# SOEN 387
# WEB-BASED ENTERPRISE APPLICATIONS DESIGN

## Tutorial 1
## Java Web & Servlets

Prepared by: Hambrsoom Baboyan

# Agenda

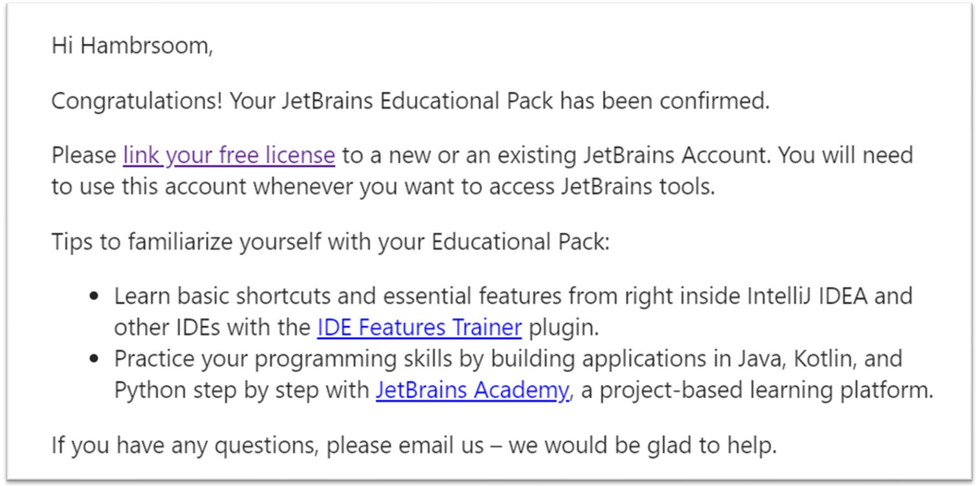
- IntelliJ IDEA Installation
- Creating a Web Project
- What is a Servlet?
- Creating a Servlet
- Building and Running the Application
- Types of Java Application
- Debugging a Servlet File

# IntelliJ IDEA Installation

Follow steps 1 & 2 if you **don't** have IntelliJ IDEA Ultimate

## Step 1: Get a Free Educational Licence For JetBrains

- Access this link and fill up the required information
- For the **Email** field, you should use your Concordia email which should be: **your_netname@live.concordia.ca**
- After completing the form, you should receive a confirmation email to your email. To access it, enter the following:
  - **Email:** your_netname@live.concordia.ca
  - **Password:** the same password as the one you use to access My Concordia
- Click on the link provided in the confirmation email and complete the creation of your educational account
- You will then receive an email (See Figure 1)



Hi Hambrsoom,

Congratulations! Your JetBrains Educational Pack has been confirmed.

Please link your free license to a new or an existing JetBrains Account. You will need to use this account whenever you want to access JetBrains tools.

Tips to familiarize yourself with your Educational Pack:

- Learn basic shortcuts and essential features from right inside IntelliJ IDEA and other IDEs with the IDE Features Trainer plugin.
- Practice your programming skills by building applications in Java, Kotlin, and Python step by step with JetBrains Academy, a project-based learning platform.

If you have any questions, please email us – we would be glad to help.

*Figure 1. Confirmation Email*

- Click on the **Link your free license** link provided in the email and sign in with your username and password.

## Step 2: Download IntelliJ IDEA Ultimate

- After signing in, click on the **Download** button and choose to download **IntelliJ IDEA Ultimate**
- Install the application in your environment
- When you will first open the application, it will ask you to insert your username and password. Type in the credentials you used to create your JetBrains account

# Creating a Web Project

- After opening IntelliJ IDEA, choose to create a **new project**
- Select **Java Enterprise** (See Figure 2)
- For the **Project SDK** field, choose **Java 8 JDK**

  Note: You should install Java 8 JDK in advance.
- Select **Maven** for **Build Tool** and **JUnit** for **Test Runner**
- Click **Next**

