CS6308- Java Programming

V P Jayachitra
Assistant Professor
Department of Computer Technology
MIT Campus
Anna University

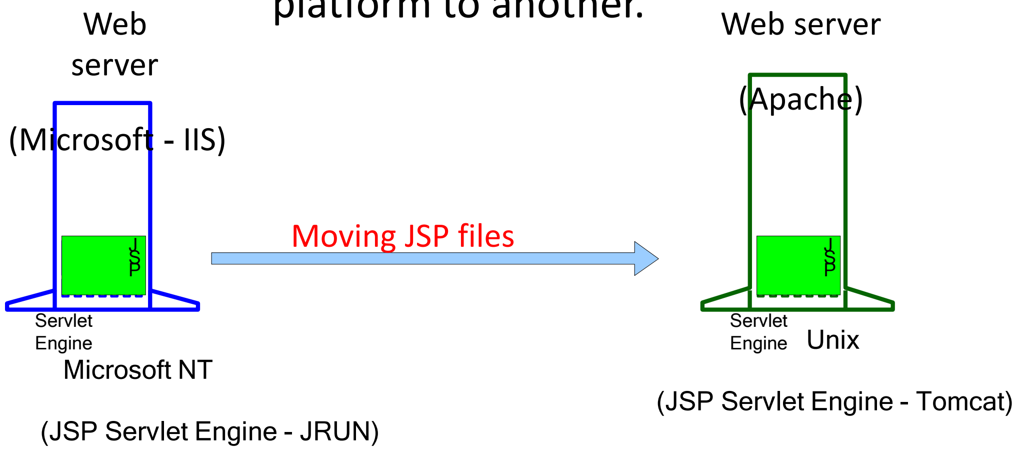
JavaServer Pages (JSP)

- JavaServer Pages (JSP) is a server-side programming technology based on the Java language and enables the development of dynamic web sites.
- JSP was developed by Sun Microsystems to allow server side development.
- JSP files are HTML files with special Tags containing Java source code that provide the dynamic content.
- JSP is a server-side technology that allows developers to embed Java code directly into HTML pages.

Why JSP?

- JSP is based on Java, an object- oriented language.
- JSP offers a robust platform for web development.
- Multi platform
- Component reuse by using Javabeans and EJB.

Moving JSP file from one platform to another.



JSP compared to **ASP**

- JSP and ASP are fairly similar in the functionality that they provide.
- JSP may have slightly higher learning curve.
- Both allow embedded code in an HTML page, session variables and database access and manipulation.
- ASP is mostly found on Microsoft platforms i.e. NT
- JSP can operate on any platform that conforms to the J2EE specification.

JSP stands for JavaServer Pages. It is a server-side technology developed by Sun Microsystems (now Oracle) as part of the Java platform. JSP allows developers to create dynamic web content by embedding Java code in HTML pages

ASP stands for Active Server Pages, a server-side scripting technology developed by Microsoft. It is part of Microsoft's web development stack and is used to create dynamic and interactive web applications.

