



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

**Web Based Application Development
CSM 3023**

**LAB REPORT 5:
JSP: JavaBeans & Java Standard Tag Library (JSTL)**

**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**



Week 5

JSP: JavaBeans & Java Standard Tag Library (JSTL)

Web Programming 2

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

Task 1: Using Scriptlet to Access a Simple JavaBeans.....	4
Task 2: Problem Solving using JavaBeans	3
Task 3: Installing JSTL Taglibs	5
Task 4: Using Java Standard Tag Library (JSTL).....	8
Task 5: Using JSP Standard Tag Library	14

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.

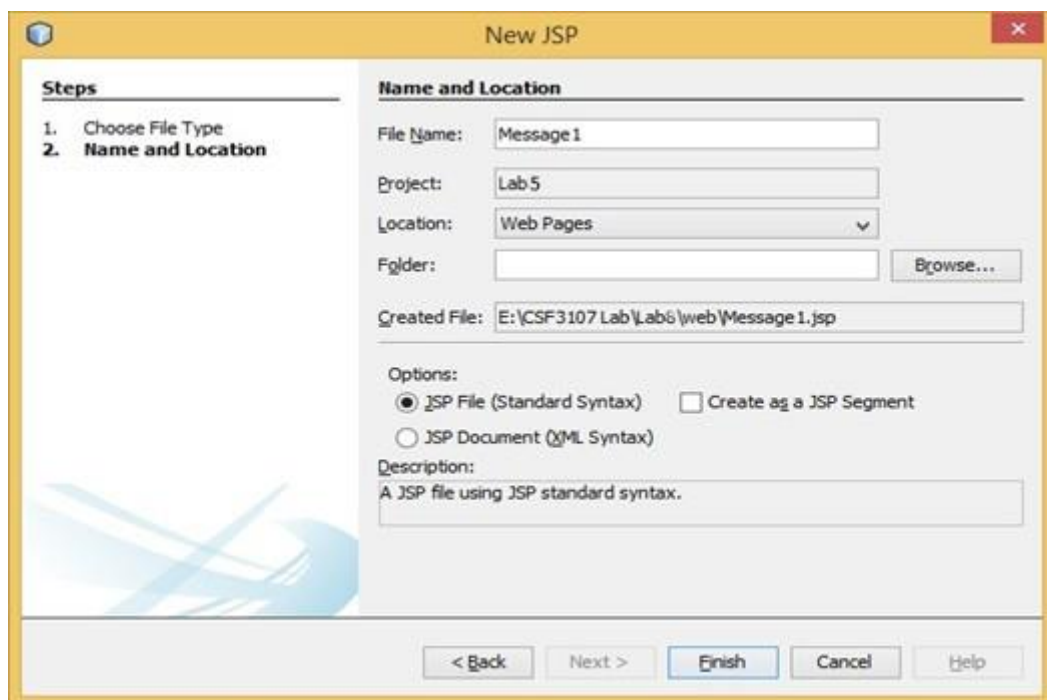
Task 1: Using Scriptlet to Access a Simple JavaBeans

Objective: Use Java Scriptlet to access JavaBeans

Problem Description: Write a JavaBeans that can display message “Welcome to CSF3107 courses....!”.

Estimated time: 20 minutes

1. Go to *Lab5*’s project.
2. Create a new JSP’s file.
3. Type file name as *Message1*.



4. Rename title as *Using JSP Scriptlet*.

5. Rename `<h1>` as *Using JSP Standard Action to call JavaBeans*.

```
9  <html>
10 <head>
11     <meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
12     <title>Using JSP Scriptlet </title>
13 </head>
14 <body>
15     <h1>Using JSP Scriptlet to call JavaBeans</h1>
16 </body>
17 </html>
```

6. Use JSP page directive to include the information such as content type, page info, language, lab8.com package and use java util API.

```
1  <!--
2      Document      : message1
3      Created on    : 18-Apr-2016, 16:13:08
4      Author       : Mohamad Nor Hassan
5  -->
6
7  <%@page contentType="text/html" pageEncoding="UTF-8"%>
8  <%@page language="java"%>
9  <%@page info="Using JSP Standard Action to call JavaBeans"%>
10 <%@page import="java.util.Date, lab5.com.Message"%>
```

7. Use a Java scriptlet to create an object for *Message* class.

```
20 <%
21     //Create an object..
22     Message objMsg = new Message();
23
24 %>
```

8. Then, assign the value as *"Welcome to CSF3107 course....!"* via setter method *setMsg(String msg)*.

```
24 //Assign value..
25 objMsg.setMsg("Welcome to CSF3107 course....!");
```

9. Use *getMsg()*'s method to display the message at paragraph.

```
27 //Display value...
28 out.println("<p>" + objMsg.getMsg() + "</p>");
```

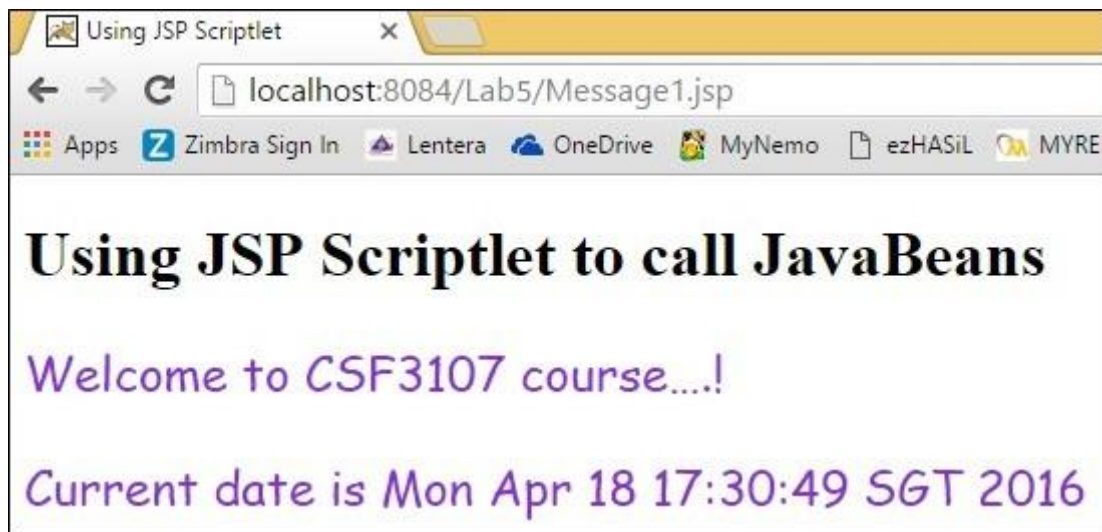
10. Add current date to the next paragraph.

```
30 //Add date..  
31 out.println("<p>Current date is " + new java.util.Date() + "</p>");
```

11. Save *Message1.jsp*

12. Compile and run *Message1.jsp*.

13. You should get the following output.



Reflection

1. What you have learnt from this exercise?

I have learnt how to use Scriptlet to access the JavaBeans

2. Explain the differences when calling JavaBeans using JSP Standard Action and Java Scriptlet.

JSP Standard Actions:

Uses specific XML-like tags like <jsp:useBean>, <jsp:setProperty>, and <jsp:getProperty>. For readability it keeps the JSP page clean and easy to read by separating Java code from HTML.

