



**UNIVERSITY MALAYSIA TERENGGANU  
FACULTY OF COMPUTER SCIENCE & MATHEMATICS**

**Web Based Application Development  
CSM 3023**

**LAB REPORT:**

**Scriptlet, Page Directive & Include Directive**

**Prepared by:  
NUR ARIFAH BINTI MOHD HANAFIAH  
S66428**

**Prepared for:  
DR MOHAMAD NOR BIN HASSAN**

**BACHELOR OF COMPUTER SCIENCE (MOBILE COMPUTING) WITH HONORS  
SEMESTER II 2023/2024**

# JSP: Scriptlet, Page Directive & Include Directive

Web Programming 2

JSP

---

Name:

Matric #:

Semester:

Lab:

Demonstrator:

---

## Lecturers

PUSAT PENGAJIAN INFORMATIK DAN MATEMATIK  
GUNAAN (PPIMG), UNIVERSITI MALAYSIA TERENGGANU  
(UMT)

## Revision History

Revision Date	Previous Revision Date	Summary of Changes	Changes Marked
		First Issue	Mohamad Nor Hassan
		Second Issue	Dr Rabiei Mamat Dr Faizah Aplop Dr Fouad Ts Dr Rosmayati Mohemad Fakhrul Adli Mohd Zaki
21/02/2019		Addition of Revision History, Table of Contents, Formatting Cover Page	Fakhrul Adli Mohd Zaki

## Table of Contents

<b>Task 1: Passing Data from Main JSP's Page to Other JSP's Page .....</b>	<b>5</b>
<b>Task 2: Using Mathematics operations in JSP.....</b>	<b>10</b>
<b>Task 3: Populate an Array Values into HTML's Table .....</b>	<b>14</b>
<b>Task 4: Perform Calculation of Car Loan .....</b>	<b>16</b>
<b>Task 5: Using JSP Page Directive to Call Java API.....</b>	<b>19</b>
<b>Task 6: Use JSP Include directive for JSP Page .....</b>	<b>25</b>

**Arahan:**

Manual makmal ini adalah untuk kegunaan pelajar-pelajar Pusat Pengajian Informatik dan Matematik Gunaan (PPIMG), Universiti Malaysia Terengganu (UMT) sahaja. Tidak dibenarkan mencetak dan mengedar manual ini tanpa kebenaran rasmi daripada penulis.

Sila ikuti langkah demi langkah sebagaimana yang dinyatakan di dalam manual. Tandakan (✓) setiap langkah yang telah selesai dibuat dan tulis kesimpulan bagi setiap aktiviti yang telah selesai dijalankan.

**Instruction:**

*This laboratory manual is for use by the students of the School of Informatics and Applied Mathematics (PPIMG), Universiti Malaysia Terengganu (UMT) only. It is not permissible to print and distribute this manual without the official authorisation of the author.*

*Please follow step by step as described in the manual. Tick (✓) each step completed and write the conclusions for each completed activity.*

## Task 1: Passing Data from Main JSP's Page to Other JSP's Page

- Objective** : To demonstrate the use of `request.getParameter("fieldName")` for passing input from one JSP's page to another JSP's page.
- Problem** : i. Create a page `memberRegister.jsp`.
- Description** ii. Page `memberRegister.jsp` consists of two (2) inputs;
- IC No (Must be in pre-formatted XXXXXXXXXX)
  - Name
3. In `memberRegister.jsp`, include two (2) buttons;  
*Submit* and *Cancel* button.
4. Create a page `memberProcessing.jsp`.
5. When user click *Submit* button, process the request and display the input key-in in `memberProcessing.jsp` page.
- Estimated time** : 30 minutes

1. Create new Project namely *Lab2*.
2. To create a JSP's page, right click *Lab2* -> *New* -> *JSP*.



3. Key-in File Name: *memberRegister*.

**New JSP**

**Steps**

1. Choose File Type
2. Name and Location

**Name and Location**

File Name:

Project:

Location:

Folder:

Browse...

Created File:

**Options:**

☒ JSP File (Standard Syntax) ☐ Create as a JSP Segment

☐ JSP Document (XML Syntax)

**Description:**

A JSP file using JSP standard syntax.

< Back   Next >   **Finish**   Cancel   Help

4. Click *Finish* button.

5. Source code for *memberRegister.jsp* will appear.

6. Write a HTML's markup to produce HTML's form

```
<body>
  <h1>Passing data from main JSP's page to other JSP's page </h1>
  <form id="memberFrm" action="memberProcessing.jsp" method="post" onsubmit="return checkICNo()">
    <fieldset>
      <legend>Member Registration</legend>
      <label for="invoiceno">Ic No *</label>
      <input type="text" id="icno" name="my_icno" size="15" placeholder="E.g. 921012101245"><br/>
      <label for="name">Name</label>
      <input type="text" id="name" name="my_name" size="45" placeholder="Key-in your name"><br/>
      <p><input type="submit" id="btnSubmit" value="Submit"/>
        <input type="reset" id="btnCancel" value="Cancel"/>
      </p>
    </fieldset>
  </form>
</body>
```

7. Save and compile *memberRegister.jsp* file.

8. Run the *memberRegister.jsp* file and you should get the interface as below:



The screenshot shows a web browser window with the address bar displaying `localhost:8084/Lab2/memberRegister.jsp`. The page title is "Passing data from main JSP's page to other JSP's page". Below the title is a form titled "Member Registration". The form contains two input fields: "Ic No \*" with a placeholder "E.g. 921012101245" and "Name" with a placeholder "Key-in your name". There are two buttons, "Submit" and "Cancel", below the input fields. At the bottom left of the page, there is a copyright notice: "©2016-Mohamad Nor".

The Output :



The screenshot shows a web browser window with the address bar displaying `localhost:8080/Lab2...`. The page title is "Passing data from main JSP's page to other JSP's page". Below the title is a form titled "Member Registration". The form contains two input fields: "Ic No\*" with a placeholder "E.g. 921012101245" and "Name" with a placeholder "Key-in your name". There are two buttons, "Submit" and "Cancel", below the input fields.

9. Repeat step 1 and step 2.

10. Key-in File Name: *memberProcessing*.

11. Click *Finish* button.

12. Source code for *memberProcessing.jsp* will appear.

13. Write a HTML code.



```
8 <!DOCTYPE html>
9 <html>
10 <head>
11 <meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
12 <title>Lab 6 - Task 6</title>
13 </head>
14 <body>
15 <h1>Passing data from main JSP's page to other JSP's page </h1>
16
17 </body>
18 </html>
```

14. Add additional HTML's tag and Java Scriptlet to retrieve the value from main's form.

```
16 <fieldset>
17 <%
18     //Define variables...
19     String myIC = null;
20     String myName = null;
21
22     //Use request.getParameter() method to retrieve data from main's form...
23     myIC = request.getParameter("my_icno");
24     myName = request.getParameter("my_name");
25 >%>
26
27 <!-- Display the output... -->
28 <p>Thank you for registering in this event..!</p>
29 <p>This is your details;</p>
30 <p>IC No : <%=myIC%></p>
31 <p>Name : <%=myName%></p>
32 </fieldset>
```

15. Compile *memberProcessing.jsp* file.

16. Run the *memberRegister.jsp* file and fill-up the input.

Lab 2

localhost:8084/Lab2/memberRegister.jsp

Apps Zimbra Sign In Lentera OneDrive HTML5: The Missing MyNemo ezHASiL Free Code Downloa