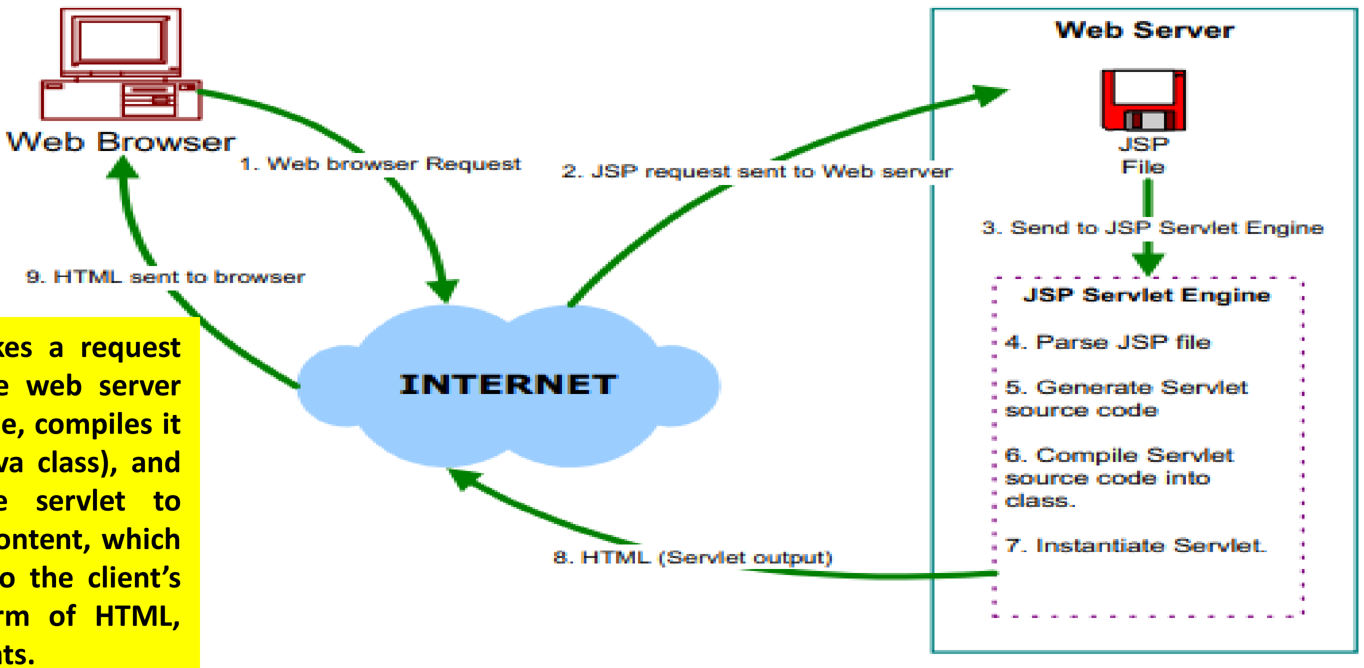
JSP compared to Servlets

- A Servlet is a Java class that provides special server side service.
- It is hard work to write HTML code in Servlets.
- In Servlets lots of println statements are required to generate HTML.

PrintWtiter out=req.getWriter();

JSP architecture

- JSPs are essential an HTML page with special JSP tags embedded.
  - These JSP tags can contain Java code.
- The JSP file extension is .jsp rather than .htm or .html.
- The JSP engine parses the .jsp and creates a Java servlet source file.
- It then compiles the source file into a class file, this is done the first time and this why the JSP is probably slower the first time it is accessed.
- Any time after this the special compiled servlet is executed and is therefore returns faster.



When a client makes a request for a JSP page, the web server processes the JSP file, compiles it into a servlet (a Java class), and then executes the servlet to generate dynamic content, which is then sent back to the client's browser in the form of HTML, XML, or other formats.

Creating first JSP page

- **JSP Comments <%-- JSP comment --%>**

```
<html>
<head>
<title>My first JSP page
</title>
</head>
<body>
<%-- This JSP comment - not visible in the page source --%>
<%  out.println(“Hello World”);  %>
</body>
</html>
```

JSP tags

- There are five main tags:
  - Declaration tag
  - Expression tag
  - Directive Tag
  - Scriptlet tag
  - Action tag

1.Declaration Tag (<%! %>)
Purpose: Used to declare variables or methods that will be available throughout the JSP page.

```
<%! data_type variable_name; %>
<%! int count = 0; %>
```

Here, count is a class-level variable and can be used anywhere in the JSP.

2. Expression Tag (<%= %>)
Purpose: Used to output the value of a Java expression directly into the JSP page.

```
<%= expression %>
<%= new java.util.Date() %>
```

This will display the current date and time in the response.

3. Directive Tag (<%@ %>)
Purpose: Provides global-level settings or configuration for the JSP page.

Types of Directives:

page: Defines page-level attributes (e.g., content type, error handling).
include: Includes another file during page translation.
taglib: Declares a custom tag library for the page.

```
<%@ directive_name attribute="value" %>

<%@ page contentType="text/html" %>
<%@ include file="header.jsp" %>
<%@ taglib uri="http://java.sun.com/jsp/jstl/core" prefix="c"
%>
```

4. Scriptlet Tag (<% %>)
Purpose: Allows embedding of Java code inside the JSP page.

```
<% java_code %>
<%
  int num = 5;
  out.println("The number is: " + num);
%>
```

Here, Java code initializes num and outputs it to the client.