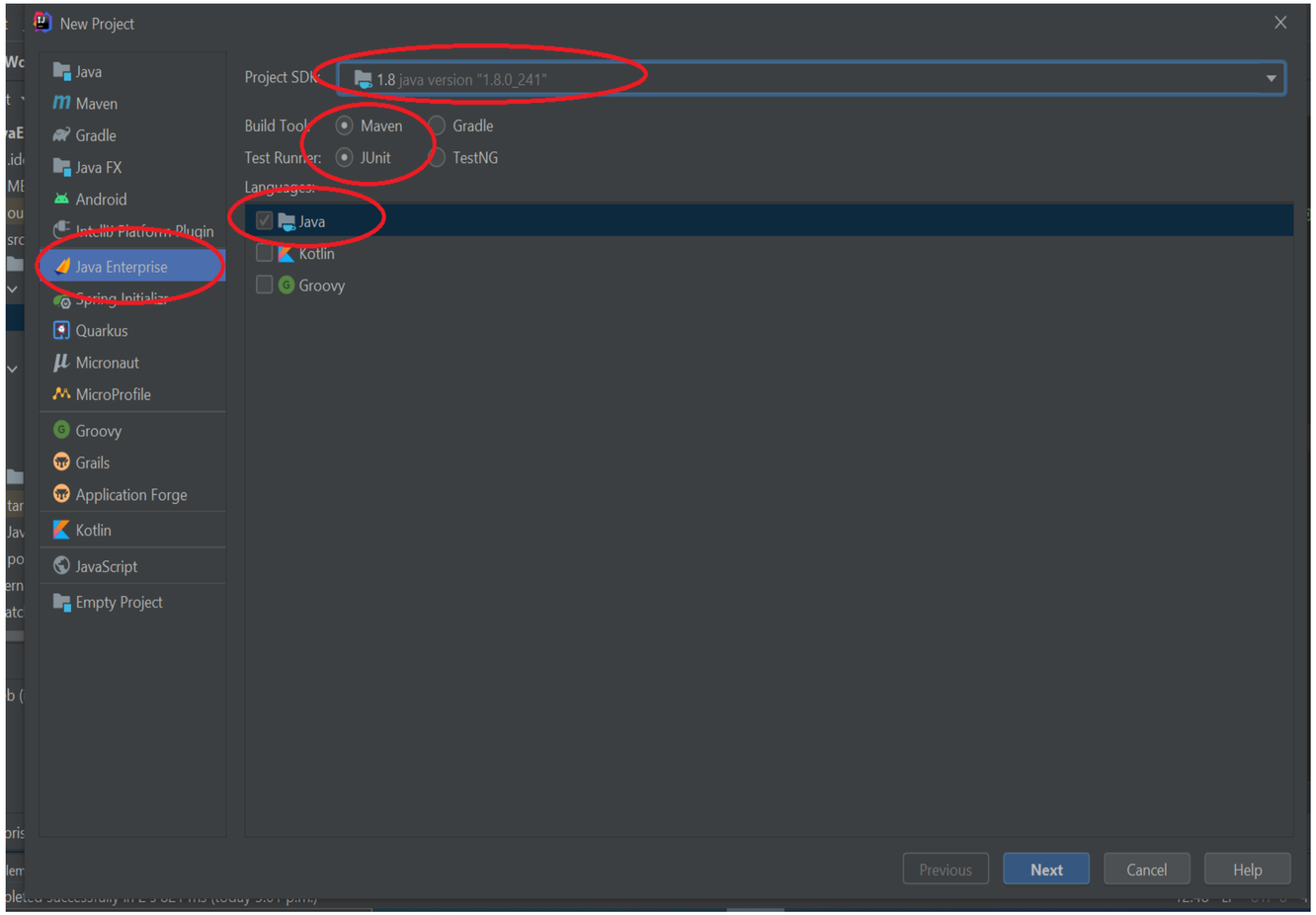



*Figure 2. Start a New Java Enterprise Project*

- From the **Libraries and Frameworks**, check **Servlet** (See Figure 3)
- Click **Next**
- Write a **Name** for the project (e.g. *HelloWorld*) (See Figure 4)
- Click **Finish**