## Passing data from main JSP's page to other JSP's page

Member Registration

Ic No \*

Name

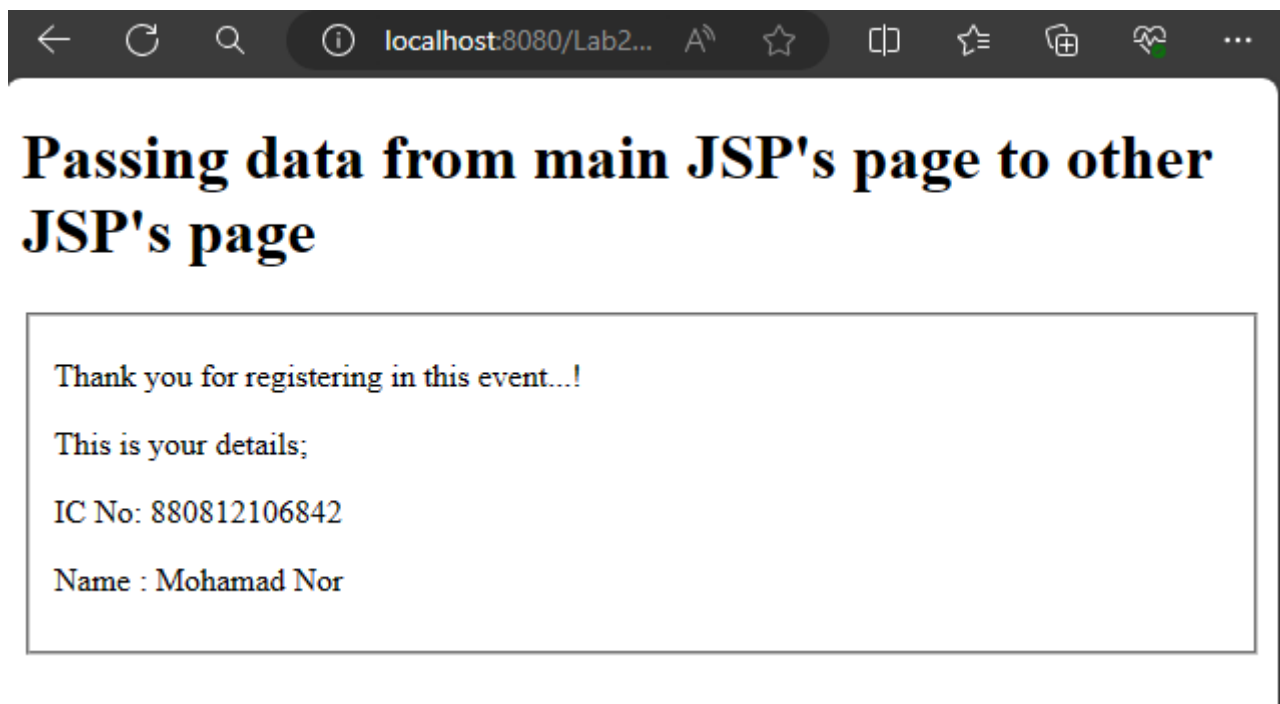
©2016-Mohamad Nor

17. Click *Submit* button to send the request.

18. These inputs will be sent to *memberProcessing.jsp* page and produce the following page.



The Output:



## Reflection

1. How do you want to submit specific information from one form to next form?
2. What happened if the field name you specify in `request.getParameter("field_name")` in second page is different from the field name you defined in first page?

## Task 2: Using Mathematics Operations in JSP

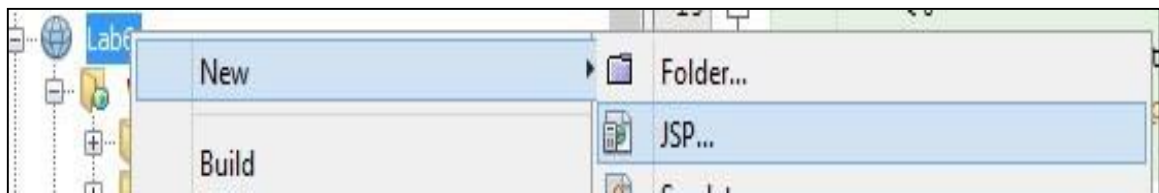
**Objective** : To demonstrate the use of *request.getParameter* (“*Mathematics operations*”) in JSP’s page.

**Problem Description** : i. Create a page *Calculator.jsp* consists of interface represent basic calculator.

ii. When user key-in inputs, process the request and display the results direct in JSP page.

**Estimated time** : 30 minutes

1. Go to Project *Lab2*.
2. To create a JSP’s page, right click *Lab2* -> *New* -> *JSP*.



3. Key-in File Name: *Calculator*.

**Steps**

1. Choose File Type
2. **Name and Location**

**Name and Location**

File Name:

Project:

Location:

Folder:

Created File:

Options:

☒ JSP File (Standard Syntax) ☐ Create as a JSP Segment

☐ JSP Document (XML Syntax)

Description:

4. Click *Finish* button.

5. Source code for *Calculator.jsp* will appear.

6. Write a HTML's markup to produce HTML's form

```
<body bgcolor= "#a00FFF" text= "gold">

<center>

<h2>Basic calculator program in jsp</h2>
<form method ="get" name ="f1">
<input type ="text" size ="20" name ="operand1" value = "" />

<select name = op size = 1>
<option value = "0" >+</option>
<option value = "1" >-</option>
<option value = "2" >*</option>
<option value = "3" >/</option>
<option value = "4" >%</option>
</select>

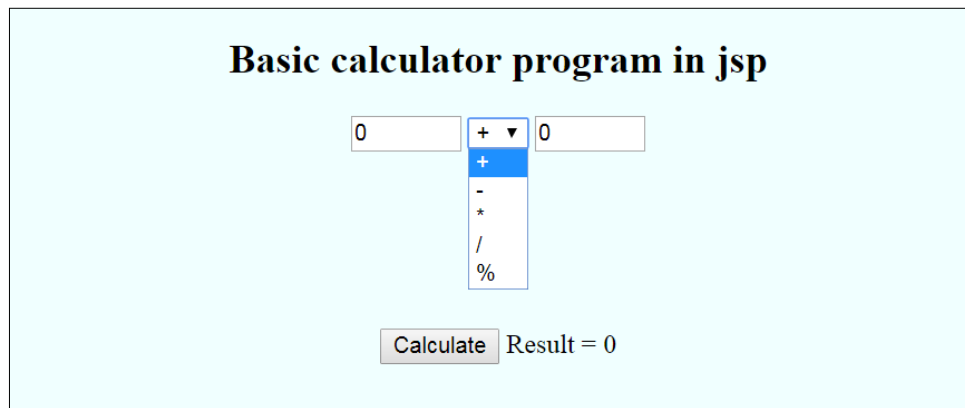
<input type ="text" size="20" name ="operand2" value = ""/>
<p><br><br><br><br>
<input type = submit value = Calculate />

</form>

</body>
```