Java Scriptlets:

Embeds Java code directly in the JSP using <% ... %> tags. Next, it can make the JSP page messy and hard to maintain by mixing Java code with HTML.

Task 2: Problem Solving using JavaBeans

Objective:	Use JavaBeans in JavaScriptlet
Problem Description:	<p>Create sample web form to register IT's training based on the these fees;</p> <p>C++ training (values as "1")= RM3000 per pax Java for beginner (values as "2")= RM3000 per pax HTML5 (values as "3")= RM2800 per pax Java EEE (values as "4")= RM5500 per pax Android Programming (values as "5")= RM3200 per pax Student = Yes (values as "1") and No (Values as "0") Student will entitle 10% discount You must cater all front-end validation.</p>
Estimated time:	60 minutes

1. Choose Project *Lab5*.
2. Create a new JSP's file.



3. Type file name as *registerTraining*.
4. Prepare the following Graphical User Interface (GUI).

A screenshot of a web form titled 'Register IT Training'. The form has a title bar with 'Register IT Training'. Below the title bar, there is a section titled 'Training Registration'. Inside this section, there are several input fields: 'IC No' with a placeholder 'E.g.: 911210-05-1234', 'Name' with a placeholder 'Enter your name', 'Type of Training' with a dropdown menu showing 'C++ training', 'No of Pax' with a placeholder 'No of pax', and 'Student' with radio buttons for 'Yes' and 'No'. At the bottom of the form, there are 'Submit' and 'Cancel' buttons. Below the form, there is a copyright notice: '©2016-Mohamad Nor'.

5. Create a *Register* JavaBeans to cater the business requirements and store it into package *lab8.com*.
6. Create a new file name known as *processTraining.jsp*.

7. Attached the *pocessTraining.jsp*'s form to *registerTraining.jsp*.
8. Open *pocessTraining.jsp*'s and use Java Scriptlet to invoke *Register* JavaBeans.
9. Output will appear in web browser.



← → ↻ 📄 localhost:8084/Lab8/processTraining.jsp

Apps Zimbra Sign In Lentera OneDrive MyNemo ezHASiL MYREN Clo

Training Registration Acknowledgement

IC No : 910710-11-2416

Name : Mohamad Nor Hassan

Type of Training : Java EEE

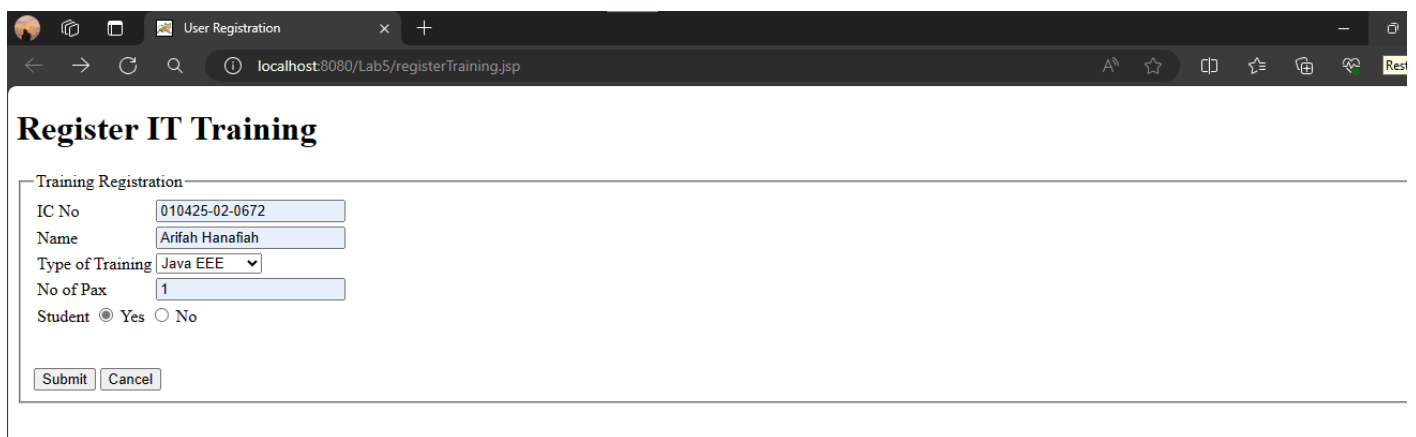
Number of Pax : 1 person/s

Student : Yes

Amount Due : RM 4950.00

©2016-Mohamad Nor

The Output :



User Registration × +

← → ↻ 🔍 📄 localhost:8080/Lab5/registerTraining.jsp

Register IT Training

Training Registration

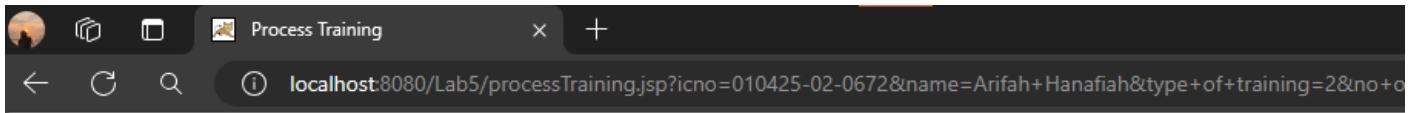
IC No

Name

Type of Training

No of Pax

Student ☒ Yes ☐ No



Training Registration Acknowledgement

IC No: 010425-02-0672

Name: Arifah Hanafiah

Type of Training: Java EEE

No of Pax: 1 person/s

Student: Yes

Amount Due: RM4950.00

Reflection

1. What you have learnt from this exercise?

I have learnt how to use JavaBeans in JavaScriptlet.

2. Describe the steps how you construct *Register* JavaBeans?

To construct a Register JavaBean, follow these steps: First, create a Java class and define private fields for the properties you want to store, such as username, password, and email. Next, provide public getter and setter methods for each property to allow access and modification. Optionally, include a no-argument constructor for creating instances and other methods for business logic if needed. Finally, save the Java class in an appropriate package, and you can use it in your JSP pages with standard actions like <jsp:useBean>, <jsp:setProperty>, and <jsp:getProperty>.

Task 3: Installing JSTL Taglibs

Objective: Installation of Apache Taglibs

Problem Description: To install JSTL Library

Estimated time: 25 minutes

1. Go to the browser and type URL

<http://tomcat.apache.org/taglibs/index.html>

Apache Taglibs

This project is an open source repository for JSP(tm) Tag Libraries.

In particular, Apache Taglibs hosts the [Apache Standard Taglib](#), an implementation of the JSTL specification. Various versions are available, and a 1.2 implementation is in the works.

