SOEN 387
WEB-BASED ENTERPRISE APPLICATIONS DESIGN

TUTORIAL – 2

Java Web & Servlets

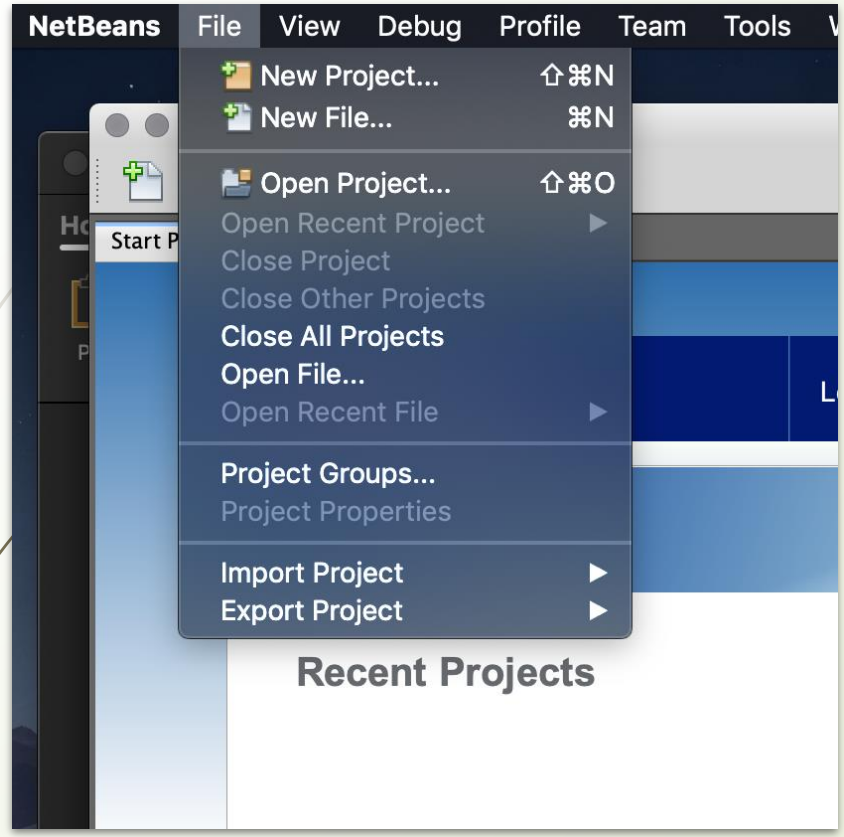
Agenda

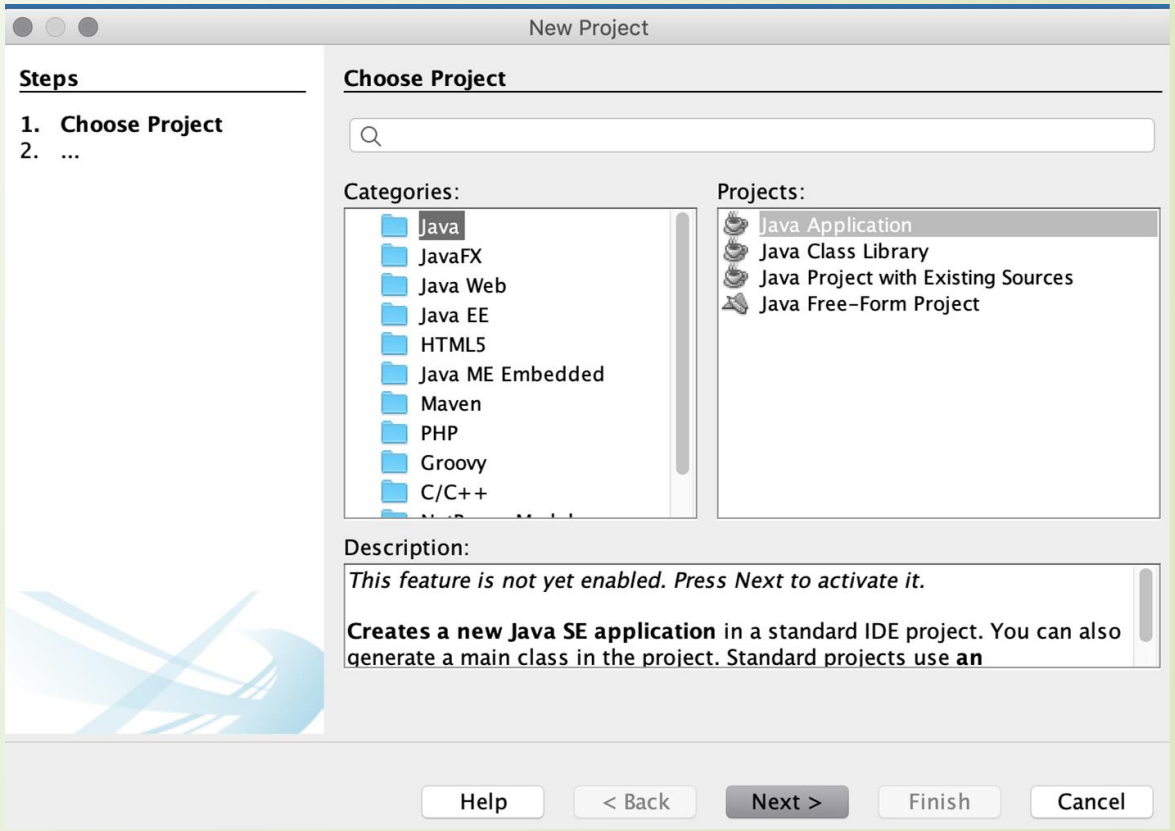
- NetBeans IDE Prerequisites
- Setting up a Java project in NetBeans
- What is servlet
- Creating web project
- Creating servlet
- Servlet API
- Building and Running the Application
- Debugging a servlet file
- File Structure
- Exercise

NetBeans IDE Prerequisites

Software or Resource	Version Required
NetBeans IDE	version 7.2, 7.3, 7.4, or 8.0
Java Development Kit (JDK)	version 6, 7, or 8