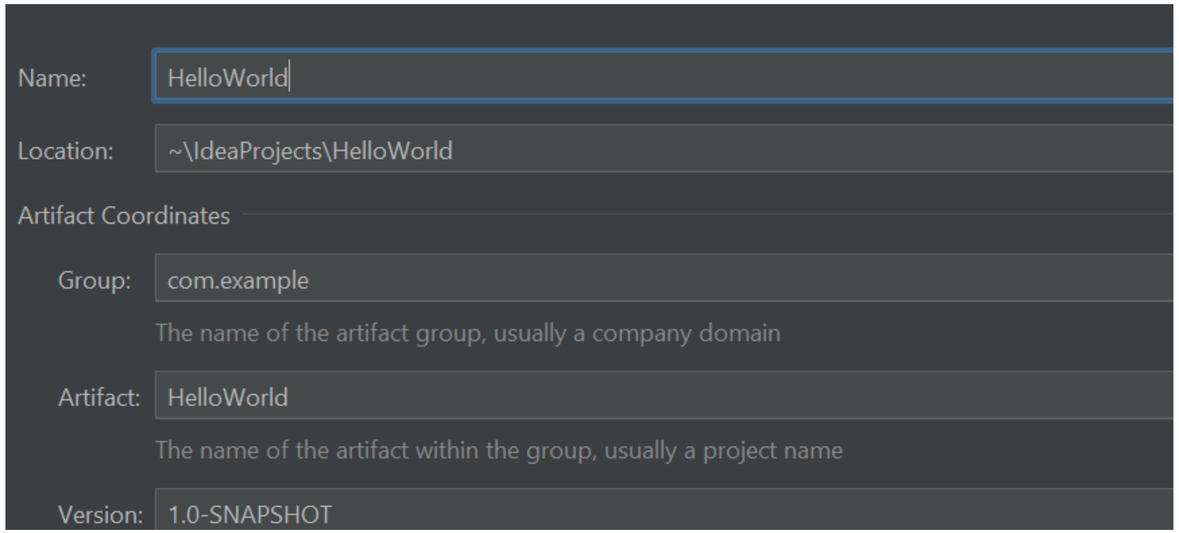



*Figure 4. Project Name*

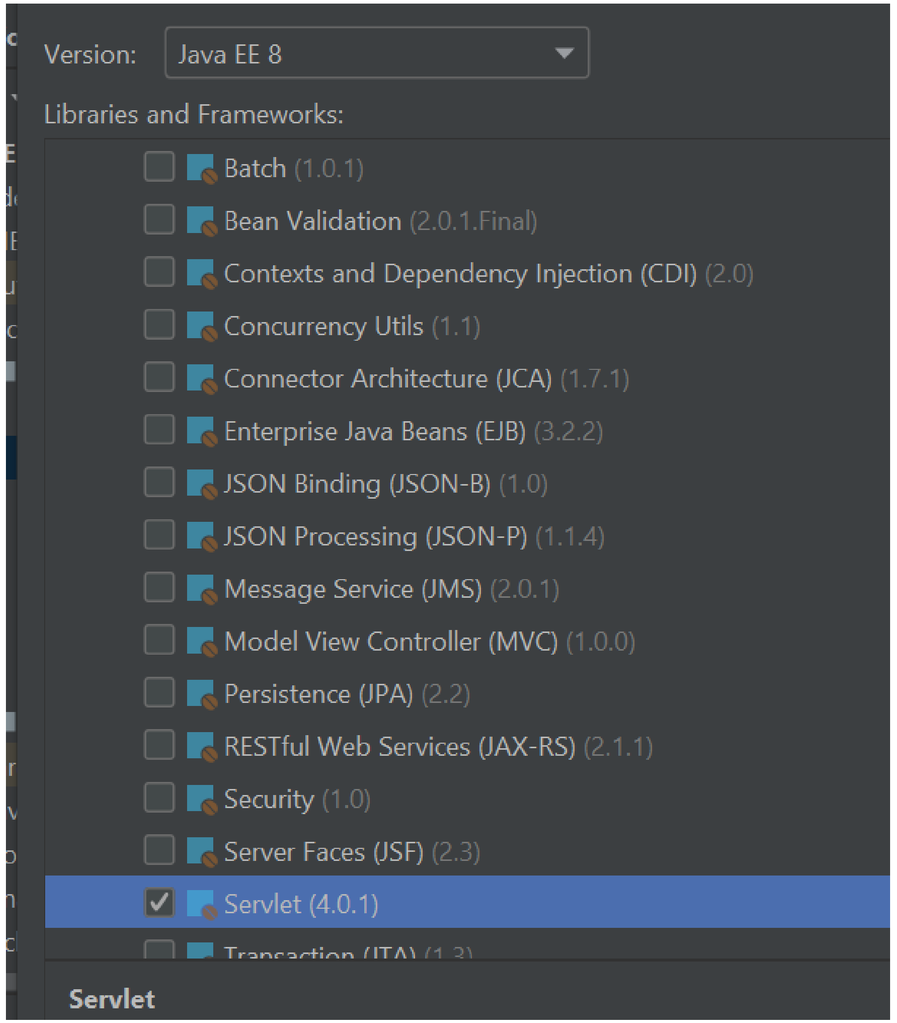



*Figure 3. Libraries & Frameworks Options*

- The complete directory structure required for the **Java Web Application** will be created automatically by **IntelliJ IDEA** (See Figure 5)
- For Tomcat Installation & Run Configuration in IntelliJ IDEA, refer to the