2. Click on *Apache Standard Taglib*.

Standard Taglib

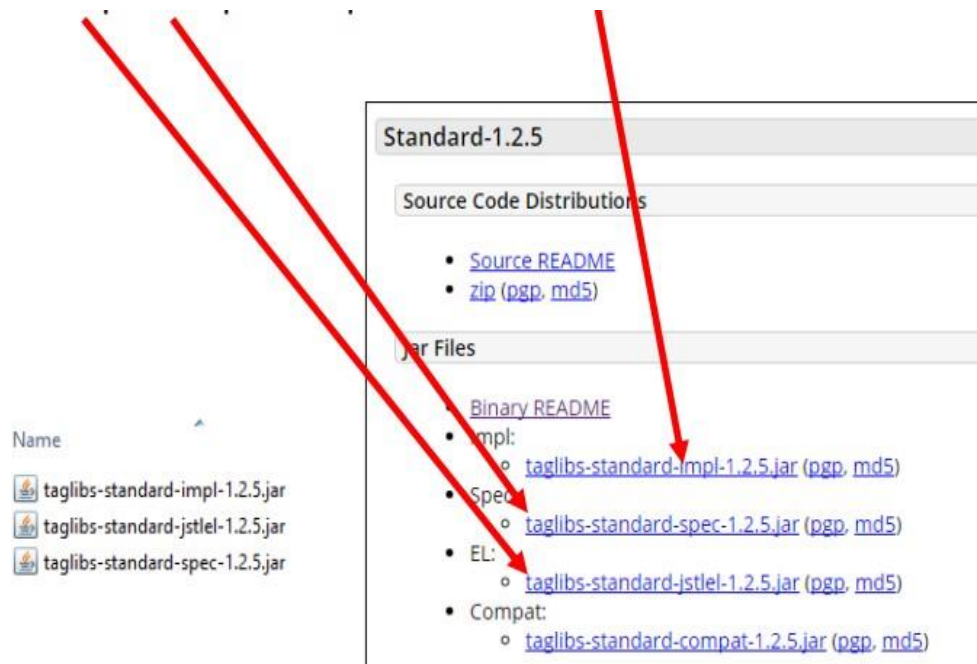
JSP(tm) Standard Tag Library implementations

Apache hosts the Apache Standard Taglib, an implementation of the JSP Standard Tag Library (JSTL) specification. Various versions are available.

Version	JSTL version	Requirements	Getting the Taglib
Standard 1.2.3	JSTL 1.2	Servlet 2.5, JavaServer Pages 2.1	download (javadoc)
Standard 1.1	JSTL 1.1	Servlet 2.4, JavaServer Pages 2.0	download (javadoc)
Standard 1.0	JSTL 1.0	Servlet 2.3, JavaServer Pages 1.2	download (javadoc)

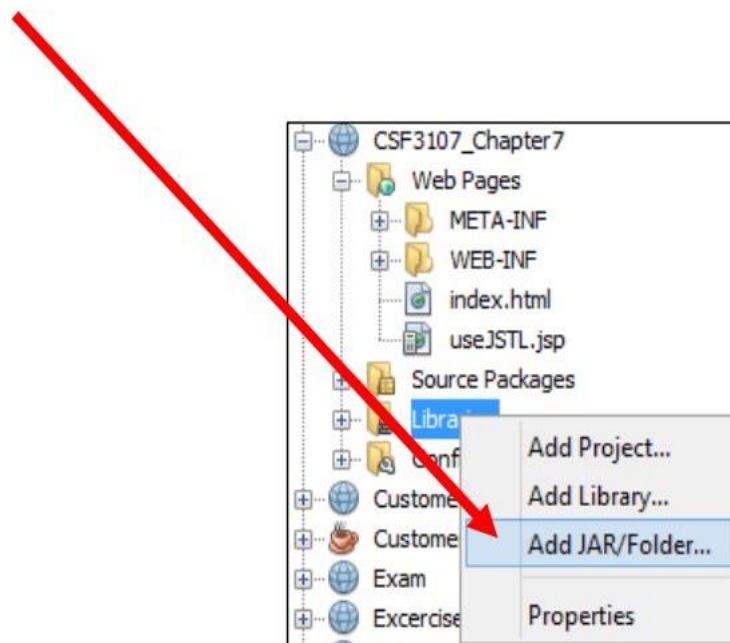
3. Choose Version 1.2 and click download link.
4. Click to jar files for Impl and save the link.

5. Repeat step 3 for Spec and EL.

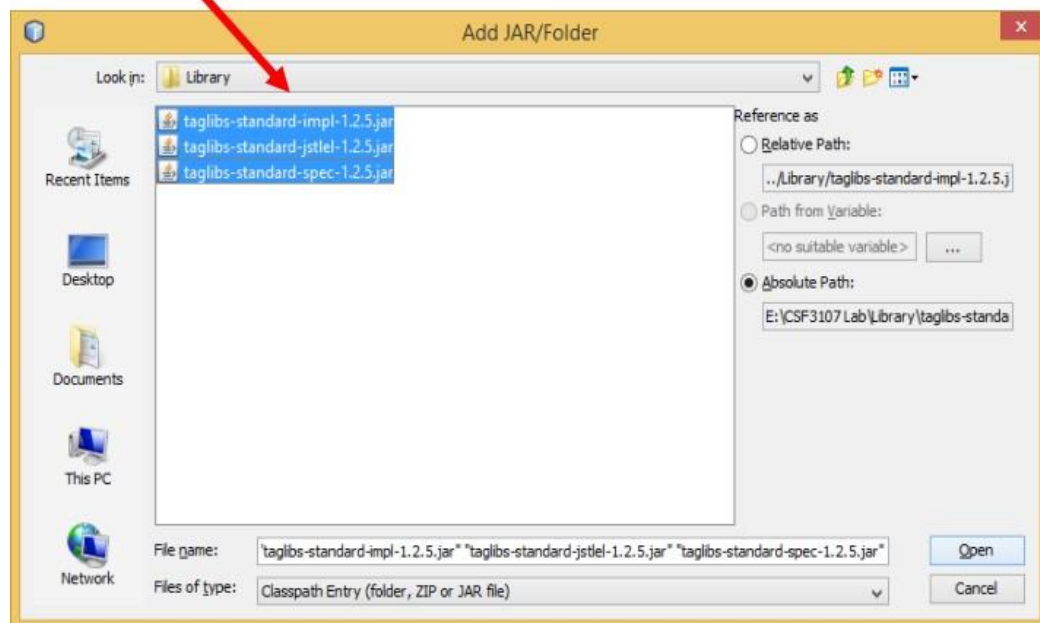


6. Go to your project -> Libraries.

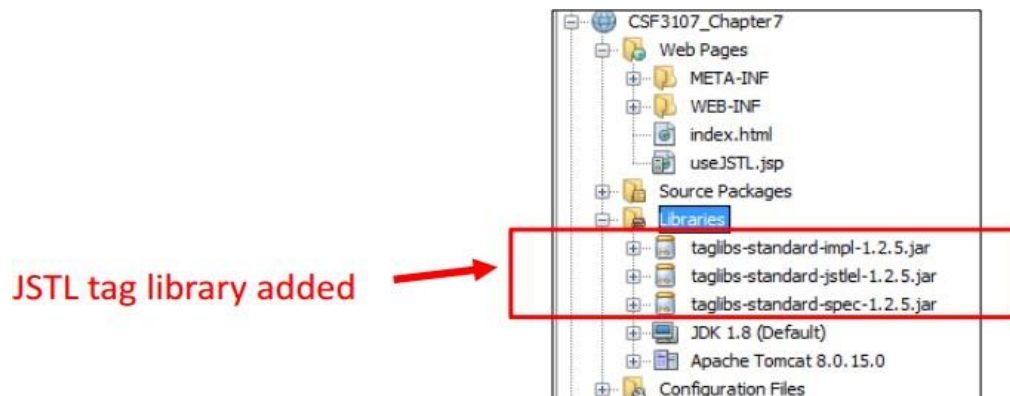
7. Right click your mouse and click to Add JAR/Folder.