Setting up a Java project in NetBeans

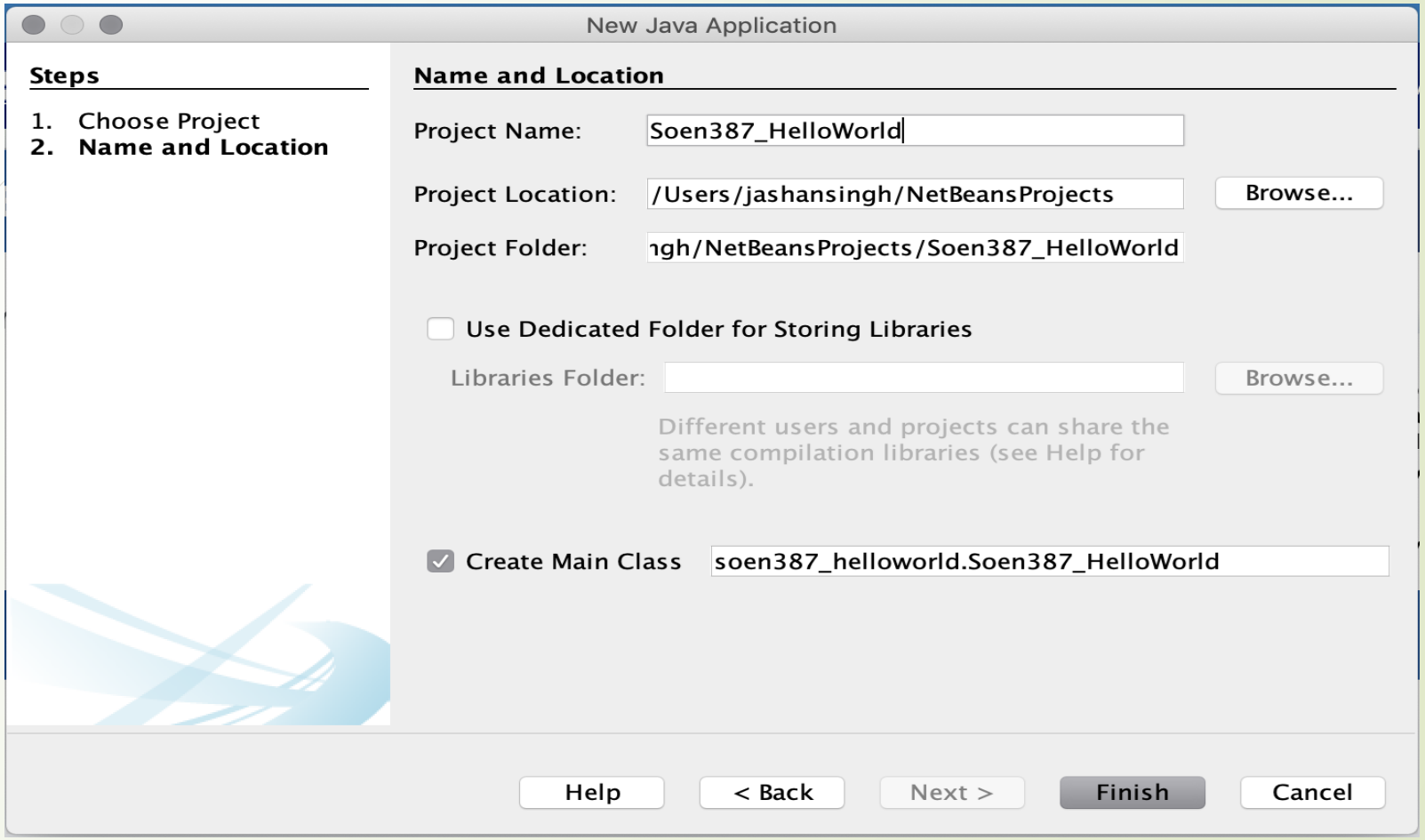




4

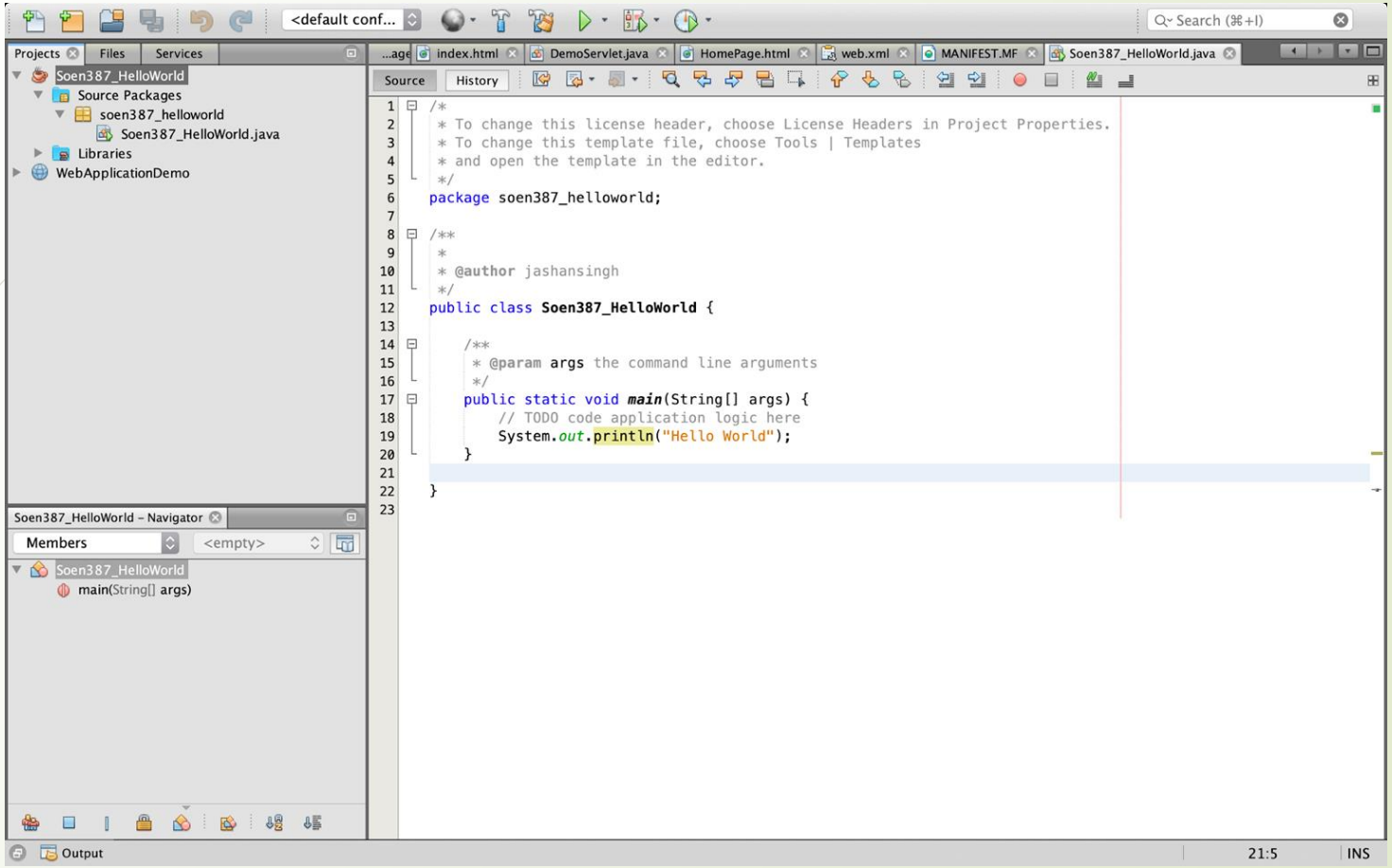
In the IDE, choose File > New Project, as shown in the figure above.

In the New Project wizard, expand the Java category and select Java Application as shown in the figure below. Click Next.



In the Name and Location page of the wizard, do the following:

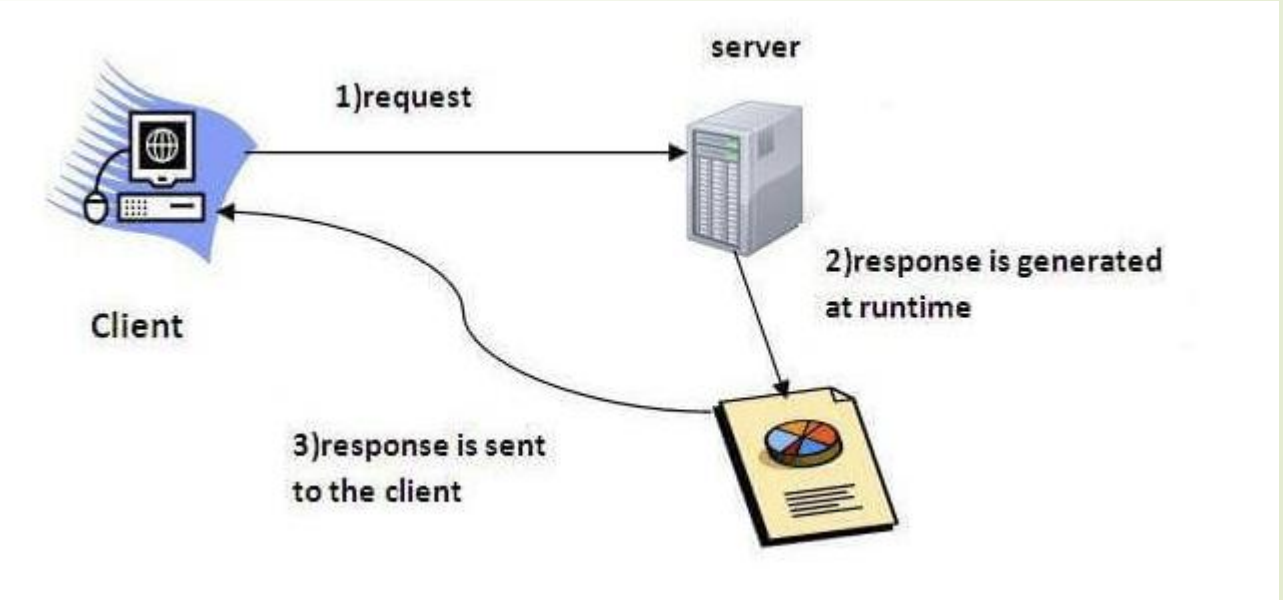
- In the Project Name field, type Soen387_HelloWorld
- Leave the Use Dedicated Folder for Storing Libraries checkbox unselected.



The project is created and opened in the IDE. You should see the following components:

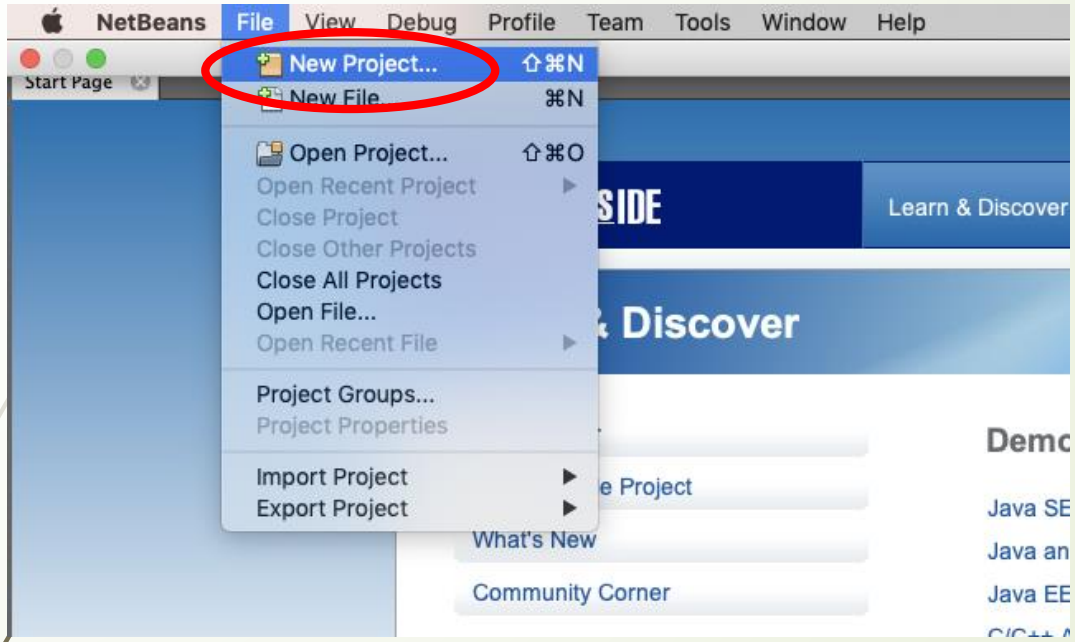
- The Projects window, which contains a tree view of the components of the project, including source files, libraries that your code depends on..
- The Source Editor window with a file called Seon387_HelloWorld open.
- The Navigator window, which you can use to quickly navigate between elements within the selected class.
- To run the program:
- Choose Run > Run Project.

