SOEN 387

WEB-BASED ENTERPRISE APPLICATIONS DESIGN

Tutorial 4

JSP and Java Beans

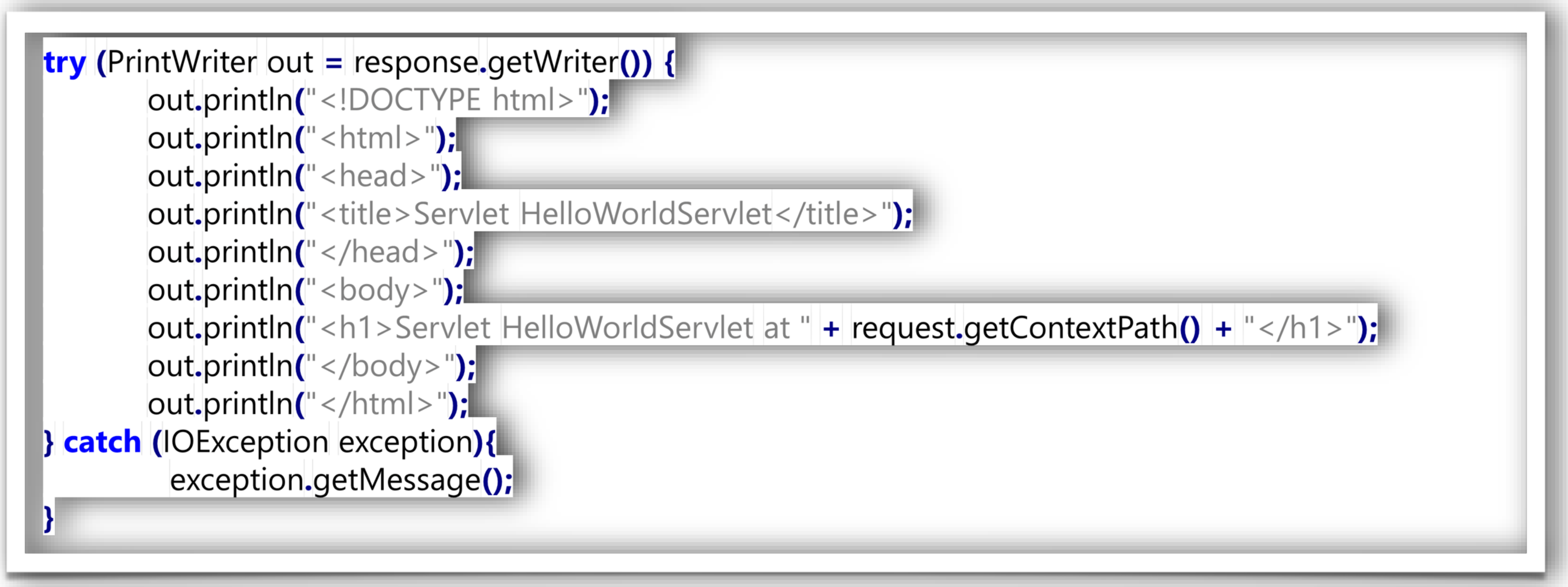
Agenda

- ➢ Why do we need JSP?
- ➢ How JSP works?
- ➢ Creating JSP
- ➢ Examples
- ➢ JSP Tags
- ➢ Java Beans

Please follow tutorial 1 to setup your project.

Why do we need JSP?

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



```
try (PrintWriter out = response.getWriter()) {
        out.println("<!DOCTYPE html>");
        out.println("<html>");
        out.println("<head>");
        out.println("<title>Servlet HelloWorldServlet</title>");
        out.println("</head>");
        out.println("<body>");
        out.println("<h1>Servlet HelloWorldServlet at " + request.getContextPath() + "</h1>");
        out.println("</body>");
        out.println("</html>");
} catch (IOException exception){
         exception.getMessage();
}
```

