Note:

      To use jstl jar supplied by tomcat vendor we refer to the following location

```
                  C:\Tomcat 9.0\webapps\examples\WEB-INF\lib
```

Core Libray
=========

```
<%@ taglib uri="http://java.sun.com/jsp/jstl/core" prefix="c"%>
   <c:out>
            It is used for writing Template text data and expression to the JSP.
    c=> prefixName
   out => tagName

<c:out value="WELCOME TO JSTL CODING...."/><br/>
The user name is :: <c:out value = "${param.user}"/><br/>
The password  is :: <c:out value = "${param.password }" default="Guest"/>
```

input

```
 http://localhost:9999/JSTLApp-01/index.jsp?user=Hyder&password=iNeuron
```

output

```
      WELCOME TO JSTL CODING....
      The user name is :: Hyder
      The password is :: iNeuron
```

2. <c:set>

            We can use to set attributes in any scope and to set map and bean properties also.

eg:

```
<c:set var="x" value="10" scope="request"/>
<c:set var="y" value="20" scope="request"/>
<c:set var="sum" value="${x+y}" scope="session"/>
<h1 style='color:red; text-align:center;'>
The result is :: <c:out value="${sum}"/>
```

3. <c:remove>

            To remove attributes in the specified scope we can use this tag.
            if the scope is not specified for removing, by default it will search in

                  a. page scope
                  b. request scope
                  c. session scope
                  d. application scope

eg::

```
<c:set var = "x" value="10" scope="page"/>
<c:set var = "y" value="20" scope="page"/>
<c:set var = "z" value="${x+y}" scope="session"/>
<h1 style='color:blue; text-align:center;'>
            The result is :: <c:out value="${z}"/>
</h1>
<c:remove var="x"/>
<c:remove var="y"/>
<c:remove var="z"/>
<h1 style='color:red; text-align:center;'>
      The result is :: <c:out value="${z}" default="1000"/>
</h1>
```

4.

```
<c:catch var="">
```

```
      //risky code
</c:catch>
      If any exception occurs, then that exception object is collected inside var
attribute vaiable which is page scope.
      if any exception is raised inside risky code, then this tag suppress that
exception and rest of the jsp wil be executed
       normally.

eg:
<h1 style='color: blue; text-align: center;'>
            UserName is :: ${param.userName}<br />
            <c:catch var="e">
                  <%
                        int age =
Integer.parseInt(request.getParameter("userAge"));
                  %>
                  UserAge  is   :: ${param.userAge }<br />
            </c:catch>

            <c:if test="${e!=null}">
                  oops... Exception raised .... : ${e}<br/>
            </c:if>
            UserHeight is :: ${param.userHeight }
</h1>

input
      http://localhost:9999/JSTLApp-01/index.jsp?
userName=sachin&userAge=ten&userHeight=5.5
output
      UserName is :: sachin
            oops... Exception raised .... : java.lang.NumberFormatException: For
input string: "ten"
      UserHeight is :: 5.5

input
      http://localhost:9999/JSTLApp-01/index.jsp?
userName=sachin&userAge=49&userHeight=5.5
output
      UserName is :: sachin
      UserAge is :: 49
      UserHeight is :: 5.5


Conditional Tags
=============
1. <c:if>
       It is used to implement core java if statement

      <c:if test="" scope="" var="">
            //body of if
      </c:if>
      if the condition evaluates to true only then body of if will be
executed,otherwise the remaining statement present
      in jsp page will be executed.

eg:
<c:set var="x" value="10"/>
<c:set var="y" value="20"/>
<c:if test="${x<y }" var="result">
```

```
      X value is ${x}<br/>
      Result is ${result}
</c:if>
<c:if test="${x eq 10 }">
      X is  equal to 10
</c:if>
```

```
output
X value is 10
Reslut is true
X is equal to 10
```

2. <c:choose>, <c:when> and <c:otherwise>
            We can use these tags for implementing if else and switch statements.

```
Implementing if-else
=================
<c:choose>
      <c:when test = "condition">
            ACTION-1
      </c:when>
      <c:otherwise>
            ACTION-2
      </c:otherwise>
</c:choose>
            if condition evaluates to tree then ACTION-1 otherwise ACTION-2
```

```
Implementing switch
=================
<c:choose>
      <c:when test = "test_condition1">
            ACTION-1
      </c:when>
      <c:when test = "test_condition2">
            ACTION-2
      </c:when>
      <c:when test = "test_condition2">
            ACTION-3
      </c:when>
            ;;;;
      <c:when test = "test_conditionN">
            ACTION-2
      </c:when>
      <c:otherwise>
            Default Action
      </c:otherwise>
</c:choose>
```

