Servlet Classes and Interfaces

Step 1: Creating a dynamic web project

- Open Eclipse
- Go the File menu. Choose New->Dynamic Web Project
- Enter the project name as ServletInterfaces. Click on Next
- Enter nothing in the next screen and click on Next
- Check the checkbox Generate web.xml deployment descriptor and click on Finish
- This will create the project files in the Project Explorer

Step 2: Creating an HTML page

- In the Project Explorer, expand the project ServletInterfaces
- Expand WebContent. Right click on WebContent . Choose New->HTML File
- Enter the filename as index.html and click on Finish
- Enter the following code:

```
<!DOCTYPE html>
<html>
<head>
<meta charset="UTF-8">
<title>Servlet Interfaces</title>
</head>
<body>

<a href="interface">Show Servlet Interface</a>

</body>
</html>
```

- Click on the Save icon

Step 3: Creating a servlet InterfaceDemo.java

- In the Project Explorer, expand ServletInterfaces->Java Resources
- Right click on src and choose New->Servlet
- In Class Name, enter InterfaceDemo and click on Finish
- Enter the following code:

```
import java.io.*
import javax.servlet.*;
import javax.servlet.annotation.WebServlet;

/**
* Servlet implementation class InterfaceDemo
*/
@WebServlet("/InterfaceDemo")
public class InterfaceDemo implements Servlet {

        ServletConfig config=null;
          public void init(ServletConfig config){
              this.config=config;
              System.out.println("Initialization complete");
          }

          public void service(ServletRequest req,ServletResponse res)
          throws IOException,ServletException{
              res.setContentType("text/html");
              PrintWriter pwriter=res.getWriter();
              pwriter.print("<html>");
              pwriter.print("<body>");
              pwriter.print("In the service() method<br>");
              pwriter.print("</body>");
```

```
            pwriter.print("</html>");
        }
        public void destroy(){
            System.out.println("In destroy() method");
        }
        public ServletConfig getServletConfig(){
            return config;
        }
        public String getServletInfo(){
            return "This is a sample servlet info";
        }
}
```

Step 4: Configuring web.xml

- In the Project Explorer, expand ServletInterfaces-> WebContent->WEB-INF
- Double click on web.xml to open it in the editor
- Enter the following script:

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

```
  <servlet>
    <servlet-name>InterfaceDemo</servlet-name>
    <servlet-class>InterfaceDemo</servlet-class>
  </servlet>

  <servlet-mapping>
    <servlet-name>InterfaceDemo</servlet-name>
    <url-pattern>/interface</url-pattern>
  </servlet-mapping>

</web-app>
```

Step 5: Checking for servlet-api.jar

- Before building the project, we need to add servlet-api.jar to the project

- Servlet-api.jar file is already present in your practice lab. (Refer FSD: Lab Guide - Phase 2)

- To add it to the project, follow the below mentioned steps:

    - In the Project Explorer, right click on ServletInterfaces and choose Properties
    - Select Java Build Path from the options on the left
    - Click on Libraries tab on the right
    - Under ClassPath, expand the node that says Apache Tomcat
    - If there is an existing entry for servlet-api.jar, then click on Cancel and exit the window
    - If it is not there, then click on Classpath entry and click on Add External JARs button on the right

- From the file list, select servlet-api.jar file and click on Ok
- Click on Apply and Close

Step 6: Building the project

- From the Project menu at the top, click on Build
- If any compile errors are shown, fix them as required

Step 7: Publishing and starting the project

- If you do not see the Servers tab near the bottom of the IDE, go to Window menu and click on Show View->Servers
- Right click on the Server entry and choose Add and Remove
- Click the Add button to move ServletInterfaces from the Available list to the Configured list
- Click on Finish
- Right click on the Server entry and click on Publish
- Right click on the Server entry and click on Start
- This will start the server

Step 8: Running the project

- To run the project, open a web browser and type: http://localhost:8080/ServletInterfaces

Step 9: Pushing the code to your GitHub repositories

- Open your command prompt and navigate to the folder where you have created your files.

  cd <folder path>

- Initialize your repository using the following command:

  git init

- Add all the files to your git repository using the following command:

  git add .

- Commit the changes using the following command:

  git commit . -m “Changes have been committed.”

- Push the files to the folder you initially created using the following command:

  git push -u origin master