JSP pages are more advantageous than Servlet

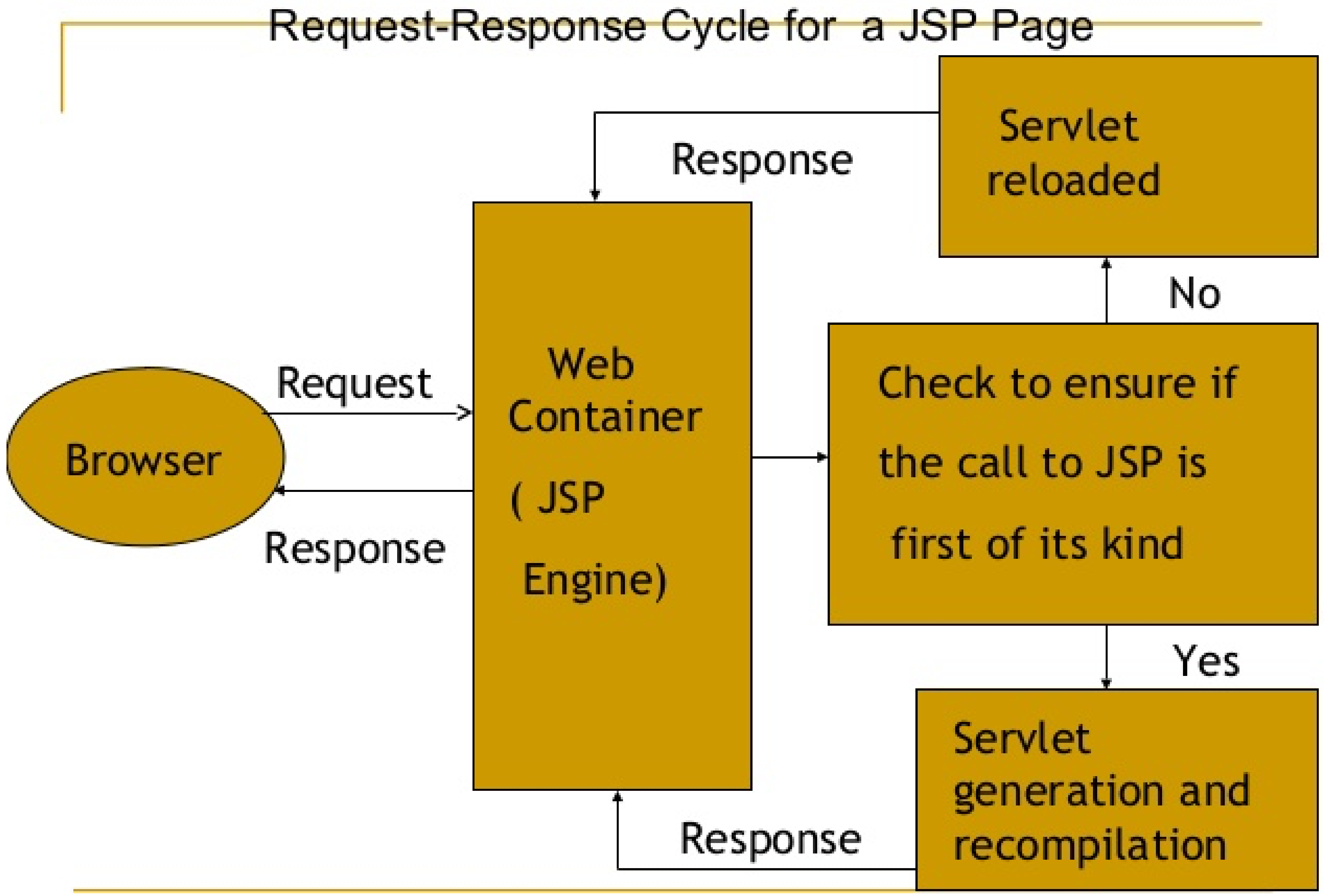
- They are easy to maintain.
- No recompilation or redeployment is required.
- JSP has access to entire API of JAVA .
- JSP are extended version of Servlet.

What is JSP?

- It stands for **Java Server Pages**.
- It is a server side technology.
- It is used for creating web application.
- It is used to create dynamic web content.
- In this JSP tags are used to insert JAVA code into HTML pages.
- It is an advanced version of Servlet Technology.
- It is a Web based technology helps us to create dynamic and platform independent web pages.
- In this, Java code can be inserted in HTML/ XML pages or both.
- JSP is first converted into servlet by JSP container before processing the client's request.

How JSP Works?

Java Server Pages (JSP)



Request-Response Cycle for a JSP Page

Browser

Request

Response

Web Container (JSP Engine)