Note:
 1. <c:when> tag explicitlly contains break statement, so no chance of fall through in switch.
 2. <c:otherwise> should always be last case only
 3. <c:choose> should compulsorily contain one <c:when> tag, but <c:otherwise> is optional.

```
eg:
<h1>
            Select one number
            <form action="./index.jsp">
                  <select name="combo">
                        <option value='1'>1</option>
                        <option value='2'>2</option>
                        <option value='3'>3</option>
                        <option value='4'>4</option>
                        <option value='5'>5</option>
                        <option value='6'>6</option>
                        <option value='7'>7</option>
                        <option value='8'>8</option>
                        <option value='9'>9</option>
                  </select> <input type='submit' />
            </form>

            <c:set var='day' value='${param.combo }' />
            <c:choose>
                  <c:when test="${day==1 }">
                        SUNDAY
                  </c:when>
                  <c:when test="${day==2 }">
                        MONDAY
                  </c:when>
                  <c:when test="${day==3 }">
                        TUESDAY
                  </c:when>
                  <c:when test="${day==4 }">
                        WEDNESDAY
                  </c:when>
                  <c:when test="${day==5 }">
                        THURSDAY
                  </c:when>
                  <c:when test="${day==6 }">
                        FRIDAY
                  </c:when>
                  <c:when test="${day==7 }">
                        SATURDAY
                  </c:when>
                  <c:otherwise>
                        SELECT NUMBER BETWEEN 1 to 7
                  </c:otherwise>
      </c:choose>
</h1>

Iteration tags
===========
1. <c:forEach  begin="" end="" step="">
            It would ressamble general purpose for loop.
            default value of step is "1", it gets incremented automatially.
            The loop body will be executed w.r.t  "begin<=end".
eg:
<c:forEach begin="1" end="10" step="2" var="count">
      <h1>Learning JSTL is very easy..${count}</h1>
</c:forEach>

eg:
<%
      String[] names = {"sachin","saurav","dhoni","kohli"};
      pageContext.setAttribute("names", names);
%>
```

```
<c:forEach  items="${names}" var="obj">
      <h1>The data is :: ${obj }<br/></h1>
</c:forEach>
```

output

```
The data is :: sachin
The data is :: saurav
The data is :: dhoni
The data is :: kohli
```

Note:
This is similar to

```
      for(String name: names)
            System.out.println(name);
```

2. <c:forTokens>
            It is a specialized version of forEach to perform StringTokenization based on some delimitor.

syntax

```
      <c:forTokens items = "" delims="" var="" begin="" end="" step="">
                  //body
      </c:forTokens>
```

eg:

```
<c:forTokens items="Sachin,Saurav,Dhoni,Dravid" delims="," var="name">
            <h1>The name is :: ${name}</h1><br/>
</c:forTokens>
```

eg:

```
<c:forTokens items="One,Two,Three,Four,Five,Six,Seven" delims="," var="data"
begin="2" end="5" step ='2'>
            <h1>The result :: ${data}</h1><br/>
</c:forTokens>
```

output

```
The result :: Three
The result :: Five
```

eg:

```
<%
            ArrayList<String> al = new ArrayList<String>();
            al.add("sachin");
            al.add("dhoni");
            al.add("kohli");
            al.add("dravid");
            al.add("rahul");
            pageContext.setAttribute("names", al);
%>
<c:forEach  items="${names}" var="name">
            <h1>${name }</h1>
</c:forEach>
```

Note:
      <c:forTokens> items attribute should be string only.
      <c:forEach> items attributes can be String,Collection object,Map etc.