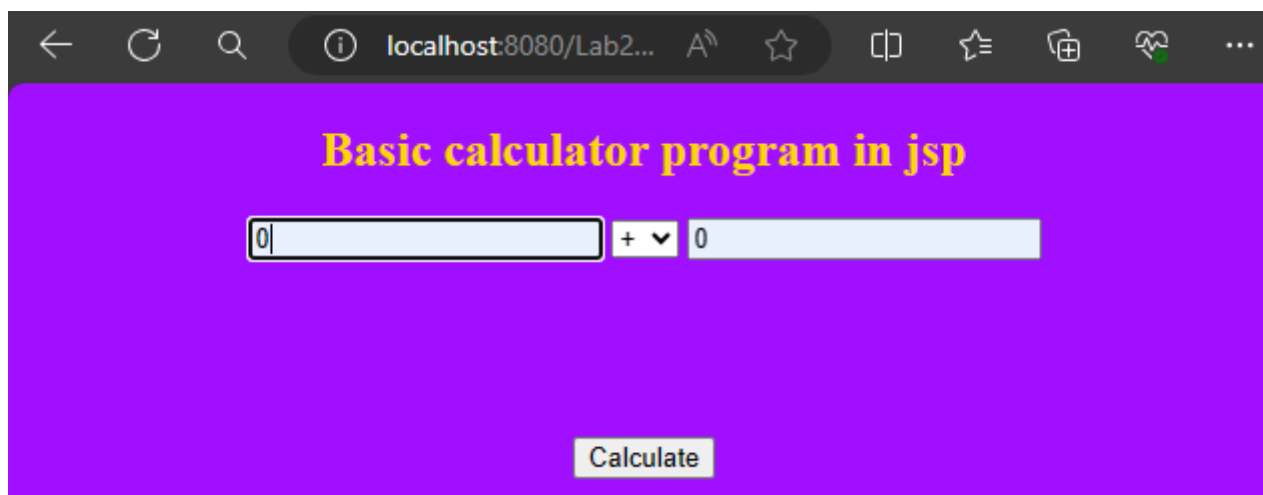
7. Save and compile *Calculator.jsp* file.

8. Run the *Calculator.jsp* file and you should get the interface as below:



The screenshot shows a web page titled "Basic calculator program in jsp". It features two input fields, both containing the number "0". Between these fields is a dropdown menu with a plus sign and a downward arrow. The dropdown menu is open, showing a list of arithmetic operators: "+", "-", "\*", "/", and "%". Below the input fields is a "Calculate" button and a text label "Result = 0".

The Output:



The screenshot shows the same calculator interface as the previous one, but rendered in a web browser. The browser's address bar shows "localhost:8080/Lab2...". The page title is "Basic calculator program in jsp". The input fields are now a single wide field containing "0|", followed by a dropdown menu with a plus sign and a downward arrow, and another input field containing "0". Below the input fields is a "Calculate" button. The background of the page is a solid purple color.

9. Add additional HTML's tag and Java scriplet to retrieve the value from users.

```

<%
String num1 = "0", num2 = "0";
int result = 0;
String op = "+";

char opchar = op.charAt(0);
if (request.getParameter("op") != null) {
    op = request.getParameter("op");
    opchar = op.charAt(0);

    num1 = request.getParameter("operand1");
    num2 = request.getParameter("operand2");

    switch(opchar) {
    case '0': result = Integer.parseInt(num1) + Integer.parseInt(num2);
    break;
    case '1': result = Integer.parseInt(num1) - Integer.parseInt(num2);
    break;
    case '2': result = Integer.parseInt(num1) * Integer.parseInt(num2);
    break;
    case '3': result = Integer.parseInt(num1) / Integer.parseInt(num2);
    break;
    case '4': result = Integer.parseInt(num1) % Integer.parseInt(num2);
    break;
    }
}
%>

Result = <%= result + " " %>

```



9. Further, add additional Java Scriplet to HTML's tag as below.

```
<body bgcolor= "#a00FFF" text= "gold">
<center>

<h2>Basic calculator program in jsp</h2>
<form method = "get" name = "f1">
<input type = "text" size = "20" name = "operand1" value = <%= num1 %> />

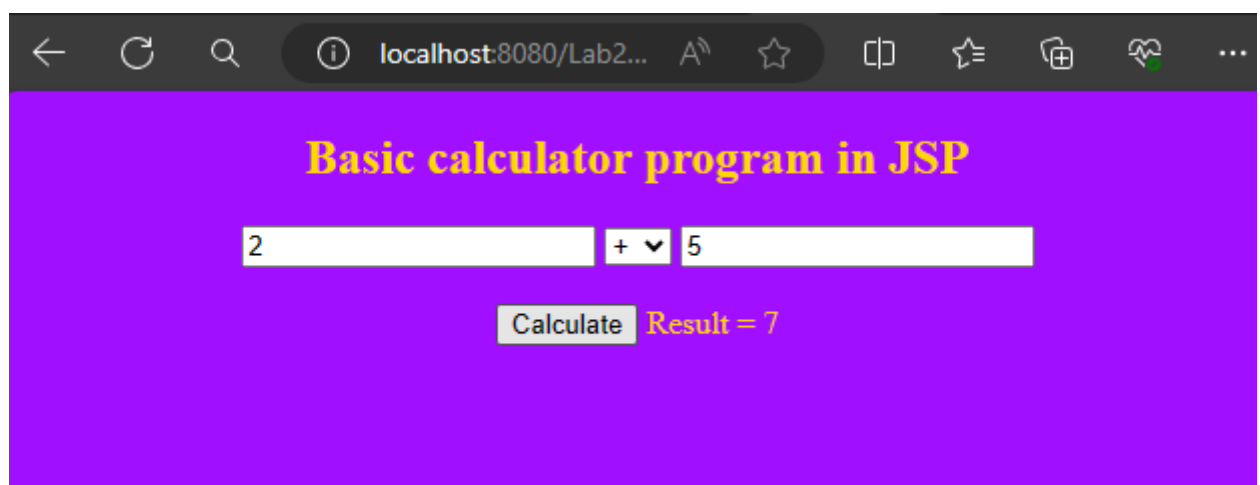
<select name = op size = 1>
<option value = "0" >+</option>
<option value = "1" >-</option>
<option value = "2" >*</option>
<option value = "3" >/</option>
<option value = "4" >%</option>
</select>

<input type = "text" size = "20" name = "operand2" value = <%= num2 %> />
<p>
<input type = submit value = Calculate />

Result = <%= result + " " %>
</form>
</body>
```

10. Compile *Calculator.jsp* file.

11. Run the *Calculator.jsp* file and test the calculator.



## Reflection

1. How do you want to submit specific information from one form to next form?

To submit specific information from one form to the next form in a JavaServer Pages (JSP) application, you can use several methods, including hidden form fields, session attributes, and URL parameters.