end of **Tutorial 1 Java Web & Servlets** video on Moodle

  Note: if you want to access **Tomcat Apache**'s page click here

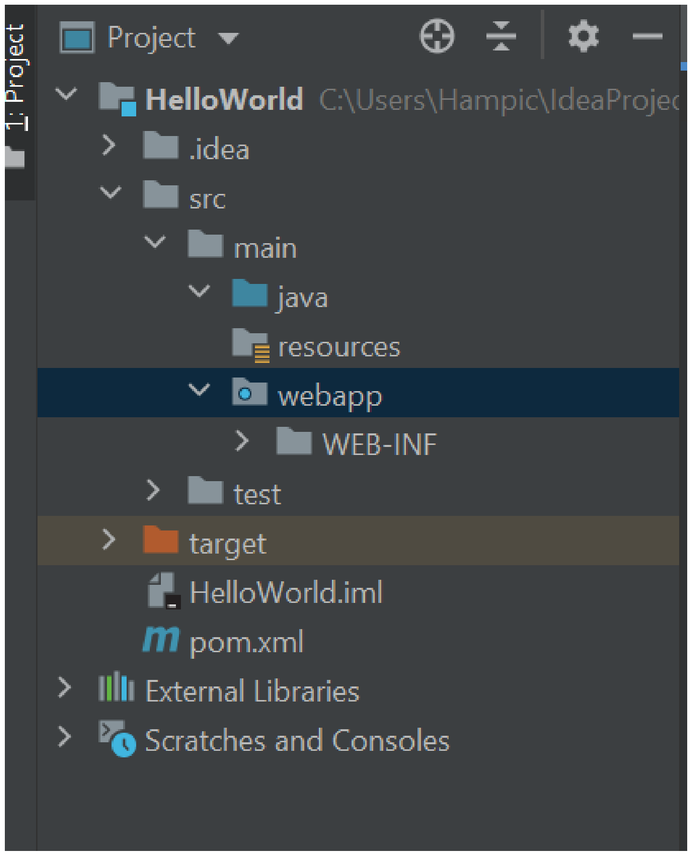



*Figure 5. Project Directory Structure*

# What is a Servlet?

# What is a 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 request
- Results returned to the client (HTML document, images, binary data)