URL related tags
----

1. <c:import>

we can use this tag for importing the response of the other pages in the current page response at the time of request processing.(ie dynamic include)

eg:

```
first.jsp
<h1>Welcome to iNeuron+Physics Wallah</h1><br/>
<c:import url="second.jsp" />

second.jsp
      <h1>The free videos are available in www.youtube.com/navinreddy</h1>
```

eg:
As noticed below the ouput of <c:import> is copied into the variable, so in the current jsp where ever the ouput is required
we can just refer to that variable.

```
first.jsp
<h1>Welcome to iNeuron+Physics Wallah</h1><br/>
      <c:import url="second.jsp" var = "x" scope="request" />
      ${x} <br/>
      ${x} <br/>
      ${x} <br/>
      ${x} <br/>
```

eg:

```
 first.jsp
<h1>Welcome to iNeuron+Physics Wallah</h1><br/>
      <c:import url="second.jsp" >
            <c:param name="java" value="hyder"/>
            <c:param name="jee" value="nitin"/>
            <c:param name="spring" value="navinreddy"/>
      </c:import>

second.jsp
<h1>The free videos are available in www.youtube.com/navinreddy</h1><br/>
<h1>Trainer name for java is :: ${param.java }</h1>
<h1>Trainer name for Jee is  :: ${param.jee }</h1>
<h1>Trainer name for spring is  :: ${param.spring }</h1>
```

2. <c:redirect>

This is similar to sendRedirect() method of ServletResponse.

eg:

```
<h1>Welcome to iNeuron+Physics Wallah</h1><br/>
      <c:redirect url="second.jsp" >
            <c:param name="java" value="hyder"/>
            <c:param name="jee" value="nitin"/>
            <c:param name="spring" value="navinreddy"/>
      </c:redirect>

second.jsp
<h1>The free videos are available in www.youtube.com/navinreddy</h1><br/>
<h1>Trainer name for java is :: ${param.java }</h1>
<h1>Trainer name for Jee is  :: ${param.jee }</h1>
<h1>Trainer name for spring is  :: ${param.spring }</h1>
```

input

```
      http://localhost:9999/JSTLApp-01/second.jsp?
java=hyder&jee=nitin&spring=navinreddy

output
The free videos are available in www.youtube.com/navinreddy
Trainer name for java is :: hyder
Trainer name for Jee is :: nitin
Trainer name for spring is :: navinreddy

3. <c:url>
   This would attach jsessionid and the query parameters to the url.

first.jsp
======
<c:url value="second.jsp" var="x" scope='request'>
            <c:param name="java" value="hyder" />
            <c:param name="jee" value="nitin" />
            <c:param name="spring" value="navinreddy" />
</c:url>
<h1>The modified url is :: ${x}</h1>
<a href="${x }">Click here to go to Next Page...</a>

second.jsp
=========
<h1>The free videos are available in www.youtube.com/navinreddy</h1><br/>
<h1>Trainer name for java is :: ${param.java }</h1>
<h1>Trainer name for Jee is  :: ${param.jee }</h1>
<h1>Trainer name for spring is  :: ${param.spring }</h1>

SQLTags
=======
      Code related to JDBC.

1. <sql:setDataSource>-> to create datasource object
2. <sql:query> -> to perform select operations
3. <sql:update>-> to perform non select operations(insert,update,delete)
4. <sql:param> -> to inject the values for preparedStatement object
5. <sql:dateParam> -> to inject data values in case of preparedstatement.

Code to perform select operation using jstl
====================================
<sql:setDataSource var="ds" url="jdbc:mysql:///enterprisejavabatch"
            driver="com.mysql.cj.jdbc.Driver" user="root" password="root123" />
<sql:query var="result" dataSource="${ds}">
            select * from student
</sql:query>
<h1>
      <c:forEach items="${result.rows}" var="row">
            ${row.sid }||${row.name }||${row.email }||${row.city }||$
{row.country}<br/>
      </c:forEach>
</h1>

Code to perform insert operation using jstl
===================================
<sql:setDataSource var="ds" url="jdbc:mysql:///enterprisejavabatch"
            driver="com.mysql.cj.jdbc.Driver" user="root" password="root123" />
      <sql:update dataSource="${ds}" var="count">
            insert into student(`name`,`email`,`city`,`country`)values(?,?,?,?)
            <sql:param value="pandya" />
            <sql:param value="pandya@gmail.com" />
            <sql:param value="GT" />
            <sql:param value="IND" />
      </sql:update>
<h1>The no of rows affected is :: ${count}</h1>
```

Note:
  It is not a good practise to write persistence logic(jdbc code) inside jsp using jstl library,becoz jsp is meant for view part that
  is presentation purpose.