What is servlet

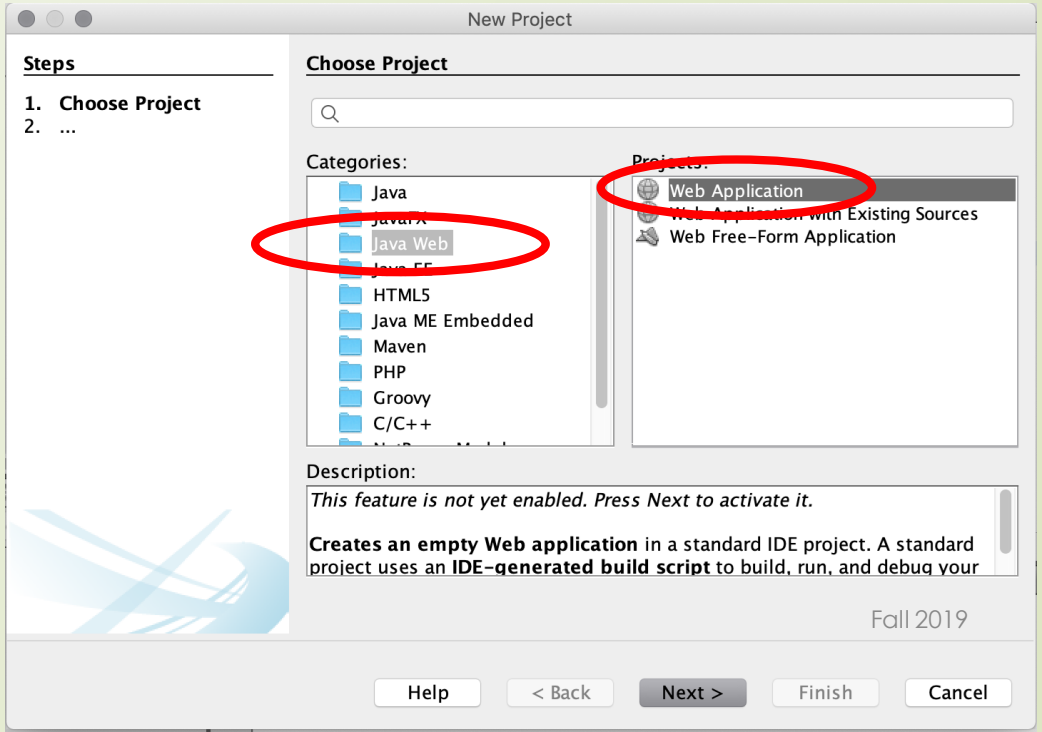


- Used to create a web application
- Resides at server side and generates a dynamic web page
- Client sends HTTP request
- Server receives request, servlets process it
- Results returned (HTML document, images, binary data)

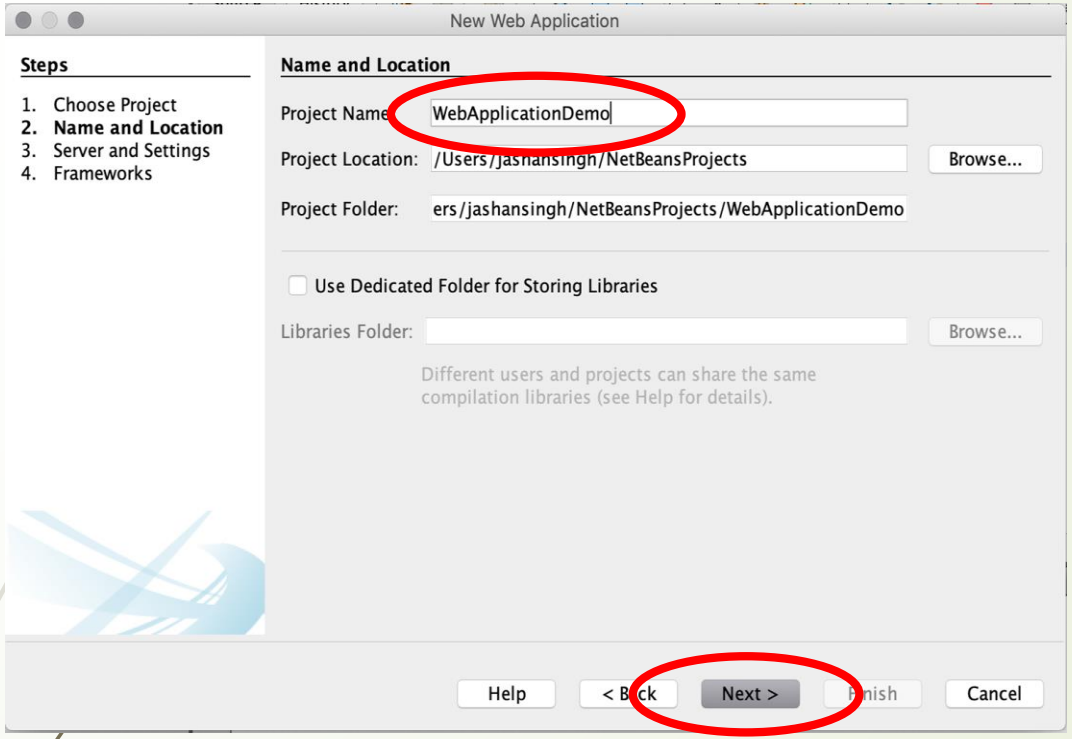
Creating web project



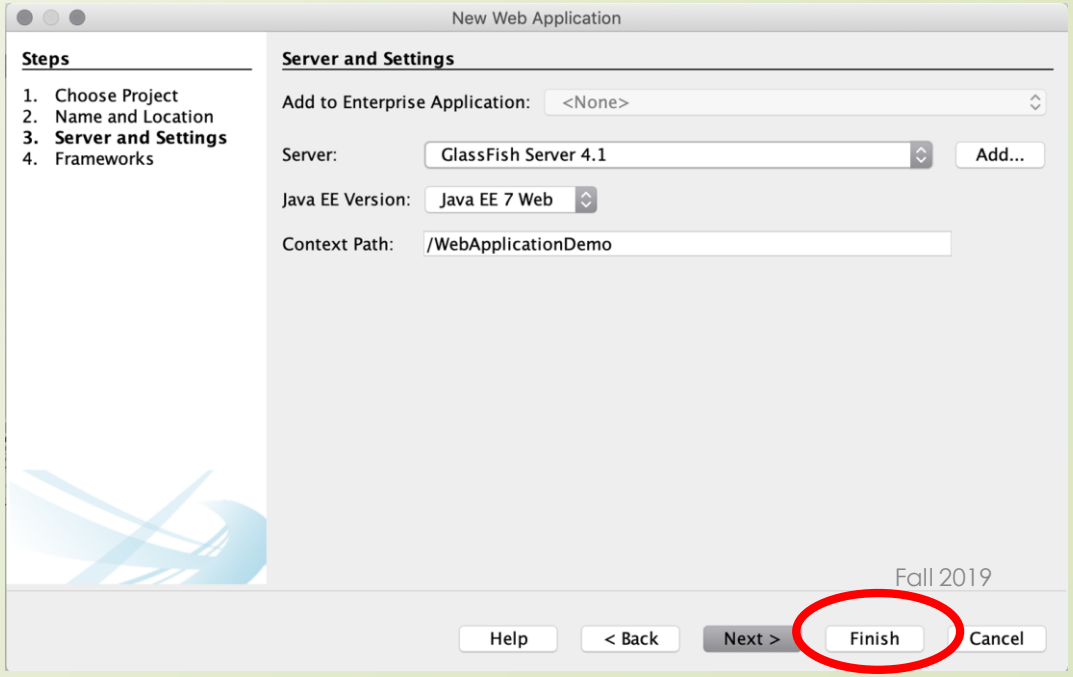
Open Netbeans IDE, Select File -> New Project



