GET and POST Requests

Request & Response Objects

Agenda

1 doGet and doPost methods

2 HttpServletRequest & HttpServletResponse

3 Create & Deploy Simple application

4 Accessing database

Objectives

At the end of this module, you will be able to:

- Describe the role of HTTP Servlet in Web Programming
- Describe and use the Servlet Life Cycle methods appropriately
- Process parameters from HTML forms
- Establish Database Connectivity through servlets

The doGet and doPost methods

- Methods doGet() and doPost() in HttpServlet class receives appropriate client request, and formats a response using 2 arguments
  - An HttpServletRequest object - encapsulates data from the client
  - An HttpServletResponse object - encapsulates response to the client

- Both these objects are created by the servlet container

- Usage:

  public void doPost(HttpServletRequest req, HttpServletResponse res) throws IOException, ServletException

  public void doGet(HttpServletRequest req, HttpServletResponse res) throws IOException, ServletException

HttpServletRequest interface

- HttpServletRequest
  - The HttpServletRequest object incorporates any communication from client to servlet
  - Provides methods that allow you to retrieve incoming information
    - For example: HTTP request headers, form data, or a client's hostname

- Methods to read parameters from a form
  - getParameter(String pname)
  - getParameterNames()
  - getParameterValues(String pname)

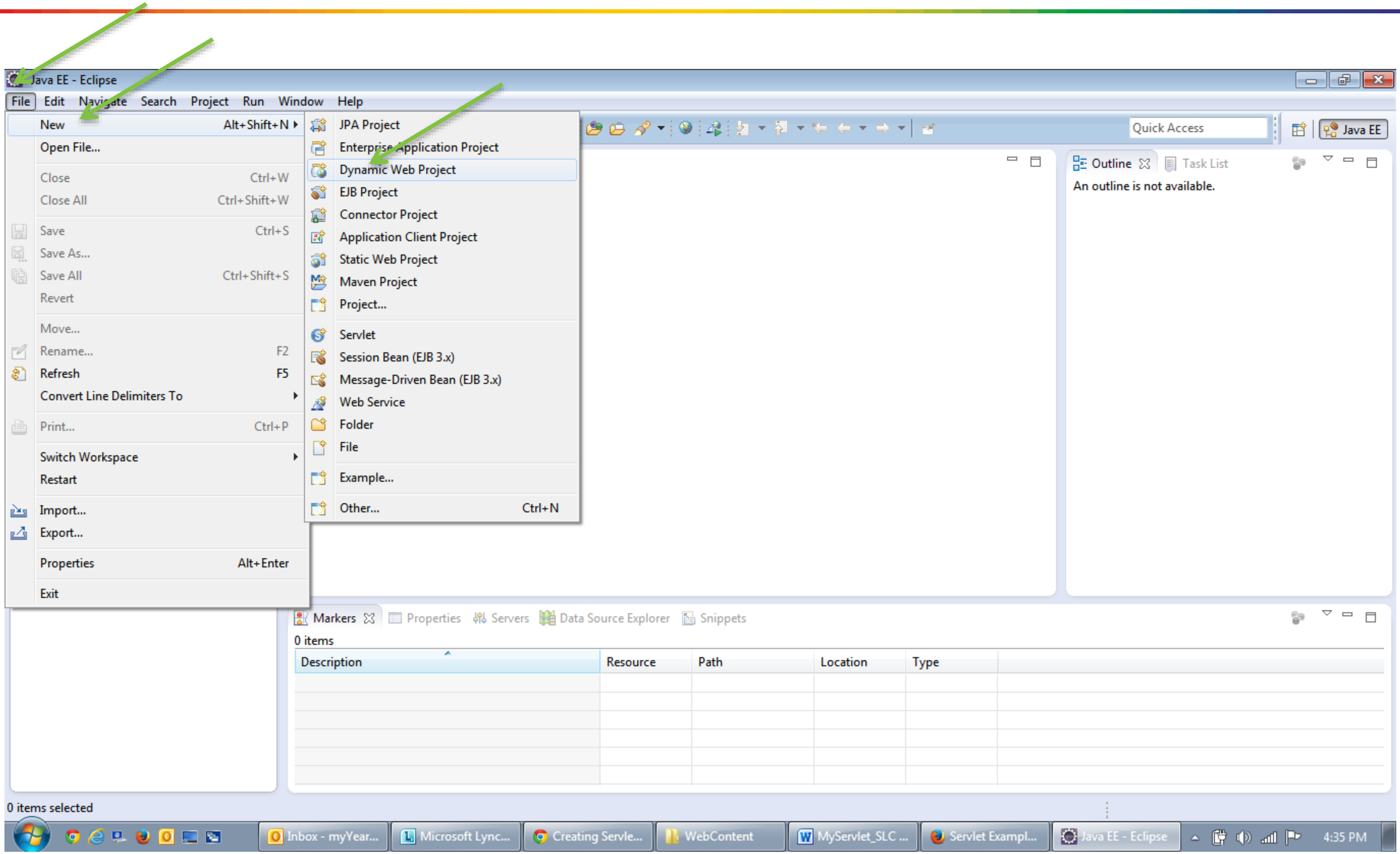
HttpServletResponse interface (Contd.).

- HttpServletResponse
  - The HttpServletResponse object incorporates any communication from servlet to client
  - Allows you to specify outgoing information
    - For example: response headers and HTTP status codes
  - Also enables you to obtain a PrintWriter object for writing output back to the client

- Methods:
  - getWriter
  - setContentType
  - sendRedirect

First Servlet Example - Demo

Create the dynamic web project

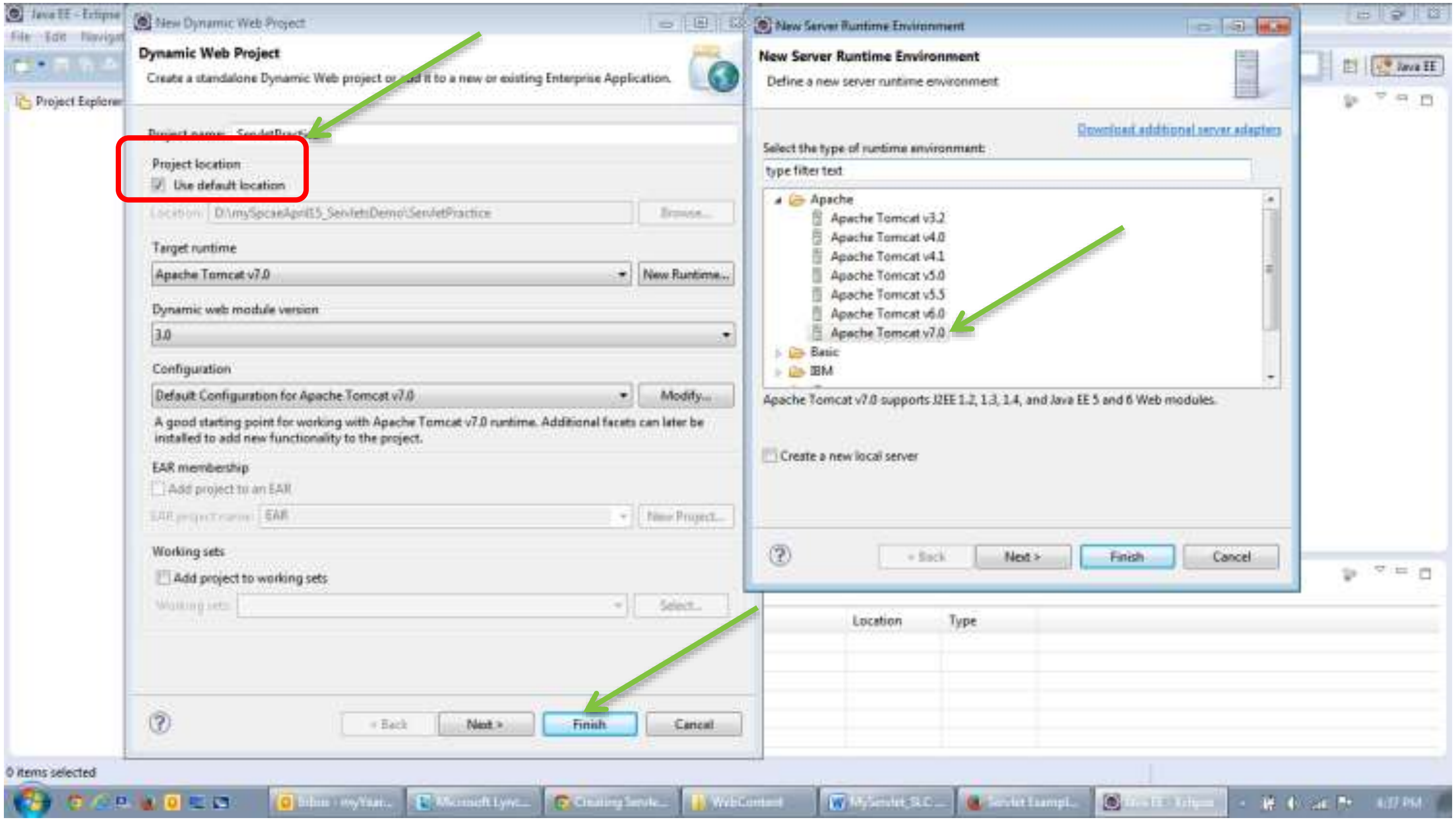


An outline is not available.

