JSTL TAGS - by RAGHU SIR

1. JSTL Core Tags :

Tags	Description
c:out	Prints given expression, similar to the way <%=...%> tag work.
c:set	Create variable in given 'scope'.
c:remove	delete variable from a particular scope.
c:if	If conditional statement
c:choose, c:when, c:otherwise	It is like switch case statement
c:forEach	Behaves like for loop and for each loop
c:forTokens	Given string separated by the supplied delimeters.
c:param	It adds a parameter in a containing URL.
c:redirect	It redirects the browser to a new URL.
c:url	It creates a new URL with optional query parameters.
c:import	Import HTML content into given variable
c:catch	It is used to store any exception raised.

AllInOneExample: index.jsp

```
<%@page import="java.util.*"%>
<%@ page language="java" contentType="text/html; charset=ISO-8859-1"
   pageEncoding="ISO-8859-1"%>
<%@taglib prefix="c" uri="http://java.sun.com/jsp/jstl/core" %>
<!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>
        <c:set var="eid" value="10" scope="request"/>

        <c:if test="${eid gt 0 }">
                <c:out value="It's ok"/>
        </c:if>

        <c:choose>
                <c:when test="${eid lt 0 }">
                        <c:out value="-ve value"/>
                </c:when>
                <c:when test="${eid gt 0}">
```

```
                <c:out value="+ve value"/>
            </c:when>
            <c:otherwise>
                <c:out value="It is zero"/>
            </c:otherwise>
        </c:choose>

        <c:forEach begin="1" end="${eid}" var="i">
            <c:out value="${i}"/>
        </c:forEach>

        <c:remove var="eid"/>
        Data is:<c:out value="${eid}"/>
        <%
            List<String> al=Arrays.asList("A","B","C");
            request.setAttribute("list", al);
        %>
        <c:forEach items="${list}" var="ob">
            <c:out value="${ob}"/>
        </c:forEach>

        <c:set value="Hi this is from sathya tech!!" var="str"/>
    <c:forTokens items="${str}" delims=" " var="s">
        <c:out value="${s}"/> <br/>
    </c:forTokens>

    <c:url var="search" value="https://www.google.co.in/search">
        <c:param name="q" value="India"/>
    </c:url>

    <%-- <c:redirect url="${search}"/> --%>

        <c:catch var="ae">
            <%
                int a=10/0;
            %>
        </c:catch>
        <c:out value="${ae}"/>

        <c:import url="hi.jsp"/>
</body>
</html>
```

2. JSTL SQL Tags

SQL Tags	Descriptions
sql:setDataSource	It is used for creating a simple data source suitable only for prototyping.
sql:query	It is used for executing the SQL query defined in its sql attribute or the body.
sql:update	It is used for executing the SQL update defined in its sql attribute or in the tag body.
sql:param	It is used for sets the parameter in an SQL statement to the specified value.
sql:dateParam	It is used for sets the parameter in an SQL statement to a specified java.util.Date value.
sql:transaction	It is used to provide the nested database action with a common connection.

AllInOneExample: index.jsp

```
<sql:setDataSource driver="oracle.jdbc.OracleDriver"
 url="jdbc:oracle:thin:@localhost:1522:ORCL"
 user="system" password="Admin" var="ds"/>
<sql:query var="rs" dataSource="${ds}" sql="select * from empstab"/>

<c:forEach items="${rs.rows}" var="emp">
        <c:out value="${emp.eid}"/>
        <c:out value="${emp.ename}"/>
        <c:out value="${emp.esal}"/>
        <c:out value="${emp.dob}"/>
</c:forEach>

<sql:update var="count" sql="insert into empstab values(18,'D',5.5,sysdate)"
dataSource="${ds}"/>
No of rows Inserted:<c:out value="${count }"/>

<sql:update var="count" sql="delete from empstab where eid>=?" dataSource="${ds}">
        <sql:param value="15"/>
</sql:update>
No of rows Deleted:<c:out value="${count }"/>


<sql:transaction dataSource="${ds}">
        <sql:update sql="insert into empstab values(22,'RR',3.3,sysdate)"/>
        <sql:update sql="insert into empstab values(23,'RT',3.3,sysdate)"/>
        <sql:update sql="insert into empstab values(24,'RZ',3.3,sysdate)"/>
</sql:transaction>

<sql:update sql="update empstab set dob=TO_DATE('2019/10/10', 'yyyy/mm/dd')"
dataSource="${ds}"/>
```

3. JSTL Function Tags :

JSTL Functions	Description
fn:contains()	It is used to test if an input string containing the specified substring
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.