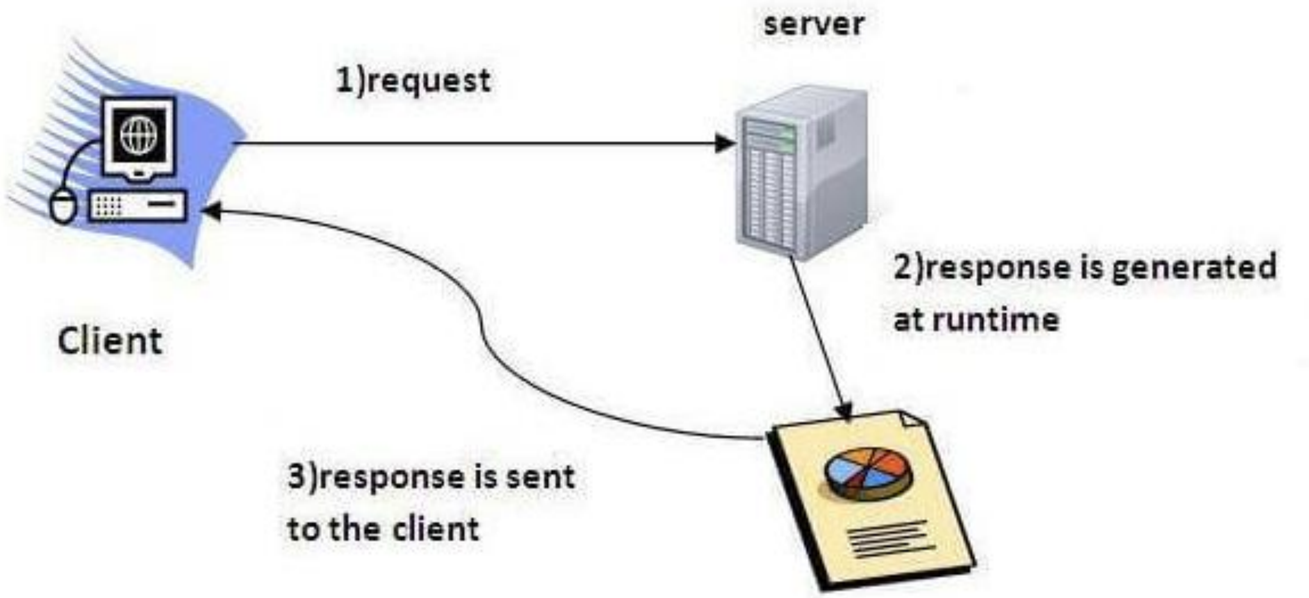



*Figure 6. The HTTP Request-Response Data Flow*

# Creating a Servlet

- Right click on **Java** and select **Create New Servlet** (See Figure 7)
- Write a **Name** for the **servlet** (e.g. *HelloWorldServlet*) (See Figure 8)
- Click **OK**