5. Action Tag (<jsp:action_name ... />)
Purpose: Used to perform specific tasks like forwarding requests, including resources, or working with JavaBeans.

```
<jsp:action_name attribute="value" />
```

Examples:
jsp:forward: Forwards the request to another resource.

```
<jsp:forward page="another.jsp" />
```

jsp:include: Includes a resource dynamically at runtime.

```
<jsp:include page="footer.jsp" />
```

jsp:useBean: Works with JavaBeans.

```
<jsp:useBean id="bean" class="com.example.BeanClass" scope="session" />
```

Declaration tag (<%! %>)

- This tag allows the developer to declare variables or methods.
- Before the declaration specify <%!
- At the end of the declaration, specify %>
- Code placed in this tag must end in a semicolon   ;

For Example,

```
<%!
   private int counter = 0 ;
%>
```

Expression tag (<%= %>)

- This tag allows the developer to embed any Java expression and is short for out.println().
- A semicolon (;) does not appear at the end of the code inside the tag.
- For example, to show the current date and time.

```
<%-- JSP comment --%>
<HTML>
<HEAD>
<TITLE>MESSAGE</TITLE>
</HEAD>
<BODY>
<% out.print("Hello, Sample JSP code"); %>
Date : <%= 5+6 %>    //prints 11
</BODY>
</HTML>
```

Directive tag (<%@ directive … %>)

- A JSP directive gives special information about the page to the JSP Engine.
- There are three main types of directives:
  1. page – processing information for this page.
  2. Include – files to be included.
  3. Tag library – tag library to be used in this page.
- Directives do not produce any visible output when the page is requested but change the way the JSP Engine processes the page.

for import we use this ig..

Page directive

//attributes!!

language	Which language the file uses.	<%@ page language = "java" %>
extends	Superclass used by the JSP engine for the translated Servlet. extends specifies the super class of the corresponding servlet code. By default, JSP pages extend HttpJspBase.	<%@ page extends = "com.example.CustomServlet" %>
import	Import all the classes in a java package into the current JSP page. This allows the JSP page to use other java classes. The following packages areimplicitly imported. java.lang.* javax.servlet.* javax.servlet.jsp.* javax.servlet.http.*	<%@ page import = "java.util.*" %> <%@ page import="java.util.ArrayList, java.util.Date" %>

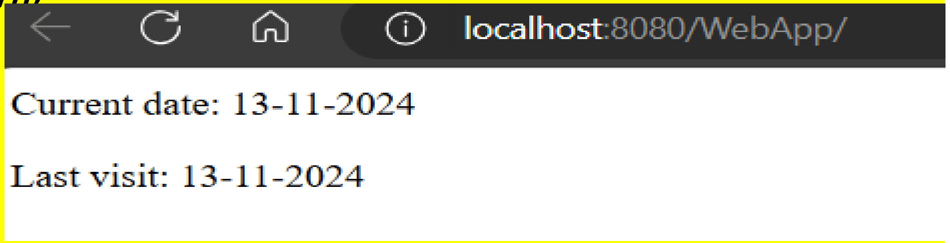
session	Does the page make use of sessions. By default all JSP pageshave session data available. There are performance benefitsto switching session to false.	Default is set to true.
		// Enable sessions (default)
		<%@ page session="true" %>
		
		// Disable sessions for performance
		<%@ page session="false" %>

buffer	1. Controls the use of buffered output for a JSP page. 2. buffer sets the size of the buffer used by our JSP page. The default value is 8kb.	<%@ page buffer = "none" %> <%@ page buffer="19kb" %>
autoFlush	Flush output buffer when full. autoFlush controls the buffer output, clearing it out when the buffer size is reached. The default value is true.	<%@ page autoFlush = "true" %>

Page directive

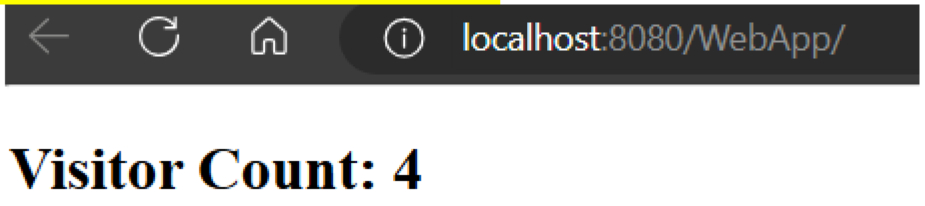
isThreadSafe	*isThreadSafe* has a default value of *true*. *isThreadSafe* determines whether or not the JSP can use Servlet multi-threading.	**<%@ page isThreadSafe=”false” %>**
info	info is used to set a text-based description for the JSP.	**<%@ page info=”This is my JSP!” %>**
errorPage	•Different page to deal with •errorPage specifies a JSP page as an error page.	**<%@ page errorPage =“/error/error.jsp” %>**

```
<%@ page language="java"
      import="java.util.* , java.text.SimpleDateFormat"   session="true“   buffer="16kb“  autoFlush="true“ %>
<!DOCTYPE html>
<html>
<head> <title>JSP Directives Demo</title></head>
<body>
  <%

     Date today = new Date();
     SimpleDateFormat formatter = new SimpleDateFormat("dd-MM-yyyy");


     if (session != null) {
        session.setAttribute("lastVisit", formatter.format(today));
     }
     // Using buffer
     out.println("Current date: " + formatter.format(today));

     //  buffer flush
     out.flush();
  %>

  <% if (session != null) { %>
     <p>Last visit: <%= session.getAttribute("lastVisit") %></p>
  <% } %>
</body>
</html>
```



```
Current date: 13-11-2024

Last visit: 13-11-2024
```