0 items

0 items selected

Configure tomcat server in Eclipse? (One time Requirement)



New Dynamic Web Project

Dynamic Web Project

Create a standalone Dynamic Web project or add it to a new or existing Enterprise Application.

Project name: ServletPractice

Project location

☑ Use default location

Location: D:\mySpcaeApril15_ServletsDemo\ServletPractice

Target runtime: Apache Tomcat v7.0

Dynamic web module version: 3.0

Configuration: Default Configuration for Apache Tomcat v7.0

A good starting point for working with Apache Tomcat v7.0 runtime. Additional facets can later be installed to add new functionality to the project.

EAR membership

□ Add project to an EAR

EAR project name: EAR

Working sets

□ Add project to working sets

Working sets:

New Server Runtime Environment

New Server Runtime Environment

Define a new server runtime environment

Download additional server adapters

Select the type of runtime environment:

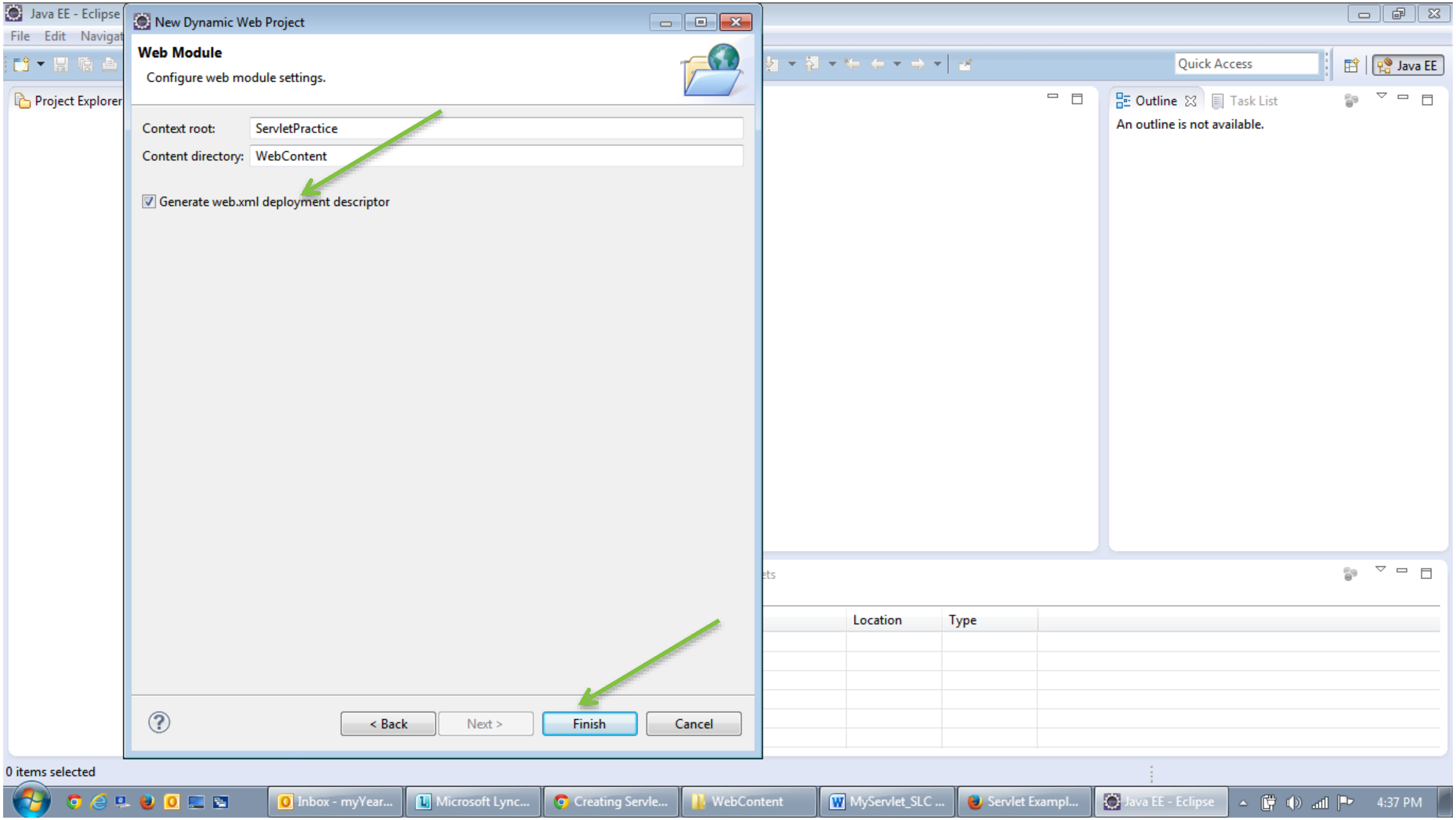
type filter text

- Apache
  - Apache Tomcat v3.2
  - Apache Tomcat v4.0
  - Apache Tomcat v4.1
  - Apache Tomcat v5.0
  - Apache Tomcat v5.5
  - Apache Tomcat v6.0
  - Apache Tomcat v7.0
- Basic
- IBM

Apache Tomcat v7.0 supports J2EE 1.2, 1.3, 1.4, and Java EE 5 and 6 Web modules.

□ Create a new local server

Enable web.xml deployment descriptor



New Dynamic Web Project

Web Module

Configure web module settings.