8. Choose taglibs-standard-impl-1.2.5.jar, taglibs-standard-jstlel-1.2.5.jar and taglibs-standard-spec-1.2.5.jar and click Open button.



9. You will see all jars files; taglibs-standard-impl-1.2.5.jar, taglibs-standardjstlel-1.2.5.jar and taglibs-standard-spec-1.2.5.jar successfully added in Libraries's folder in NetBeans.



Reflection

What you have learnt from this exercise?

I learn how to add dependencies for the three taglibs standard whis is impl, jstlel and spec jar.

Task 4: Using Java Standard Tag Library (JSTL)

Objective: Using JSTL core tags directive to retrieve information from one page and display it in another JSP page and format number, currency.

Problem Description:

1. To set the value “Welcome to CSF3107 - Web Programming courses..!” and display the value using JSTL core.
2. To passing information from one page to another using JSTL core
 - i. Create userRegistration.html to key-in information.
 - ii. Create userHandler.jsp to receive information key-in from userRegistration.html.
3. Using JSTL’s fmt to format number and currency.

Estimated time: 30 minutes

Prerequisite: Must attached JSTL’s library to current Project’s directory (Refer to Chapter 7’s notes).

Task 1 Assign value “Welcome to CS F3107-Web Programming 2 courses” and display the value inside JSP page

1. Go to project *Lab5*.
2. Create a new JSP’s file as *jstlCore1*.

3. Rename the title as *Using JSTL tag library* and `<h1>` as *Use JSTL's features*.

```
<html>
  <head>
    <meta http-equiv="Content-Type" content="text/html; charset=UTF-8" />
    <title>Using JSTL tag library</title>
  </head>
  <body>
    <h1>Use JSTL's features</h1>
  </body>
</html>
```

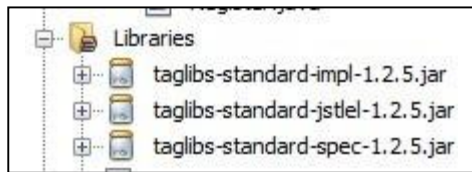
4. Before using JSTL's tag, we need to declare JSTL's library tag using Taglib directive.

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

5. Using JSTL core tag, assign the value *"Welcome to CSF3107 - Web Programming courses..!"* to specific variable and finally, display the message.

```
<body>
  <h1>Use JSTL's features</h1>
  <c:set var="message" value="Welcome to CSF3107 - Web Programming courses..!" />
  <p> <c:out value="${message}" /> </p>
</body>
```


- Before you compile *jstlCore1.jsp*, please ensure you already attach jar file for JSTL tag library.

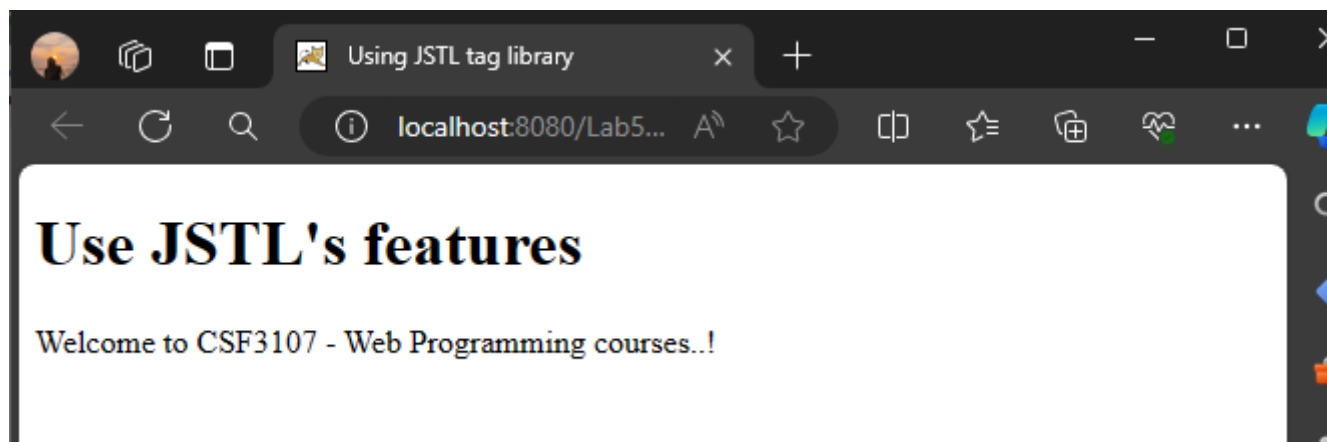


- Compile and run *jstlCore1.jsp* page.

- You will get the following output.



The Output:



Task 2-Using JSTL's core to request information from one page and display the information

Step 1

- Go to Lab8's project.
- Create HTML's file and rename as userRegistration.
- Produce the following HTML's form.

← → ↻ localhost:8084/Lab5/userRegistration.html

Apps Zimbra Sign In Lentera OneDrive MyNemo ezHASiL

User Details

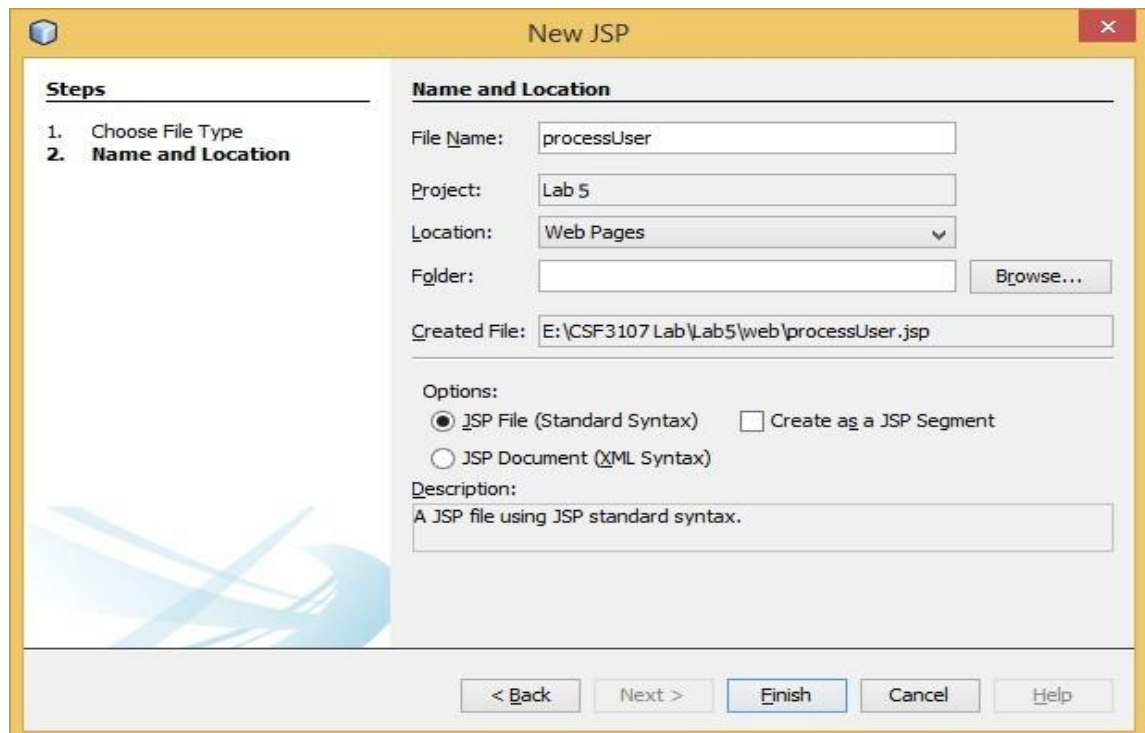
Name	<input type="text" value="Enter firstname"/>
Surname	<input type="text" value="Enter firstname"/>
Password	<input type="text" value="Max 10 charaters"/>
Gender	Male <input checked="" type="radio"/> Female <input type="radio"/>
Type of User	<input type="text" value="Beginner"/>
Prefer Language	<input checked="" type="checkbox"/> Malay <input type="checkbox"/> English <input type="checkbox"/> Mandarin <input type="checkbox"/> Tamil