Check to ensure if the call to JSP is first of its kind

No

Servlet reloaded

Response

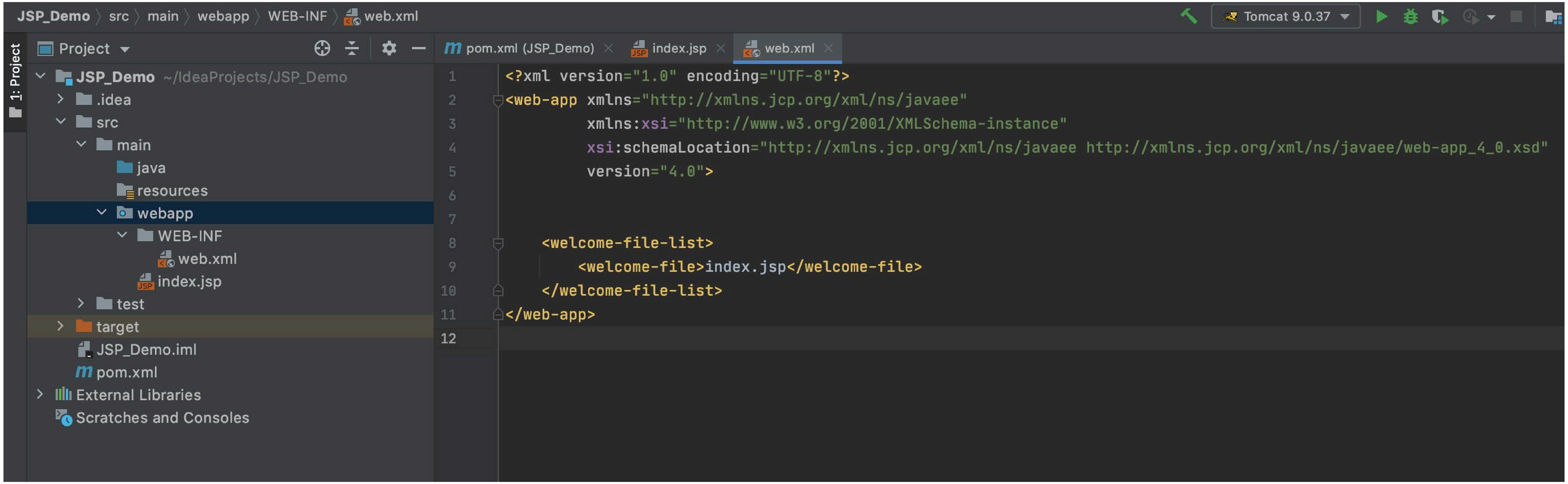
Yes

Servlet generation and recompilation

Response

Creating a JSP

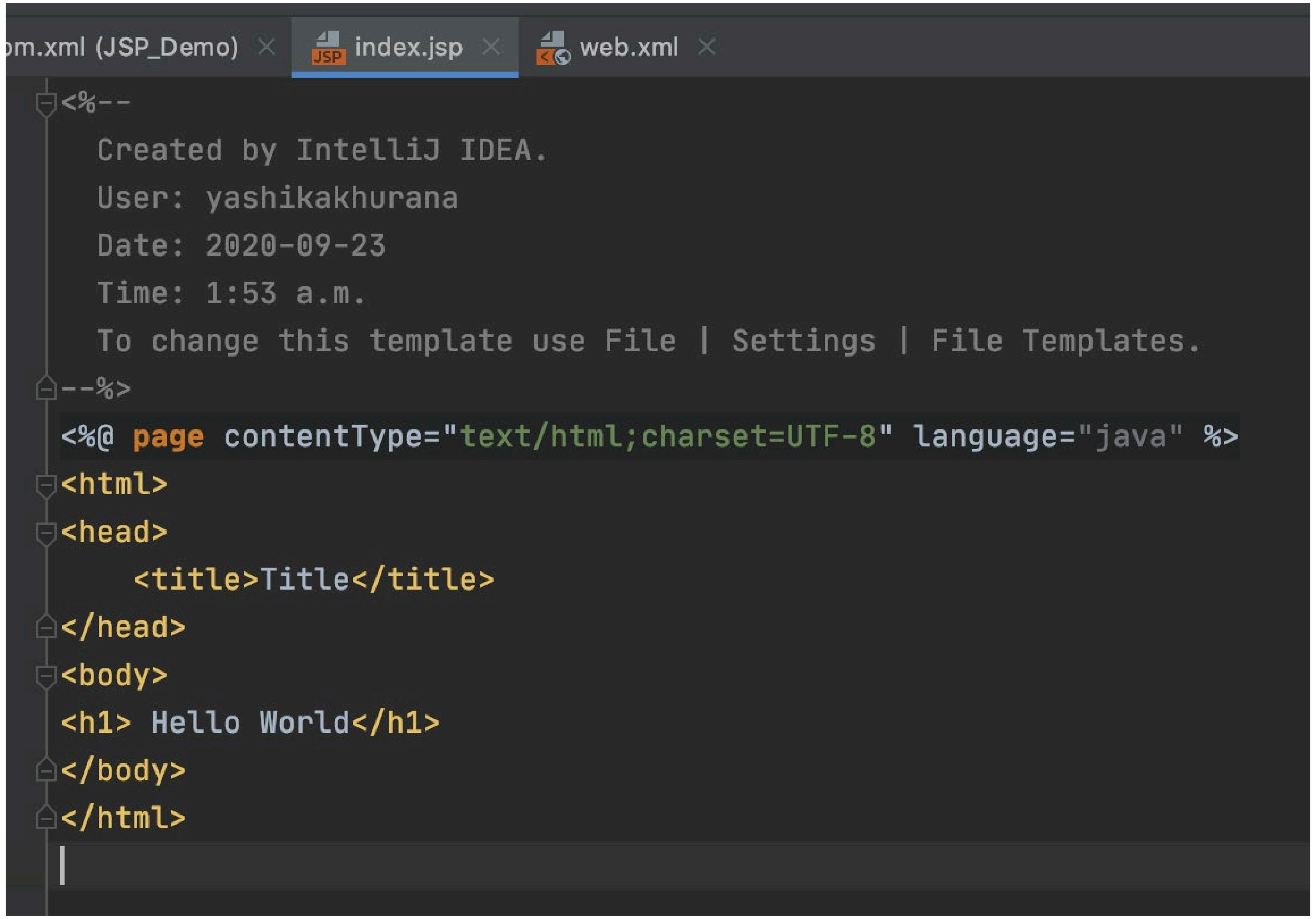
Web.xml



```
<?xml version="1.0" encoding="UTF-8"?>
<web-app xmlns="http://xmlns.jcp.org/xml/ns/javaee"
         xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
         xsi:schemaLocation="http://xmlns.jcp.org/xml/ns/javaee http://xmlns.jcp.org/xml/ns/javaee/web-app_4_0.xsd"
         version="4.0">


    <welcome-file-list>
        <welcome-file>index.jsp</welcome-file>
    </welcome-file-list>
</web-app>