AllInOneExample: index.jsp

```
<%@ page language="java" contentType="text/html; charset=ISO-8859-1"
    pageEncoding="ISO-8859-1"%>
<%@taglib prefix="c" uri="http://java.sun.com/jsp/jstl/core" %>
<%@taglib prefix="fn" uri="http://java.sun.com/jsp/jstl/functions"
%>

<html>
<body>
<pre>
<c:set value="Hello from Sathya Technologies" var="str"/>
STR: ${str }
SUB STR: ${fn:substring(str,2,7) }
SUB STR: ${fn:substringAfter(str,'from') }
SUB STR: ${fn:substringBefore(str,'Sathya') }

<c:if test="${fn:contains(str,'Sathya') }">
     Hello Sathya Tech#1
</c:if>
<c:if test="${fn:containsIgnoreCase(str,'sathya') }">
     Hello Sathya Tech#2
</c:if>
<c:if test="${fn:startsWith(str,'Hel') }">
     Hello Sathya Tech#3
</c:if>
```

```
<c:if test="${fn:endsWith(str,'ies') }">
     Hello Sathya Tech#4
</c:if>
Index: ${fn:indexOf(str,'from') }
<c:set var="str2" value="  Hello RAM  "></c:set>
LEN : ${fn:length(str2) }
<c:set var="str3" value="${fn:trim(str2) }"></c:set>
LEN : ${fn:length(str3) }
LOWER : ${fn:toLowerCase(str3) }
UPPER : ${fn:toUpperCase(str3) }

REPLACE : ${fn:replace(str3,'Hello','hi') }

<c:set var="str4" value="${fn:split(str3,' ') }"></c:set>
${str4[0]}
${str4[1]}

<c:set var="str5" value="Hi Employee <ename> AJAY </ename>"></c:set>
${str5}
${fn:escapeXml(str5) }
</pre>
</body>
</html>
```

4. JSTL Formatting tags:-

Formatting Tags	Descriptions
fmt:parseNumber	It is used to Parses the string representation of a currency, percentage or number.
fmt:formatNumber	It is used to format the numerical value with specific format or precision.
fmt:formatDate	It formats the time and/or date using the supplied pattern and styles.

AllInOneExample:index.jsp

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

<html>
<body>
<pre>
<c:set var="cost" value="12255.669858"/>
<fmt:parseNumber var="i" value="${cost}" integerOnly="true"/>
Cost is: ${i}
US: <fmt:formatNumber currencyCode="USD" value="${cost}"
type="currency"/>
INDIA: <fmt:formatNumber currencyCode="INR" value="${cost}"
type="currency"/>
GROUP: <fmt:formatNumber value="${cost}" type="currency"
groupingUsed="true"/>
```

```
<fmt:formatNumber value="${cost}" type="currency"
maxIntegerDigits="3" maxFractionDigits="3"/>

<c:set value="<%=new java.util.Date()%>" var="str"/>
<fmt:formatDate value="${str }"/>
<fmt:formatDate value="${str }" type="time"/>
<fmt:formatDate value="${str }" type="date"/>
<fmt:formatDate value="${str }"  type="both" dateStyle="short"
timeStyle="short" />
<fmt:formatDate value="${str }"  type="both" dateStyle="medium"
timeStyle="medium" />
<fmt:formatDate value="${str }"  type="both" dateStyle="Long"
timeStyle="long" />
</pre>
</body>
</html>
```

5. JSTL XML tags:

XML Tags	Descriptions
x:out	Similar to <%= ... > tag, but for XPath expressions.
x:parse	It is used for parse the XML data specified either in the tag body or an attribute.
x:set	It is used to sets a variable to the value of an XPath expression.
x:choose	It is a conditional tag that establish a context for mutually exclusive conditional operations.
x:when	It is a subtag of that will include its body if the condition evaluated be 'true'.
x:otherwise	It is subtag of that follows tags and runs only if all the prior conditions evaluated be 'false'.
x:if	It is used for evaluating the test XPath expression and if it is true, it will processes its body content.

Required Jars Download Link:
https://www.mediafire.com/file/f44hqus638mcxe7/JSTL_XML_JARS.rar/file

AllInOneExample:index.jsp

```
<%@ page language="java" contentType="text/html; charset=ISO-8859-1"
    pageEncoding="ISO-8859-1"%>
<%@taglib prefix="c" uri="http://java.sun.com/jsp/jstl/core" %>
<%@taglib prefix="x" uri="http://java.sun.com/jsp/jstl/xml" %>

<html>
<body>
<h2>Hello</h2>
```

```
<c:set var="emps">
<employees>
      <employee>
            <empId>10</empId>
            <empName>AJAY</empName>
            <empSal>366.36</empSal>
      </employee>
      <employee>
            <empId>11</empId>
            <empName>VIJAY</empName>
            <empSal>855.36</empSal>
      </employee>
      <employee>
            <empId>12</empId>
            <empName>JAI</empName>
            <empSal>985.36</empSal>
      </employee>
</employees>
</c:set>

<pre>
<x:parse xml="${emps}" var="data"/>
<x:set var="ename" select="$data/employees/employee[2]/empName"/>
<x:out select="$ename"/>
<x:out select="$data/employees/employee[1]/empId"/>

<x:choose>
      <x:when select="$data/employees/employee[1]/empSal < 20000">
            Good, But not OK
      </x:when>
      <x:when select="$data/employees/employee[1]/empSal > 20000">
            Nice, Be happy
      </x:when>
      <x:otherwise>
            Unable to get Data
      </x:otherwise>
</x:choose>
<x:if select="$data/employees/employee[2]/empName = 'VIJAY'">
      Hello VIJAY
</x:if>
</pre>
</body>
</html>
```


FB : https://www.facebook.com/groups/thejavatemple/
email : javabyraghu@gmail.com