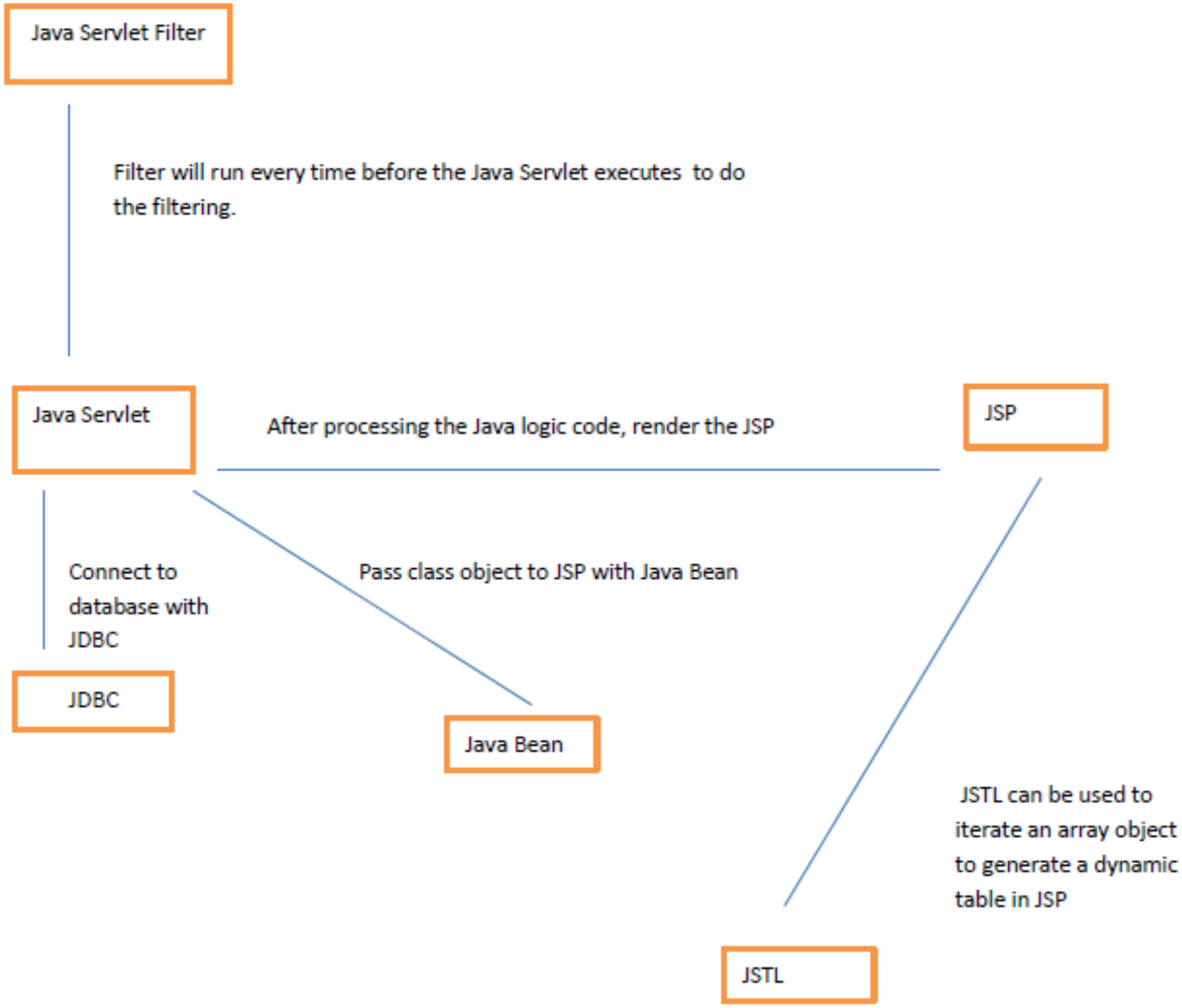
A Tutorial on Project Assignment No. 3

Arian Niaki

Overview

- Eclipse
- Apache Tomcat
- Java Servlet
- Java Servlet Filter
- JSP
- Java Bean
- JSTL
- JDBC

All the below components runs on top of Apache Tomcat.



Download Eclipse IDE for Java EE (Neon)

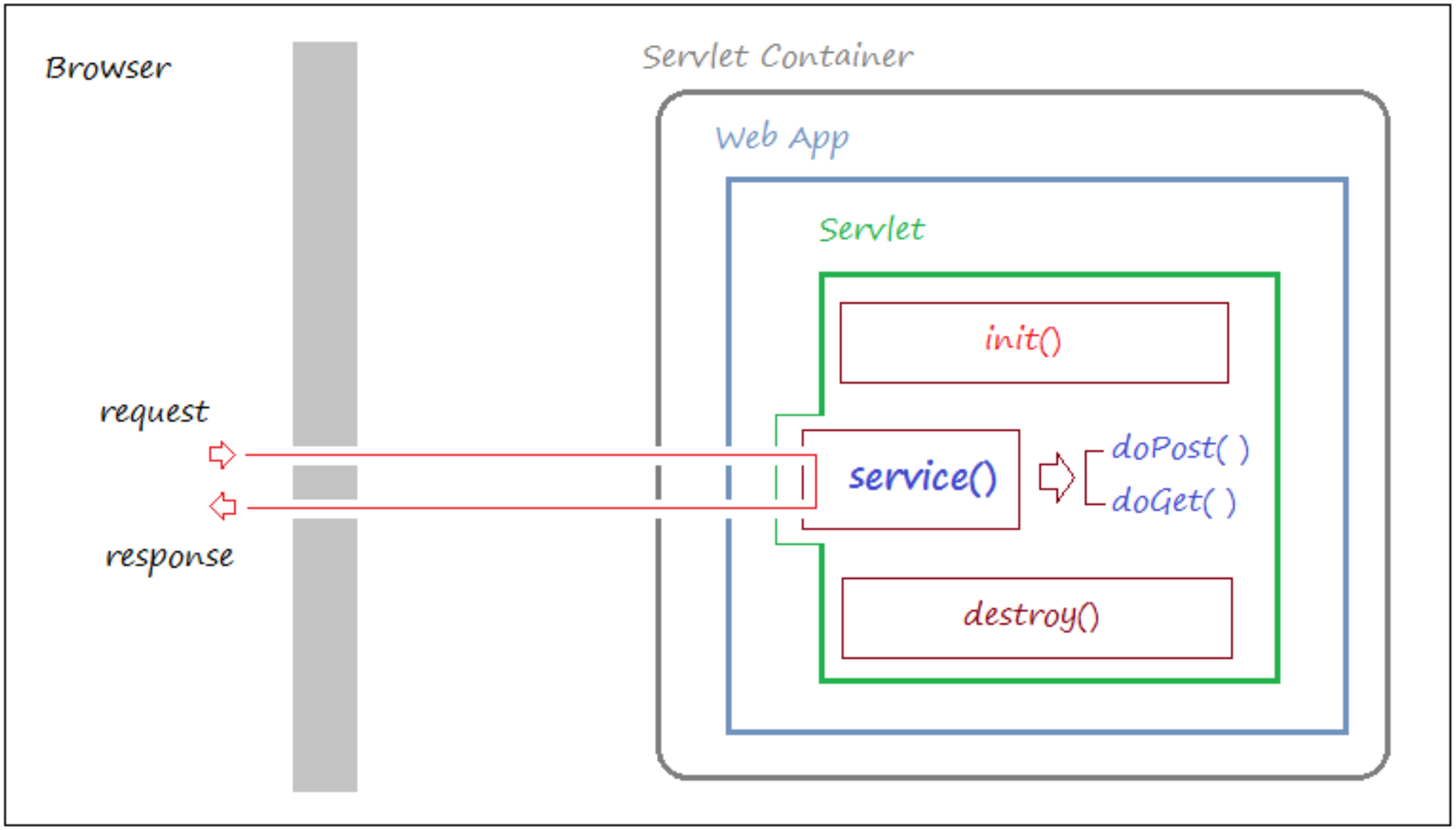
Apache Tomcat

- Application Server
- Renders webpage which includes Java Server Page coding
- Download: http://tomcat.apache.org/download-80.cgi
- Installation and tutorial: http://o7planning.org/web/fe/default/en/document/20397/java-jsp-tutorial-for-beginners#a750360