```

```
<welcome-file-list>
    <welcome-file>index.jsp</welcome-file>
</welcome-file-list>
```

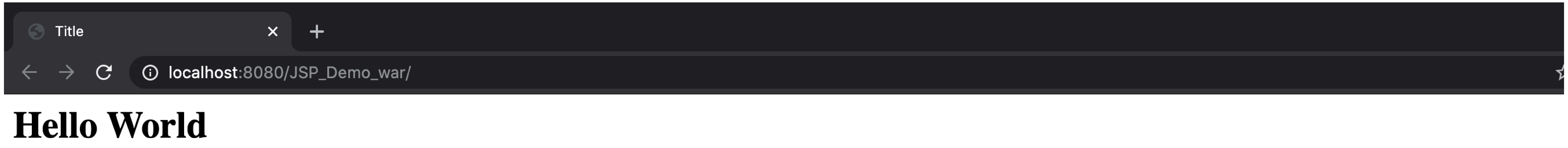
index.jsp



```
<%--
  Created by IntelliJ IDEA.
  User: yashikakhurana
  Date: 2020-09-23
  Time: 1:53 a.m.
  To change this template use File | Settings | File Templates.
--%>
<%@ page contentType="text/html;charset=UTF-8" language="java" %>
<html>
<head>
    <title>Title</title>
</head>
<body>
<h1> Hello World</h1>
</body>
</html>
```

```
<%@ page contentType="text/html;charset=UTF-8"
language="java" %>
<html>
<head>
    <title>Title</title>
</head>
<body>
<h1> Hello World</h1>
</body>
</html>
```