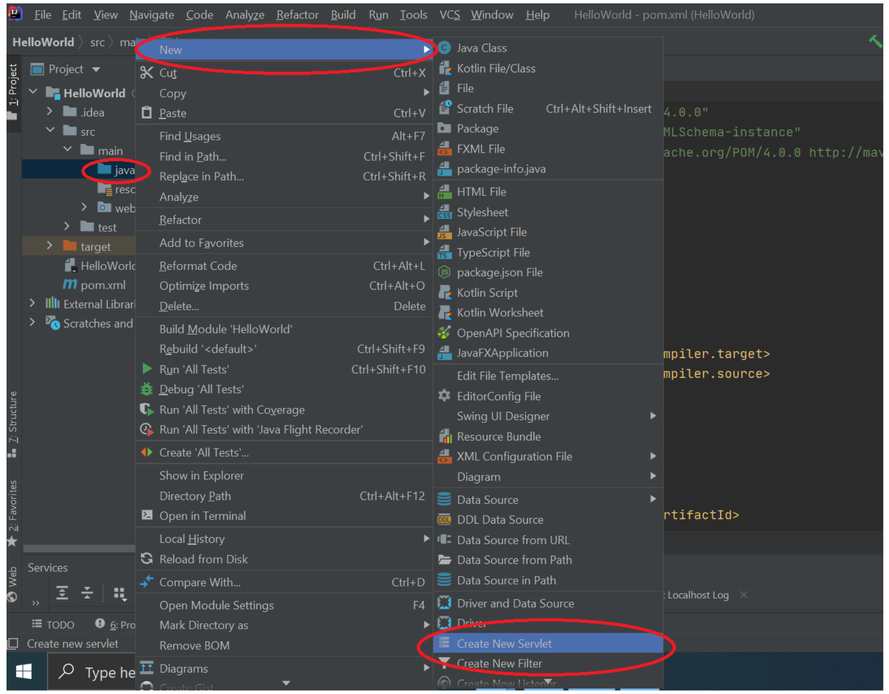



*Figure 7. Create a New Servlet*

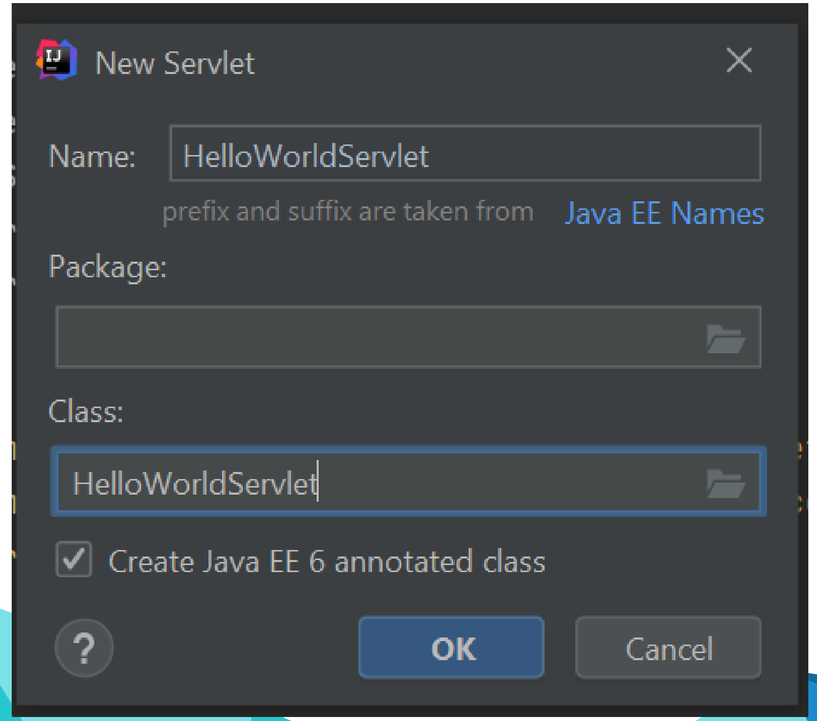



*Figure 8. Set Servlet Name*

Notes:

- By leaving the **Package** field empty, the servlet will reside in the **default package**
- By having the checkmark on **Create Java EE 6 annotated class**, it will generate the **WebServlet** annotation (See next slide)

# Servlet Class

- In the servlet class, **@WebServlet** annotation is used to declare a java class as a servlet.
- This servlet class must extend HttpServlet and override the methods (e.g. doGet & doPost)
  - **doGet(HttpServletRequest request, HttpServletResponse response):** used for getting information from the server.
  - **doPost(HttpServletRequest request, HttpServletResponse response):** used for sending information to the server.

```
@WebServlet(name = "HelloWorldServlet")
public class HelloWorldServlet extends HttpServlet {
    protected void doPost(HttpServletRequest request,

    }

    protected void doGet(HttpServletRequest request, H
        |
    }
}
```

*Figure 9. Autogenerated Servlet By IntelliJ IDEA*

- In the **doGet** method in **HelloWorldServlet**, add this piece of code:

```
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();
}
```