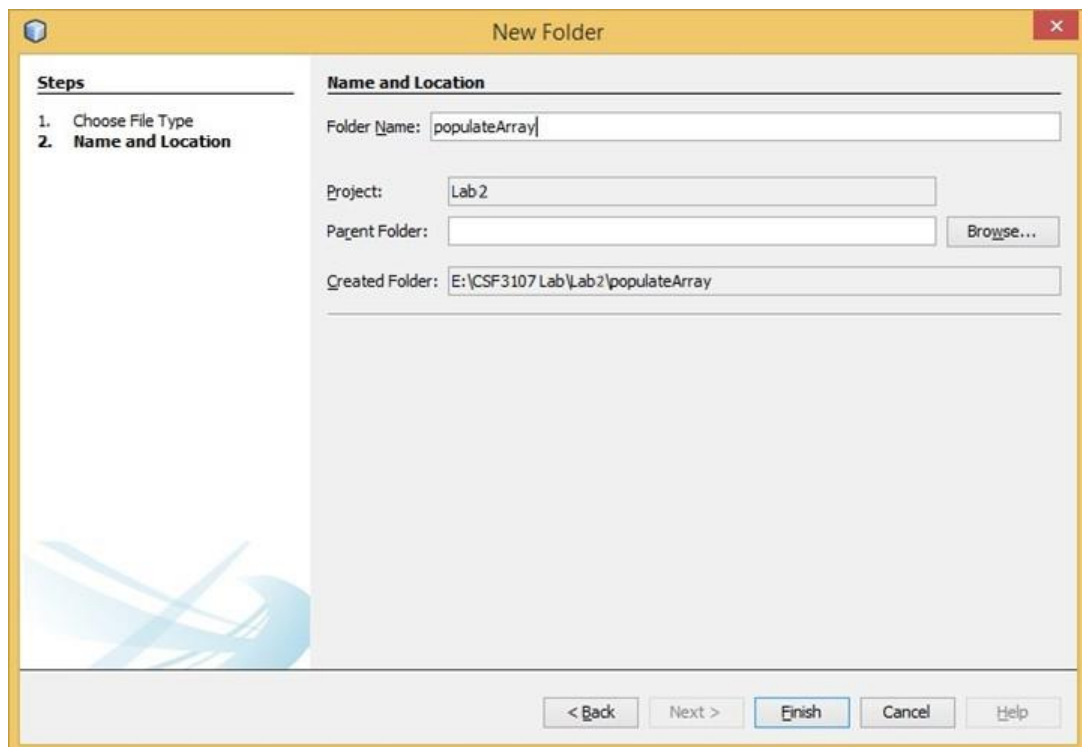
2. What happened if the field name you specify in `request.getParameter("field_name")` in second page is different from the field name you defined in first page?

**It will be not be able to retrieve the parameter value**

## Task 3: Populate an Array Values into HTML's Table

- Objective** : Read Java array and populate it into HTML's table.
- Problem** : i. Create 2D array that store sales data.
- Description** ii. Then, read an array and populate into HTML's table.
- Estimated time** : 50 minutes

1. Go to Project *Lab2*.
2. To create a JSP's page, right click *Lab2* -> *New* -> *JSP*.
3. Key-in File Name: *populateArray*.



4. Click *Finish* button.

5. Prepare standard HTML's a markup for page *populateArray.jsp*.
6. Write a Java Scriptlet and store the following information into an array;

	Jan	Feb	Mac
Salesman 1	2500	2100	2200
Salesman 2	2000	1900	2400
Salesman 3	1800	2200	2450

7. Read the array and populate its value into HTML's table.
8. Save and compile *populateArray.jsp* file.
9. Run the *populateArray.jsp* file and sample of output is shown below:

Read Java array and populate it into HTML's table			
Salesman	Jan	Feb	Mac
Salesman 1	2500	2100	2200
Salesman 2	2000	1900	2400
Salesman 3	1800	2200	2450

©2016-Mohamad Nor

## The Output:

Read Java array and populate it into HTML's table			
Salesman	Jan	Feb	Mar
Salesman 1	2500	2100	2200
Salesman 2	2000	1900	2400
Salesman 3	1800	2200	2450

## Reflection

1. Write a sample syntax to declare 2D Java array.  
`int[][] matrix = new int[3][4];`
2. Define a sequence of steps on how you accomplish Task 7.

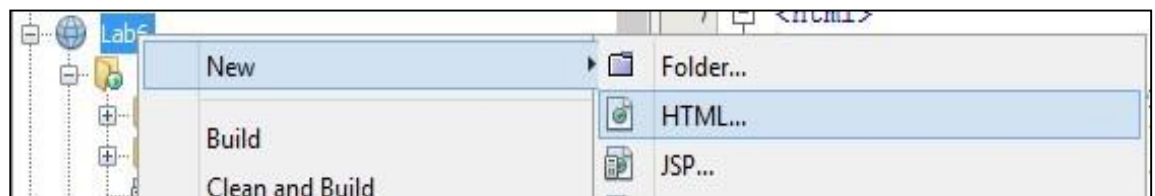
3. What is the difference between HTML's page and JSP's page

JSP is a technology which is used to create dynamic web applications. HTML is a standard markup language which is used to create the structure of web pages. JSP allow to place the custom tag or third party tag. It does not allow to place the custom tag or third party tag.

## Task 4: Perform Calculation of Car Loan

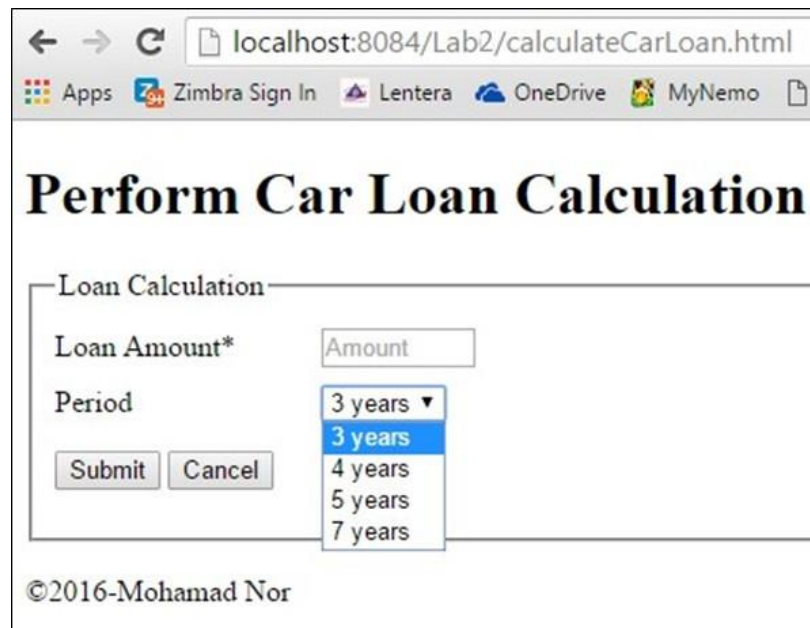
<b>Objective</b>	: Passing input to next page for further processing.
<b>Problem Description</b>	: i. Create simple interface in HTML that consists of Loan Amount and Loan Period. ( Loan Period < 5 years, interest is 2.8% per year, and > 5 years interest is 4.5% per year).  ii. Submit the form and perform calculation based on user input and, finally, display the result.
<b>Estimated time</b>	: 50 minutes

1. Go to Project Lab2.
2. To create a HTML's page, right click Lab2 -> New -> JSP



3. Key-in File Name: *calculateCarLoan*
4. Create a standard HTML's markup for form.
5. In your form, create two (2) fields; *Loan Amount* and *Loan Period*.
6. Save *calculateCarLoan.html* and run the file.