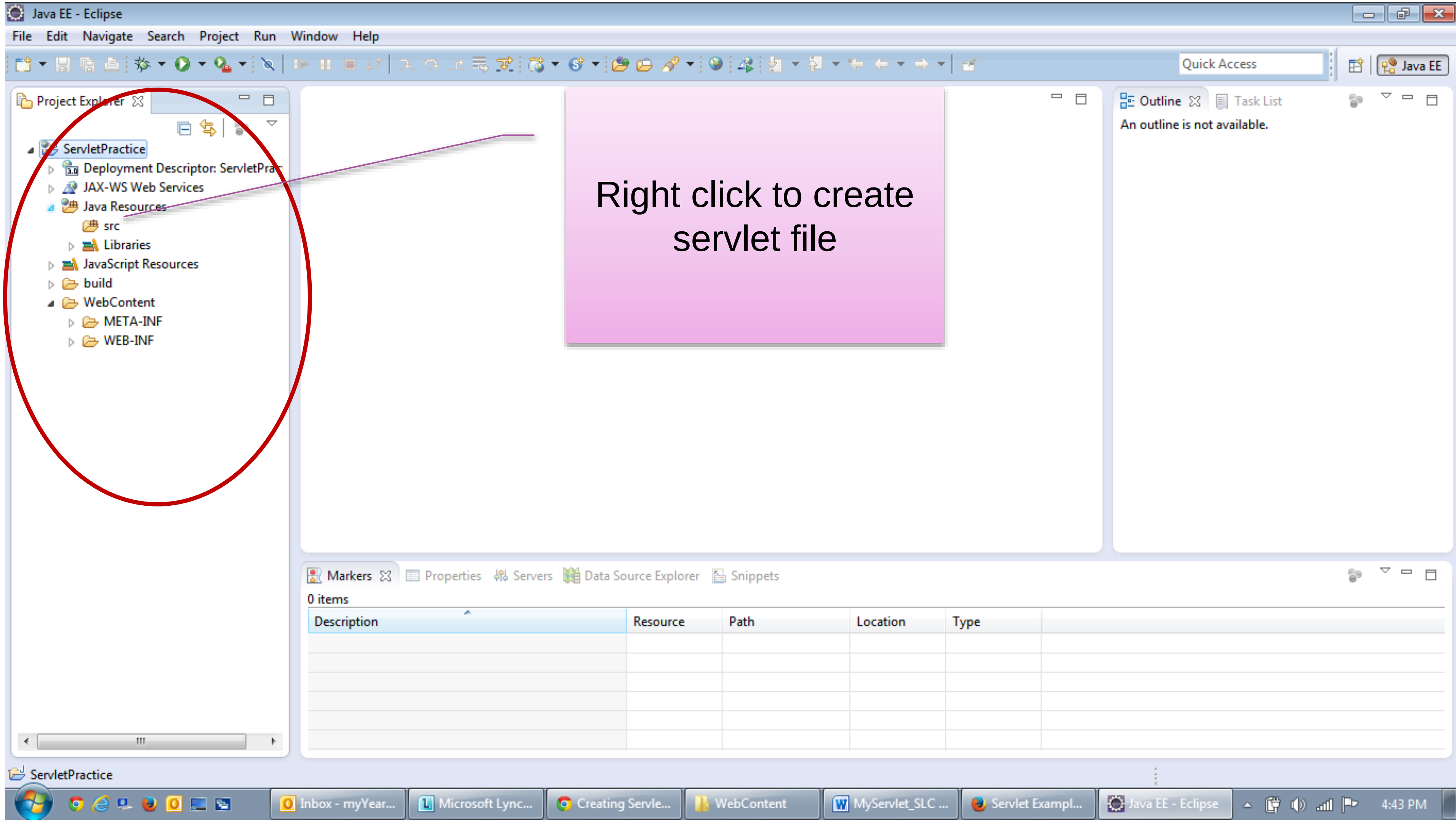
Context root: ServletPractice

Content directory: WebContent

☑ Generate web.xml deployment descriptor

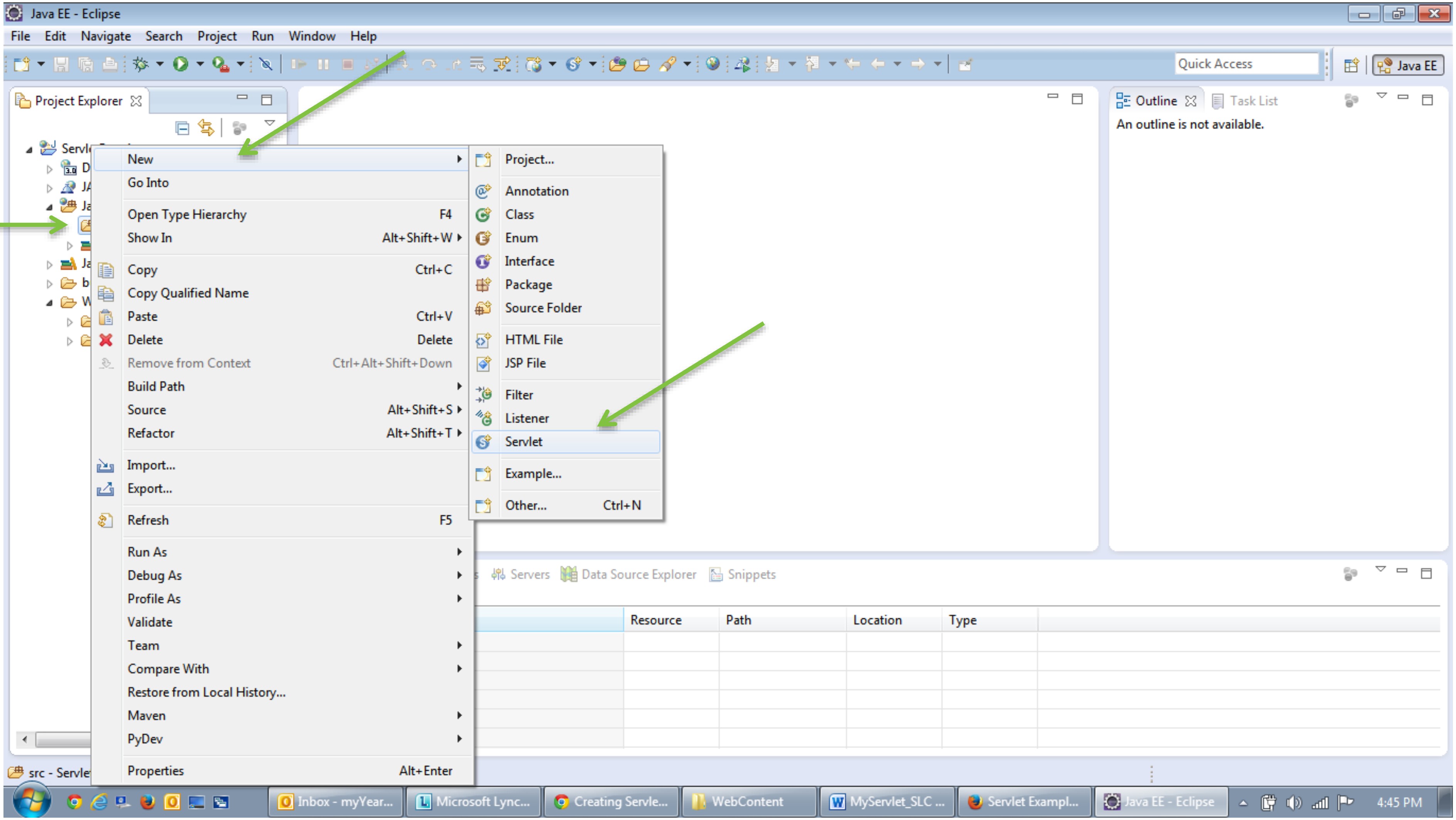
An outline is not available.