Java Servlet

- Are programs that run on a Web or Application server.
- Act as a middle layer between a request coming from a HTTP client and databases or applications on the HTTP server.
- Goal: separate the logic code(Java) from the HTML
- Tutorial: http://o7planning.org/web/fe/default/en/document/12760/java-servlet-tutorial-for-beginners

Java Servlet



Servlet Example

```java
package org.o7planning.tutorial.servlet;

import java.io.IOException;

import javax.servlet.ServletException;
import javax.servlet.ServletOutputStream;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;


public class HelloServlet extends HttpServlet {

   private static final long serialVersionUID = 1L;


   public HelloServlet() {
   }

   @Override
   protected void doGet(HttpServletRequest request,
           HttpServletResponse response) throws ServletException, IOException {

       ServletOutputStream out = response.getOutputStream();

       out.println("<html>");
       out.println("<head><title>Hello Servlet</title></head>");

       out.println("<body>");
       out.println("<h3>Hello World</h3>");
       out.println("This is my first Servlet");
       out.println("</body>");
       out.println("<html>");
   }

   @Override
   protected void doPost(HttpServletRequest request,
           HttpServletResponse response) throws ServletException, IOException {
       this.doGet(request, response);
   }

}
```

(1)

localhost:8080/ServletTutorial/hello

browser

web.xml

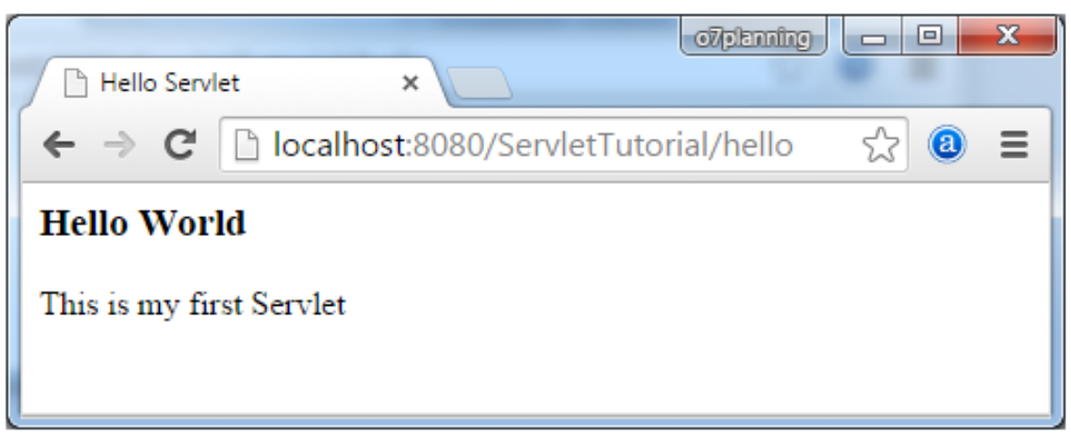
```
<servlet>
    <servlet-name>helloServlet</servlet-name>
    <servlet-class>org.o7planning.tutorial.servlet.HelloServlet</servlet-class>
</servlet>

<servlet-mapping>
    <servlet-name>helloServlet</servlet-name>
    <url-pattern>/hello</url-pattern>
</servlet-mapping>
```

(2)

(3)

HelloServlet.java

```
@Override
protected void doGet(HttpServletRequest request,
        HttpServletResponse response) throws ServletException, IOException {

    ServletOutputStream out = response.getOutputStream();

    out.println("<html>");
    out.println("<head><title>Hello Servlet</title></head>");

    out.println("<body>");
    out.println("<h3>Hello World</h3>");
    out.println("This is my first Servlet");
    out.println("</body>");
    out.println("<html>");
}
```



Page source

```
<html>
<head><title>Hello Servlet</title></head>
<body>
<h3>Hello World</h3>
This is my first Servlet
</body>
<html>
```

Java Servlet Filter

- Servlet Filters are Java classes that can be used in Servlet Programming for the following purposes:
    - To intercept requests from a client before they access a resource at back end.
    - To manipulate responses from server before they are sent back to the client
- Tutorial: http://o7planning.org/web/fe/default/en/document/753859/java-servlet-filter-tutorial

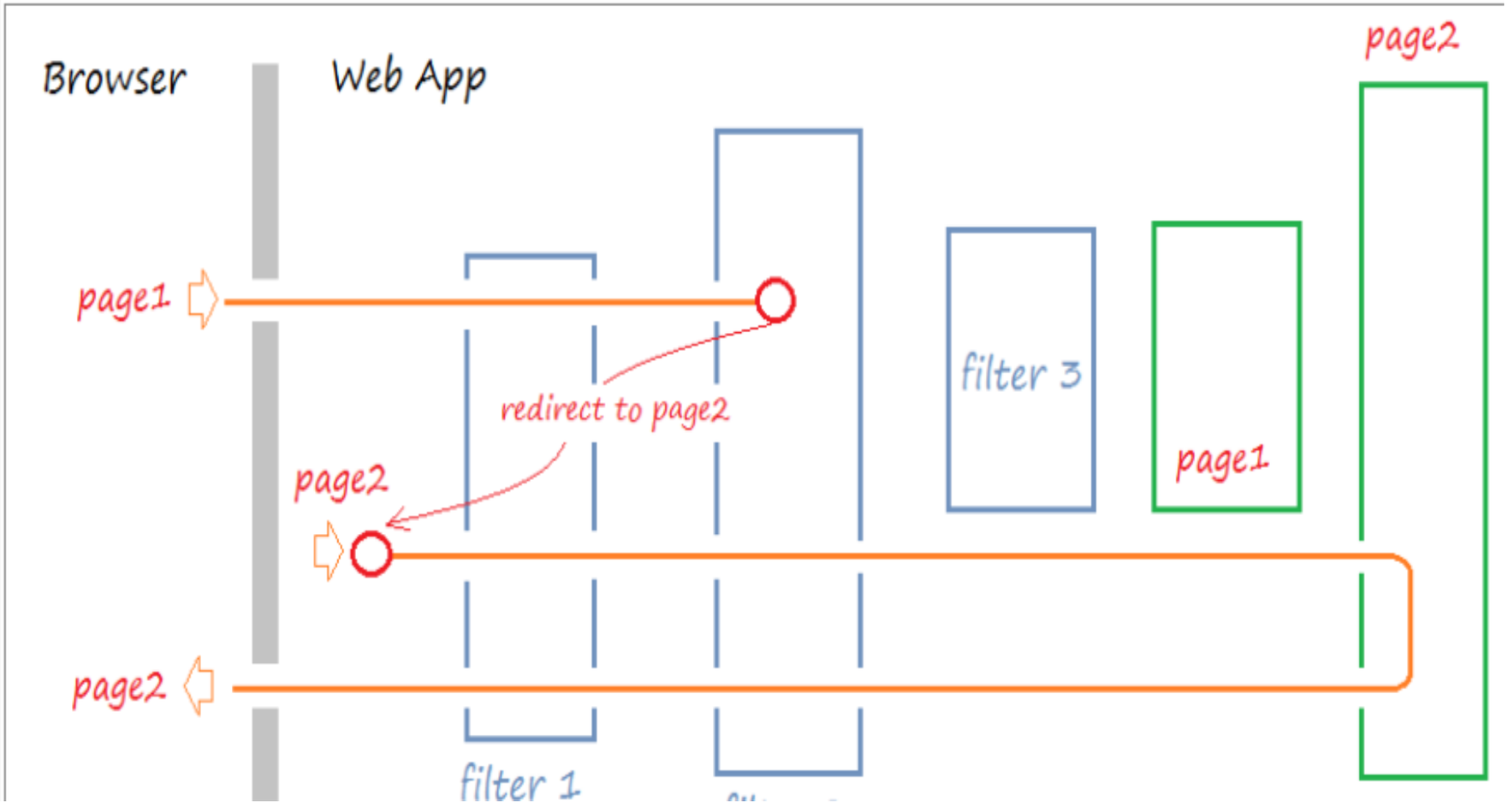
Java Servlet Filter

- Example:
  - Authentication Filter: instead of writing code to verify the user in each JSP, you can put the verification code in one Filter. And then each time you make request to Server for getting a JSP, the Filter will run.
  - Image not found: If you request an image and the image is not found it will be redirected to the default image (Sorry no Image found.jpg).