Select Java Web -> Web Application, then click on Next,

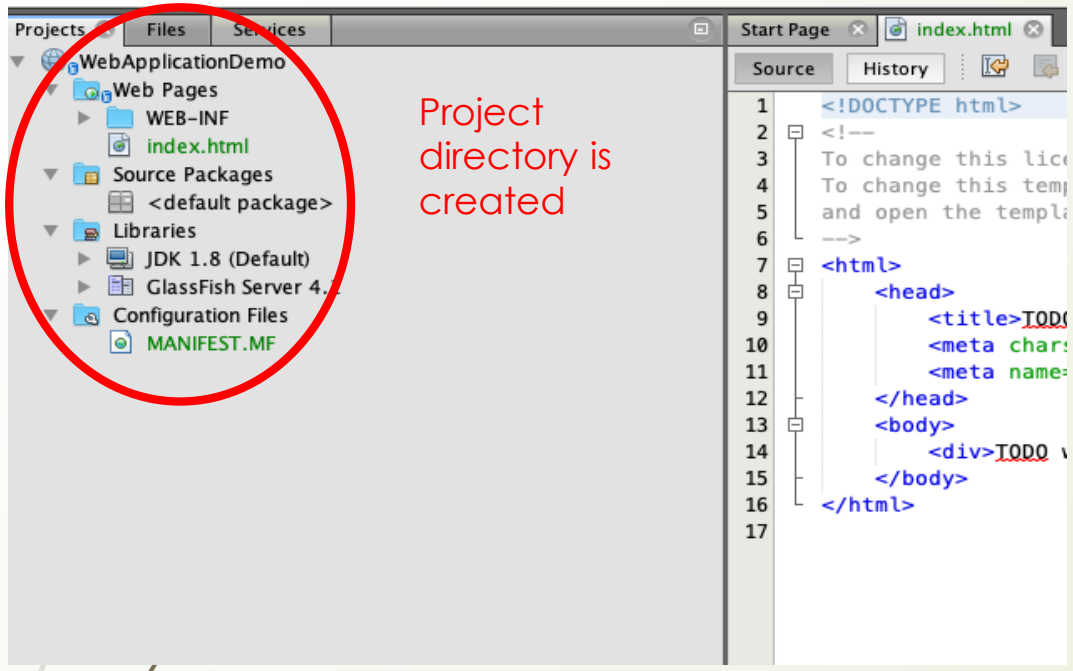


Give a name to your project and click on Next,



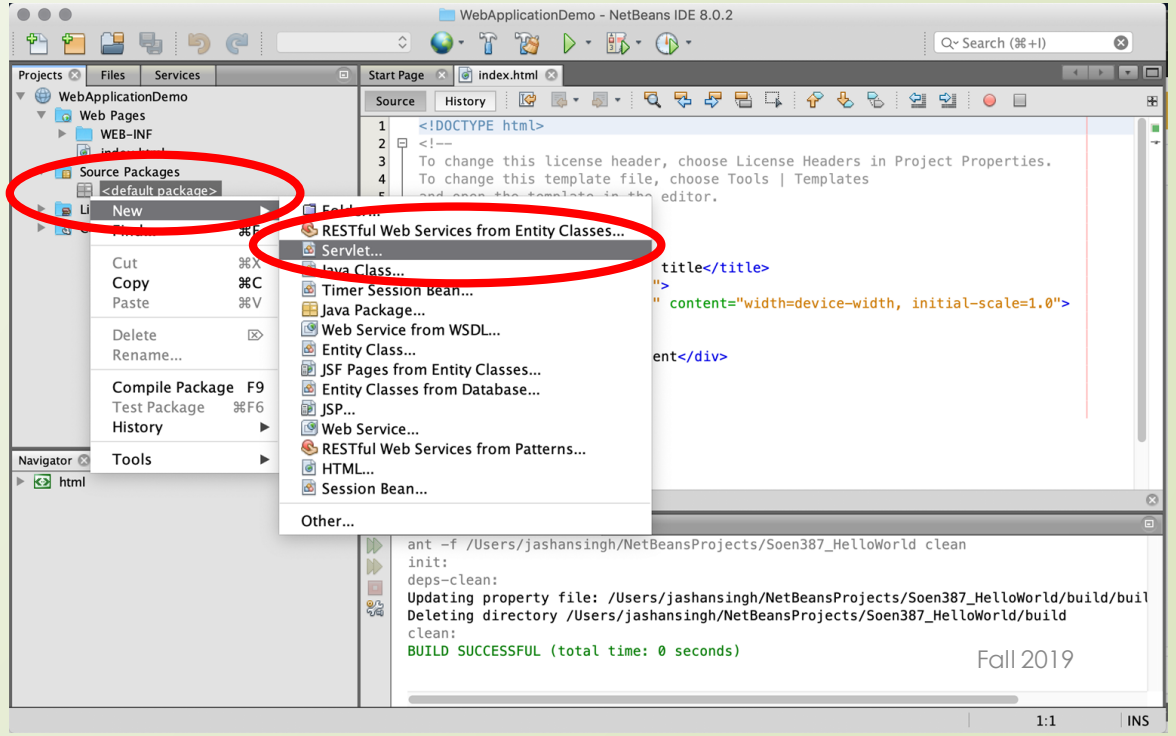
and then, Click Finish

Creating servlet

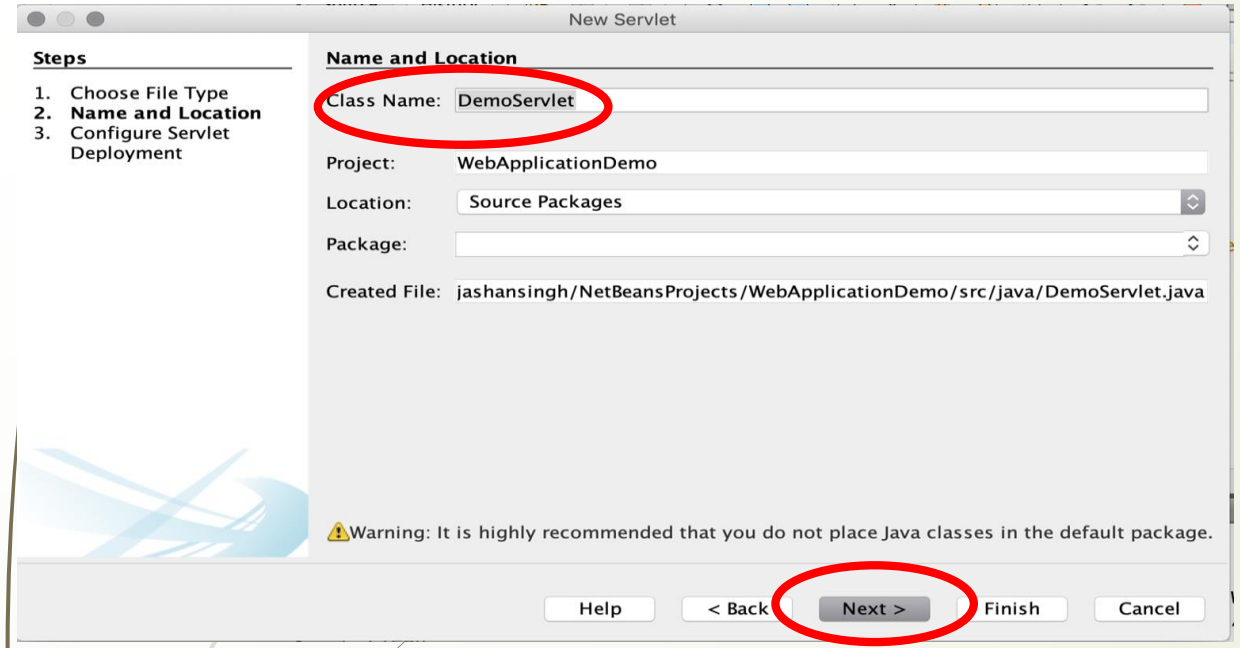


Project directory is created

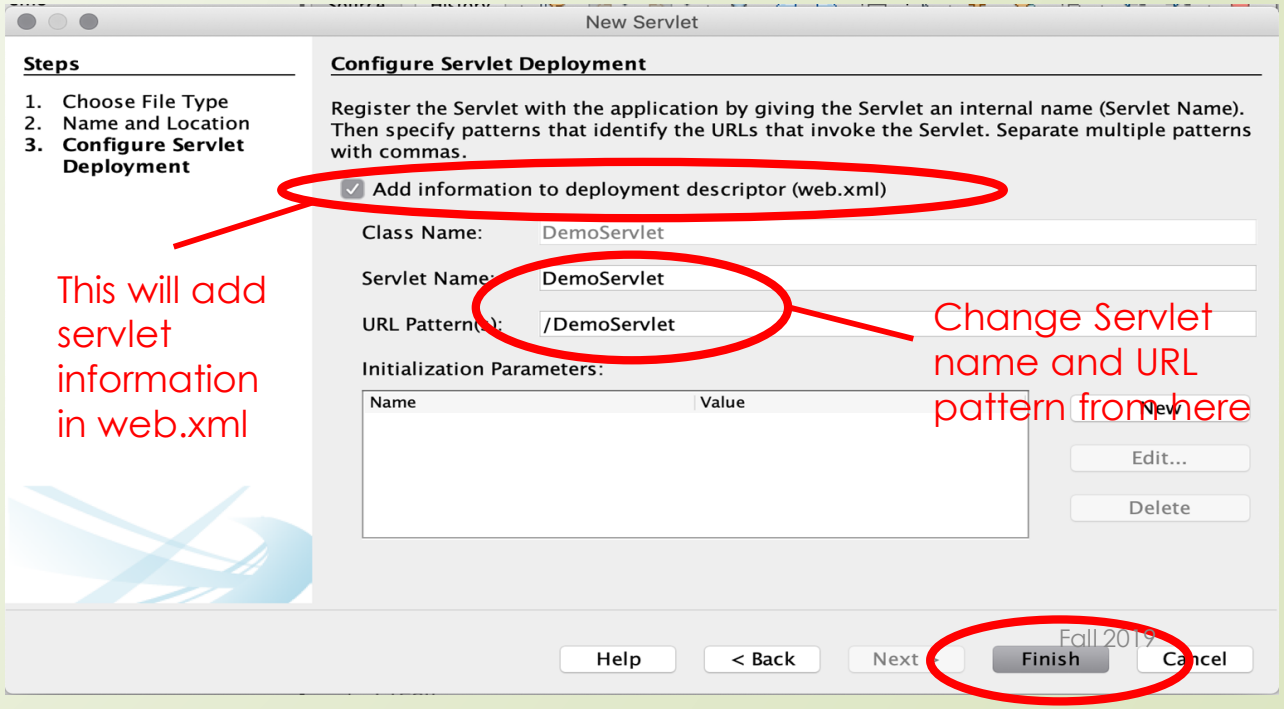
The complete directory structure required for the Servlet Application will be created automatically by the IDE.