Create the servlet

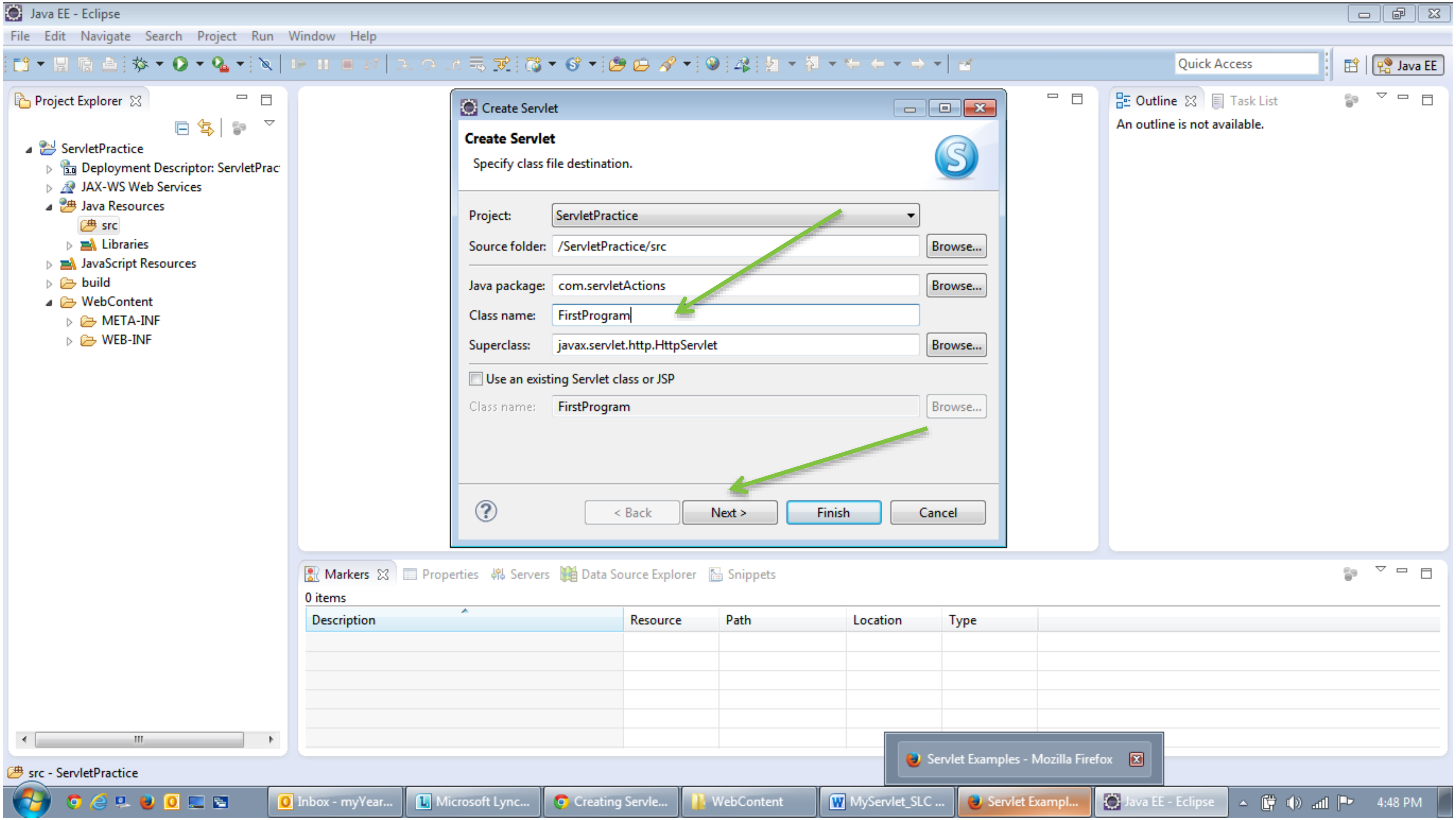


Right click to create servlet file

Select the servlet option



Type class name



Create Servlet

Specify class file destination.

Project: ServletPractice

Source folder: /ServletPractice/src

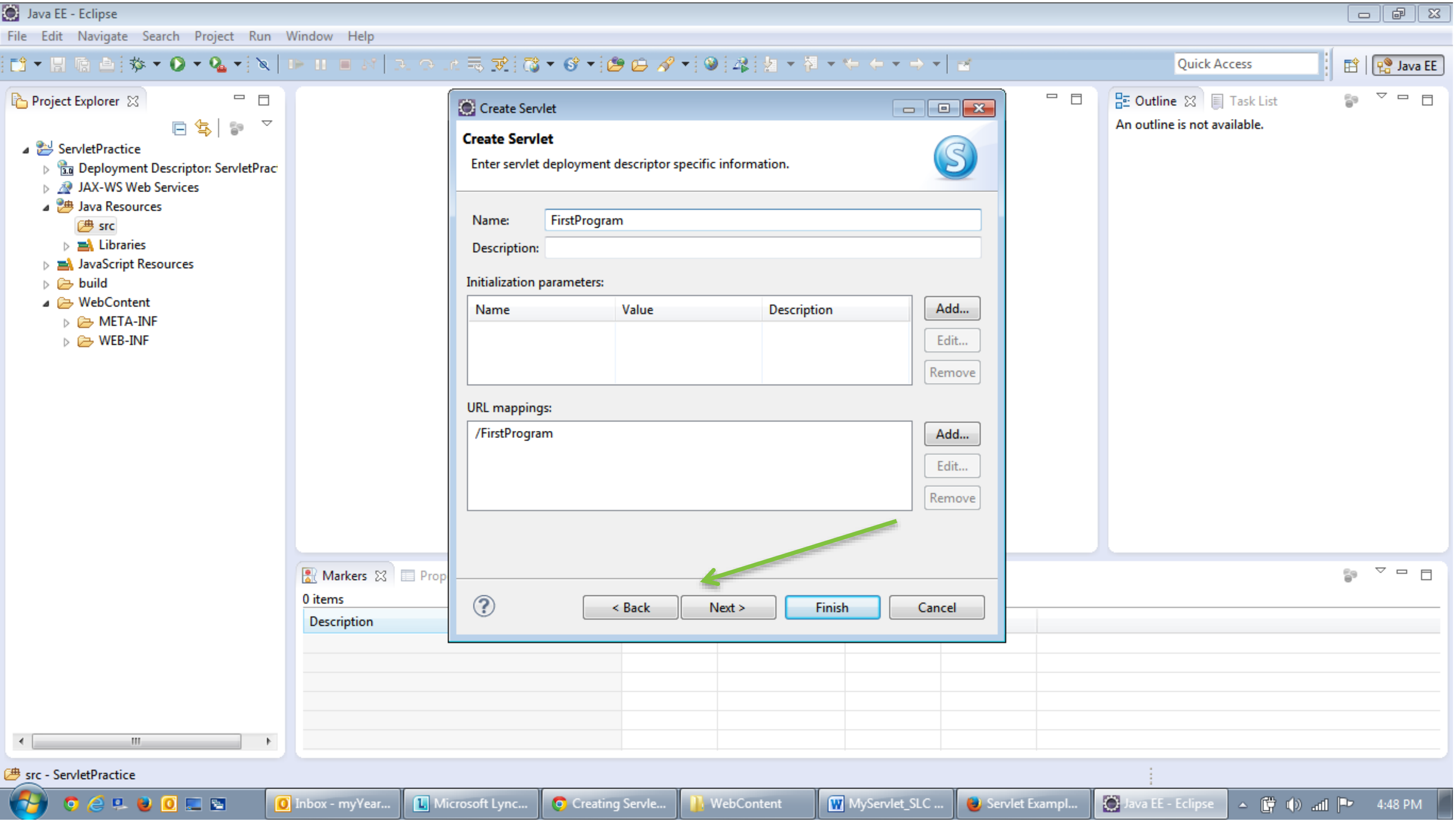
Java package: com.servletActions

Class name: FirstProgram

Superclass: javax.servlet.http.HttpServlet

□ Use an existing Servlet class or JSP

Class name: FirstProgram



Create Servlet

Enter servlet deployment descriptor specific information.

Name: FirstProgram

Description:

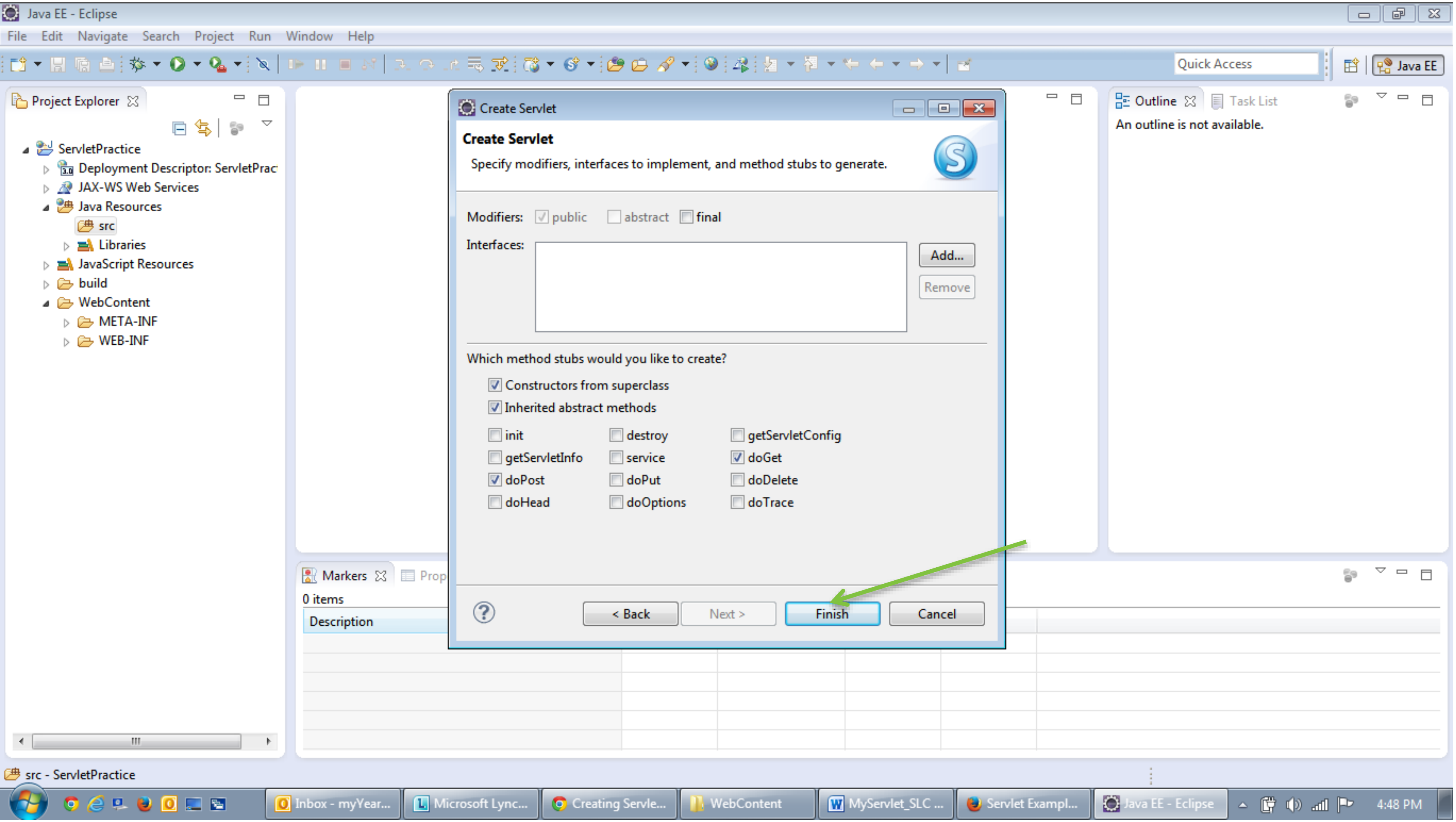
Initialization parameters:

Name	Value	Description

URL mappings:

/FirstProgram

Select Finish



Create Servlet

Specify modifiers, interfaces to implement, and method stubs to generate.

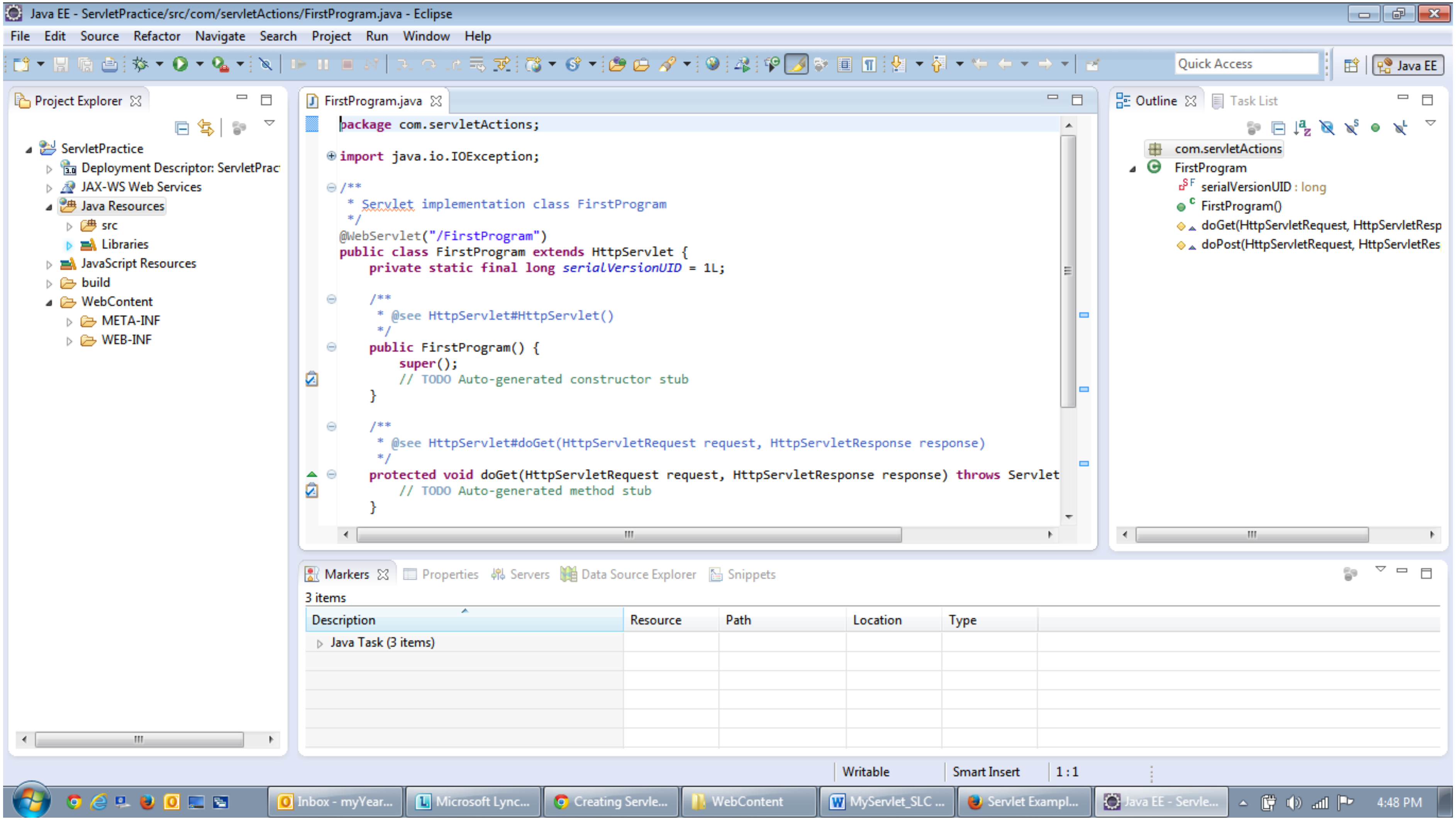
Modifiers: ☑ public □ abstract □ final

Interfaces:

Which method stubs would you like to create?

☑ Constructors from superclass
☑ Inherited abstract methods

□ init □ destroy □ getServletConfig
□ getServletInfo □ service ☑ doGet
☑ doPost □ doPut □ doDelete
□ doHead □ doOptions □ doTrace

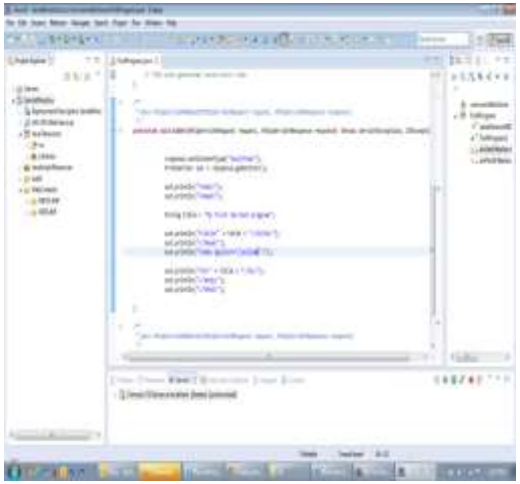


```java
package com.servletActions;

import java.io.IOException;

/**
 * Servlet implementation class FirstProgram
 */
@WebServlet("/FirstProgram")
public class FirstProgram extends HttpServlet {
    private static final long serialVersionUID = 1L;

    /**
     * @see HttpServlet#HttpServlet()
     */
    public FirstProgram() {
        super();
        // TODO Auto-generated constructor stub
    }

    /**
     * @see HttpServlet#doGet(HttpServletRequest request, HttpServletResponse response)
     */
    protected void doGet(HttpServletRequest request, HttpServletResponse response) throws Servlet
        // TODO Auto-generated method stub
    }
```