7. You will get the following output.



← → ↻ localhost:8084/Lab2/calculateCarLoan.html

Apps Zimbra Sign In Lentera OneDrive MyNemo

## Perform Car Loan Calculation

Loan Calculation

Loan Amount\*

Period

©2016-Mohamad Nor

8. Create JSP's file and rename the file as *processCalculateCarLoan*.

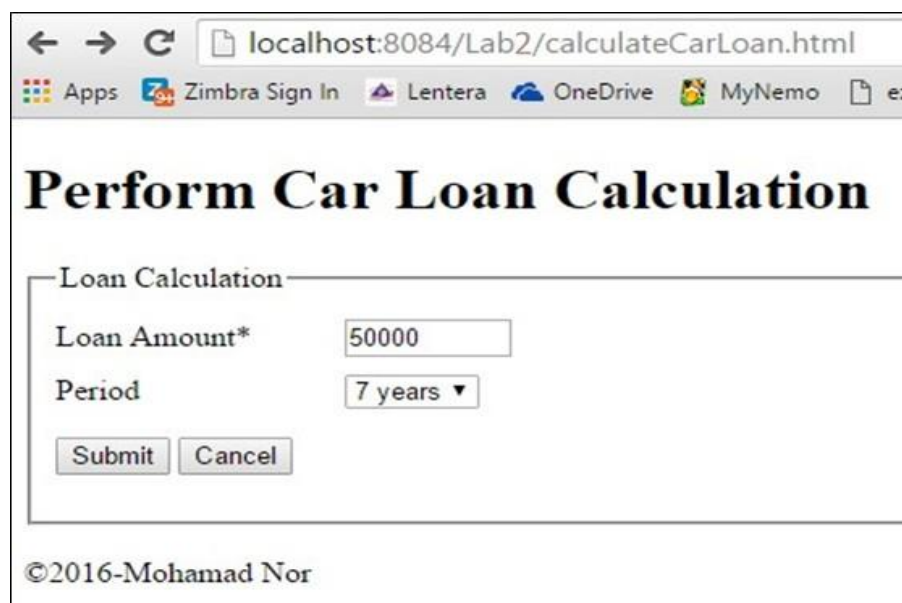
9. Construct the logic for calculating car loan and display the result.

10. Save and compile *processCalculateCarLoan.jsp*.

11. Run *calculateCarLoan.html* file and fill-up the input.

12. Then, submit your result.

13. You should get the following output;



← → ↻ localhost:8084/Lab2/calculateCarLoan.html

Apps Zimbra Sign In Lentera OneDrive MyNemo

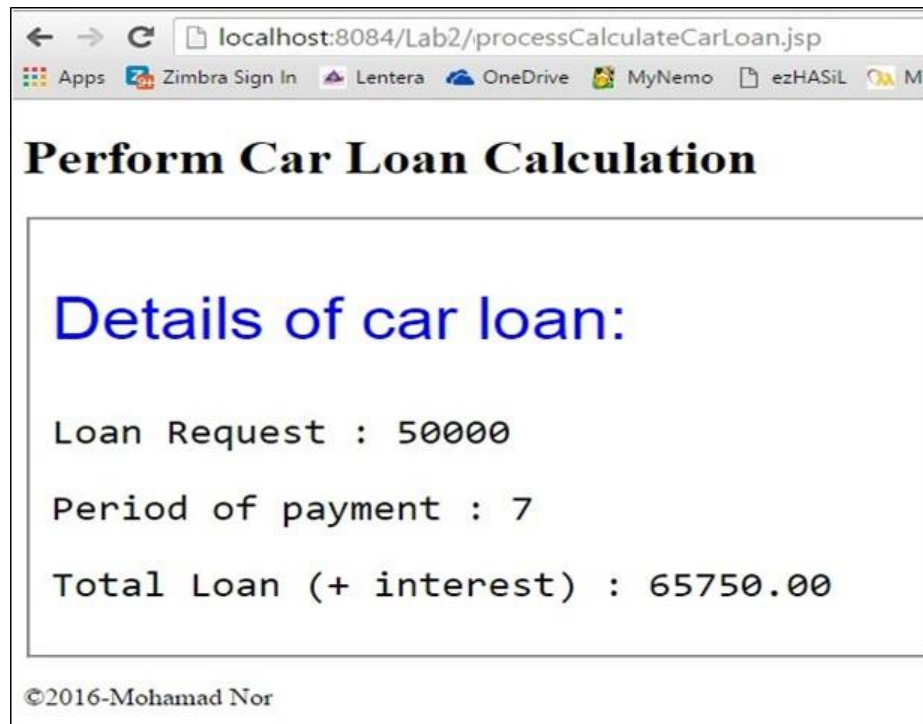
## Perform Car Loan Calculation

Loan Calculation

Loan Amount\*

Period

©2016-Mohamad Nor



## Reflection

1. How you want to retrieve data from previous page?

use request parameters or session attributes, depending on how the data was sent from the previous page. Example, using HTTP GET or POST methods or request parameter.

2. Where the construction of logic occur for calculating Total Loan ( + interest) ?



## Task 5: Using JSP Page Directive to Call Java API

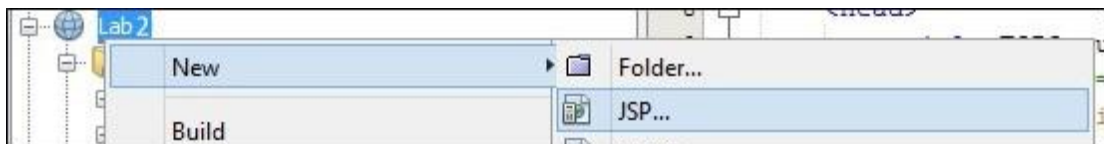
**Objective** : Use JSP page directive elements to call certain Java API.

**Problem** : Using Java *ArrayList* object to store data and retrieve it via JSP page.

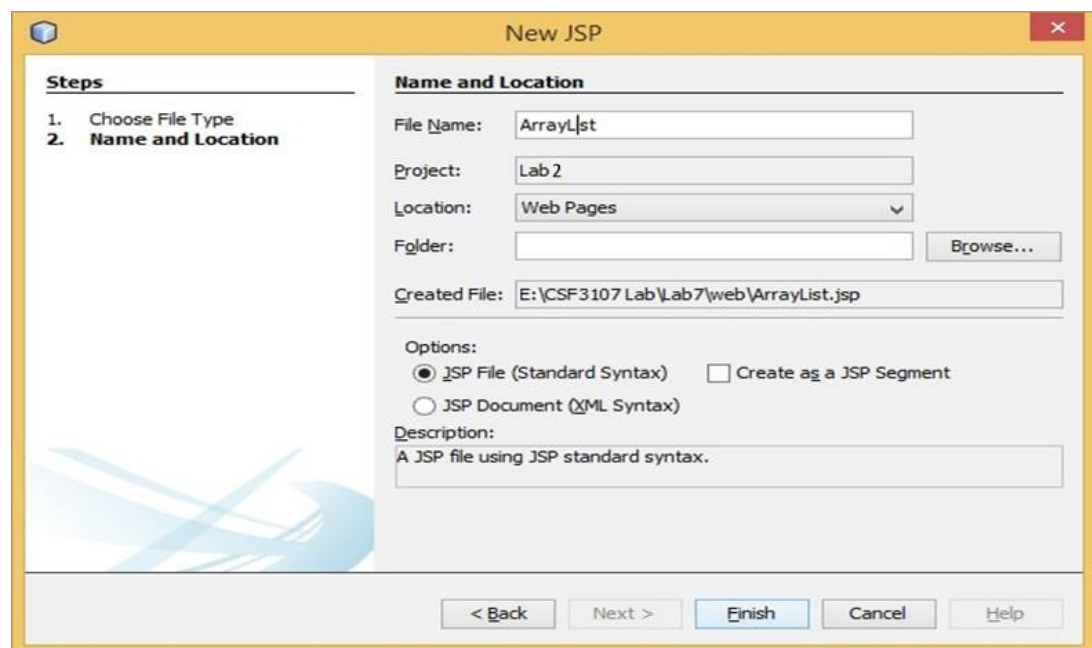
**Description**

**Estimated time** : 20 minutes

1. Create a new JSP's file.



2. Type file name as *ArrayList*.



2. Click *Finish* button.

4. Type title as *Use Java ArrayList*.

5. Type header1 as *Use JSP Page Directive*

```
1 <%--
2     Document    : ArrayList
3     Created on  : 10-Apr-2016, 09:24:46
4     Author     : Mohamad Nor Hassan
5 --%>
6
7 <%@page contentType="text/html" pageEncoding="UTF-8"%>
8 <!DOCTYPE html>
9 <html>
10 <head>
11     <meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
12     <title>Use Java ArrayList</title>
13 </head>
14 <body>
15     <h1>Use JSP Page Directive</h1>
16 </body>
17 <br/>
18 <footer>&copy;2016-Mohamad Nor</footer>
19 </html>
```