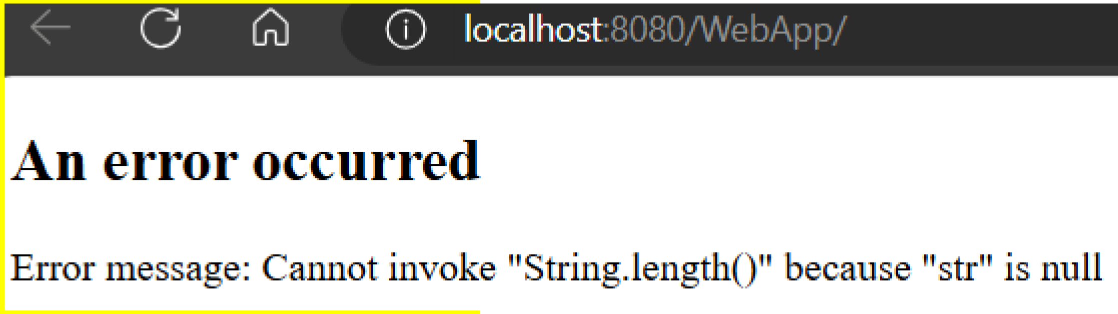
```
<%@ page language="java" isThreadSafe="true" %>
<!DOCTYPE html>
<html>
<body>
  <%!
    // Thread-safe counter using synchronized
    private int counter = 0;
    synchronized int getCount() {
      return ++counter;
    }
  %>

  <h2>Visitor Count: <%= getCount() %></h2>
</body>
</html>
```



Visitor Count: 4

```
<%@ page language="java"
        info="Login page that handles user authentication" %>
<!DOCTYPE html>
<html>
<body>
    <form action="login" method="post">
        Username: <input type="text" name="username"><br>
        Password: <input type="password" name="password"><br>
        <input type="submit" value="Login">
    </form>
</body>
</html>
```

```
<%@ page language="java"
      errorPage="error.jsp"
      isThreadSafe="true"
      info="Main application page" %>
<!DOCTYPE html>
<html>
<body>
  <%
     String str = null;
     str.length(); // NullPointerException
  %>
</body>
</html>
```


localhost:8080/WebApp/

An error occurred

Error message: Cannot invoke "String.length()" because "str" is null

isErrorPage="true“ makes the exception object available

```
<%@ page isErrorPage="true"
      language="java"
      info="Error handling page" %>
<!DOCTYPE html>
<html>
<body>
  <h2>An error occurred</h2>
  <p>Error message: <%= exception.getMessage() %></p>
</body>
</html>
```

Include directive

- Allows a JSP developer to include contents of a file inside another.
- Typically include files are used for navigation, tables, headers and footers that are common to multiple pages.

This includes the html from privacy.html found in the include directory into the current jsp page.

```
<%@ include file = “include/privacy.html %>
```

or to include a naviagation menu (jsp file) found in the current directory.

```
<%@ include file = “navigation.jsp %>
```

Scriptlet tag (<% … %>)

- Between <% and %> tags, any valid Java code is called a Scriptlet. This code can access any variable declared.
- For example, to print a variable.

```
<%
   String username = “JSP";
   System.out.println (username);
%>
```

Tag Lib directive

- A tag lib is a collection of custom tags that can be used by the page.
-
```
<%@ taglib uri = “tag library URI” prefix = “tag Prefix” %>
```

Action tag

- There are three main roles of action tags :
  - enable the use of server side Javabeans
  - transfer control between pages
  - browser independent support for applets.
- **JSP Custom Tags**
  - Custom tag in JSP is know as user defined tag. In JSP a user can create its own tag for performing the specific task.

Implicit Objects

1. request
Type: javax.servlet.http.HttpServletRequest
Purpose: Represents the HTTP request sent by the client to the server.
Key Methods:
getParameter(String name): Retrieve request parameters (e.g., form data).
getAttribute(String name): Get request-level attributes.
getSession(): Retrieve the current session.