3 items

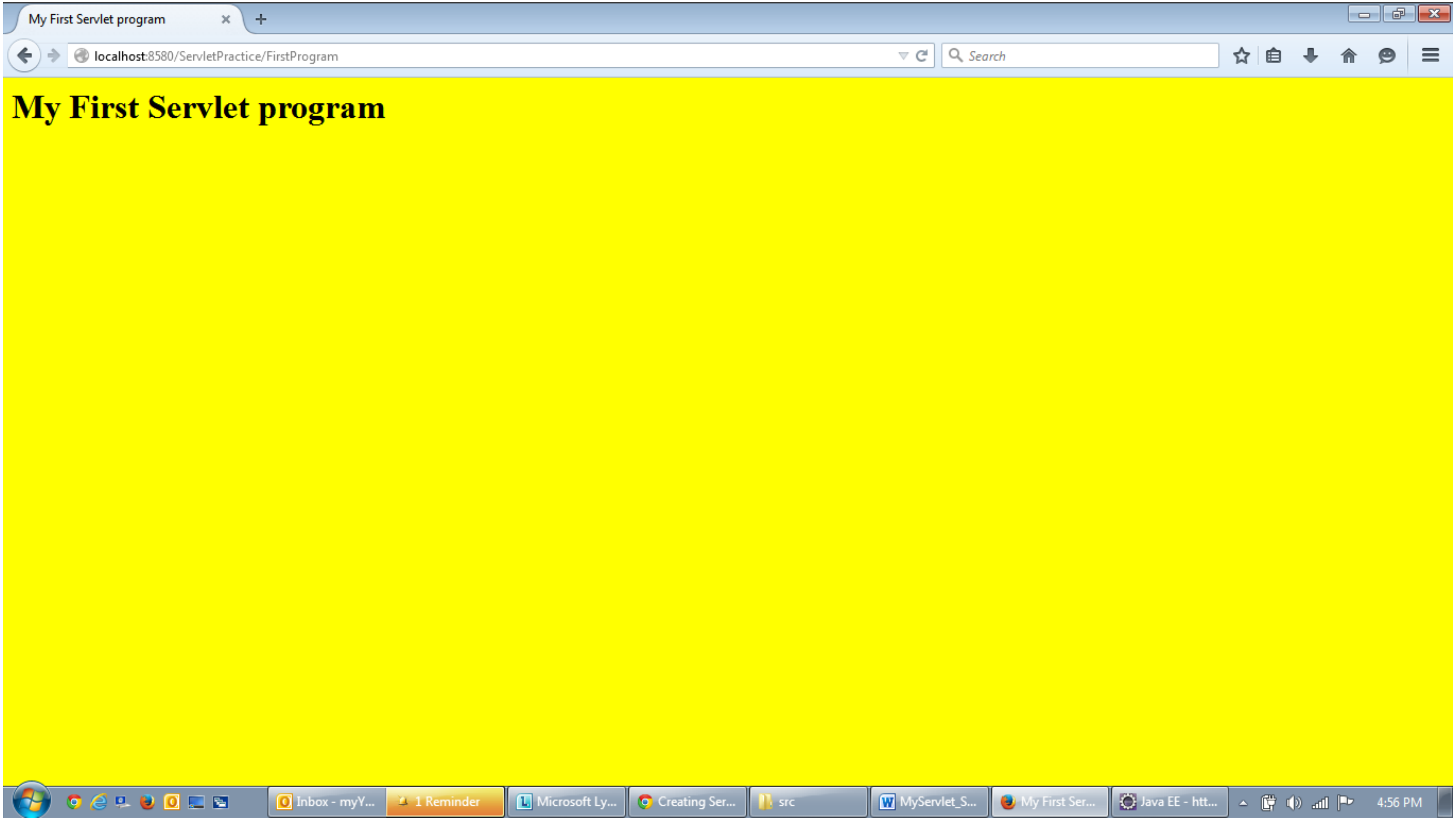
Description	Resource	Path	Location	Type
Java Task (3 items)				

Source Code for First_Servlet_Program Example



```
import java.io.*;
import javax.servlet.*;
import javax.servlet.http.*;

public class First_Servlet_Program extends HttpServlet {

    public void doGet(HttpServletRequest request, HttpServletResponse
response)
    throws IOException, ServletException
    {
        response.setContentType("text/html");
        PrintWriter out = response.getWriter();
        out.println("<html>");
        out.println("<head>");
        out.println("<title> First Servlet Program!</title>");
        out.println("</head>");
        out.println("<body>");
        out.println("<h1> Welcome To First Servlet Program !!!</h1>");
        out.println("</body>");
        out.println("</html>");
    }
}
```

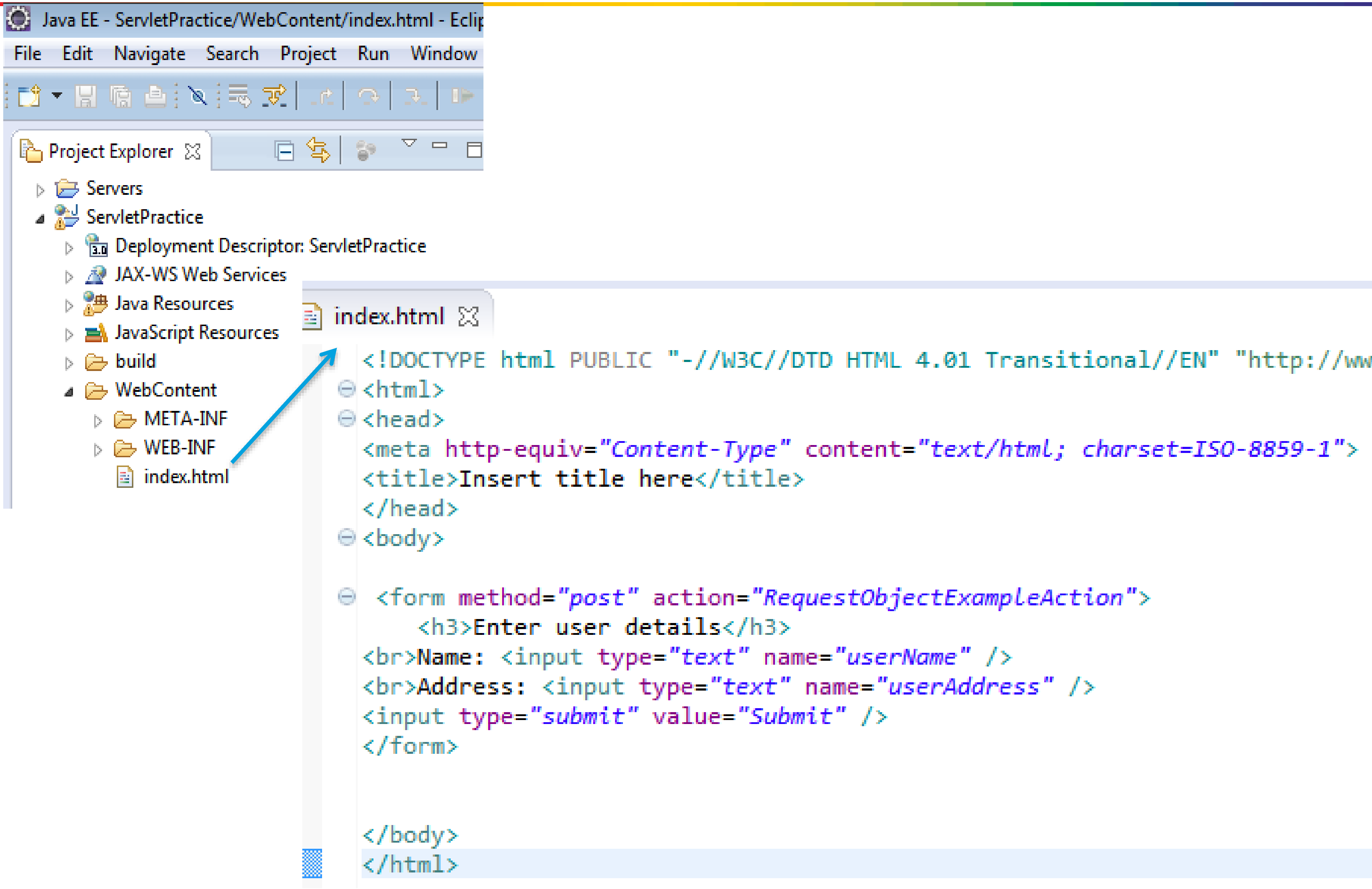
Output response



My First Servlet program

Request Object

Create HTML file – index.html



```html
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN" "http://ww
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=ISO-8859-1">
<title>Insert title here</title>
</head>
<body>

 <form method="post" action="RequestObjectExampleAction">
    <h3>Enter user details</h3>
<br>Name: <input type="text" name="userName" />
<br>Address: <input type="text" name="userAddress" />
<input type="submit" value="Submit" />
</form>


</body>
</html>
```