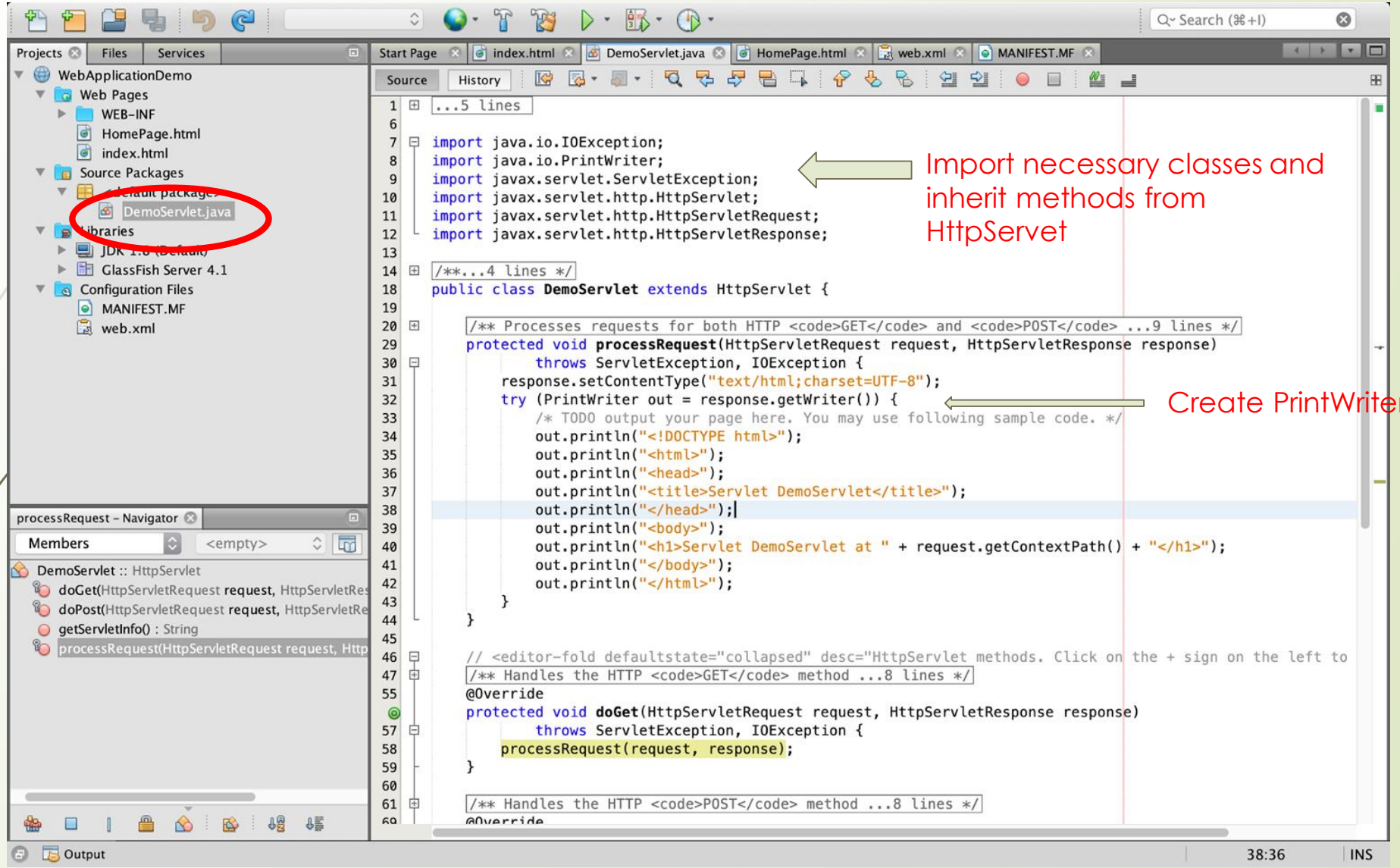


To create a Servlet, open Source Package, right click on default packages -> New -> Servlet.