2.response
Type: javax.servlet.http.HttpServletResponse
Purpose: Represents the HTTP response sent from the server to the client.
Key Methods:
sendRedirect(String location): Redirect the client to another resource.
setContentType(String type): Set the content type of the response.

Variable	Of type
request	javax.servlet.http.httpservletrequest
response	javax.servlet.http. httpservletresponse
out	javax.servlet.jsp.jspwriter
session	javax.servlet.http.httpsession
pagecontent	javax.servlet.jsp.pagecontext
application	javax.servlet.http.servletcontext
config	javax.servlet.http.servletconfig
page	java.lang.object
exception	java.lang.throwable

3.response
Type: javax.servlet.http.HttpServletResponse
Purpose: Represents the HTTP response sent from the server to the client.
Key Methods:
sendRedirect(String location): Redirect the client to another resource.
setContentType(String type): Set the content type of the response.

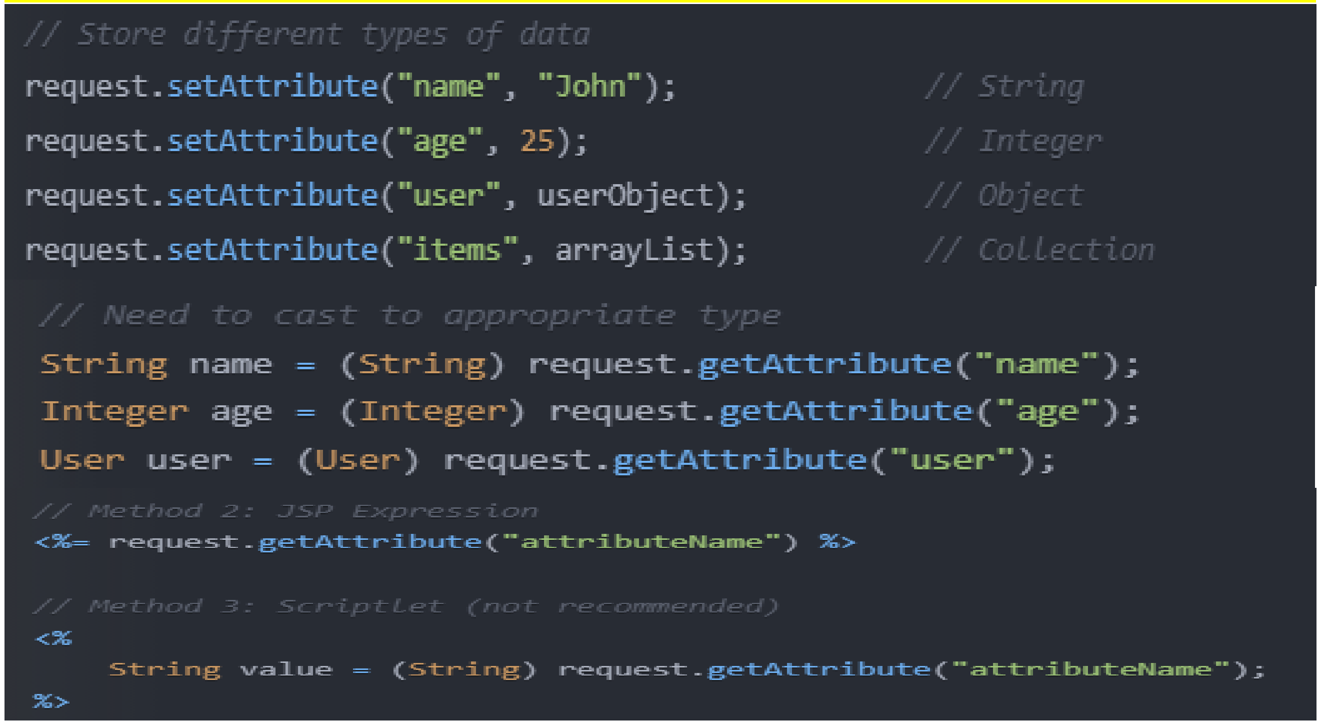
4. session
Purpose: Represents the user's session. It stores information for the duration of a session across multiple requests.
Key Methods:
setAttribute(String name, Object value): Store an attribute in the session.
getAttribute(String name): Retrieve an attribute from the session.

Implicit Objects

<u>request object</u>

Access to information associated with a request. This object is normally used in looking up parameter values and cookies.