Request object Parameters – Form Data

```
protected void doPost(HttpServletRequest request, HttpServletResponse response) throws ServletException, 
IOException {

 PrintWriter out = response.getWriter();

 response.setContentType("text/html");
        out.println("<html>");
        out.println("<head>");
    out.println("<title> Request Object example </title>");
        out.println("</head>");

        out.println("<body bgcolor=\"yellow\">");
                out.println("<h1>"+"Request object Parameters: " + "</h1>"+"<br>");
        out.println("<h2>" + "User Details" + "</h2>");
        Enumeration<String> reqParams = request.getParameterNames();

            while (reqParams.hasMoreElements()) {
                String name = (String)reqParams.nextElement();
                String value = request.getParameter(name);
                out.println("<b> "+name + "</b>"+ " = " +value+"<br>");
            }

        out.println("The Name you entered was: " + request.getParameter("userName")+"<br>");
        out.println("The Address you entered was: " + request.getParameter("userAddress"));

        out.println("</body>");
        out.println("</html>");

}
```

Request object Info

```
protected void doPost(HttpServletRequest request, HttpServletResponse response)
throws ServletException, IOException {

 PrintWriter out = response.getWriter();

 response.setContentType("text/html");
        out.println("<html>");
        out.println("<head>");
    out.println("<title> Request Object example </title>");
        out.println("</head>");

        out.println("<body bgcolor=\"yellow\">");

        out.println("<h1>"+"Request object Info: " + "</h1>"+"<br>");
        out.println("<b>Method:</b> "+ request.getMethod()+"<br>");
        out.println("<b>Request URI: </b>"+ request.getRequestURI()+"<br>");
        out.println("<b>Protocol: </b>" +  request.getProtocol()+"<br>");
        out.println("<b>PathInfo: </b>" +  request.getPathInfo()+"<br>");
        out.println("<b>Remote Address: </b>" +  request.getRemoteAddr()+"<br>");

        out.println("</body>");
        out.println("</html>");

}
```

Request object Headers details

```
protected void doPost(HttpServletRequest request, HttpServletResponse response) throws
ServletException, IOException {

 PrintWriter out = response.getWriter();

 response.setContentType("text/html");
        out.println("<html>");
        out.println("<head>");
        out.println("<title> Request Object example </title>");
        out.println("</head>");

        out.println("<body bgcolor=\"yellow\">");

        out.println("<h1>"+"Request object Headers details: " + "</h1>"+"<br>");
        Enumeration<String> reqHeaders = request.getHeaderNames();
        out.println("<table border=0>");

        while (reqHeaders.hasMoreElements()) {
            String name = (String)reqHeaders.nextElement();
            String value = request.getHeader(name);
            out.println("<tr><td bgcolor=\"#CCCCCC\">");
            out.println("<b>" + name + "</b>" +" = " + value);
            out.println("</td><td>");
            out.println("</td></tr>");
        }
        out.println("</table>");

        out.println("</body>");
        out.println("</html>");

}
```

Handling Form Data

Handling Form Data

- Write a Servlet that retrieves form parameters from the HTML form

- Simpleform.html

```
<html>
<head><title>Simple Form</title></head>
<body>
    <form method=“post” action=“SimpleFormServlet”>
        <h3>Enter user details</h3>
        <br>Name: <input type=“text” name=“userName” />
        <br>Address: <input type=“text” name=“userAddress” />
        <input type=“submit” value=“Submit” />
    </form>
</body>
```

The getParameter() method

- Syntax: public String getParameter(String name)

- To get request parameters sent as an extra information with the request, invoke getParameter method of ServletRequest (HttpServletRequest extends ServletRequest)
  - Provide parameter name as an argument
  - Returns a string that contains URL-decoded value of first occurrence of that parameter name
    - If parameter exists but has no value, then an empty string is returned
    - If parameter does not exist, then null is returned

- Use this method when you are sure the parameter has only one value

- Example: String name = request.getParameter("userName");

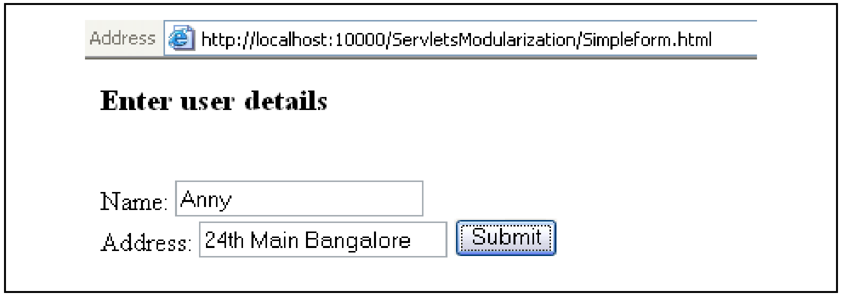
Example Servlet: Handling Form Data

- SimpleFormServlet's doPost method retrieves request parameters such as user’s name and address having a single value from the form

```
public class SimpleFormServlet  extends HttpServlet {
  public void doPost(HttpServletRequest request, HttpServletResponse response)
    throws IOException, ServletException {
    response.setContentType("text/html");
    PrintWriter out = response.getWriter();
    String name = request.getParameter("userName");
    String address = request.getParameter("userAddress");
    out.println("<html>");
    out.println("<h1>" + “User Details" + "</h1>");
    out.println("<p>The Name you entered was: " + name + "</p>");
    out.println("<p> The Address you entered was: " + address + "</p>");
   }
}
```

Demo for Handling Simple Form Data

- Demonstrate a SimpleFormServlet with a doPost method that retrieves request parameters such as user’s name and address having a single value from the form (getParameter method)

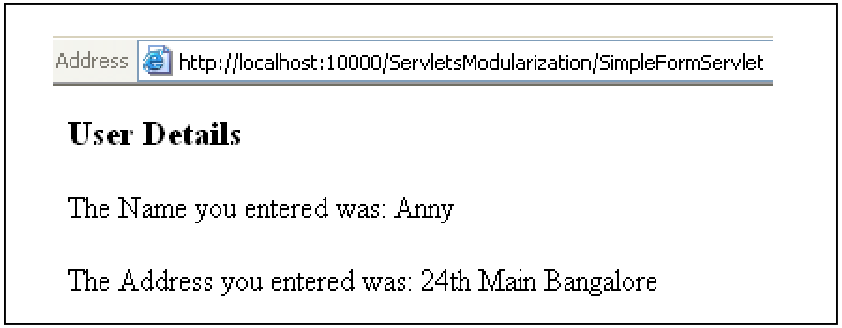


http://localhost:10000/ServletsModularization/Simpleform.html

Enter user details

Name: Anny

Address: 24th Main Bangalore



http://localhost:10000/ServletsModularization/SimpleFormServlet

User Details

The Name you entered was: Anny

The Address you entered was: 24th Main Bangalore

Reason it out

- What difference does it make if SimpleFormServlet used doGet method and accordingly method="GET" in the html form?

Form Data for different HTML Components

- Suppose a Job seeker Company needs basic information about the user’s name, address, state, highest qualification and skills

comp.html

```
<html>
  <head><title>Information Details</title></head>
  <body>
    <form action=“differentCompServlet" method=POST>
      <BR>Name: <input type=text name=”name” />
      <BR>Address: <textarea name=”address” rows=5 cols=20></textarea>
      <BR>State: <select name="state">
              <option value="Andhra Pradesh"> Andhra Pradesh </option>
              <option value="Karnataka"> Karnataka </option>
              <option value="Uttar Pradesh"> Uttar Pradesh</option>
              </select>
```

Form Data for different HTML Components (Contd.).

comp.html

```
<BR><BR>Highest Qualification:<BR>
            Under Graduate<input type=radio name=qualification value=”UG”>
            Post Graduate<input type=radio name=qualification value=”PG”>
      <BR><BR>Skills:<BR>
            Java<input type=checkbox name=skills value=Java>
            Servlets<input type=checkbox name=skills value=Servlets>
            JSPs<input type=checkbox name=skills value=JSPs>
            EJB 3.0<input type=checkbox name=skills value=EJB>
      <BR><BR><input type=submit value=submit><input type=reset>
    </form>
  </body>
</html>
```

Methods: getParameterNames() and getParameterValues()

- Syntax: public Enumeration getParameterNames()
  - Returns a full list of parameter names as an Enumeration of String objects, each String containing the name of a request parameter
  - Returns an empty Enumeration if the request has no parameters