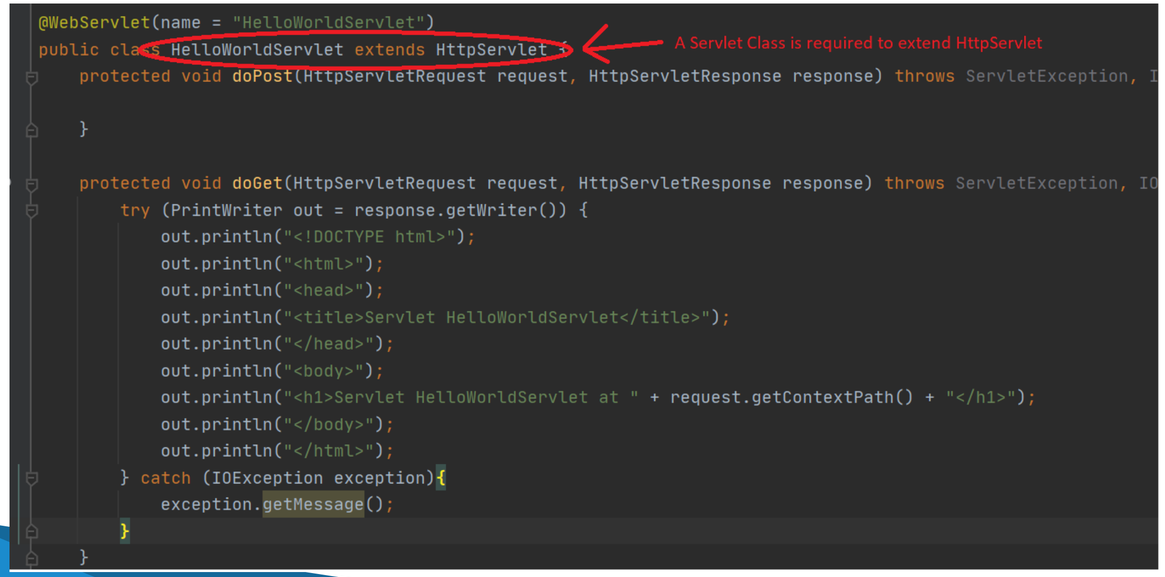



*Figure 10. Servlet after Adding the PrintWriter Code*

- Right click on the **webapp** folder (See Figure 11)
- Select **New → HTML File**
- Write a **Name** for the **HTML** file (e.g. *index*) (See Figure 12)
- Select **HTML 5 file**