<% String NAME = request.getParameter(“UNAME”); %>
 Welcome : <%out.println(NAME) %>
Return Type: Always returns String



```
// Store different types of data
request.setAttribute("name", "John");           // String
request.setAttribute("age", 25);                // Integer
request.setAttribute("user", userObject);       // Object
request.setAttribute("items", arrayList);       // Collection

// Need to cast to appropriate type
String name = (String) request.getAttribute("name");
Integer age = (Integer) request.getAttribute("age");
User user = (User) request.getAttribute("user");

// Method 2: JSP Expression
<%= request.getAttribute("attributeName") %>

// Method 3: Scriptlet (not recommended)
<%
    String value = (String) request.getAttribute("attributeName");
%>
```

Implicit Objects

Session object

A session object uses a key/value combination to store information.
To retrieve information from a session,

```
session.putValue(“visitcounter”, totalvisits)

session.getValue(“visitcounter”)
```

Servlet code

CODE-1(Servlet)

Calculator!!

```
import java.io.IOException;
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("/CalculatorServlet")   url-pattern
public class CalculatorServlet extends HttpServlet {
    protected void doPost(HttpServletRequest request, HttpServletResponse
response)
            throws ServletException, IOException {
        int num1 = Integer.parseInt(request.getParameter("num1"));
        int num2 = Integer.parseInt(request.getParameter("num2"));
        String operator = request.getParameter("operator");

        int result = 0;
        switch (operator) {
            case "add":
                result = num1 + num2;
                break;
            case "subtract":
                result = num1 - num2;
                break;
            case "multiply":
                result = num1 * num2;
                break;
            case "divide":
                result = num1 / num2;
                break;
        }
         request.setAttribute("result", result);
        request.getRequestDispatcher("calculator.jsp").forward(request, response);
    }
}
```

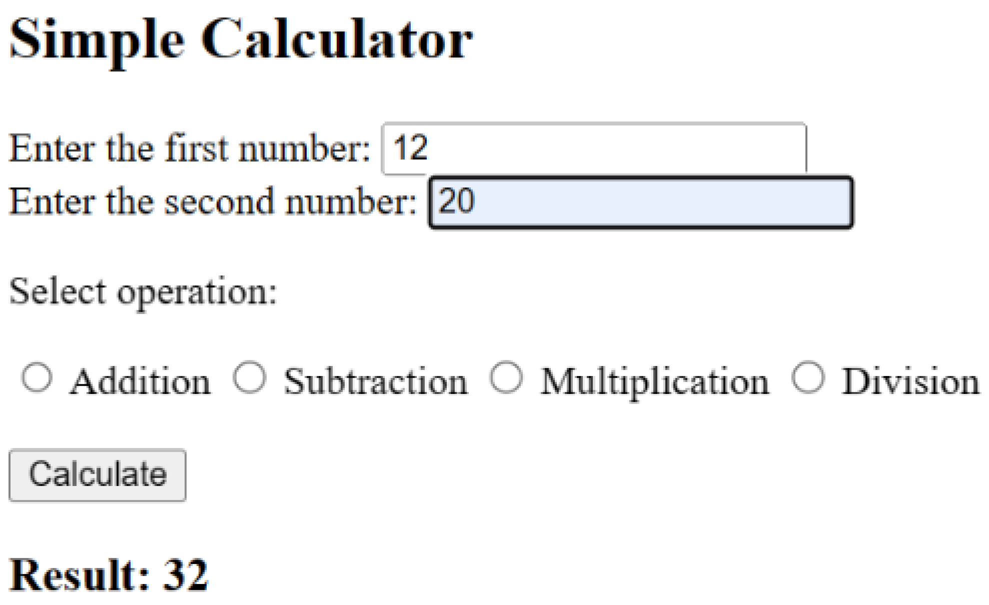
JSP code

CODE-2(JSP)

```
<%@ page language="java" contentType="text/html; charset=UTF-8" pageEncoding="UTF-8"%>
<!DOCTYPE html>
<html>
<body>
   <h2>Simple Calculator</h2>
   <form action="CalculatorServlet" method="post">
           Enter the first number:      <input type="text" name="num1"><br>
           Enter the second number: <input type="text" name="num2"><br>

      <p>Select operator:</p>
      <input type="radio" name="operator" value="add"> Addition
      <input type="radio" name="operator" value="subtract"> Subtraction
      <input type="radio" name="operator" value="multiply"> Multiplication
      <input type="radio" name="operator" value="divide"> Division

      <br><br>

      <input type="submit" value="Calculate">

   </form>

   <%
      // Display the result if it exists

      Integer result = (Integer) request.getAttribute("result");
      if (result != null) {
   %>
      <h3>Result: <%= result %></h3>
      <!-- <p>Result: ${result} <p> -- >
   <%
      }
   %>
</body>
</html>
```