Java Servlet Filter

Example situation:

1. Users submit a request for personal info.
2. Request will be sent to the Server.
3. It goes through Filter that records log information.
4. It goes to Filter to check user logined or not, this filter found that the user is not logined, it will redirect the request of the user to the log page.



JSP

- JSP: HTML + Java
- Tutorial: http://o7planning.org/web/fe/default/en/document/20397/java-jsp-tutorial-for-beginners#a36971
- http://www.w3schools.com/

JSP

```
1  <%@ page language="java" contentType="text/html; charset=ISO-8859-1"
2    pageEncoding="ISO-8859-1"%>
3  <!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
4              "http://www.w3.org/TR/html4/loose.dtd">
5  <html>
6  <head>
7  <meta http-equiv="Content-Type" content="text/html; charset=ISO-8859-1">
8  <title>Insert title here</title>
9  </head>
10 <body>
11
12   <h1>Hello JSP</h1>
13
14   <%
15       java.util.Date date = new java.util.Date();
16   %>
17
18   <h2>
19       Now is
20       <%=date.toString()%>
21   </h2>
22 </body>
23 </html>
```

JSP

Client

Browser:

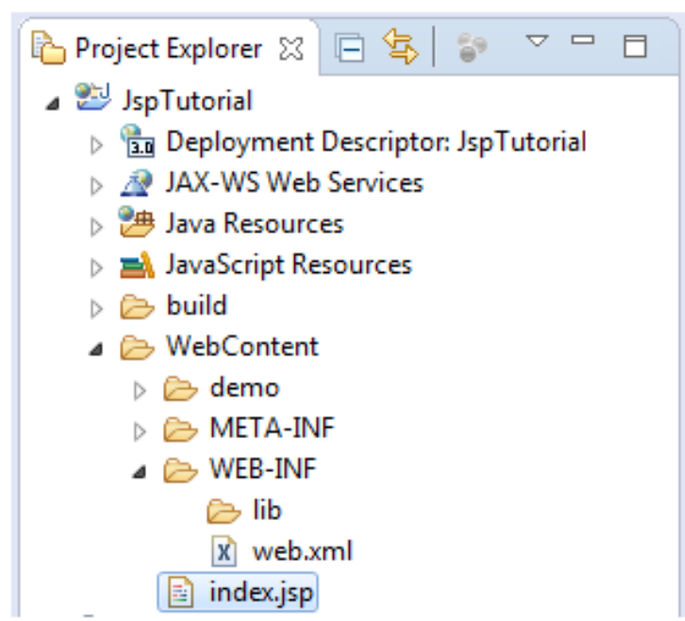
http://localhost:8080/JspTutorial/

index.jsp

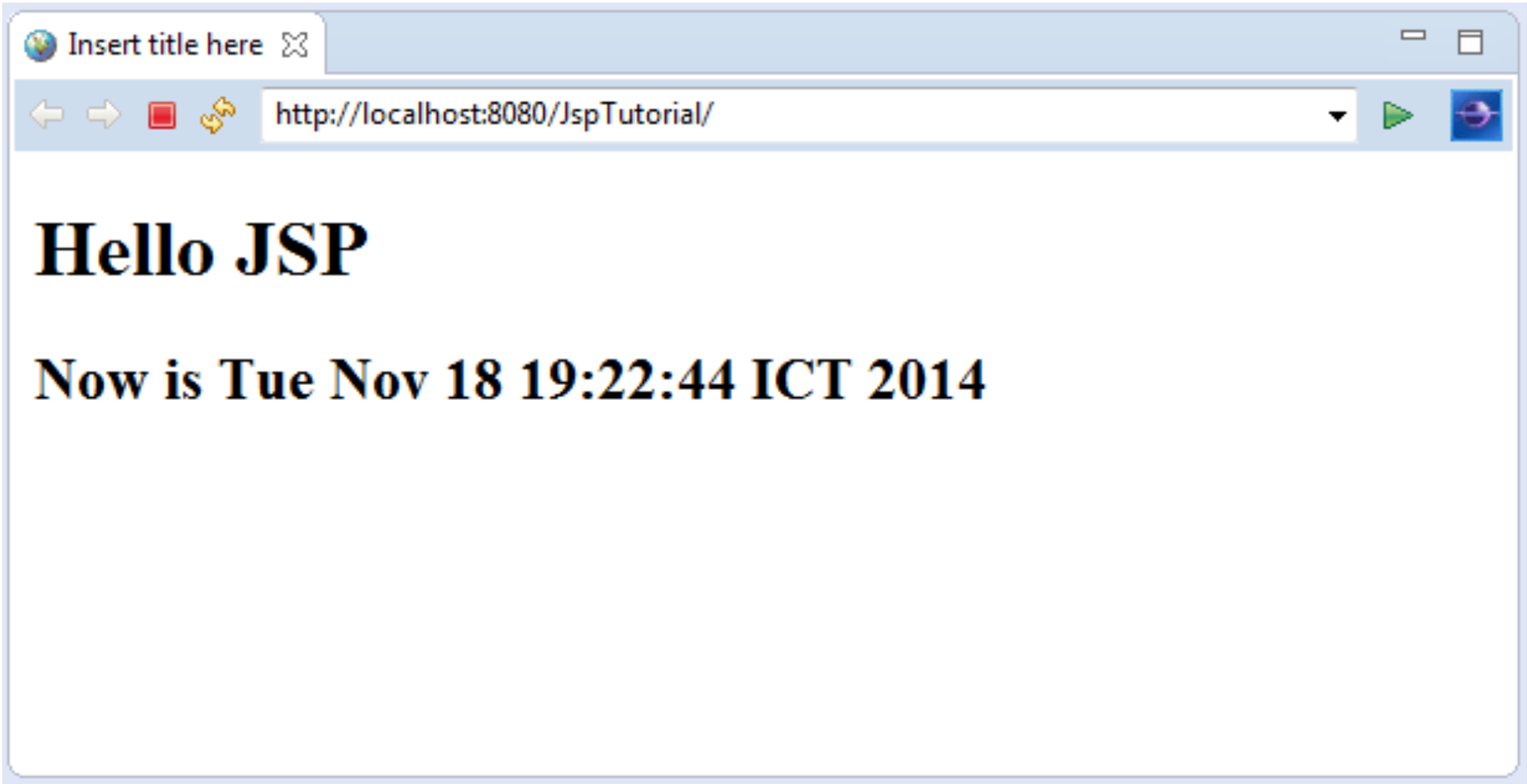
Server

web.xml

```
<welcome-file-list>
  <welcome-file>index.html</welcome-file>
  <welcome-file>index.htm</welcome-file>
  <welcome-file>index.jsp</welcome-file>
  <welcome-file>default.html</welcome-file>
  <welcome-file>default.htm</welcome-file>
  <welcome-file>default.jsp</welcome-file>
</welcome-file-list>
```