©2016-Mohamad Nor

Note: Types of user is *Beginner*, *Intermediate* and *Advanced*

Step 2

1. Create JSP's file and rename as *processUser*.



2. Add JSTL taglib directive into the *processUser.jsp*'s page.
3. Retrieve the information by using *c:param* tag and display the information by using *c:out* tag.
4. Compile *userHandler.jsp* page.
5. Run *userRegistration.html* page and key-in the information.

User Details

Name: Mohamad Nor

Surname: Hassan

Password:

Gender: Male ☒ Female ☐

Type of User: Advanced

Prefer Language: ☒ Malay ☐ English ☐ Mandarin ☐ Tamil

Submit Cancel

©2016-Mohamad Nor

6. Click *Submit* button.

7. You will get the following output.

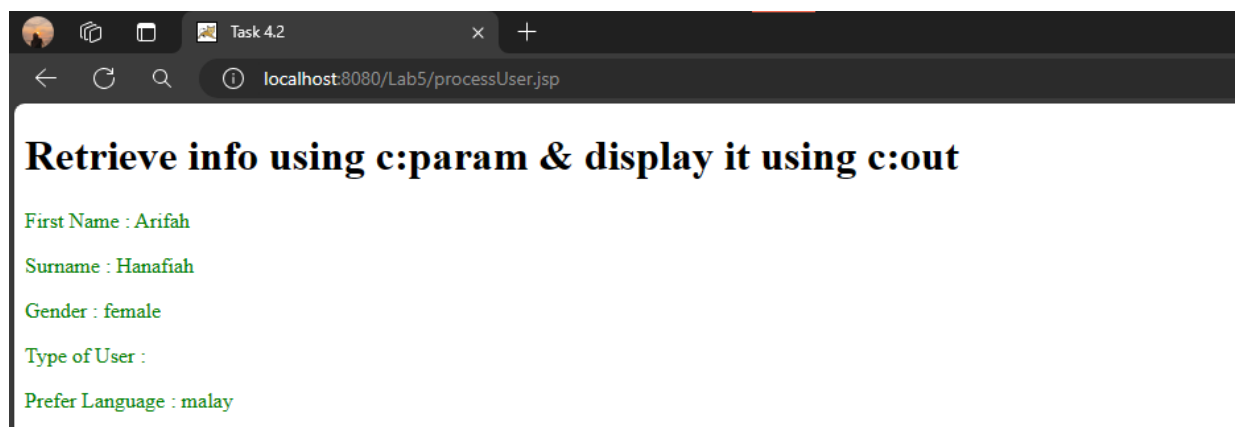


The Output :

userRegistration.jsp:

A screenshot of a web browser window showing a form titled 'User Registration'. The form is titled 'User Details' and contains the following fields: 'Name' (Arifah), 'Surname' (Hanafiah), 'Password' (010425), 'Gender' (radio buttons for Male and Female, with Female selected), 'Type of User' (a dropdown menu showing 'Beginner'), and 'Prefer Language' (checkboxes for Malay, English, Mandarin, and Tamil, with Malay selected). There are 'Submit' and 'Cancel' buttons at the bottom.

processUser.jsp:



Task 3 - Using JSTL's fmt to format number and currency

Step 1

1. Create JSP's file and rename as *jstlFormat1*.

New JSP

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:

< Back Next > **Finish** Cancel Help

2. Add JSTL taglib directive for JSTL *core* and JSTL *formatting* into the current code.
3. Add the following code in your page.

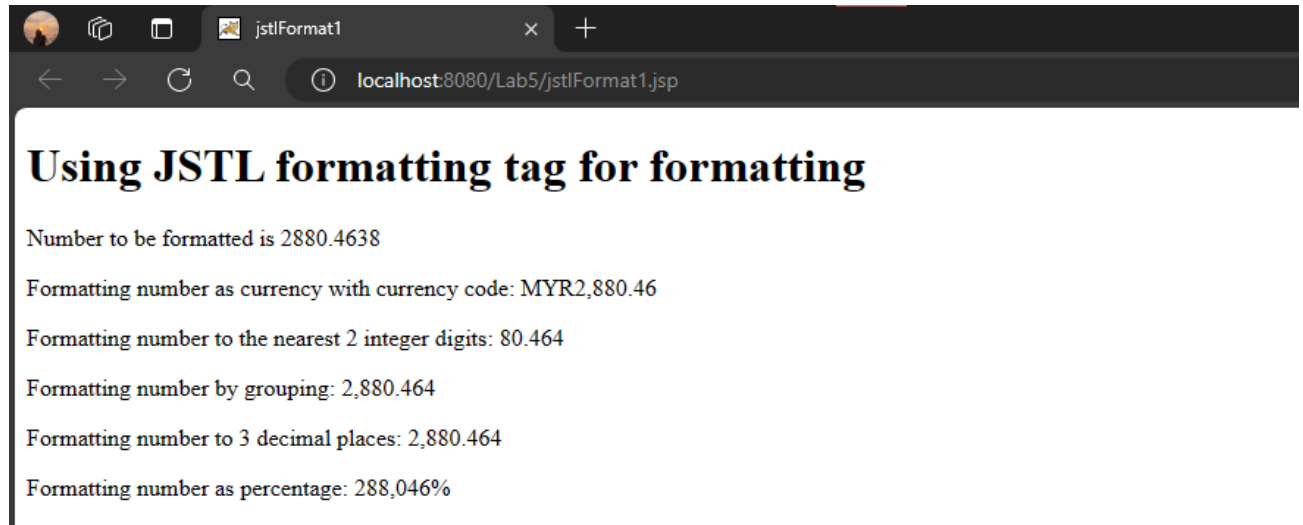
```
<body>
  <h1>Using JSTL formattig tag for formatting</h1>

  <!-- Assign specific number to variable -->
  <c:set var="total" value="2880.4638"/>
  <p>Number to be formatted is <c:out value="${total}" /></p>
  <p>Formatting number as currency with currency code : <fmt:formatNumber type="currency" currencyCode="MYR" value="${total}" /></p>
  <p>Formatting number to the nearest 2 integer digit : <fmt:formatNumber type="number" maxIntegerDigits="2" value="${total}" /></p>
  <p>Formatting number by grouping : <fmt:formatNumber type="number" groupingUsed="true" value="${total}" /></p>
</body>
```

4. Continue to do formatting for
 - 3 decimal places
 - percentage
5. Save *jstlFormat1.jsp*
6. Compile and run *jstlFormat1.jsp*.
7. You will get the following output.