6. In order to use Java *ArrayList*'s object, we need to use JSP Page Directive and import the related API.

```
7 <%@page contentType="text/html" pageEncoding="UTF-8"%>
8 <%@page import="java.util.ArrayList"%>
```

7. In order to use Java syntax, create a Java Scriptlet notation.

```
15 <body>
16     <h1>Use JSP Page Directive</h1>
17     <%
18
19     %>
20 </body>
```

8. Create an object *ArrayList* to store a list of student name.

```
17 <%
18     //Create ArrayList object ...
19     ArrayList<String> studentList = new ArrayList<String>();
20 %>
```

9. Add the following name to ArrayList's object.

- ✓ Mohamad Azam
- ✓ Peter Chong
- ✓ Rahimah Mansor
- ✓ Sri Devi
- ✓ Ng Hue Ween
- ✓ S. Nagarajan

```
21 //Store student name..  
22 studentList.add(0, "Mohamad Azam");  
23 studentList.add(1, "Peter Chong");  
24 studentList.add(2, "Rahimah Mansor");  
25 studentList.add(3, "Sri Devi");  
26 studentList.add(4, "Ng Hue Ween");  
27 studentList.add(5, "S. Nagarajan");
```

10. Display the number of records for an ArrayList's object.

```
28  
29 //Display the number of records..  
30 out.println("<p>The number of records in ArrayList are " +  
31 studentList.size() + "</p>");
```

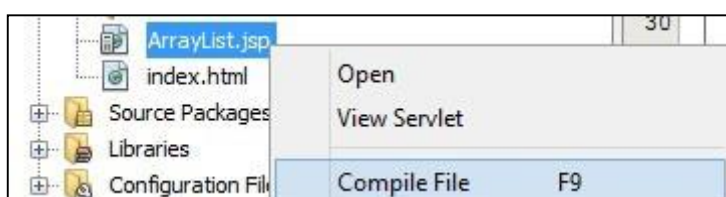
11. Finally, populate the list of students.

```
38  
39 //Populate a list of students..  
40 for (int i=0; i < studentList.size(); i++ )  
41 out.println("<p>Record " + (i+1) + " is " + studentList.get(i) + "</p>");
```

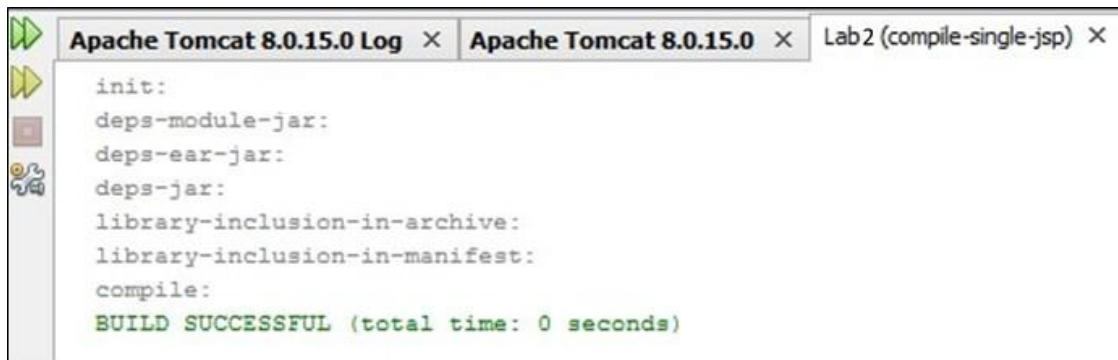
12. Click *SaveAll* icon.



13. Right click file *ArrayList.jsp* and click *Compile File* (F9).

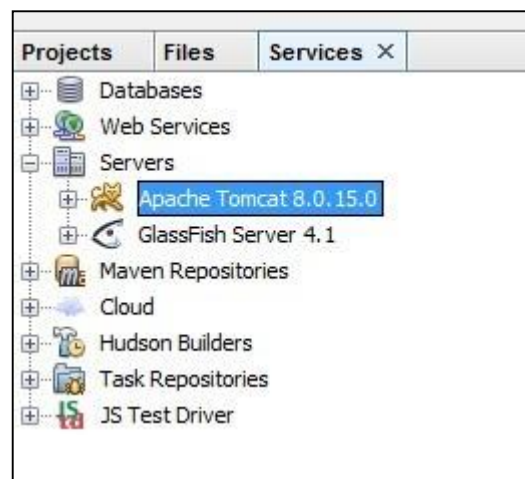


14. You will get notification message the the bottom of Netbeans IDE with the green colour.

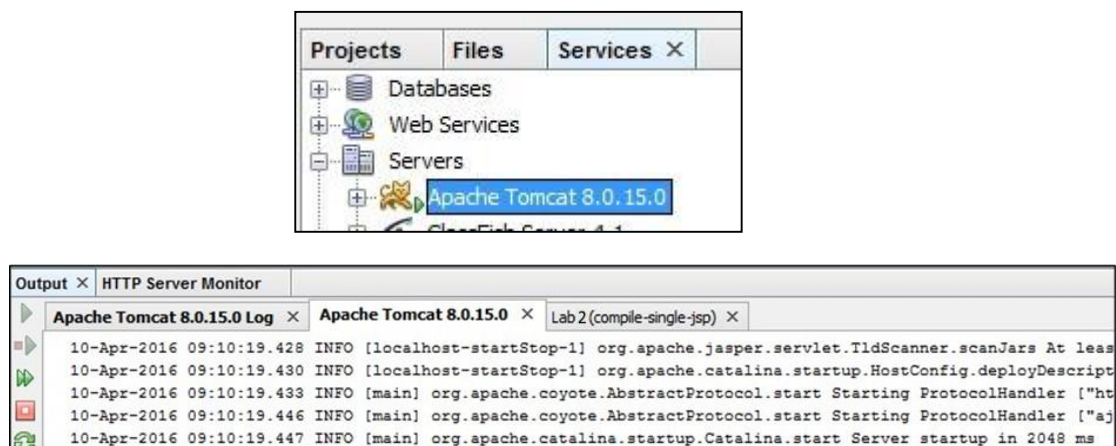


15. Before running any JSP's files for first time upon opening your Netbeans IDE, you need to start your web server (i.e; Apache Tomcat).