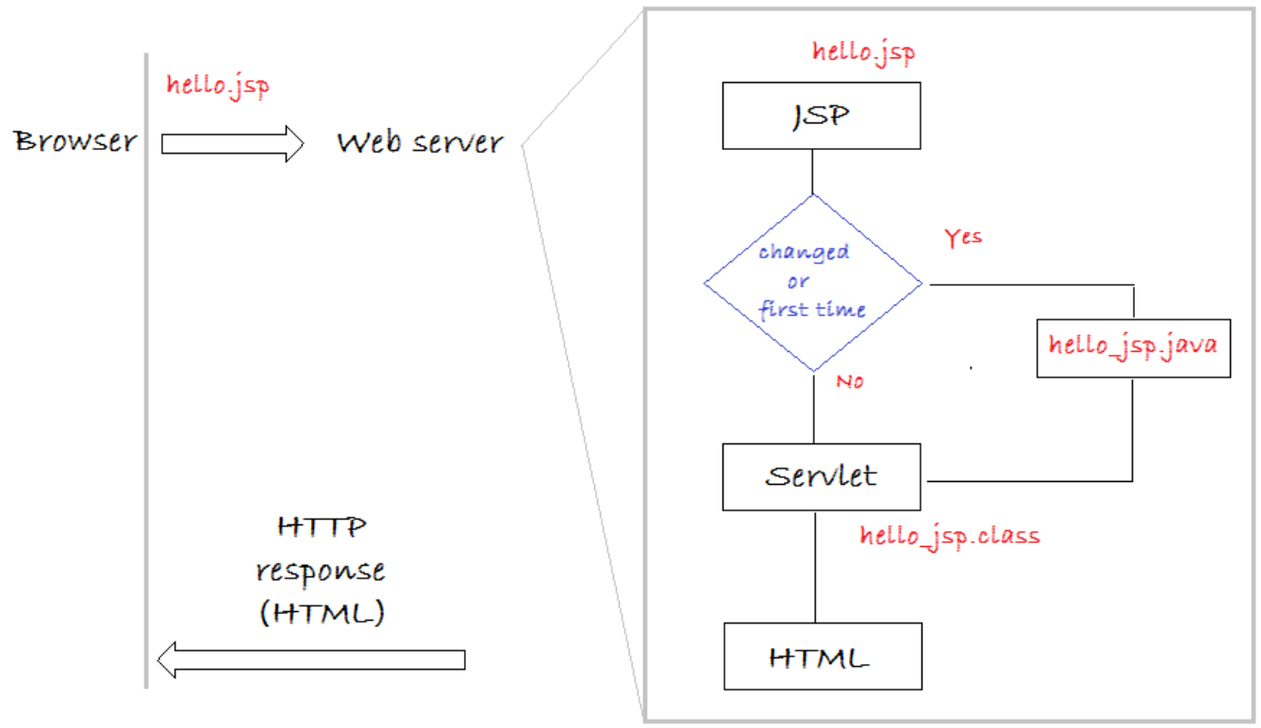


JSP


Insert title here
http://localhost:8080/JspTutorial/
Hello JSP
Now is Tue Nov 18 19:22:44 ICT 2014

JSP and Servlet

- When user sends a request to a JSP page, for example, ***hello.jsp***:
- At the first time, **Web Server** will change ***hello.jsp*** page to ***hello_jsp.java*** file, and compile it to ***hello_java.class***. This is a **Servlet**, and it will create a **HTML** in response to user's request.
- From second request onwards, it will check ***hello.jsp*** file for any change. If there is no change, it will call servlet (***hello_jsp.class***) and reply with HTML data to user. If there is some change, it will recreate ***hello_jsp.java*** and recompile it to ***hello_jsp.class*** file.
- So when you change JSP files you do not need to rerun the **Web Server**.



JSP:example

header.html

```html
<div style="background: #E0E0E0; height: 80px; padding: 5px;">
  <div style="float: left">
      <h1>My Site</h1>
  </div>
  <div style="float: right; padding: 10px;">
      Search <input name="search">
  </div>
</div>
```

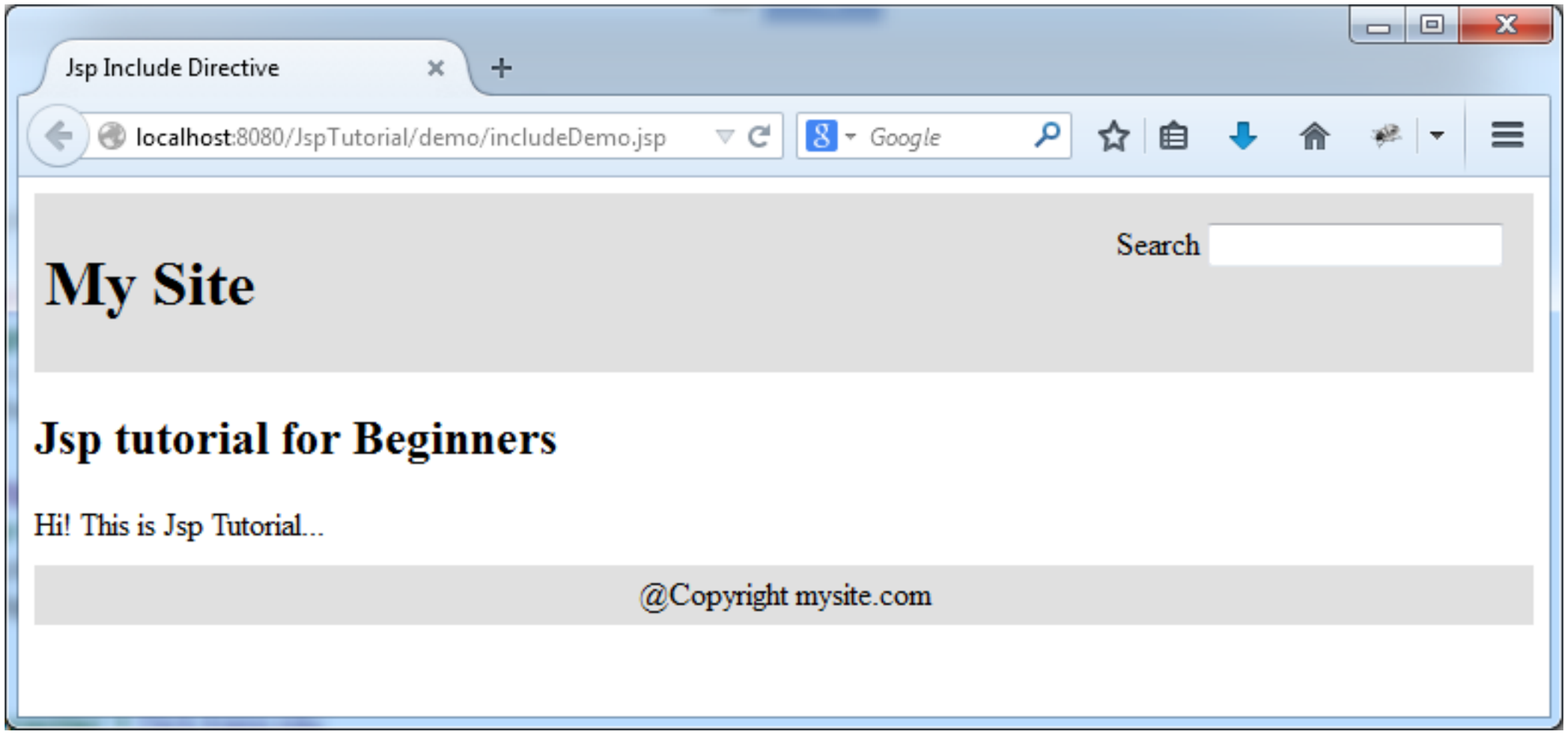
footer.html

```html
<div
  style="background: #E0E0E0; text-align: center; padding: 5px; margin-top: 10px;">
  @Copyright mysite.com
</div>
```

includeDemo.jsp

```jsp
<%@ page import="java.util.Random,java.text.*"%>
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
<title>Jsp Include Directive</title>
</head>
<body>

  <%@ include file="../fragment/header.html"%>


  <h2>Jsp tutorial for Beginners</h2>

  Hi! This is Jsp Tutorial...


  <%@ include file="../fragment/footer.html"%>
</body>
</html>
```