//Deployment descriptor!

```
<!-- Servlet Configuration -->
   <servlet>
      <servlet-name>calculatorServlet</servlet-name>
      <servlet-class>CalculatorServlet</servlet-class>
   </servlet>

   <servlet-mapping>
      <servlet-name>calculatorServlet</servlet-name>
      <url-pattern>/CalculatorServlet</url-pattern>
   </servlet-mapping>

   <!-- Welcome File -->
   <welcome-file-list>
      <welcome-file>calculator.jsp</welcome-file>
   </welcome-file-list>
```



Simple Calculator

Enter the first number: 12
Enter the second number: 20

Select operation:

Result: 32

Index.jsp

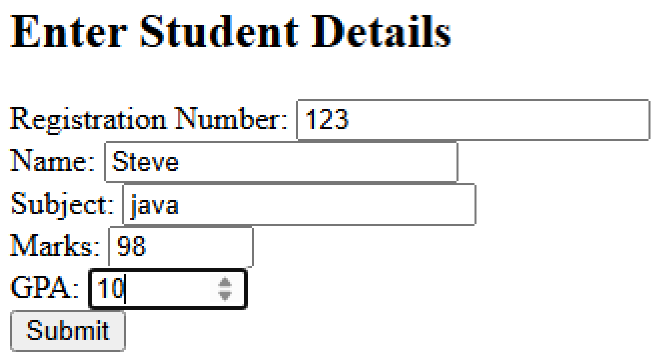
```
<%@ page language="java" contentType="text/html; charset=ISO-8859-1"%>
<!DOCTYPE html>
<html>
<head>
    <meta charset="ISO-8859-1">
    <title>Student Details Form</title>
</head>
<body>
    <h2>Enter Student Details</h2>

    <form action="StudentDetailsServlet" method="post">
        Registration Number: <input type="text" name="regno" required><br>
        Name: <input type="text" name="name" required><br>
        Subject: <input type="text" name="subject" required><br>
        Marks: <input type="number" name="marks" required><br>
        GPA: <input type="number" step="0.01" name="gpa" required><br>
        <input type="submit" value="Submit">
    </form>
</body>
</html>
```

StudentDetails.jsp

```
<%@ page language="java" contentType="text/html; charset=ISO-8859-1"%>
<!DOCTYPE html>
<html>
<head>
    <meta charset="ISO-8859-1">
    <title>Student Details Form</title>
</head>
<body>
    <h2>Enter Student Details</h2>

    <form action="StudentDetailsServlet" method="post">
        Registration Number: <input type="text" name="regno" required><br>
        Name: <input type="text" name="name" required><br>
        Subject: <input type="text" name="subject" required><br>
        Marks: <input type="number" name="marks" required><br>
        GPA: <input type="number"  name="gpa" required><br>
        <input type="submit" value="Submit">
    </form>
</body>
</html>
```

StudentDetailServlet.java

```
import java.io.IOException;
import java.util.ArrayList;
import java.util.List;
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("/StudentDetailsServlet")
public class StudentDetailsServlet extends HttpServlet {
    private List<Student> students = new ArrayList<>();
    protected void doGet(HttpServletRequest request, HttpServletResponse response)
         throws ServletException, IOException {
      // Forward the request to the JSP for displaying the form
      request.getRequestDispatcher("studentDetailsForm.jsp").forward(request, response);
   }
    protected void doPost(HttpServletRequest request, HttpServletResponse response)
         throws ServletException, IOException {
      // Retrieve student details from the form submission
      String regno = request.getParameter("regno");
      String name = request.getParameter("name");
      String subject = request.getParameter("subject");
      int marks = Integer.parseInt(request.getParameter("marks"));
      double gpa = Double.parseDouble(request.getParameter("gpa"));
       // Create a new Student object
      Student newStudent = new Student(regno, name, subject, marks, gpa)
      // Add the new student to the list
      students.add(newStudent);
      // Set the student details as a request attribute
      request.setAttribute("students", students);
      // Forward the request to the JSP for displaying the student details
      request.getRequestDispatcher("studentDetails.jsp").forward(request, response);
   }
}
```