The Output:



Reflection

1. What the purpose of using JSTL's tag library?

JavaServer Pages Tag Library (JSTL) is a set of tags that can be used for implementing some common operations such as looping, conditional formatting, and others.

2. List FIVE(5) categories of JSTL library.

1. Core tag library
2. Xml tag library
3. Internationalization tag library
4. SQL tag library
5. Functions tag library

Task 5: Using JSP Standard Tag Library

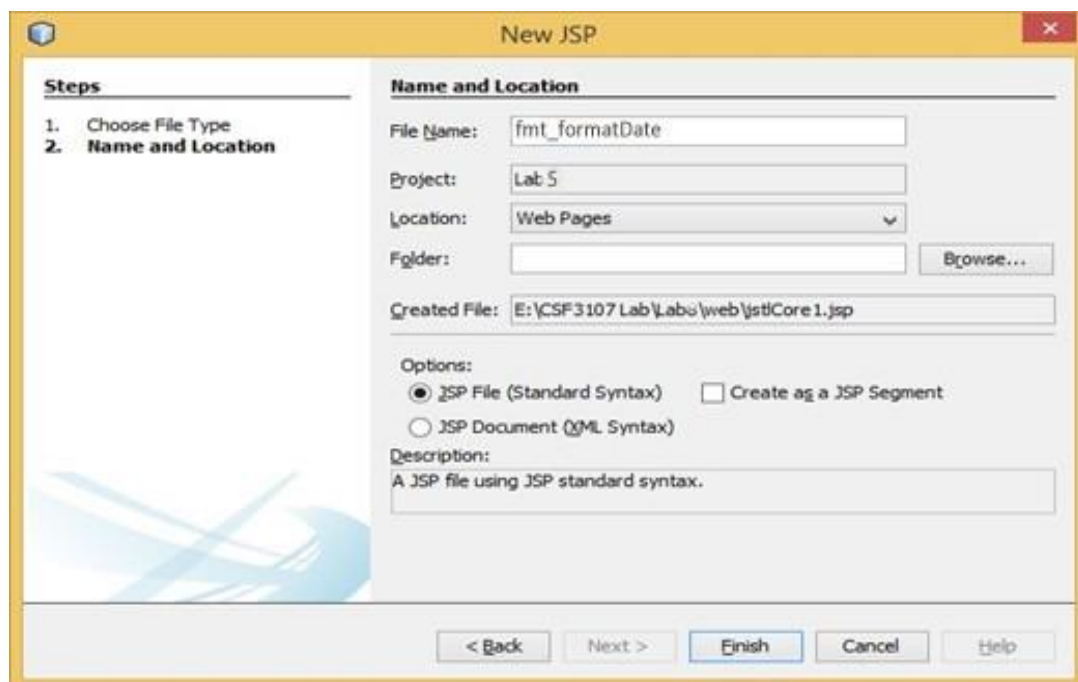
Objective: Using JSTL tags for parsing and formatting the dates.

Problem Description: 1. Using JSTL's `fmt` to format Dates.
2. Using JSTL's `fmt` to parse Dates.

Estimated time: 30 minutes

Task 1 Using JSTL's `fmt` to format Date

1. Go to project *Lab5*.
2. Create a new JSP file as *fmt_formatDate*.



2. Rename the title as *Using JSTL tag library* and `<h2>` as *Use fmt_formatDate features*.

```
<title>fmt:parseDate feature</title>
</head>
<body>
  <h2>fmt:parseDate feature</h2>
  <!-- a Date time string -->
```

4. Before using JSTL's tag, we need to declare JSTL's library tag using Taglib directive.

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

5. Add JSTL Taglib directive for JSTL core and JSTL formatting into the current code.
6. Add the following code in your page.

```
<c:set var="now" value="<%=new java.util.Date()%>" />
<p>
  Time (fmt:formatDate type="time"):
  <strong>
    <fmt:formatDate type="time" value="{now}" />
  </strong>
</p>
<p>
  Date (fmt:formatDate type="date"):
  <strong>
    <fmt:formatDate type="date" value="{now}" />
  </strong>
</p>
<p>
  Date, Time (fmt:formatDate type="both"):
  <strong>
    <fmt:formatDate type="both" value="{now}" />
  </strong>
</p>
<p>
  Date, Time Short (fmt:formatDate type="both" dateStyle="short"):
  <strong>
    <fmt:formatDate type="both" dateStyle="short" timeStyle="short" value="{now}" />
  </strong>
</p>
```