Output

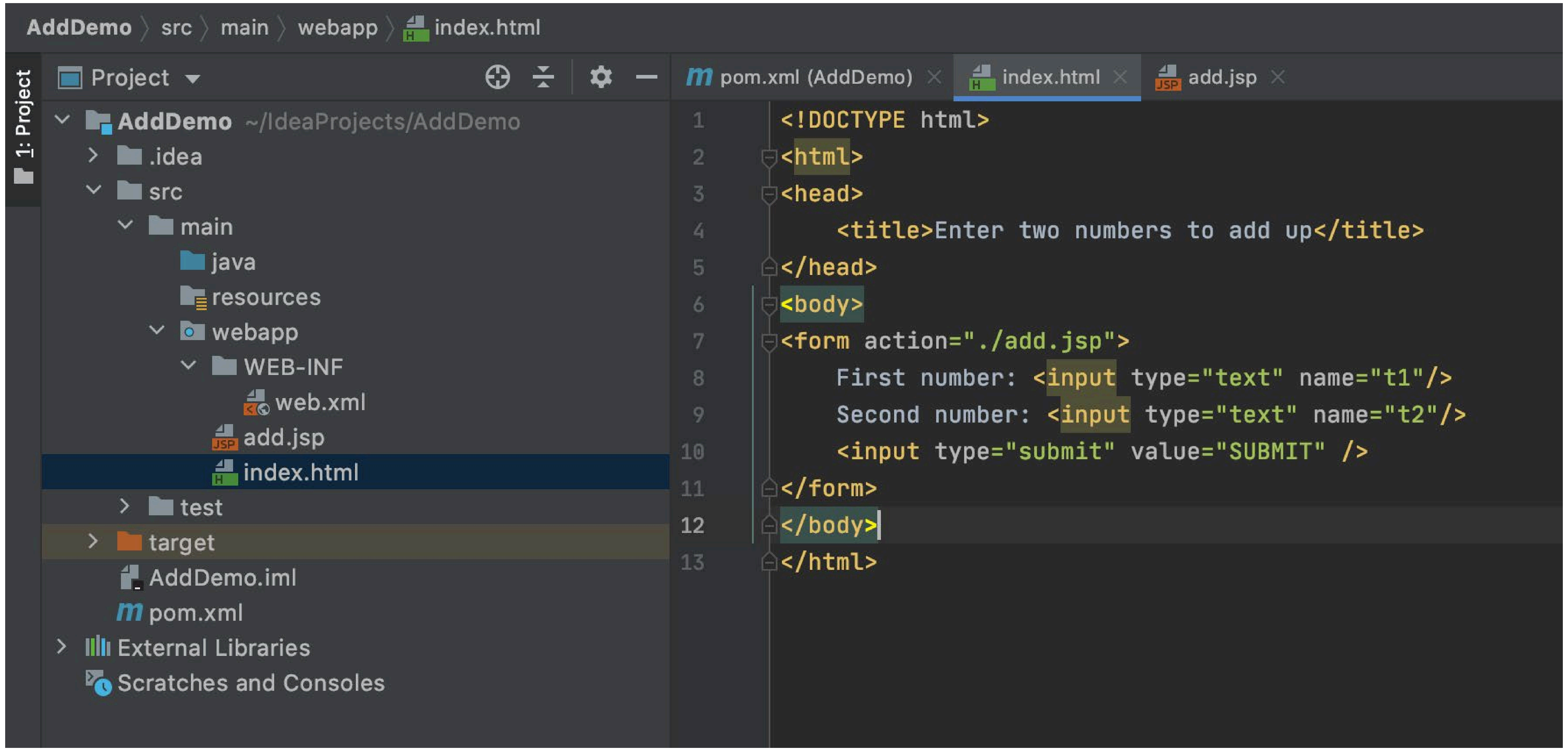


Hello World

Example to Add two no. using JSP

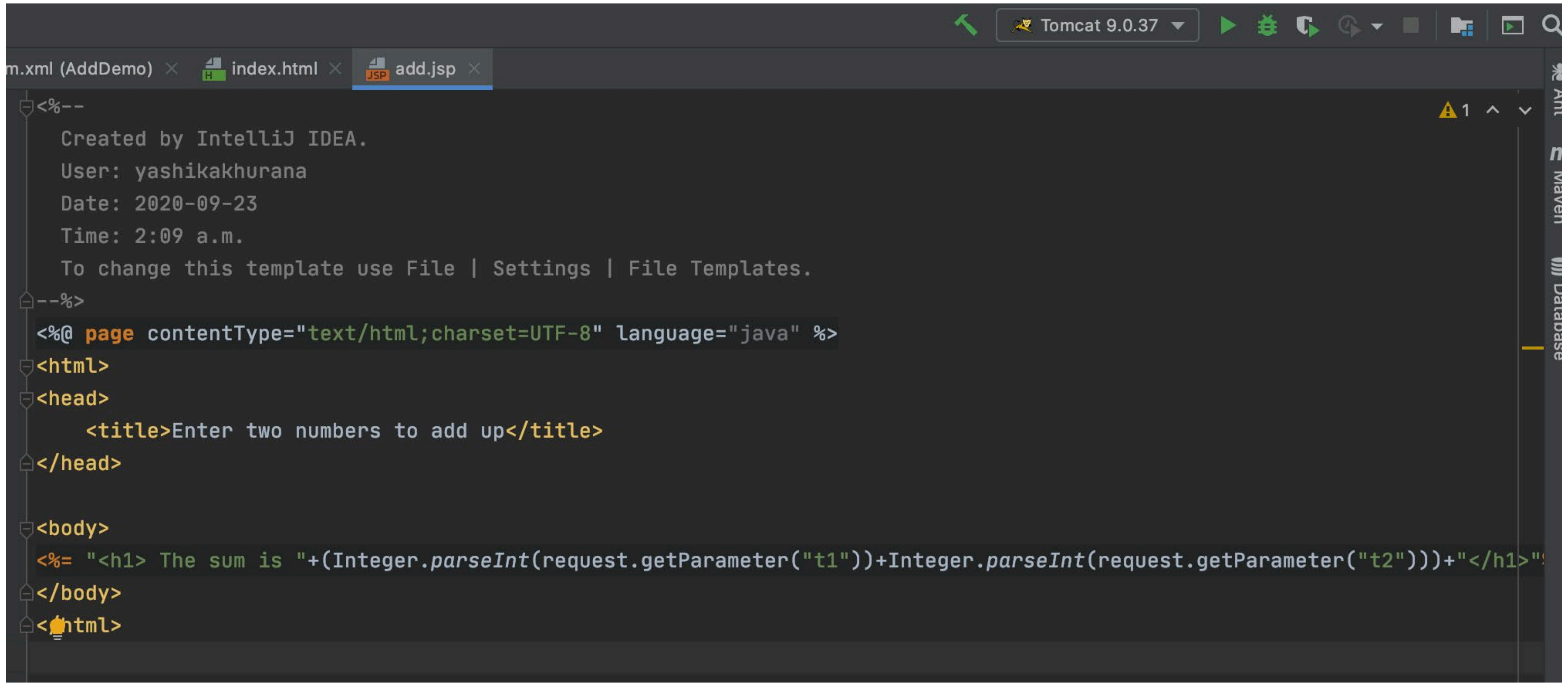
index.html

```html
<!DOCTYPE html>
<html>
<head>
    <title>Enter two numbers to add up</title>
</head>
<body>
<form action="./add.jsp">
    First number: <input type="text" name="t1"/>
    Second number: <input type="text" name="t2"/>
    <input type="submit" value="SUBMIT" />
</form>
</body>
</html>
```



```html
<!DOCTYPE html>
<html>
<head>
    <title>Enter two numbers to add up</title>
</head>
<body>
<form action="./add.jsp">
    First number: <input type="text" name="t1"/>
    Second number: <input type="text" name="t2"/>
    <input type="submit" value="SUBMIT" />
</form>
</body>
</html>
```

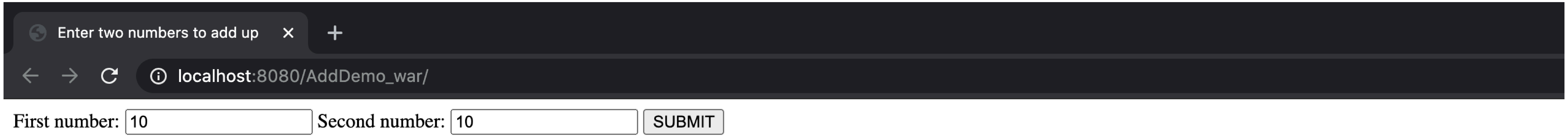
add.jsp



```
<%--
  Created by IntelliJ IDEA.
  User: yashikakhurana
  Date: 2020-09-23
  Time: 2:09 a.m.
  To change this template use File | Settings | File Templates.
--%>
<%@ page contentType="text/html;charset=UTF-8" language="java" %>
<html>
<head>
    <title>Enter two numbers to add up</title>
</head>

<body>
<%= "<h1> The sum is "+(Integer.parseInt(request.getParameter("t1"))+Integer.parseInt(request.getParameter("t2")))+"</h1>"
</body>
</html>
```