JSP:example

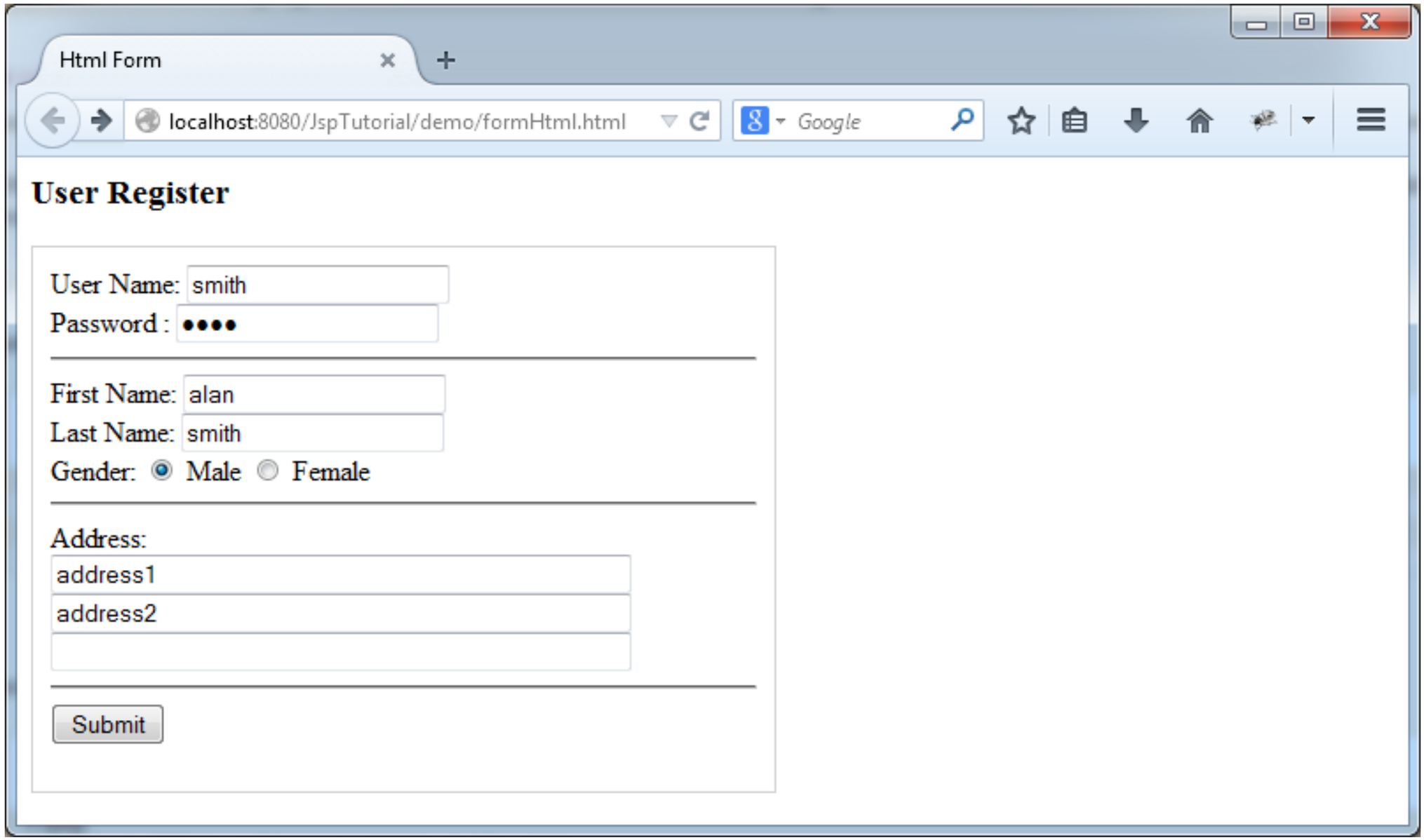

Jsp Include Directive
localhost:8080/JspTutorial/demo/includeDemo.jsp
Google
Search
My Site
Jsp tutorial for Beginners
Hi! This is Jsp Tutorial...
@Copyright mysite.com

JSP

- Form Processing
- http://localhost:8080/JspTutorial/demo/formProcessing.jsp?userName=smith&password=1234&firstName=alan&lastName=smith&gender=M&address=address1&address=address2&address=

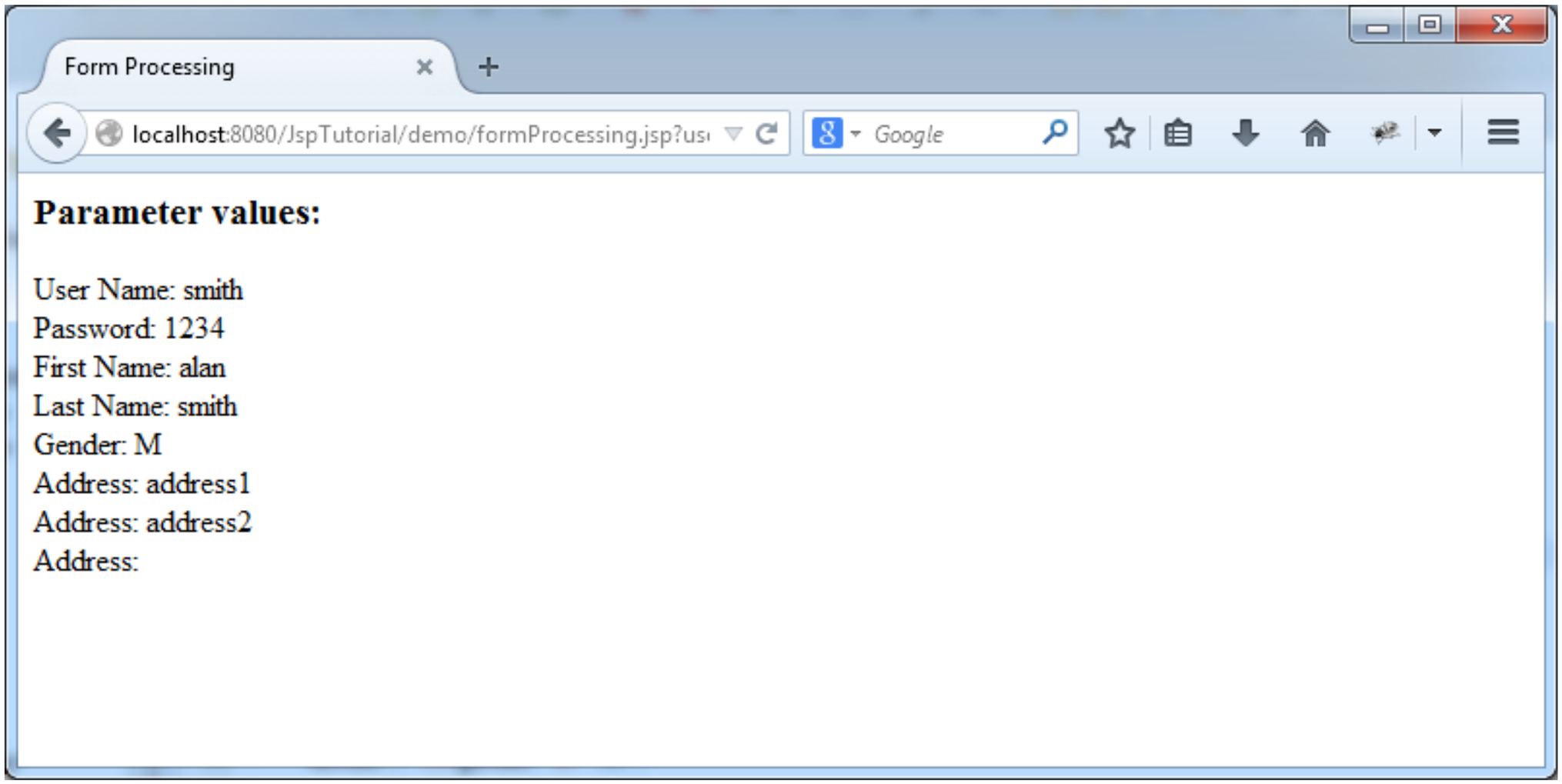
JSP

- Form Processing



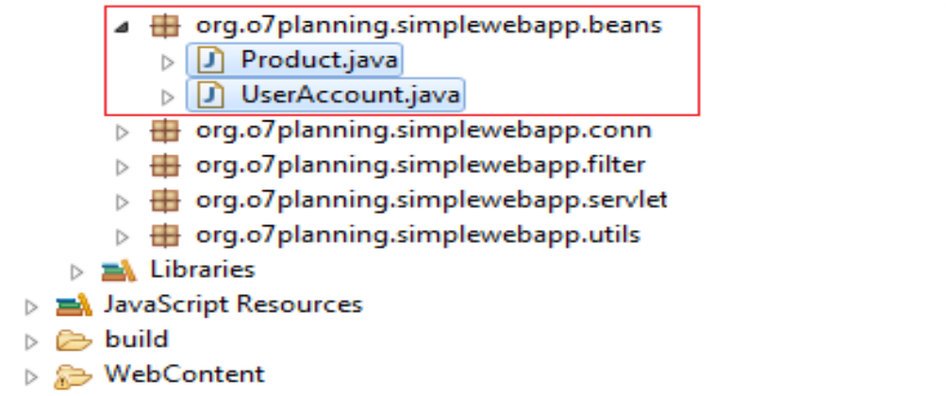
JSP

- Form Processing



Java Bean

- After the servlet processes the Java logic code, it needs to pass the result to the JSP.
- JSP only supports simple data type like String, Integer.
- We need JavaBean to support class data type
- Tutorial: http://o7planning.org/web/fe/default/en/document/72162/create-a-simple-web-application-using-servlet-jsp-and-jdbc#a812142