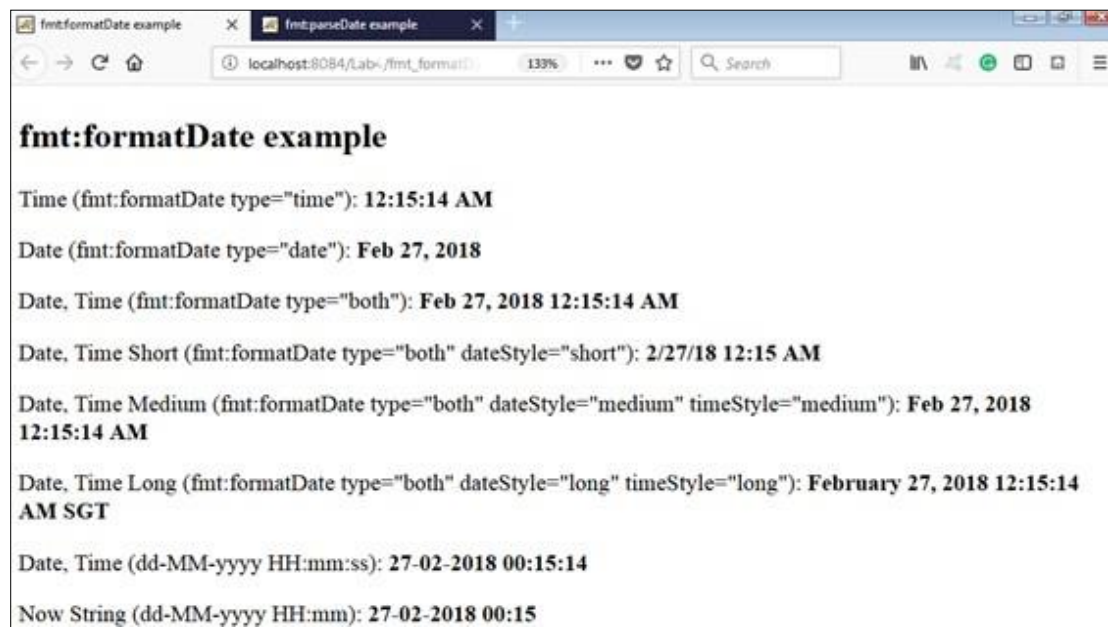
```

<p>
  Date, Time Medium (fmt:formatDate type="both" dateStyle="medium" timeStyle="medium"):
  <strong>
    <fmt:formatDate type="both" dateStyle="medium" timeStyle="medium" value="{now}" />
  </strong>
</p>
<p>
  Date, Time Long (fmt:formatDate type="both" dateStyle="long" timeStyle="long"):
  <strong>
    <fmt:formatDate type="both" dateStyle="long" timeStyle="long" value="{now}" />
  </strong>
</p>
<p>
  Date, Time (dd-MM-yyyy HH:mm:ss):
  <strong>
    <fmt:formatDate pattern="dd-MM-yyyy HH:mm:ss" value="{now}" />
  </strong>
</p>
  <!-- Store in variable -->
<fmt:formatDate pattern="dd-MM-yyyy HH:mm" value="{now}" var="nowString"/>
<p>
  Now String (dd-MM-yyyy HH:mm):
  <strong>
    <c:out value="{nowString}" />
  </strong>
</p>

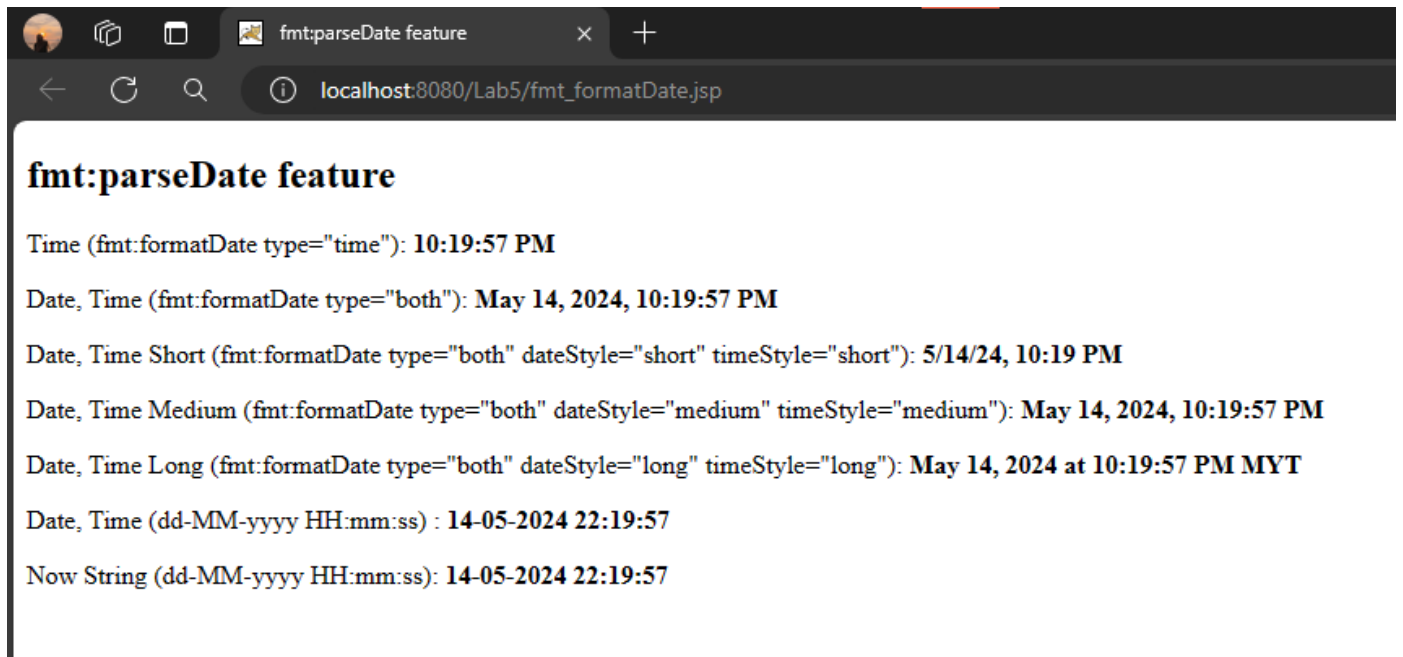
```

7. Compile and run *fmt_formatDate.jsp* page.

8. You will get the following output.



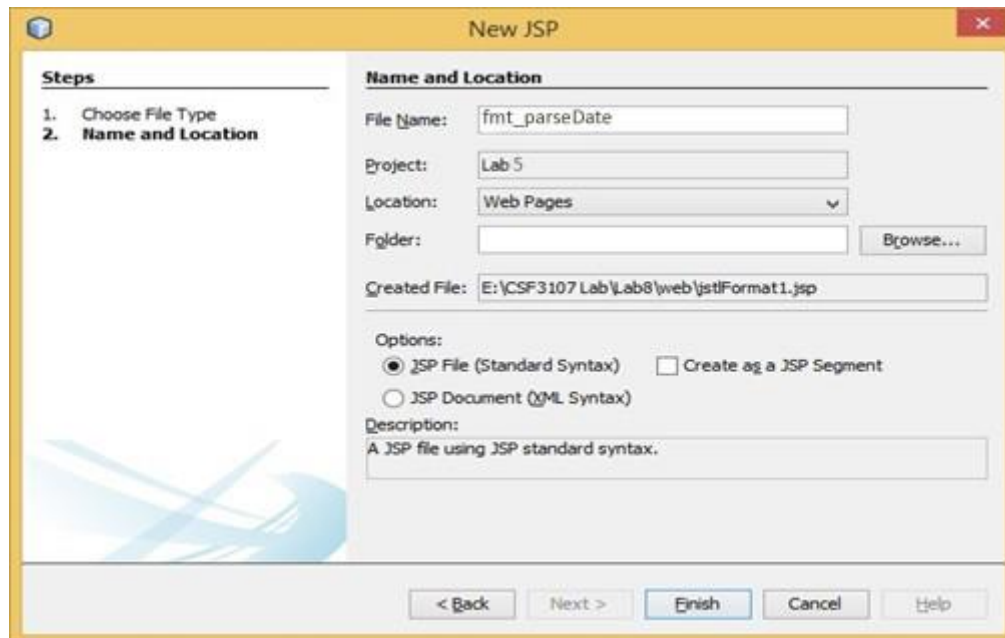
The Output :