- Use this method if a servlet has to get a full list of all request parameters

- Syntax: public String[ ] getParameterValues(String name)
  - Returns an array of String objects containing all values the given request parameter has
  - Returns null if the parameter does not exist
  - If the parameter has a single value, the array has a length of 1

- Use this method if a parameter has more than one value. Ex: checkbox

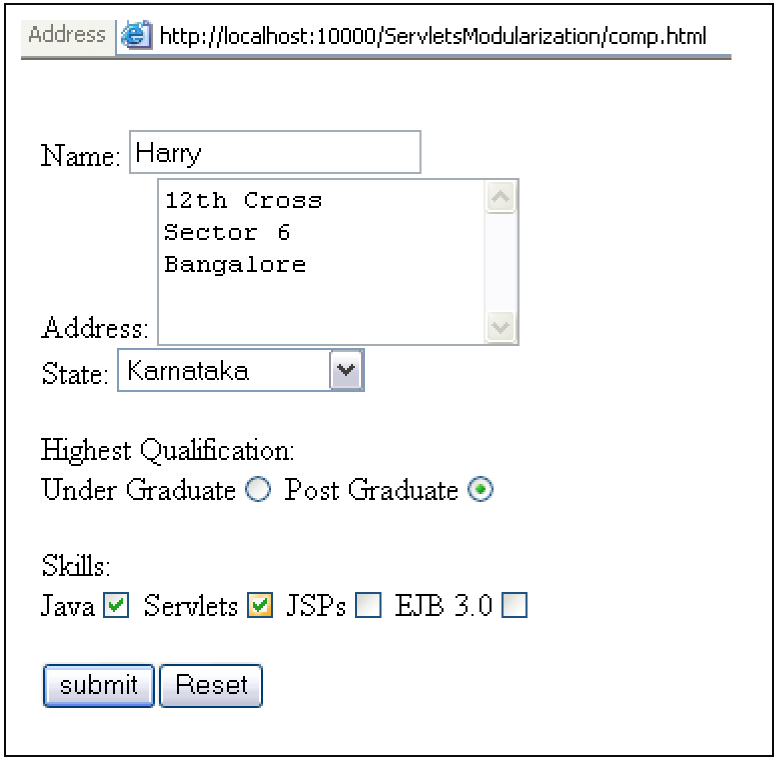
Example Servlet: Listing different Form Data

- DifferentCompServlet's doPost method retrieves request parameters using getParameterNames and getParameterValues

```
public class DifferentCompServlet extends HttpServlet {
public void doPost(HttpServletRequest req, HttpServletResponse res)
           throws ServletException, IOException {
  res.setContentType("text/html");
  PrintWriter pw = res.getWriter();
  Enumeration e = req.getParameterNames();
  // Get enumeration of parameter names

  while(e.hasMoreElements()) {
  String pname = (String) e.nextElement();
  String pvalues[] = req.getParameterValues(pname);
  pw.println(pname+" : ");
  // print parameter values by iterating through array
  for(int count = 0; count < pvalues.length; count++)
  {
        pw.println(pvalues[count]);
  }
        pw.println("<br>");
  }
           pw.close();   }   }
```

Demo for Handling Different Form Data

- Demonstrate a DifferentFormServlet with a doPost method that retrieves request parameters of various form data using getParameterNames and getParameterValues



http://localhost:10000/ServletsModularization/comp.html

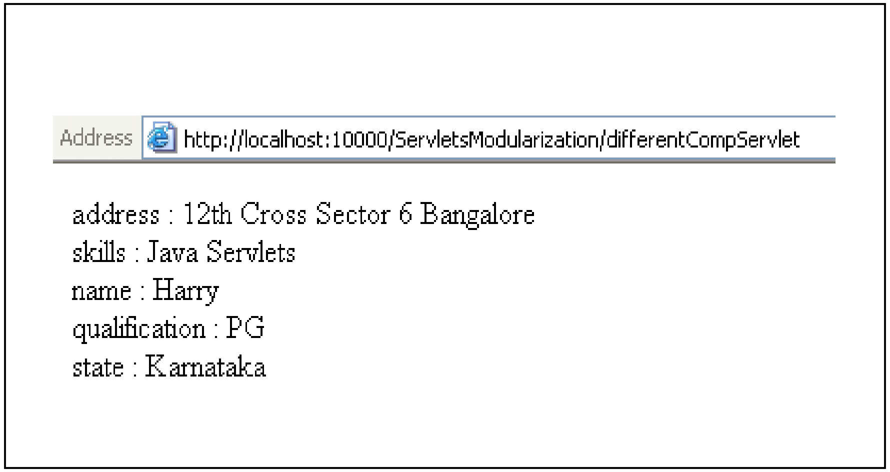
Name: Harry

Address:
```
12th Cross
Sector 6
Bangalore
```

State: Karnataka

Highest Qualification:
Under Graduate ○ Post Graduate ●

Skills:
Java ☑ Servlets ☑ JSPs □ EJB 3.0 □

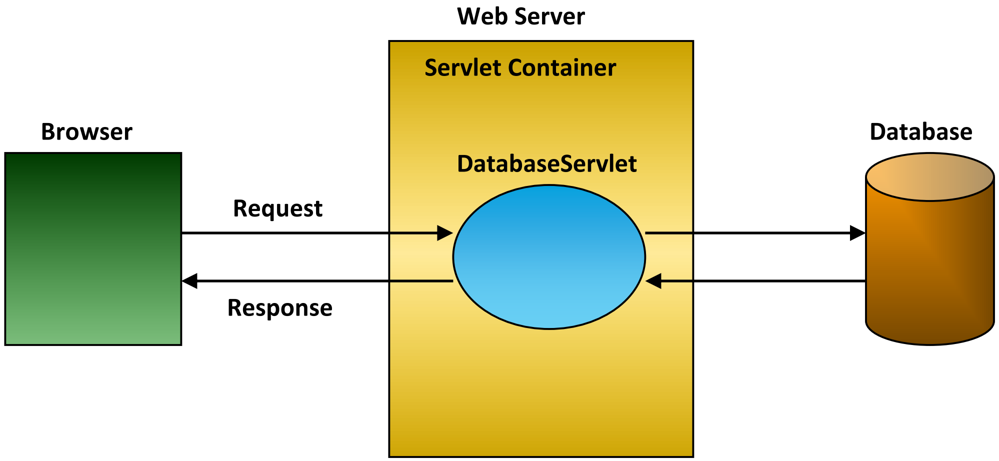


http://localhost:10000/ServletsModularization/differentCompServlet

```
address : 12th Cross Sector 6 Bangalore
skills : Java Servlets
name : Harry
qualification : PG
state : Karnataka
```

Using JDBC in a Servlet

- A servlet can retrieve information from a database or perform update/delete/insert queries to/from a database



Demo for accessing database

- A servlet to display records from database table

```
import java.sql.*;
import javax.servlet.*;
import javax.servlet.http.*;
import java.io.*;
public class DatabaseServlet extends HttpServlet {
Connection con;
PreparedStatement st;
Statement stmt;
ResultSet rs;
public void init(ServletConfig config) throws ServletException {
try {
Class.forName("sun.jdbc.odbc.JdbcOdbcDriver");
con = DriverManager.getConnection("Jdbc:Odbc:vdsn2", "scott",
  "tiger");
System.out.println("Connected..");
} catch (Exception e) {
  System.out.println("Error in connection..");         }}
```

Demo for accessing database (Contd.).

```
public void doGet(HttpServletRequest req, HttpServletResponse
  res) throws ServletException, IOException {
res.setContentType("text/html");
PrintWriter pw = res.getWriter();
//Displaying records
try {
      stmt = con.createStatement();
      rs = stmt.executeQuery("select * from books");

      pw.println("Displaying Book Records...");
while (rs.next()) {
pw.println("<p>" + rs.getInt(1) + " " + rs.getString(2) + " "
  +rs.getString(3));
  }
  } catch (Exception e) {
   System.out.println("Error..." + e);
   } } }
```

Summary

In this module, you were able to:

- Process parameters from HTML forms

Quiz

1. The doGet() or doPost() method of a Servlet are invoked by --------------------
    1. init() method
    2. service() method
    3. destroy() method

- --------------- is the deployment descriptor file for Servlets
    1. servlet-config.xml
    2. web.xml
    3. struts-config.xml

WIPRO
Applying Thought

Thank You