- **UserAccount.java**

```java
package org.o7planning.simplewebapp.beans;

public class UserAccount {

    public static final String GENDER_MALE ="M";
    public static final String GENDER_FEMALE = "F";

    private String userName;
    private String gender;
    private String password;


    public UserAccount() {

    }

    public String getUserName() {
        return userName;
    }

    public void setUserName(String userName) {
        this.userName = userName;
    }

    public String getGender() {
        return gender;
    }

    public void setGender(String gender) {
        this.gender = gender;
    }

    public String getPassword() {
        return password;
    }

    public void setPassword(String password) {
        this.password = password;
    }

}
```

JSTL

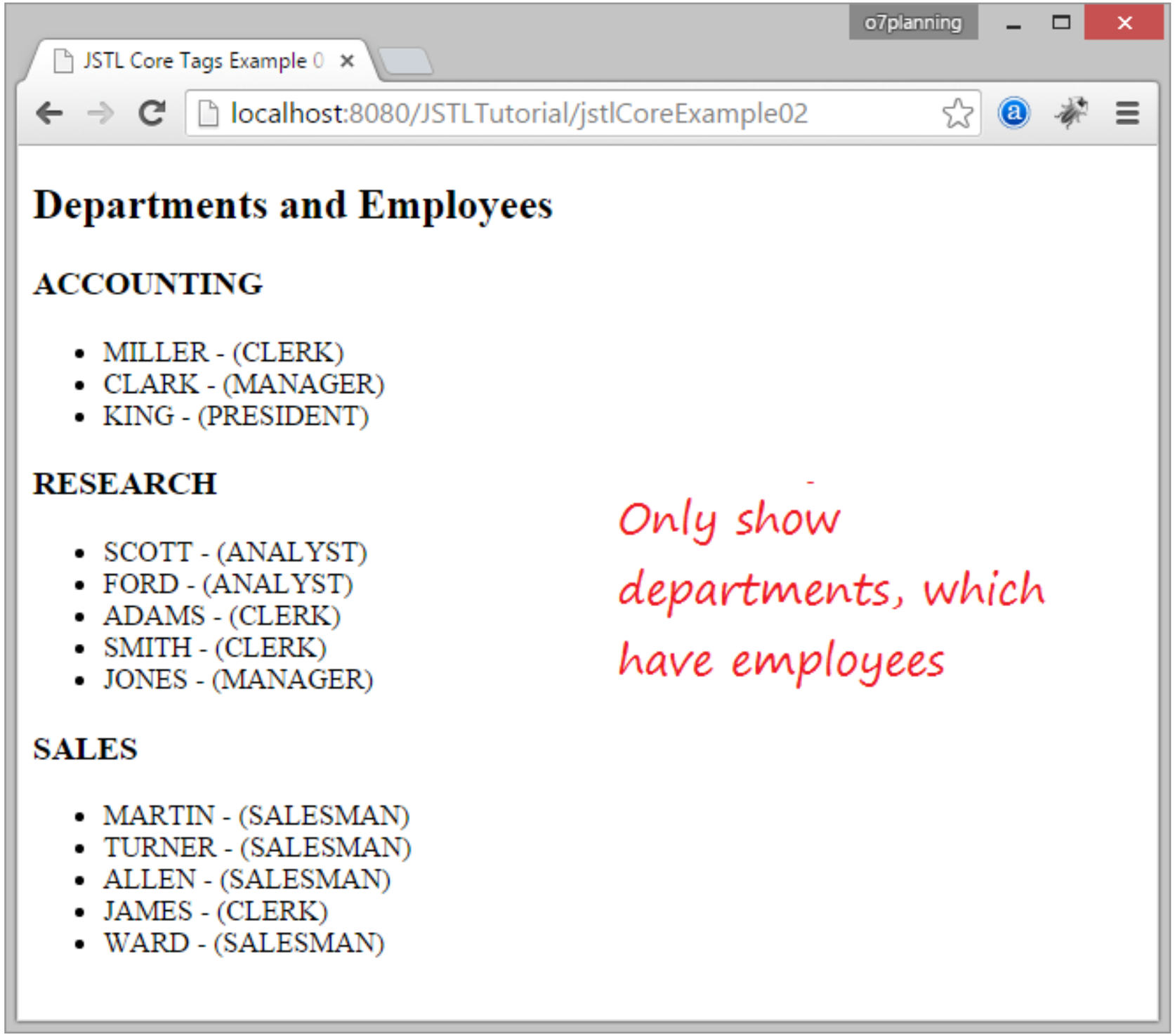
- JSTL is a collection of useful JSP tags which encapsulates core functionality common to many JSP applications.

- Tutorial: http://o7planning.org/web/fe/default/en/document/72162/create-a-simple-web-application-using-servlet-jsp-and-jdbc#a812094
- Tutorial: http://www.avajava.com/tutorials/lessons/how-do-i-use-jstl-on-my-jsps.html

JSTL

```
1   <%@ page language="java" contentType="text/html; charset=UTF-8"
2     pageEncoding="UTF-8"%>
3
4
5   <%@ taglib uri="http://java.sun.com/jsp/jstl/core" prefix="c" %>
6
7   <!DOCTYPE html>
8   <html>
9   <head>
10  <meta charset="UTF-8">
11  <title>JSTL Core Tags Example 01</title>
12  </head>
13  <body>
14
15  <h2>Departments and Employees</h2>
16
17
18  <c:forEach items="${departments}" var="dept">
19
20  <!-- Check if collection is not null or not empty -->
21  <c:if test="${not empty dept.employees}">
22       <h3>${dept.deptName}</h3>
23       <ul>
24
25         <c:forEach items="${dept.employees}" var="emp">
26             <li>
27                ${emp.empName} - (${emp.job})
28             </li>
29         </c:forEach>
30       </ul>
31   </c:if>
32
33  </c:forEach>
34
35
36  </body>
37  </html>
```

JSTL


o7planning
JSTL Core Tags Example 0
localhost:8080/JSTLTutorial/jstlCoreExample02
Departments and Employees
ACCOUNTING
MILLER - (CLERK)
CLARK - (MANAGER)
KING - (PRESIDENT)
RESEARCH
SCOTT - (ANALYST)
FORD - (ANALYST)
ADAMS - (CLERK)
SMITH - (CLERK)
JONES - (MANAGER)
SALES
MARTIN - (SALESMAN)
TURNER - (SALESMAN)
ALLEN - (SALESMAN)
JAMES - (CLERK)
WARD - (SALESMAN)
Only show departments, which have employees