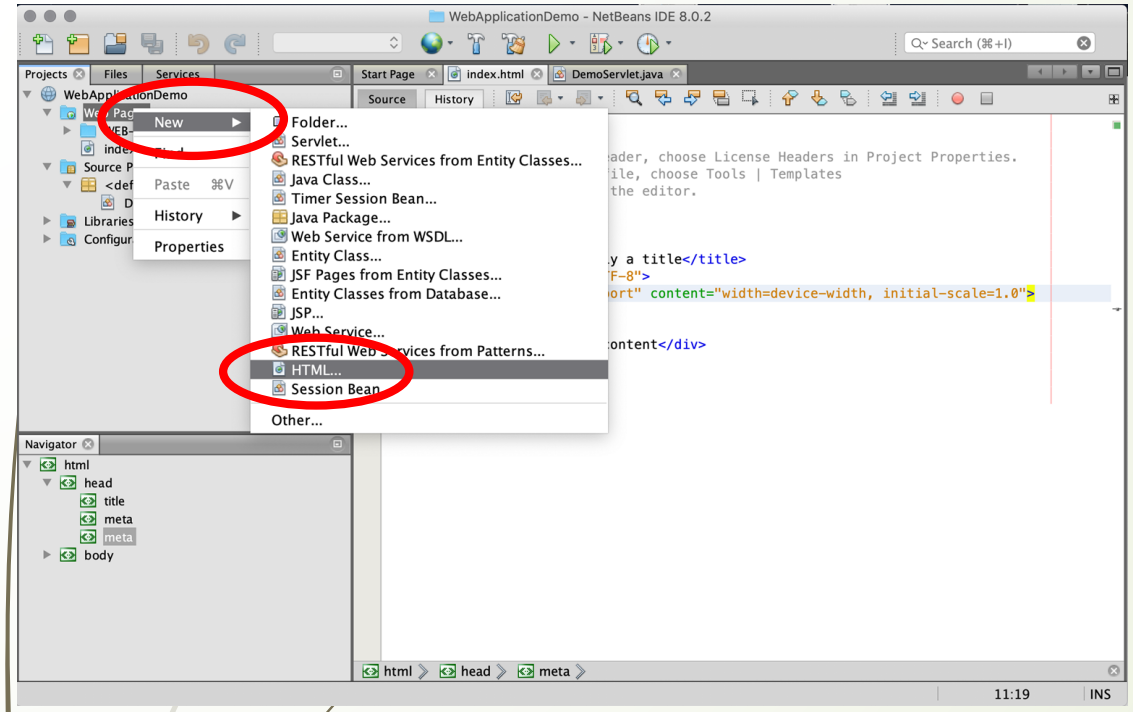


Give a Name to your Servlet class file

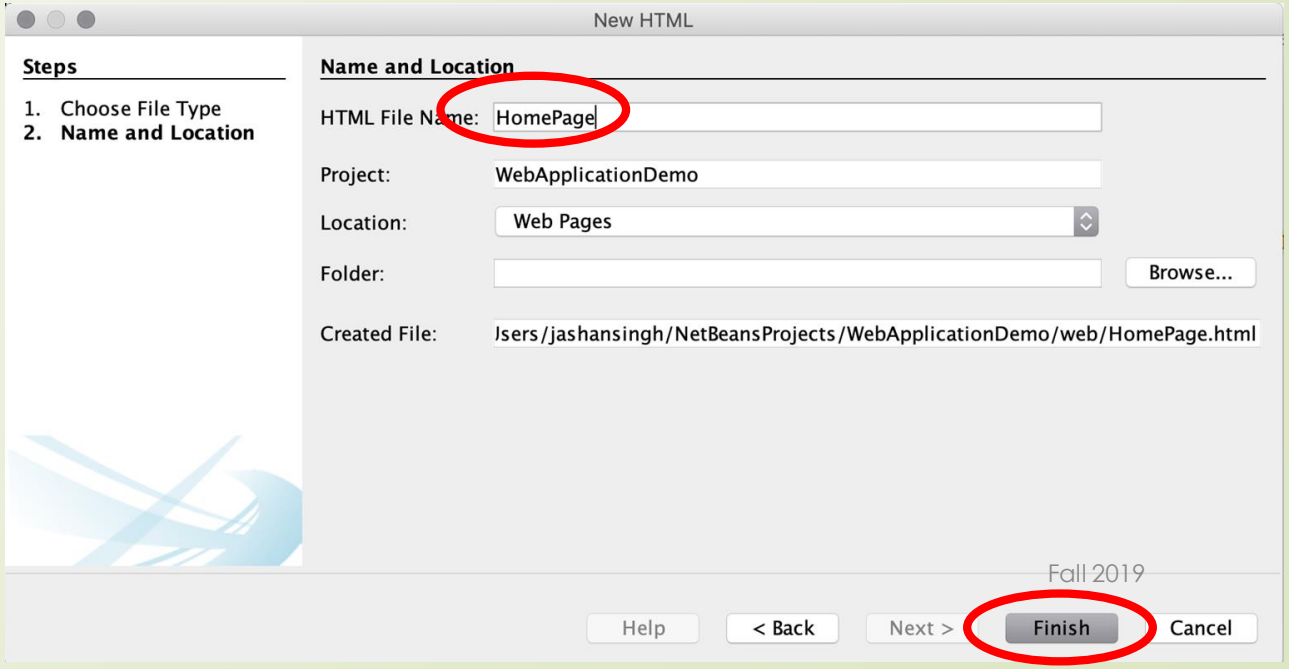




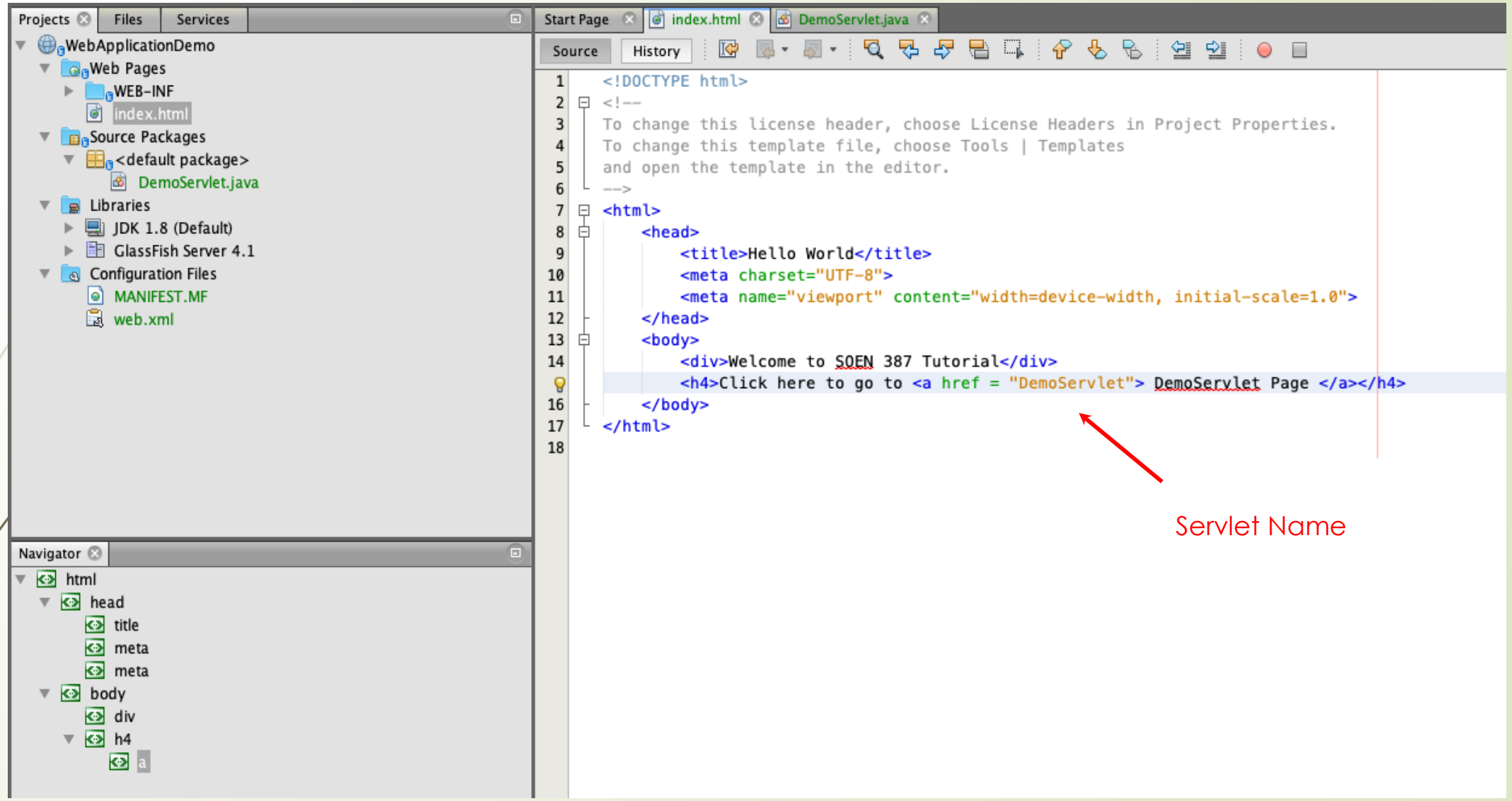
Now, your Servlet class is ready



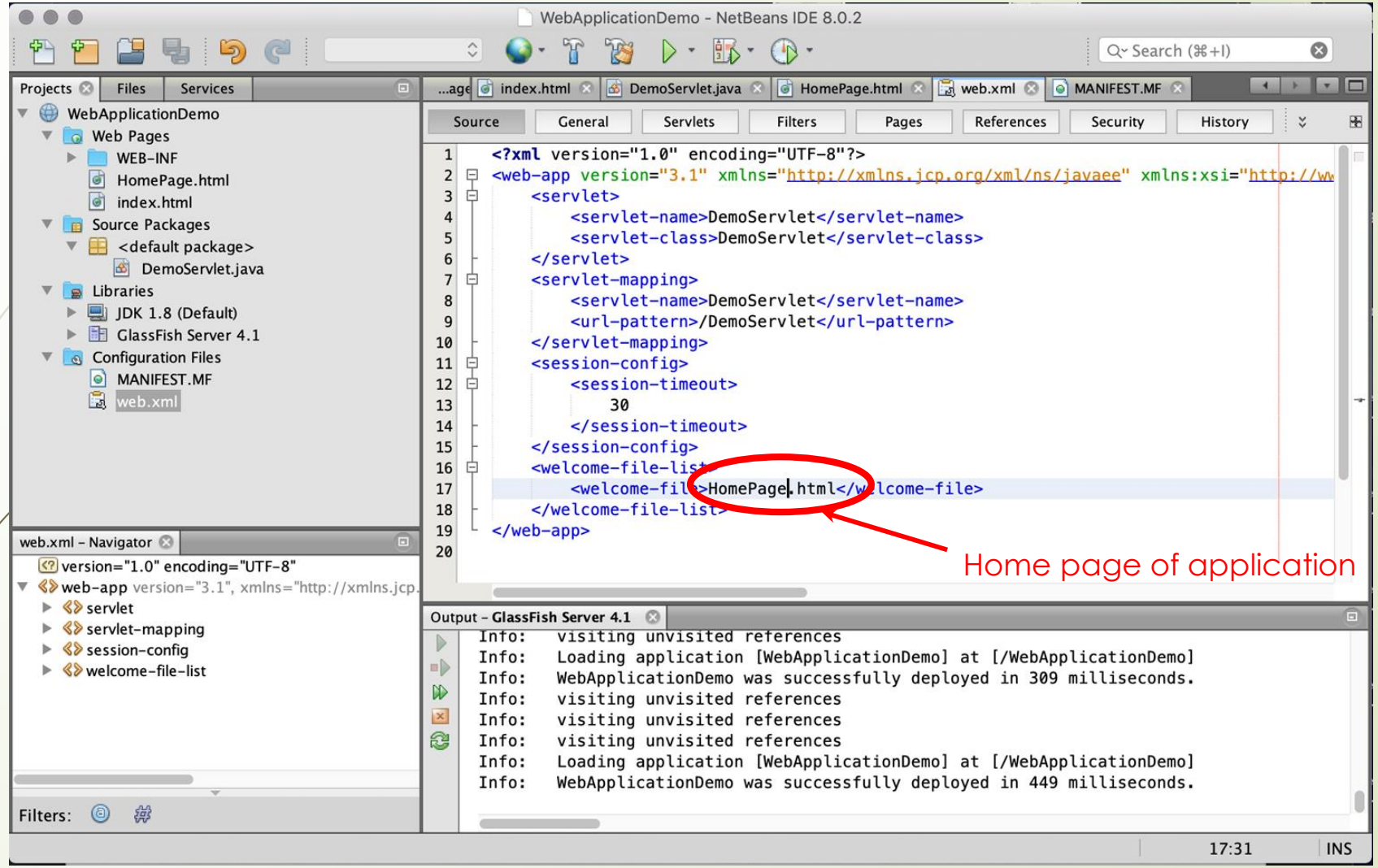
Create an HTML file, right click on Web Pages -> New -> HTML