Task 2 - Using JSTL's fmt to Parse Date

Step 1

1. Create JSP's file and rename as *fmt_parseDate*.



2. Add JSTL taglib directive for JSTL *core* and JSTL *formatting* into the current code.
3. Add the following code in your page.

```
<body>
  <c:set var="dateTimeString" value="17-11-2015 11:49" />
  <h4>
    dateTimeString:
    <c:out value="${dateTimeString}" />
  </h4>

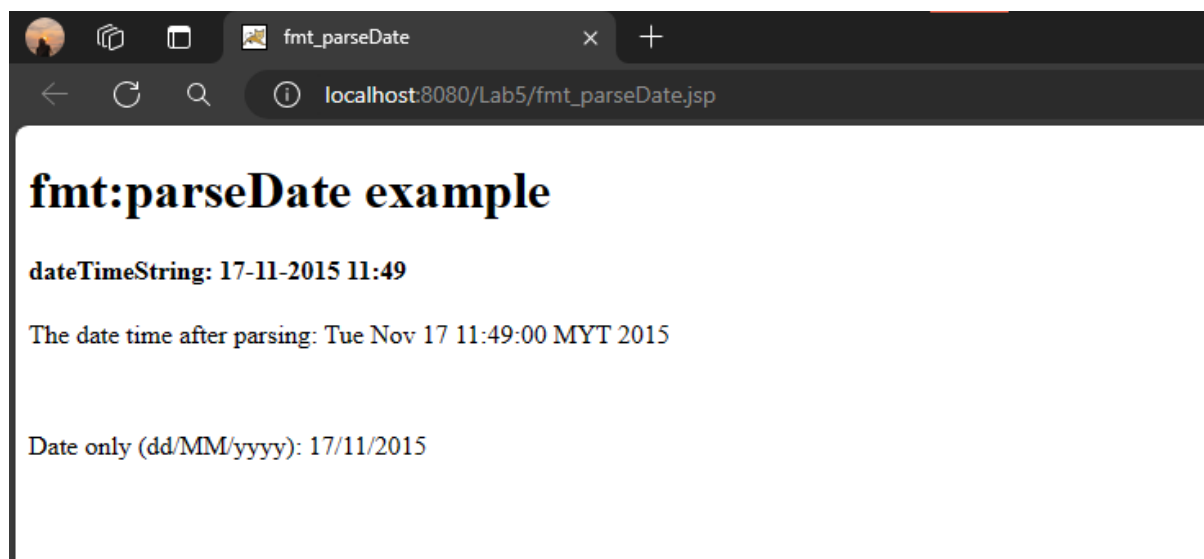
  <!-- Parsing a date time string, and store in a variable type of java
  <fmt:parseDate value="${dateTimeString}"
    type="both" var="parsedDatetime" pattern="dd-MM-yyyy HH:mm" />
  <p>
    The date time after parsing:
    <c:out value="${parsedDatetime}" />
  </p>
  <br/>
  <p>
    Date only (dd/MM/yyyy):
    <fmt:formatDate value="${parsedDatetime}" pattern="dd/MM/yyyy" />
  </p>
</body>
```

5. Save *fmt_parseDate.jsp*
6. Compile and run *fmt_parseDate.jsp*.

7. You will get the following output.



The Output :



Reflection

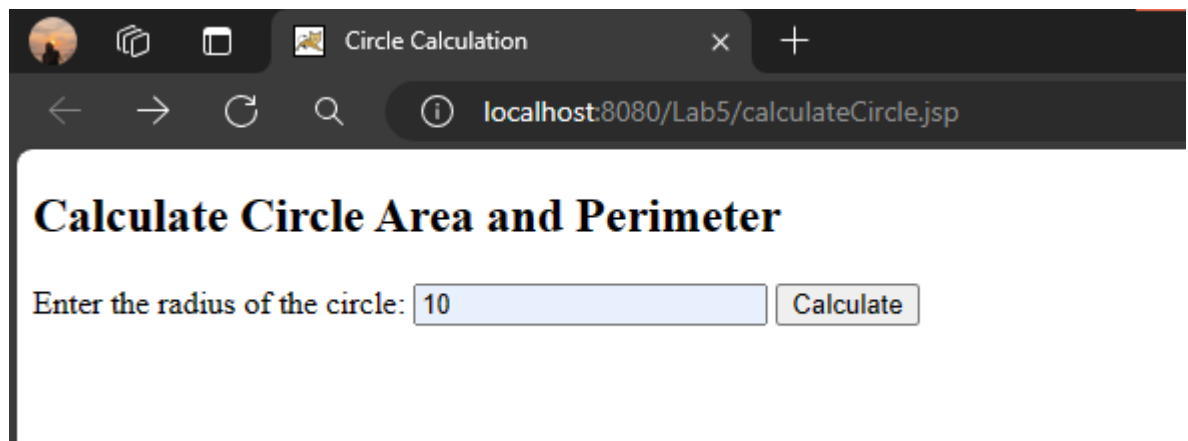
1. What you have learnt from this exercise?

From this exercise I learnt how to use JSTL tags for parsing the formatting dates.

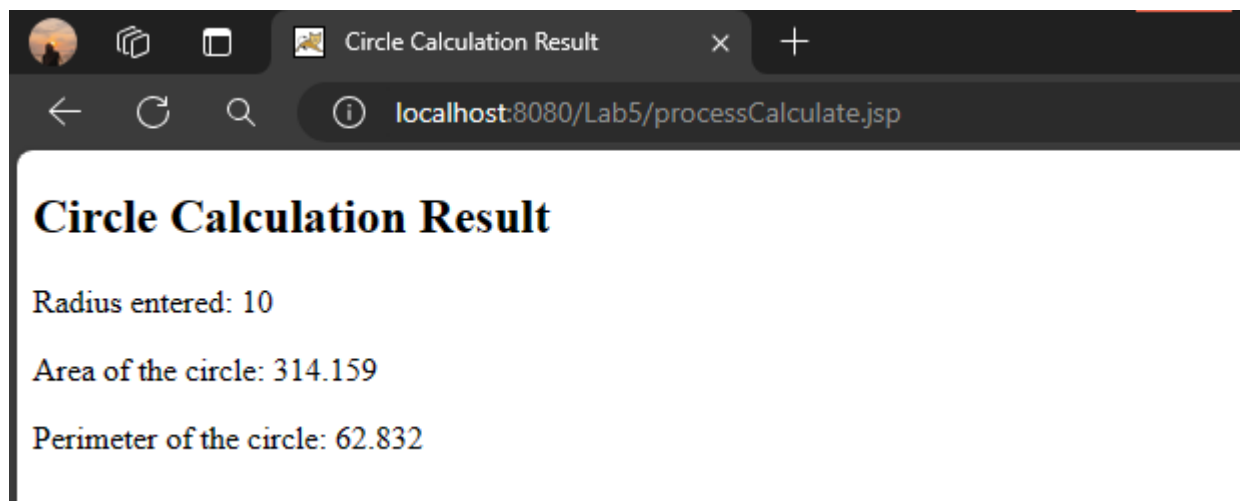
Exercise

1. Write a JSP's form that asks the user to key-in the radius of circle. The program should calculate the area of circle and the perimeter of circle. Finally, use JSTL library to format your result into 3 decimal places.

The Output:



A screenshot of a web browser window. The title bar shows 'Circle Calculation'. The address bar shows 'localhost:8080/Lab5/calculateCircle.jsp'. The page content has a heading 'Calculate Circle Area and Perimeter'. Below the heading is a form with the text 'Enter the radius of the circle:' followed by a text input field containing the value '10' and a 'Calculate' button.



A screenshot of a web browser window. The title bar shows 'Circle Calculation Result'. The address bar shows 'localhost:8080/Lab5/processCalculate.jsp'. The page content has a heading 'Circle Calculation Result'. Below the heading, the results are displayed: 'Radius entered: 10', 'Area of the circle: 314.159', and 'Perimeter of the circle: 62.832'.

2. Rahim bought 800 shares of stock at a price of RM10.50 per share. He must pay her stock broker a 5 percent commission for the transaction. Write a web based program that calculates and displays the following:
- The amount paid for the stock alone without the commission.
 - The amount of the commission.
 - The total amount paid (for the stock plus the commission).

You should use JavaBeans to implement business logic and JSTL for display purposes.

The Output :

Stock Transaction Calculator

Enter the number of shares:

Enter the price per share (RM):

Enter the commission rate (%):

Transaction Summary:

Amount paid for stock alone:	RM 46000.000
Commission amount:	RM 2300.000
Total amount paid:	RM 48300.000