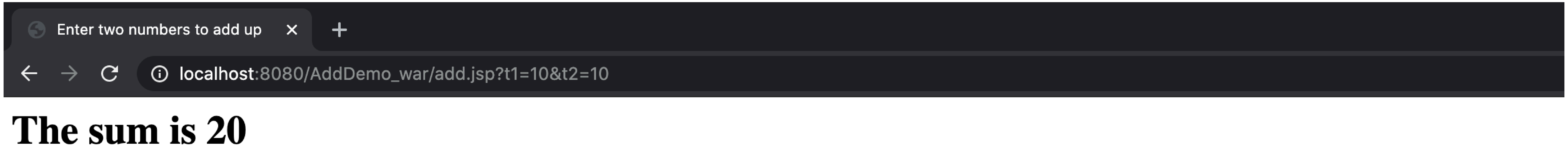
```
<%@ page contentType="text/html;charset=UTF-8" language="java" %>
<html>
<head>
    <title>Enter two numbers to add up</title>
</head>

<body>
<%= "<h1> The sum is "+(Integer.parseInt(request.getParameter("t1"))+Integer.parseInt(request.getParameter("t2")))+"</h1>"%>
</body>
</html>
```

Output



localhost:8080/AddDemo_war/

First number: 10 Second number: 10



localhost:8080/AddDemo_war/add.jsp?t1=10&t2=10

The sum is 20

JSP TAGS

JSP TAGS

- Declaration tag
- Expression tag
- Directive tag
- Scriplet tag
- Action tag

<%

……

%>

Declaration Tag (<%! %>)

- Declaration tag allows the developer to declare variables or methods
- Before the declaration you must use <%! And at the end of the declaration %>
- Code placed in this must end with a semicolon (;)
- 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
- Example <%= new java.util.Date() %>

Directive Tag (<%@ directive %>)

- JSP directive gives special information about the jsppage, to the JSP engine
- There are three main types of directives:

1. Page- Use for processing the information of that page
2. Include- Use for to include all the files
3. Tag library-Use to inform the page about the Tags library

Page Directive (<%@ directive %>)

- This directive has 11 optional attributes that provides the JSP Engine with special processing information
- It includes- buffer, autoFlush, contentType, errorPage, isErrorPage, extends, import, info, isThreadSafe, language, session, isELIgnored, isScriptingEnabled
- Example: To import classes
- <%@ page import =“java.util.*”%>
- Example: info Directive
- <%@page info=“abc.com testpage,copyright 2020.”%>

Include Directive (<%@ include file %>)

- This tag allows to include the contents of a file inside another.
- Typically include files are used for navigation, tables, headers, footers that are common to multiple pages
- You can include html/jsp page
- Example: <%@ include file =“header.html” %>
- <%@ include file =“navigation.jsp” %>

Scriplet tag(<% %>)