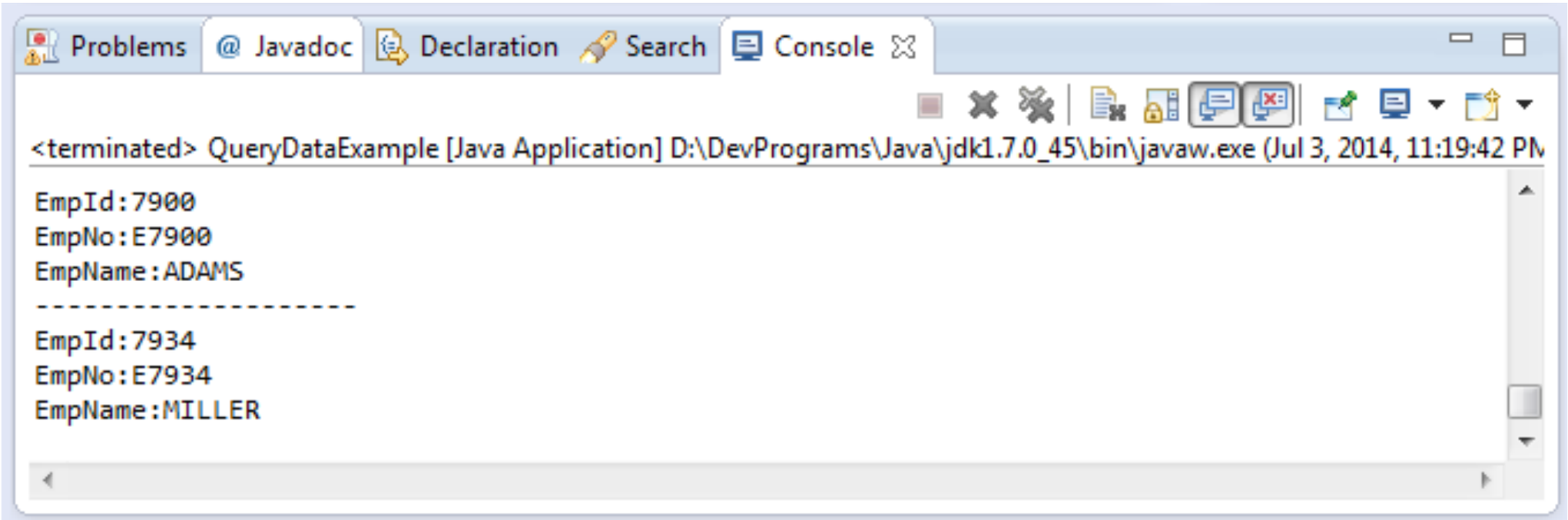
JDBC

- (**JDBC**) is a standard Java API to interact with relational databases form Java
- Connect your database to the Java Code
- Tutorial: http://o7planning.org/web/fe/default/en/document/12562/java-jdbc-tutorial

JDBC

```java
package org.o7planning.tutorial.jdbc.basic;

import java.sql.Connection;
import java.sql.ResultSet;
import java.sql.SQLException;
import java.sql.Statement;

import org.o7planning.tutorial.jdbc.ConnectionUtils;

public class QueryDataExample {

    public static void main(String[] args) throws ClassNotFoundException,
            SQLException {

        // Get Connection
        Connection connection = ConnectionUtils.getMyConnection();

        // Create statement
        Statement statement = connection.createStatement();

        String sql = "Select Emp_Id, Emp_No, Emp_Name from Employee";

        // Execute SQL statement returns a ResultSet object.
        ResultSet rs = statement.executeQuery(sql);

        // Fetch on the ResultSet
        // Move the cursor to the next record.
        while (rs.next()) {
            int empId = rs.getInt(1);
            String empNo = rs.getString(2);
            String empName = rs.getString("Emp_Name");
            System.out.println("--------------------");
            System.out.println("EmpId:" + empId);
            System.out.println("EmpNo:" + empNo);
            System.out.println("EmpName:" + empName);
        }

        // Close connection.
        connection.close();
    }

```

JDBC


Problems
Javadoc
Declaration
Search
Console
<terminated> QueryDataExample [Java Application] D:\DevPrograms\Java\jdk1.7.0_45\bin\javaw.exe (Jul 3, 2014, 11:19:42 PM
EmpId:7900
EmpNo:E7900
EmpName:ADAMS

EmpId:7934
EmpNo:E7934
EmpName:MILLER

Connecting it all together

Connecting it all together

- Our Database: Product table

	code character varying(20)	name character varying(128)	price double precision
1	P001	Java Core	100
2	P002	C# Core	90

Connecting it all together

- Our JDBC: query Product

```
public static List<Product> queryProduct(Connection conn) throws SQLException {
    String sql = "Select a.Code, a.Name, a.Price from Product a ";

    PreparedStatement pstm = conn.prepareStatement(sql);

    ResultSet rs = pstm.executeQuery();
    List<Product> list = new ArrayList<Product>();
    while (rs.next()) {
        String code = rs.getString("Code");
        String name = rs.getString("Name");
        float price = rs.getFloat("Price");
        Product product = new Product();
        product.setCode(code);
        product.setName(name);
        product.setPrice(price);
        list.add(product);
    }
    return list;
}
```

Connecting it all together

- Our Servlet: Product Servlet

```
@WebServlet(urlPatterns = { "/productList" })
public class ProductListServlet extends HttpServlet {
    private static final long serialVersionUID = 1L;

    public ProductListServlet() {
        super();
    }

    @Override
    protected void doGet(HttpServletRequest request, HttpServletResponse response)
            throws ServletException, IOException {
        Connection conn = MyUtils.getStoredConnection(request);

        String errorString = null;
        List<Product> list = null;
        try {
            list = DBUtils.queryProduct(conn);
        } catch (SQLException e) {
            e.printStackTrace();
            errorString = e.getMessage();
        }

        // Store info in request attribute, before forward to views
        request.setAttribute("errorString", errorString);
        request.setAttribute("productList", list);


        // Forward to /WEB-INF/views/productListView.jsp
        RequestDispatcher dispatcher = request.getServletContext()
                .getRequestDispatcher("/WEB-INF/views/productListView.jsp");
        dispatcher.forward(request, response);
    }
```

Connecting it all together

- Our JSP: Product view JSP

```
<html>
 <head>
    <meta charset="UTF-8">
    <title>Product List</title>
 </head>
 <body>

    <jsp:include page="_header.jsp"></jsp:include>
    <jsp:include page="_menu.jsp"></jsp:include>

    <h3>Product List</h3>

    <p style="color: red;">${errorString}</p>

    <table border="1" cellpadding="5" cellspacing="1" >
       <tr>
          <th>Code</th>
          <th>Name</th>
          <th>Price</th>
          <th>Edit</th>
          <th>Delete</th>
       </tr>
       <c:forEach items="${productList}" var="product" >
          <tr>
             <td>${product.code}</td>
             <td>${product.name}</td>
             <td>${product.price}</td>
             <td>
                <a href="editProduct?code=${product.code}">Edit</a>
             </td>
             <td>
                <a href="deleteProduct?code=${product.code}">Delete</a>
             </td>
          </tr>
       </c:forEach>
    </table>
```

JSTL

Connecting it all together

- Result:

My Site

Home | Product List | My Account Info | Login

Product List

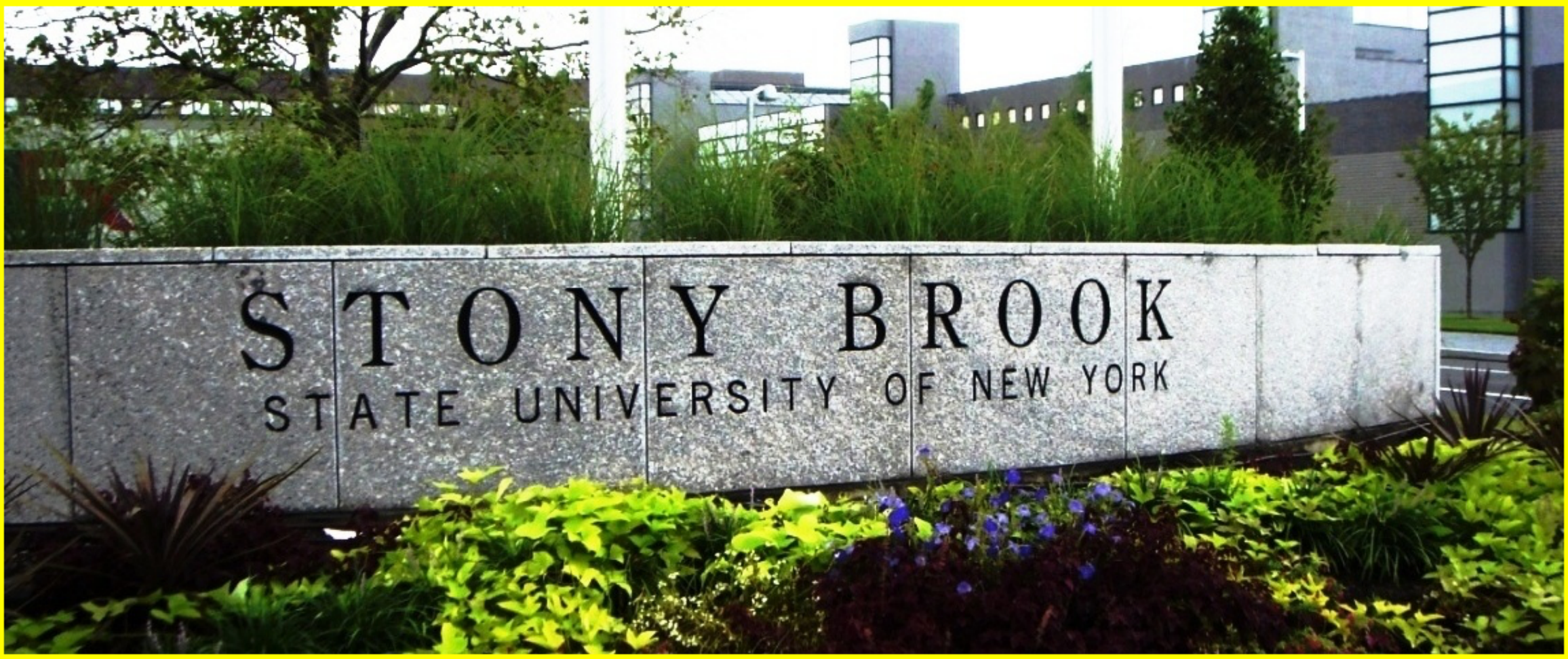
Code	Name	Price	Edit	Delete
P001	Java Core	100.0	Edit	Delete
P002	C# Core	90.0	Edit	Delete