Enter Student Details

Registration Number: 123
Name: Steve
Subject: java
Marks: 98
GPA: 10

```
%@ page language="java" contentType="text/html; charset=UTF-8" pageEncoding="UTF-8"%>
<%@ page import="java.util.List" %>
<%@ page import="Students" %>
<!DOCTYPE html>
<html>

<body>
    <h2>Student Details List</h2>

    <table>
        <tr>
            <th>Registration Number</th>
            <th>Name</th>
            <th>Subject</th>
            <th>Marks</th>
            <th>GPA</th>
        </tr>
        <%
       List<Students>studentss = (List<Students>)request.getAttribute("students");
        if (studentss != null) {
            for (Students student : studentss) {
        %>
            <tr>
                <td><%= student.getRegno() %></td>
                <td><%= student.getName() %></td>
                <td><%= student.getSubject() %></td>
                <td><%= student.getMarks() %></td>
                <td><%= student.getGpa() %></td>
            </tr>
        <%
            }
        }
        %>
    </table>
    <a href="StudentsDetails.jsp">Add Another Student</a>
</body></html>
```

```
// StudentDetailsServlet.java
import java.io.IOException;
import java.util.ArrayList;
import java.util.List;
import jakarta.servlet.ServletException;
import jakarta.servlet.annotation.WebServlet;
import jakarta.servlet.http.HttpServlet;
import jakarta.servlet.http.HttpServletRequest;
import jakarta.servlet.http.HttpServletResponse;


public class StudentDetailsServlet extends HttpServlet {
    // Make the list static to maintain data across requests
    private static List<Students> studentss = new ArrayList<>();

    protected void doGet(HttpServletRequest request, HttpServletResponse response)
            throws ServletException, IOException {
        // Set the existing students list as an attribute to display in the form page
        request.setAttribute("students1", studentss);
        // Forward the request to the JSP for displaying the form
        request.getRequestDispatcher("studentDetailsForm.jsp").forward(request, response);
    }
    protected void doPost(HttpServletRequest request, HttpServletResponse response)
            throws ServletException, IOException {
        try {
            // Retrieve student details from the form submission
            String regno = request.getParameter("regno");
            String name = request.getParameter("name");
            String subject = request.getParameter("subject");
            int marks = Integer.parseInt(request.getParameter("marks"));
            double gpa = Double.parseDouble(request.getParameter("gpa"));
            // Create a new Student object
            Students newStudent = new Students(regno, name, subject, marks, gpa);
            // Add the new student to the list
            studentss.add(newStudent);
            // Set the student details as a request attribute
            request.setAttribute("students1", studentss);
            for (Students student : studentss) {
            student.getRegno() ;
                student.getName() ;
                student.getSubject() ;
                 student.getMarks() ;
                student.getGpa() ;
            }
            // Forward the request to the JSP for displaying the student details
            request.getRequestDispatcher("StudentsForm.jsp").forward(request, response);
        } catch (NumberFormatException e) {
            System.out.println(e.getMessage());       }    }}
```

```
//class student with getter and setters!
public class Students {
    String regno;
    String name;
    String subject;
    int marks;
    double gpa;

    public Students(String regno, String name,
String subject, int marks, double gpa) {
        this.regno = regno;
        this.gpa = gpa;
        this.name = name;
        this.marks = marks;
        this.subject = subject;
    }
    public String getRegno() { return regno; }
    public String getName() { return name; }
    public String getSubject() { return subject; }
    public int getMarks() { return marks; }
    public double getGpa() { return gpa; }
}
```

```
<%@ page language="java" contentType="text/html; charset=UTF-8" pageEncoding="UTF-8"%>
<%@ page import="java.util.List" %>
<%@ page import="Students" %>
<!DOCTYPE html>
<html>

<body>
  <h2>Student Details List</h2>

  <table>
    <tr>
      <th>Registration Number</th>
      <th>Name</th>
      <th>Subject</th>
      <th>Marks</th>
      <th>GPA</th>
    </tr>
    <%
    List<Students>studentss = (List<Students>)request.getAttribute("students");
    if (studentss != null) {
      for (Students student : studentss) {
    %>
      <tr>
        <td><%= student.getRegno() %></td>
        <td><%= student.getName() %></td>
        <td><%= student.getSubject() %></td>
        <td><%= student.getMarks() %></td>
        <td><%= student.getGpa() %></td>
      </tr>
    <%
      }
    }
    %>
  </table>

  <a href="StudentsDetails.jsp">Add Another Student</a>
</body>
</html>
```