>

</body>
</html>
```

Output:

```
Found kalibermind string
```