Create Product

Your Demo War File

User Clicks Login

Student Registration System



User ID : stud1 **Password:** ••••• Log In Register

User Login

HTML code for mainpage

```
<form name="myForm" action="login" method="post">

<span style="font-size: 10pt">User ID :</span>
<input id="Text1" name="username" type="text" />
  <span style="font-size: 10pt">Password:</span>
<input id="Password1" name="userpasswd" type="password" />
 
<input id="Button1" style="width: 70px" type="button" value="Log In" onclick="return Button1_onclick()" />
 
<input id="Button2" style="width: 70px" type="button" value="Register" onclick="return Button2_onclick()" /><br />
<br />
</form>
```

User Login

Web. XML

```
  <servlet-name>login</servlet-name>
  <servlet-class>loginServlet</servlet-class>
</servlet>

<servlet-mapping>
  <servlet-name>login</servlet-name>
  <url-pattern>/login</url-pattern>
</servlet-mapping>
```

User Login

loginServlet.java do post:

```
String username = request.getParameter("username");
String userpasswd = request.getParameter("userpasswd");
String mysJDBCDriver = "com.mysql.jdbc.Driver";
String mysURL = "jdbc:mysql://127.0.0.1:3306/cse305";
String mysUserID = "root";
String mysPassword = "1234";

HttpSession session = request.getSession();
session.putValue("login", "");
if ((username != null) && (userpasswd != null)) {
    if (username.trim().equals("") || userpasswd.trim().equals("")) {
        response.sendRedirect("index.htm");
    } else {
        // code start here
        java.sql.Connection conn = null;
        try {
            Class.forName(mysJDBCDriver).newInstance();
            java.util.Properties sysprops = System.getProperties();
            sysprops.put("user", mysUserID);
            sysprops.put("password", mysPassword);

            // connect to the database
            conn = java.sql.DriverManager.getConnection(mysURL, sysprops);
            System.out.println("Connected successfully to database using JConnect");

            conn.setAutoCommit(false);
            java.sql.Statement stmt1 = conn.createStatement();
            java.sql.ResultSet rs = stmt1.executeQuery(
                    " select * from Student where Id='" + username + "' and Pswd='" + userpasswd + "'");
            if (rs.next()) {
                // login success
                session.putValue("login", username);
                System.out.println("RequestDispatcher rd= context.getRequestDispatcher;");
                ServletContext context = getServletContext();
                RequestDispatcher rd = context.getRequestDispatcher("/studentinfo");
                rd.forward(request, response);
```

User Login

Web. XML

```
<servlet>
  <servlet-name>studentinfo</servlet-name>
  <servlet-class>StudentInfoServlet</servlet-class>
</servlet>

<servlet-mapping>
  <servlet-name>studentinfo</servlet-name>
  <url-pattern>/studentinfo</url-pattern>
</servlet-mapping>
```

Student Information

Studentinfoservlet.java do post:

```
69          java.sql.Statement stmt1 = conn.createStatement();
70          java.sql.ResultSet rs = stmt1
71                  .executeQuery("select Course.CrsCode,Course.CrsName,Course.DeptID,Professor.Name,"
72                          + "Transcript.Grade from Course, Professor,Transcript where Course.CrsCode=Transcript.CrsCode "
73                          + "and Professor.Id=Course.InsNo and Transcript.StudId='"
74                          + stuId + "'");
75          String strGrade;
76
77          while (rs.next()) {
78              strGrade = rs.getString(5);
79              if (rs.getString(5).trim().equals("-1")) {
80                  strGrade = "N/A";
81              }
82              DataTypeB data = new DataTypeB();
83              data.setItem1(rs.getString(1));
84              data.setItem2(rs.getString(2));
85              data.setItem3(rs.getString(3));
86              data.setItem4(rs.getString(4));
87              data.setItem5(strGrade);
88              System.out.println(rs.getString(1));
89              System.out.println(rs.getString(2));
90              list.add(data);
91          }
92      } catch (Exception e) {
93          e.printStackTrace();
94      } finally {
95          try {
96              conn.close();
97          } catch (Exception ee) {
98          };
99      }
100     System.out.println("length:"+list.size());
101     session.setAttribute("list2", list);
102     RequestDispatcher view = request.getRequestDispatcher("StudentInformation.jsp");
103     view.forward(request, response);
```

Student Information

StudentInformation.jsp and usage of JSTL

```
21                  <table border="8" id="TABLE1" onclick="return TABLE1_onclick()">
22                  <tr>
23                    <td style="width: 84px">
24                        <span style="font-size: 10pt">
25                        Course Code</span></td>
26                    <td style="width: 187px">
27                        <span style="font-size: 10pt">Course Name</span></td>
28                      <td style="width: 74px">
29                          <span style="font-size: 10pt">Department</span></td>
30                      <td>
31                          <span style="font-size: 10pt">Professor</span></td>
32                      <td style="width: 7px">
33                          <span style="font-size: 10pt">Grade</span></td>
34                      <td style="width: 7px">
35                          <span style="font-size: 10pt">Oper</span></td>
36                  </tr>
37
38  <%@taglib prefix="c" uri="http://java.sun.com/jsp/jstl/core" %>
39
40  <c:forEach items="${list2}" var="item">
41          <% String stuId = "" + session.getValue("login"); %>
42          <tr>
43                      <td style="width: 84px">
44                          <span style="font-size: 10pt">${item.item1}</span></td>
45                      <td style="width: 187px">
46                          <span style="font-size: 10pt">${item.item2}</span></td>
47                      <td style="width: 74px">
48                          <span style="font-size: 10pt">${item.item3}</span></td>
49                      <td>
50                          <span style="font-size: 10pt">${item.item4}</span></td>
51                      <td style="width: 7px">
52                          <span style="font-size: 10pt">${item.item5}</span></td>
53                      <td>
54                          <form name="myForm" action="delcourse" method="post">
```

Result of User Login

Result HTML

This is the table for the courses in which you enrolled.
You can delete the ones which do not have a grade.
And you can search new courses by click the "Search" button below.
You may engage in at most 10 courses.

Course Code	Course Name	Department	Professor	Grade	Oper
101	Algorithms	CSE	Alice	23	Delete
103	Cryptography	CSE	Cindy	63	Delete

Summary

- Eclipse: a SDK to develop the web project
- Apache Tomcat: application server, renders web page
- Java Servlet: separate the logic code (Java) from the HTML
- Java Servlet Filter:
    - To intercept requests from a client before they access a resource at back end
        - To manipulate responses from server before they are sent back to the client
- JSP: HTML page with Java code
- Java Bean: used to support class type data
- JSTL: a collection of useful JSP tags which encapsulates core functionality common to many JSP applications
- JDBC: connect your database to Java

Attention!!

Attention

- Run the Demo war file uploaded to BlackBoard.
- Please go through this tutorial:
  - http://o7planning.org/web/fe/default/en/document/72162/create-a-simple-web-application-using-servlet-jsp-and-jdbc#a812142

Thank you

- Try to figure out each component independently and help each other to integrate these components into the project.