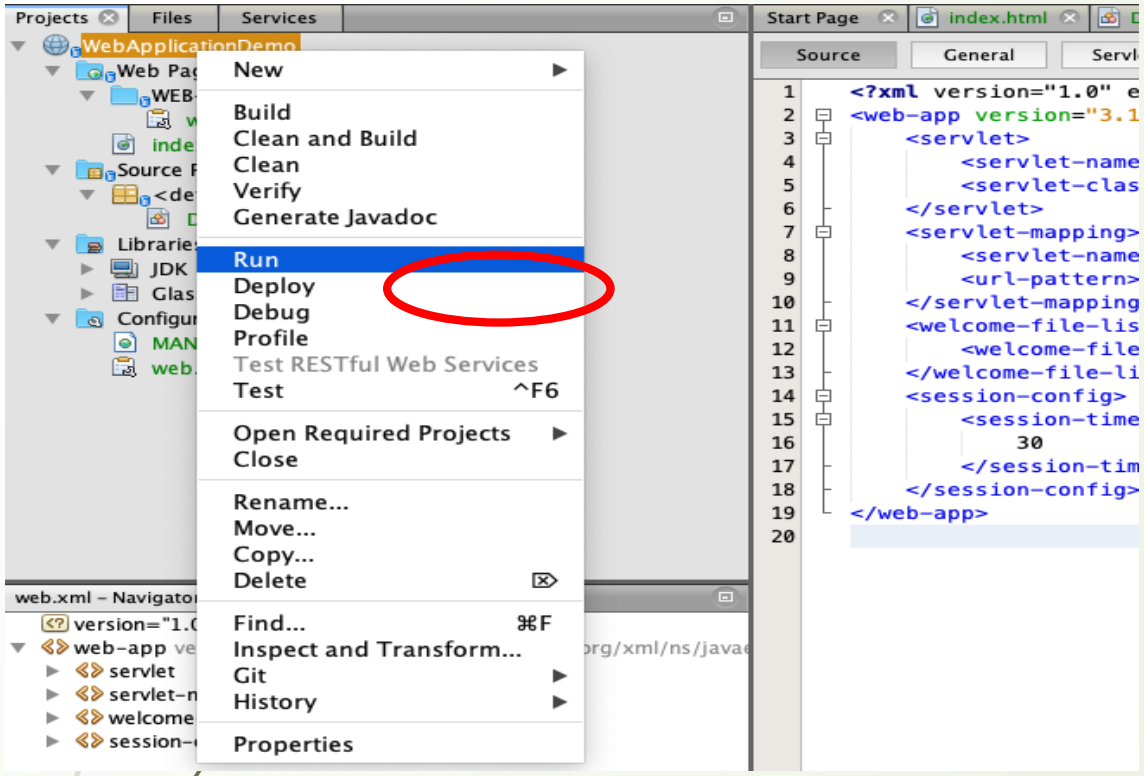
Give it a name.



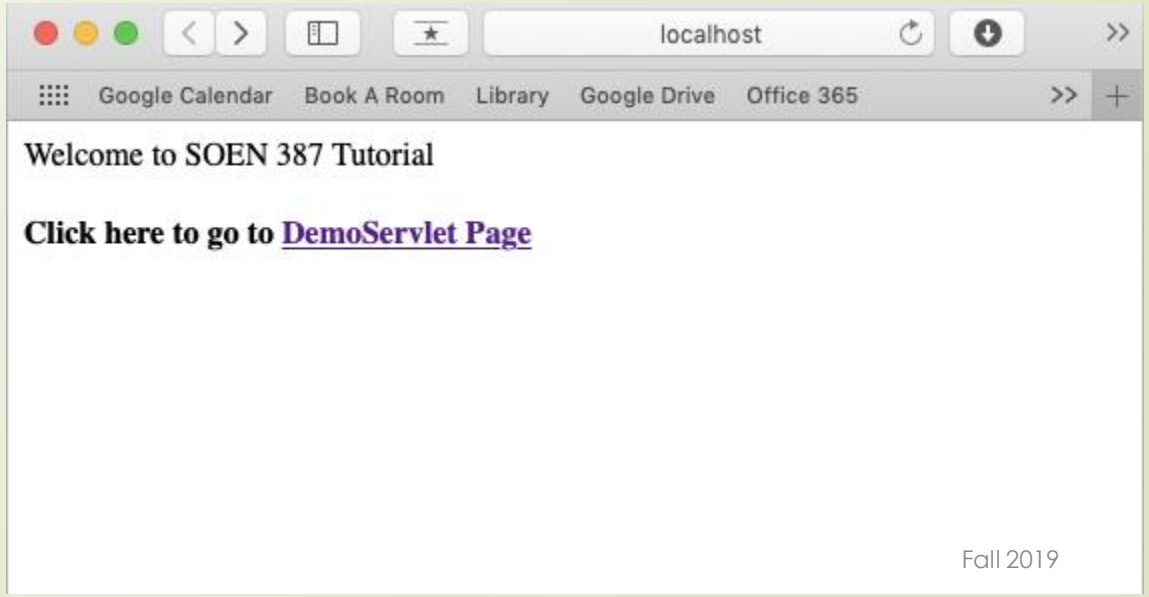
Write some code inside your HTML file. We have created a hyperlink to our Servlet in our HTML file.



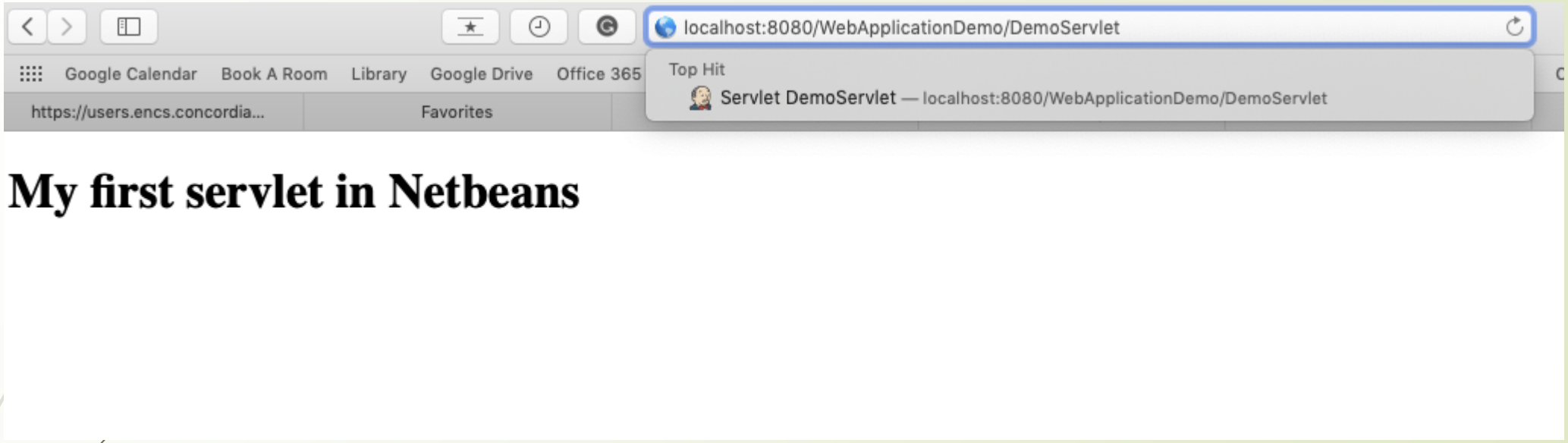
Edit web.xml file. In the web.xml file you can see, we have specified the url-pattern and the servlet-name, this means when DemoServlet url is accessed our Servlet file will be executed.



Run your application, right click on your Project and select Run



open localhost:
http://localhost:8080/WebApplicationDemo/



Hurray! Our First Servlet class is running.

Servlet API

- All servlets must import the following packages:

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

- All servlets must extend the HttpServletor Servlet class

```
public class DemoServlet extends HttpServlet
```

- Servlets must provide and implementation for doGet, doPostor both

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

- HTML can be written as text to the response

- Obtain the PrintWriterobject from the response and write text to it

```
    try (PrintWriter out = response.getWriter()) {
            /* TODO output your page here. You may use following sample
code. */
            out.println("<!DOCTYPE html>");
            out.println("<html>");
            out.println("<head>");
            out.println("<title>Servlet DemoServlet</title>");
            out.println("</head>");
            out.println("<body>");
            out.println("<h1>Servlet DemoServlet at " +
request.getContextPath() + "</h1>");
            out.println("</body>");
            out.println("</html>");
        }
```

Building and Running the Application

Once you have written and test run your application, you can use the Clean and Build command to build your application for deployment. When you use the Clean and Build command, the IDE runs a build script that performs the following tasks:

- Deletes any previously compiled files and other build outputs.
- Recompiles the application and builds a JAR file containing the compiled files.

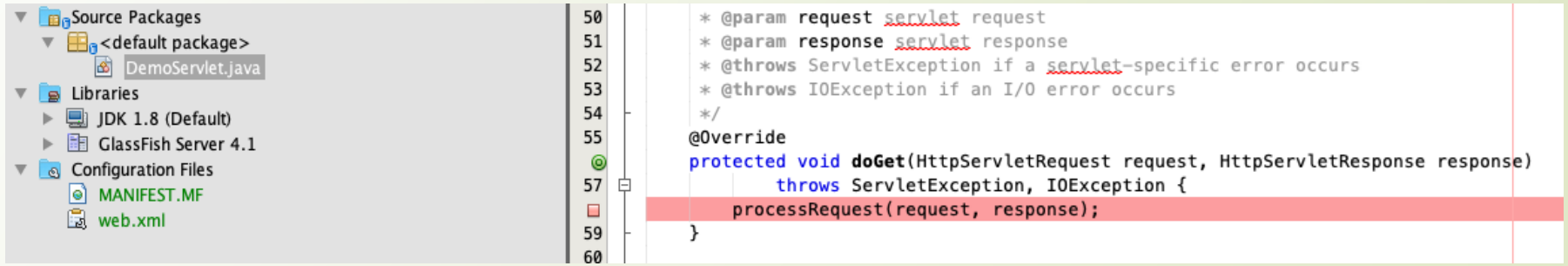
To build your application:

- Choose Run > Clean and Build Project. You can view the build outputs by opening the Files window and expanding the SOEN387_HelloWorld node.
- The compiled bytecode file SOEN387_HelloWorld.class is within the build/classes/SOEN387_HelloWorld subnode.