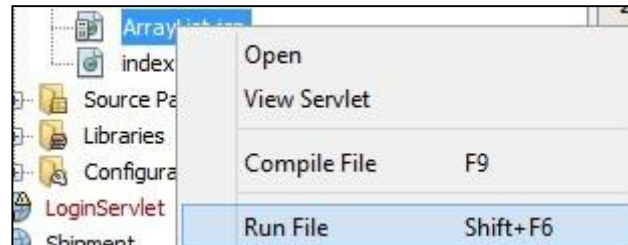
16. To perform this, go to *Services -> Servers -> Apache Tomcat*.



17. You should get the green indicator at *Apache Tomcat's* icon and *Apache Tomcat* output message with the time taken to start specified time to start *Apache Tomcat* web server.

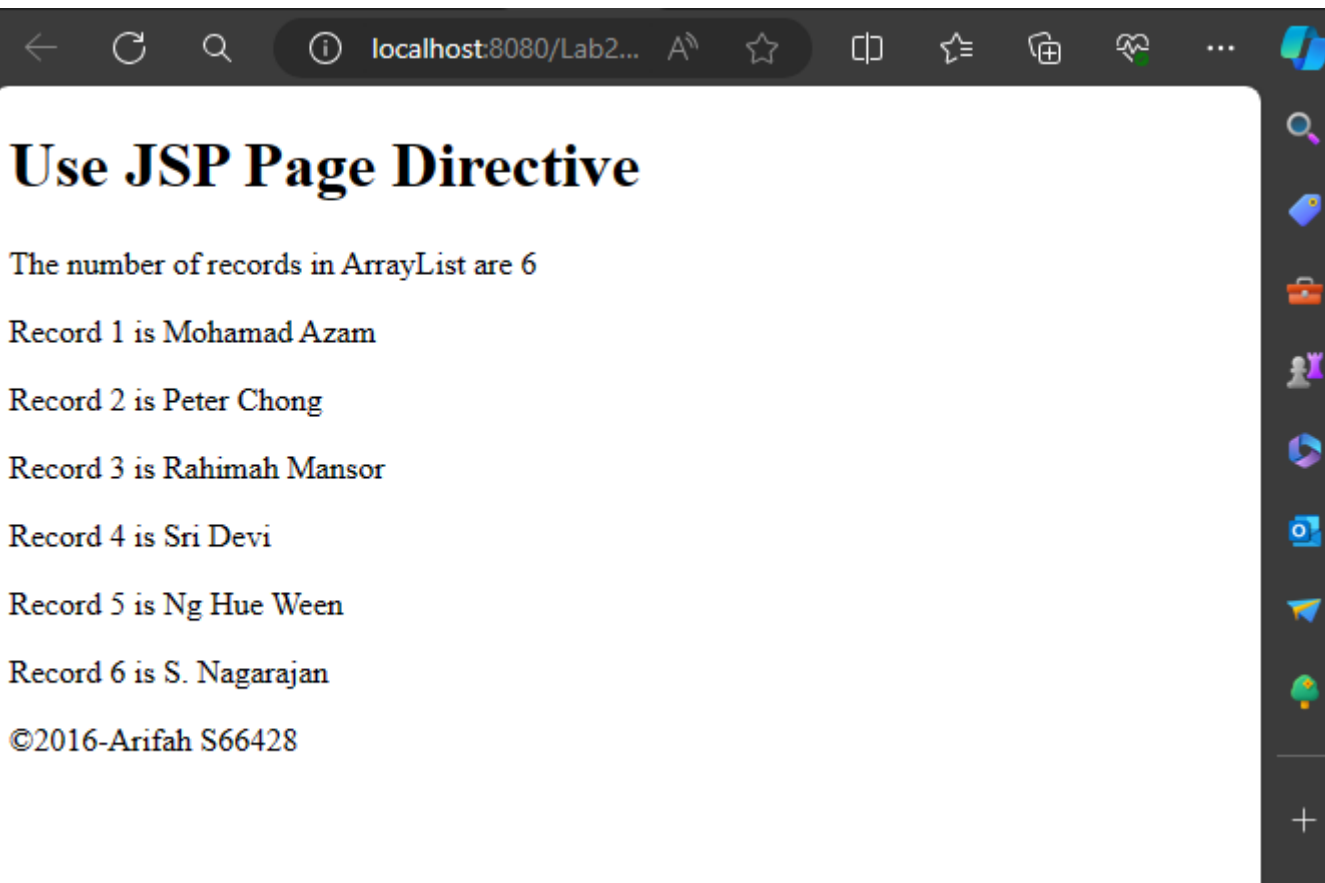


18. Go to *Project's* tab. Then right click file *ArrayList.jsp* and click *Run File* (Shift+F6).



19. Output will appear in web browser.





## Use JSP Page Directive

The number of records in ArrayList are 6

Record 1 is Mohamad Azam

Record 2 is Peter Chong

Record 3 is Rahimah Mansor

Record 4 is Sri Devi

Record 5 is Ng Hue Ween

Record 6 is S. Nagarajan

©2016-Arifah S66428

### Reflection

1. What you have learnt from this exercise?

**I have learnt how to Using Java *ArrayList* object to store data and retrieve it via JSP page.**

2. Write a sample syntax how you want to use java *Math* object in JSP?

```
<%-- Use the Math object to perform operations --%>

<%
double number = 25;
double squareRoot = Math.sqrt(number);

int randomNumber = (int) (Math.random() * 10) + 1; %>
```

3. List and write a sample syntax for THREE (3) of JSP page directive.

1. **<%@ page contentType="text/html; charset=UTF-8" %>**
2. **<%@ page errorPage="errorPage.jsp" %>**
3. **<%@ page language="java" %>**



## Task 6: Use JSP Include directive for JSP Page

<b>Objective</b>	: Demonstrate the use of JSP Page Include directive.
<b>Problem</b>	: Create a JSP master page that displays the header, main contents and footer.
<b>Description</b>	
<b>Estimated time</b>	: 30 minutes

1. Create a new JSP's file.
2. Type file name as *mainPage*.
3. Create content for *mainPage.jsp* as below.

### Using JSP Include directive

Java Server Page (JSP) is a technology for controlling the content or appearance of Web pages through the use of servlets, small programs that are specified in the Web page and run on the Web server to modify the Web page before it is sent to the user who requested it.

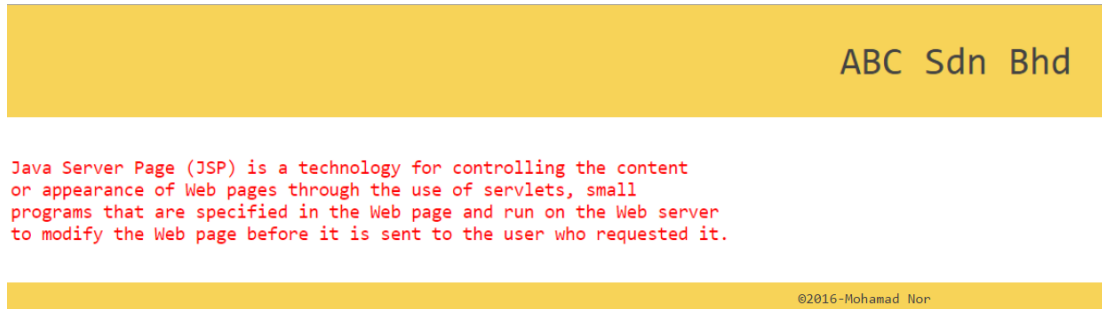
4. Create a header file as *headerPage.jsp* and display the following output.