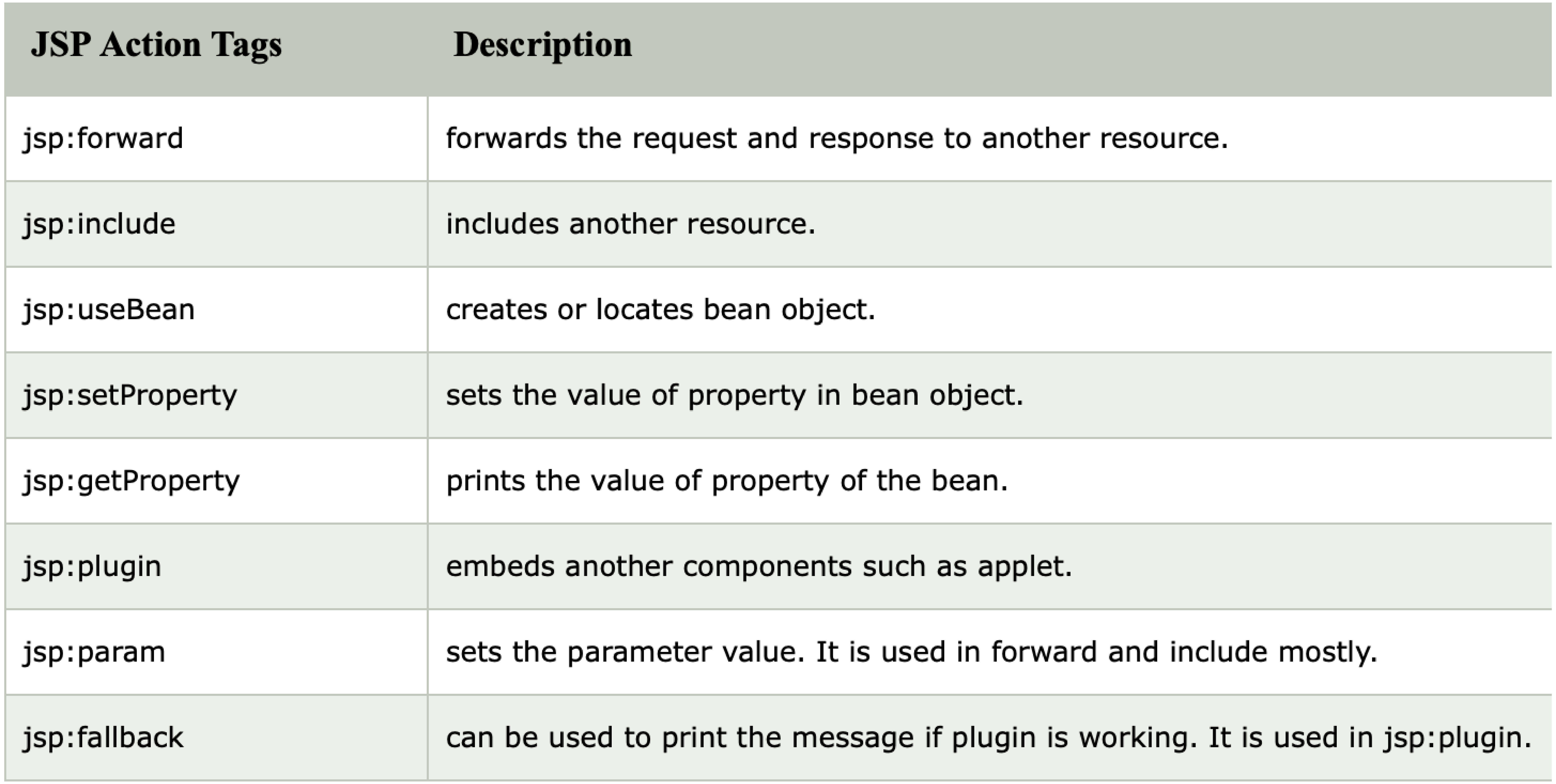
- Between <% and %> tags, any valid java code is called a Scriplet
- This code can access any variable or been declared
- Example:

```
<%
String message="Demo";
out.println(message);
%>
```

Action Tag

- There are many JSP action tags or elements.
- Each JSP action tag is used to perform some specific tasks
- The action tags are used to control the flow between the pages
- It also enables to use of the server side Javabeans

Action Tag



JSP Action Tags	Description
jsp:forward	forwards the request and response to another resource.
jsp:include	includes another resource.
jsp:useBean	creates or locates bean object.
jsp:setProperty	sets the value of property in bean object.
jsp:getProperty	prints the value of property of the bean.
jsp:plugin	embeds another components such as applet.
jsp:param	sets the parameter value. It is used in forward and include mostly.
jsp:fallback	can be used to print the message if plugin is working. It is used in jsp:plugin.

JAVA BEANS

Java Beans

- Reusability is the main concept in any programming language. A JavaBean is a software component that has been designed to be reusable in a variety of environments.
- JavaBeans are classes which encapsulate several objects into a single object.
- It helps in accessing these object from multiple places.

JavaBeans has several conventions that should be followed:

- Beans should have a default constructor (no arguments)
- Beans should provide getter and setter methods
  1. A getter method is used to read the value of a readable property
  2. To update the value, a setter method should be called
- Beans should implement java.io.serializable, as it allows to save, store and restore the state of a JavaBean you are working on

Access Java Beans in JSP

<jsp:useBean> tag is used to instantiate a JavaBean or to locate existing bean instance and assign it to a variable name. Synatx of <jsp:useBean> action tag :

```
<jsp:useBean
  id= "instanceName"
  scope= "page | request | session | application"
  class= "packageName.className"
</jsp:useBean>
```

Where

- id is the identifier used to identify the bean in a given scope.
- class attribute is used to instantiate the specified bean. Bean must have zero argument constructor and must not be an abstract class.
- scope is scope of bean and can have four possible values.