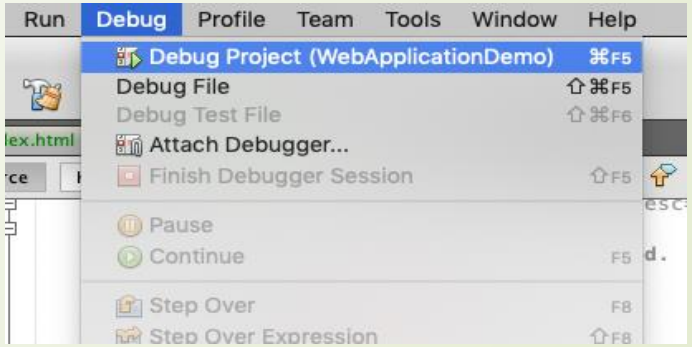
Debugging a servlet file

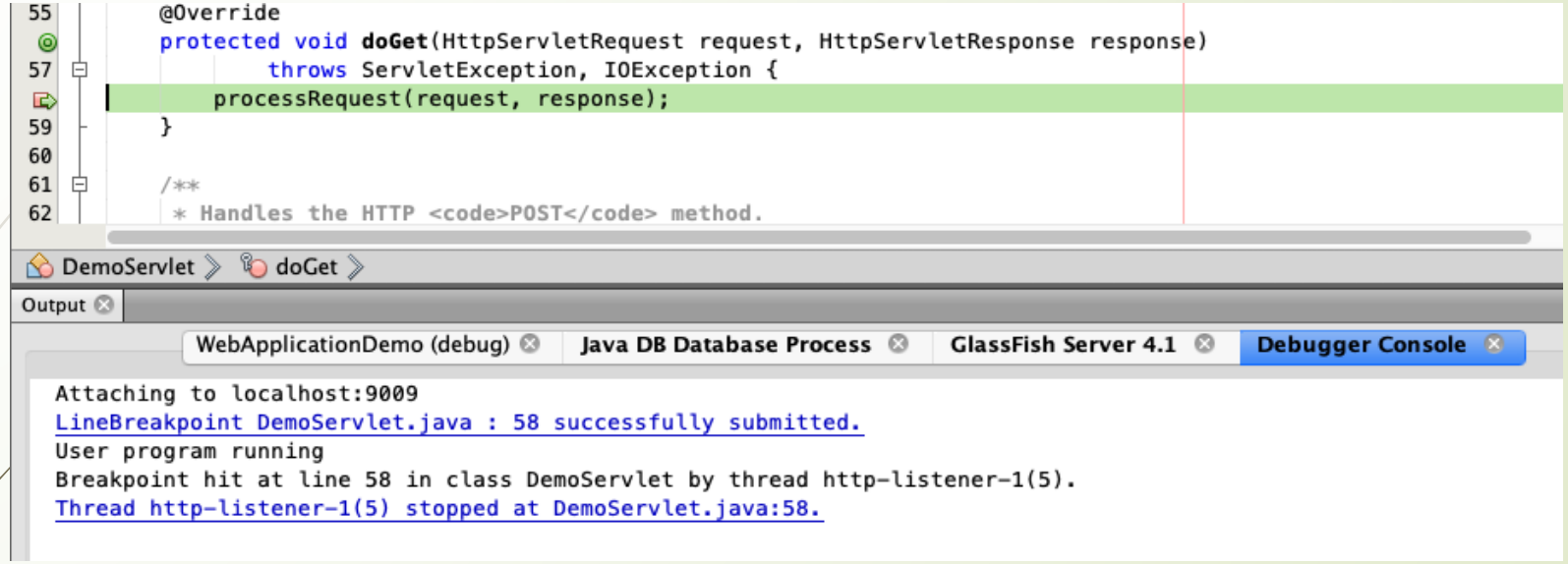
When you execute a servlet in NetBeans, it will try to start the internal Tomcat. Since Tomcat is running inside of NetBeans, you will be able to set breakpoints in the code and stop execution of the servlet.

- Place a breakpoint on a line inside of doGet



- From the Debug menu select Start Session and Run In Debugger

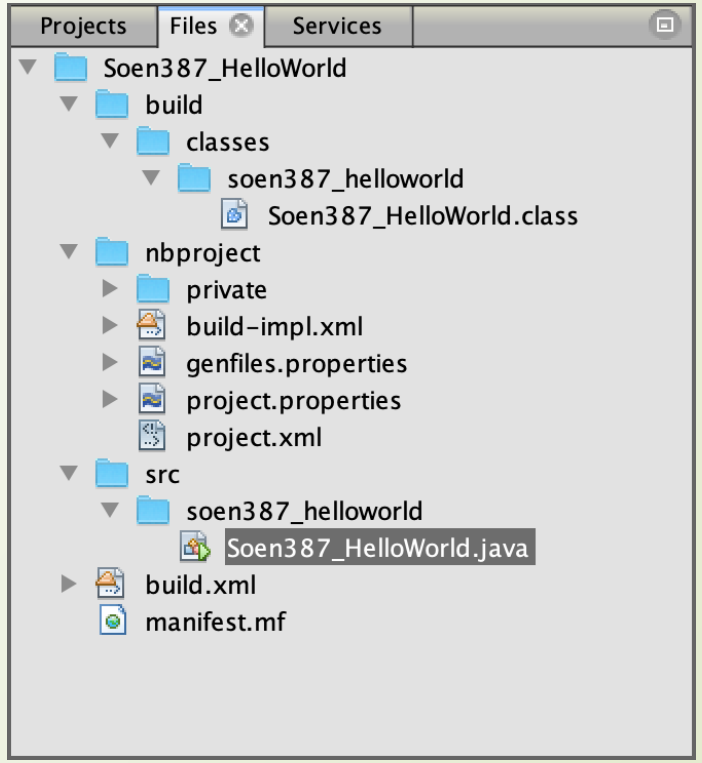


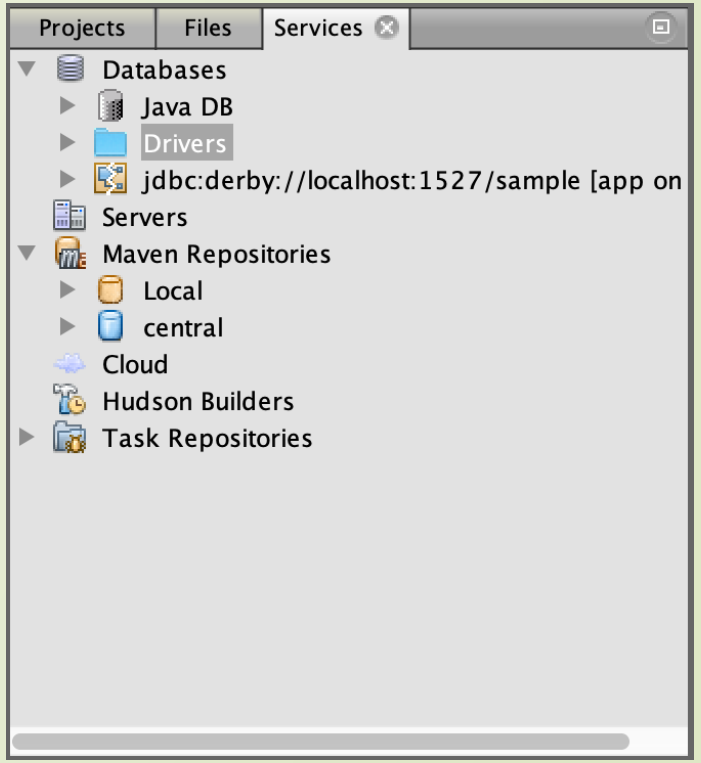


NetBeans will start the internal Tomcat and will display the servlet in the browser.

Control should pass to the breakpoint in NetBeans.

Project File Structure





Exercise

Setting the HTTP status code

It is sometimes necessary to set a status code. An HTTP status code— sometimes called a response code— effectively tells the client what the output represents. Or in other words, it indicates the status of the request: successful, page not found, access denied etc.

Write a servlet to send the appropriate HTTP status codes based on request parameter.
For instance, your may look for a request parameter called 'document' and generate the following response codes:

1. status code 200 SC_OK, *when document equals 'document1'*
2. status code 301 SC_MOVED_PERMANENTLY , *when document equals 'document2'*
3. Status code 404 SC_NOT_FOUND , *when document equals 'document3'*
4. Status code 400 SC_BAD_REQUEST, for any other values