ABC Sdn Bhd

5. Create a header file as *footerPage.jsp* and display the following output

©2016-Mohamad Nor

6. Include your *headerPage.jsp* and *footerPage.jsp* inside your *mainPage.jsp*.
7. Save *mainPage.jsp*
8. Compile and run *mainPage.jsp*.
9. You should get the following output.



## Reflection

1. What you have learnt from this exercise?

**I learnt how to insert the headerPage.jsp and footerPage.jsp into the mainPage.jsp**

2. Write a syntax how you want to include *common.html* file that located at a directory known as *master*.

**<%@ include file="master/common.html" %>**



The Ouput:

**ABC Sdn Bhd**

## Using JSP Include Directive

Java Server Page (JSP) is a technology for controlling the content or appearance of Web pages through the use of servlets, small programs that are specified in the Web page and run on the Web server to modify the Web page before it is sent to the user who requested it.

Copyright 2024 - Arifah S66428

## Exercise

1. Write a JSP's page to convert temperatures to Fahrenheit temperatures and via versa. The formula is given as:

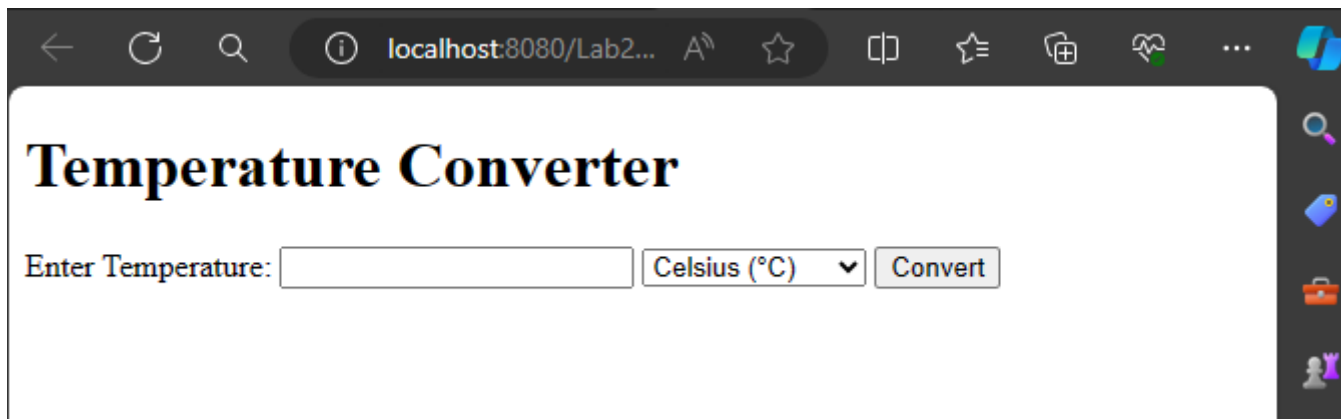
$$F = (9/5)C + 32$$

Your program should ask the user to enter a temperature in Celsius, and then display the temperature converted to Fahrenheit.

### The Code:

```
1  <!--
2      Document   : convertTemperature
3      Created on : 20 Apr 2024, 12:26:28 am
4      Author    : Arifah S66428
5  -->
6
7  <%@page contentType="text/html" pageEncoding="UTF-8"%>
8  <!DOCTYPE html>
9  <html>
10 <head>
11 <meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
12 <title>Convert Temperature</title>
13 </head>
14 <body>
15 <h1>Temperature Converter</h1>
16
17 <form method="post">
18 <label for="temperature">Enter Temperature:</label>
19 <input type="number" id="temperature" name="temperature" required>
20 <select name="unit">
21 <option value="celsius">Celsius (°C)</option>
22 <option value="fahrenheit">Fahrenheit (°F)</option>
23 </select>
24 <button type="submit">Convert</button>
25 </form>
26
27
28 <%
29     if (request.getMethod().equalsIgnoreCase("POST")) {
30         double temp = Double.parseDouble(request.getParameter("temperature"));
31         String unit = request.getParameter("unit");
32         double convertedTemp = 0;
33         String convertedUnit = "";
34
35         if (unit.equals("celsius")) {
36             convertedTemp = (temp * 9 / 5) + 32;
37             convertedUnit = "Fahrenheit";
38         } else if (unit.equals("fahrenheit")) {
39             convertedTemp = (temp - 32) * 5 / 9;
40             convertedUnit = "Celsius";
41         }
42     }
43
44     <h2>Conversion Result:</h2>
45     <p><%= temp %> <%= unit.toUpperCase() %> is equivalent to <%= convertedTemp %> <%= convertedUnit %>.</p>
46
47 <%
48 }
```

### The Output :



2. Write a JSP's form that asks for the length and width of two rectangles. The program should tell the user which rectangle has the greater area, or if the areas are the same. [Note: All result must be in 2 decimal places].

### The Code:

```

1  <!--
2      Document   : rectangleArea
3      Created on : 20 Apr 2024, 12:30:00 am
4      Author    : Arifah S66428
5  -->
6
7  <%@page contentType="text/html" pageEncoding="UTF-8"%>
8  <!DOCTYPE html>
9  <html>
10 <head>
11     <meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
12     <title>Rectangle Area Comparison</title>
13 </head>
14 <body>
15     <h1>Rectangle Area Comparison</h1>
16
17     <form method="post">
18         <label for="length1">Rectangle 1 Length:</label>
19         <input type="number" id="length1" name="length1" step="0.01" required>
20         <label for="width1">Width:</label>
21         <input type="number" id="width1" name="width1" step="0.01" required><br>
22
23         <label for="length2">Rectangle 2 Length:</label>
24         <input type="number" id="length2" name="length2" step="0.01" required>
25         <label for="width2">Width:</label>
26         <input type="number" id="width2" name="width2" step="0.01" required><br>
27
28         <button type="submit">Compare Areas</button>
29     </form>

```

```

30
31 <%
32     if (request.getMethod().equalsIgnoreCase("POST")) {
33         double length1 = Double.parseDouble(request.getParameter("length1"));
34         double width1 = Double.parseDouble(request.getParameter("width1"));
35         double length2 = Double.parseDouble(request.getParameter("length2"));
36         double width2 = Double.parseDouble(request.getParameter("width2"));
37
38         double areal = length1 * width1;
39         double area2 = length2 * width2;
40
41         String resultMessage = "";
42         if (areal > area2) {
43             resultMessage = "Rectangle 1 has a greater area than Rectangle 2.";
44         } else if (areal < area2) {
45             resultMessage = "Rectangle 2 has a greater area than Rectangle 1.";
46         } else {
47             resultMessage = "Both rectangles have the same area.";
48         }
49     }
50     <h2>Area Comparison Result:</h2>
51     <p><%= resultMessage %></p>
52     <p>Area of Rectangle 1: <%= String.format("%.2f", areal) %> square units</p>
53     <p>Area of Rectangle 2: <%= String.format("%.2f", area2) %> square units</p>
54     <%
55     }
56     %>
57 </body>
58 </html>

```

The Output:

## Rectangle Area Comparison

Rectangle 1 Length:	<input type="text"/>	Width:	<input type="text"/>
Rectangle 2 Length:	<input type="text"/>	Width:	<input type="text"/>
<input type="button" value="Compare Areas"/>			

### Area Comparison Result:

Rectangle 2 has a greater area than Rectangle 1.

Area of Rectangle 1: 13.53 square units

Area of Rectangle 2: 24.00 square units