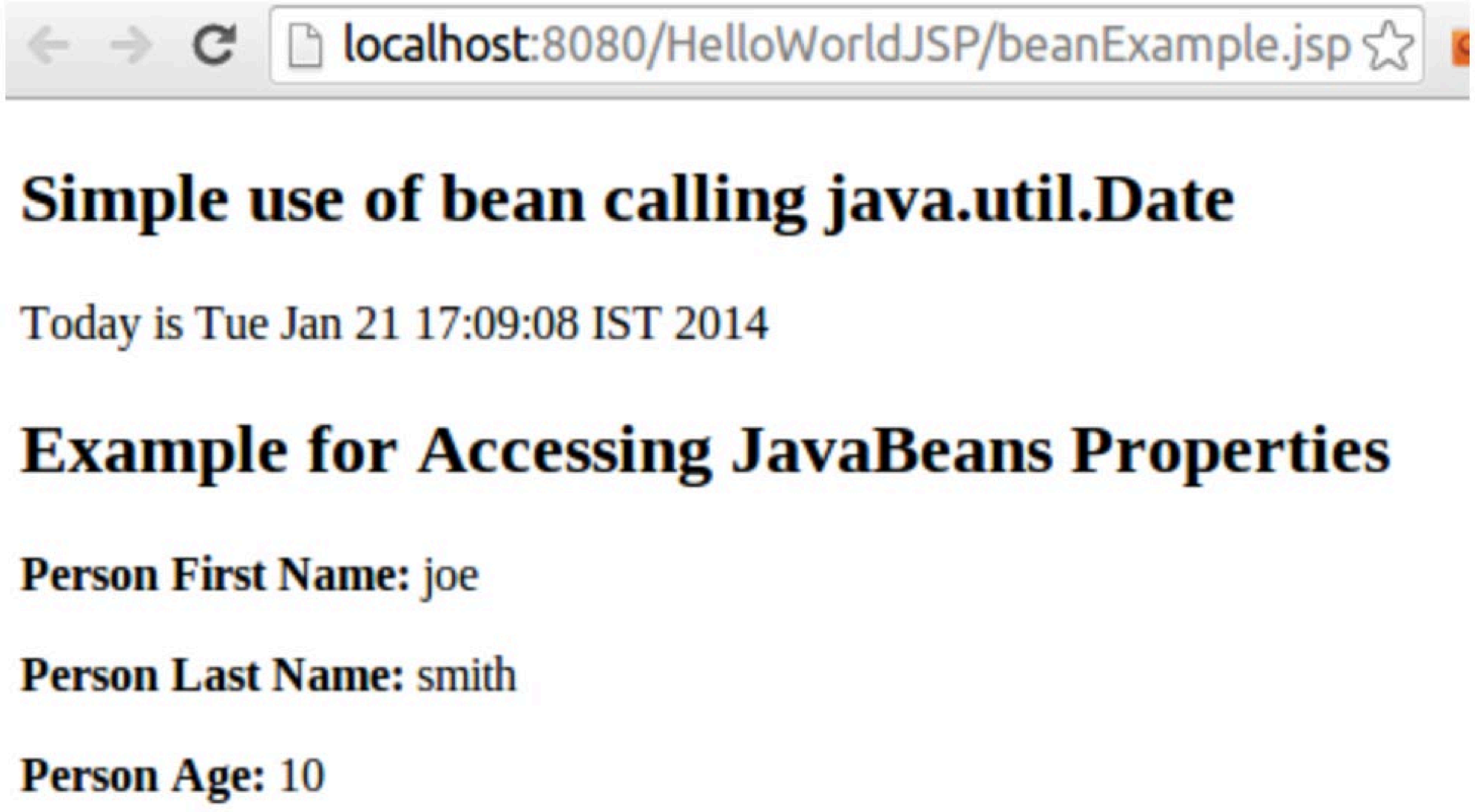
o request- specifies that bean is available in all jsp (sharing same request).
o page- specifies that this bean can be used within the page only and this is default scope.
o session- bean is available in session scope.
o application- bean is available in application scope.

Create Person.java

```
package javabeat.net.jsp.beans;
public class Person {
        private String firstName = null;
        private String lastName = null;
        private int age = 0;

        public String getFirstName() {
                return firstName;}

        public void setFirstName(String firstName) {
                this.firstName = firstName;}

        public String getLastName() {
                return lastName;}

        public void setLastName(String lastName) {
                this.lastName = lastName;}

       public int getAge() {
                return age;}

       public void setAge(int age) {
                this.age = age;
        }
}
```

beanExample.jsp

```
<%@ page language='java' contentType='text/html; charset=UTF-8'
    pageEncoding='UTF-8'%>
<!DOCTYPE html PUBLIC '-//W3C//DTD HTML 4.01 Transitional//EN' 'http://www.w3.org/TR/html4/loose.dtd'>
<html>
<head>
<meta http-equiv='Content-Type' content='text/html; charset=UTF-8'>
<title>Use Bean Example</title>
</head>
<body>
<jsp:useBean id='date'  class='java.util.Date'/>
<jsp:useBean id='person'  class='javabeat.net.jsp.beans.Person'>
   <jsp:setProperty name='person' property='firstName'
                  value='joe'/>
   <jsp:setProperty name='person' property='lastName'
                  value='smith'/>
   <jsp:setProperty name='person' property='age'
                  value='10'/>
</jsp:useBean>

<h2>Simple use of bean calling java.util.Date</h2>
Today is <%=date%>
<h2>Example for Accessing JavaBeans Properties</h2>
<p><b>Person First Name:</b>
   <jsp:getProperty name='person' property='firstName'/>
</p>
<p><b>Person Last Name:</b>
   <jsp:getProperty name='person' property='lastName'/>
</p>
<p><b>Person Age:</b>
   <jsp:getProperty name='person' property='age'/>
</p>
</body>
</html>
```



Simple use of bean calling java.util.Date

Today is Tue Jan 21 17:09:08 IST 2014

Example for Accessing JavaBeans Properties

Person First Name: joe

Person Last Name: smith

Person Age: 10

Thank You!

References

- https://slideplayer.com/slide/16349711/
- https://www2.cs.siu.edu/~rahimi/cs412/slides/JSP_tutorial.pdf
- https://www.javatpoint.com/jsp-tutorial
- https://javabeat.net/javabeans-jsp/