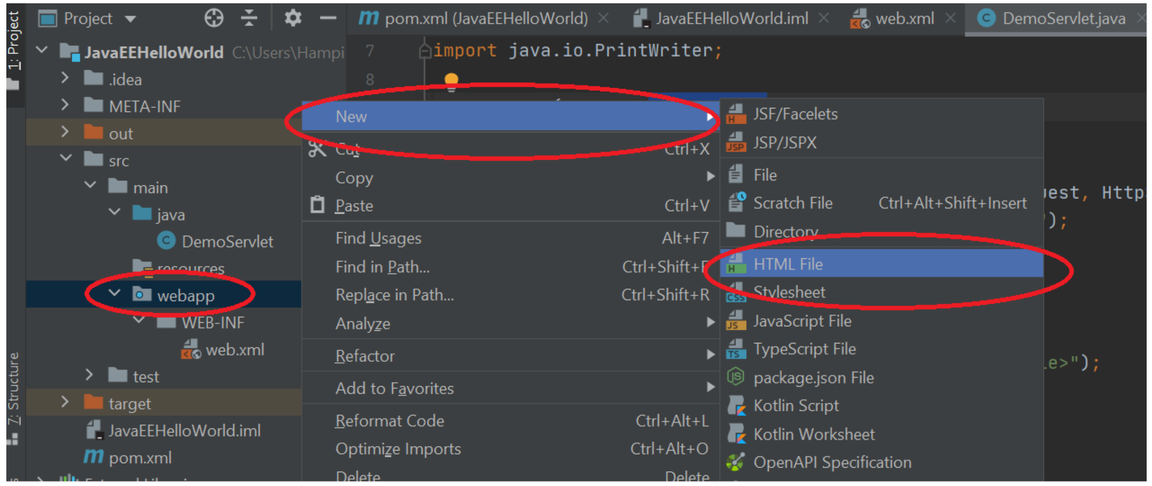



*Figure 11. Create the Home Page*

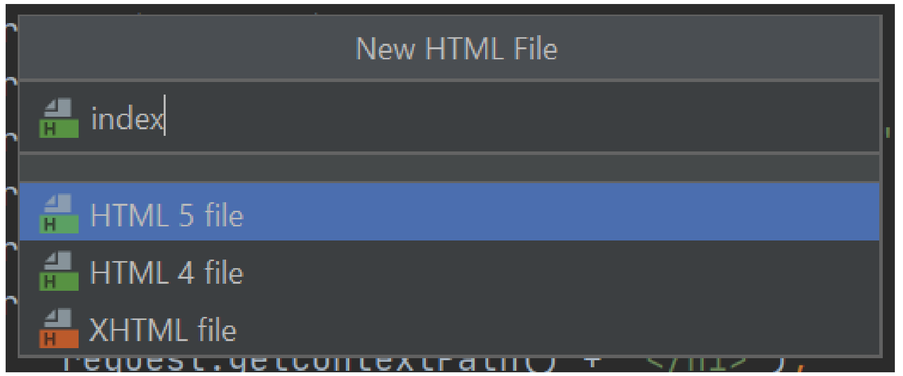



*Figure 12. Set Name & Type of the Home Page Html File*

- **Replace** the code inside the **index.html** file by the following code snippet:

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title>Hello World</title>
</head>
<body>
<div>Welcome to SOEN 387 Tutorial</div>
<div> Click here to go to <a href="HelloWorldServlet">Hello World</a></div>
</body>
</html>
```

- The tag <a href="HelloWorldServlet">Hello World</a> will invoke the servlet **HelloWorldServlet** by calling the **doGet** method implemented in this servlet.

- Open the **web.xml** file which resides in this directory *HelloWorld/src/main/webapp/WEB-INF/web.xml*

  (See Figure 13)

- Insert the code below inside the **<web-app></web-app>** element

```
<servlet>
        <servlet-name>HelloWorldServlet</servlet-name>
        <servlet-class>HelloWorldServlet</servlet-class>
</servlet>
<servlet-mapping>
            <servlet-name>HelloWorldServlet</servlet-name>
            <url-pattern>/HelloWorldServlet</url-pattern>
</servlet-mapping>
<welcome-file-list>
            <welcome-file>index.html</welcome-file>
</welcome-file-list>
```

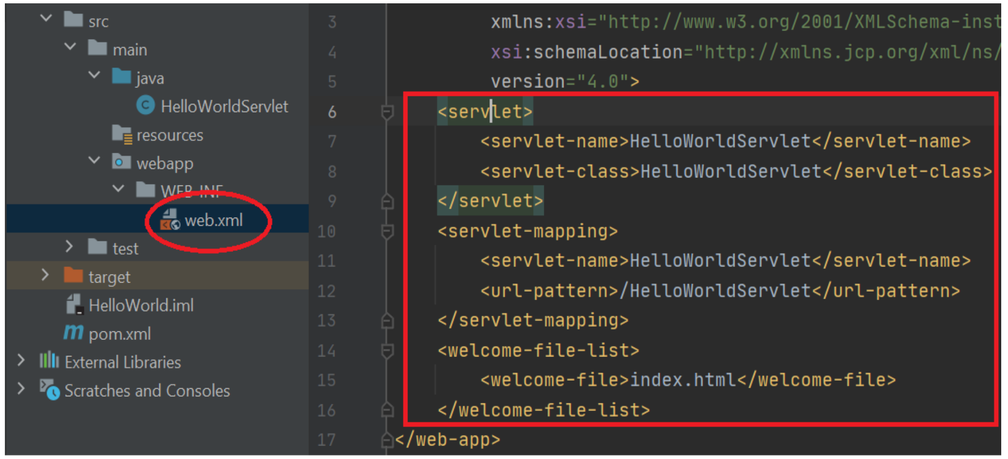



*Figure 13. Web.xml Content*

Notes:

- The **<servlet-mapping>** element defines a mapping between a **servlet** and a **URL** pattern.
- The **<welcome-file-list>** element is used to specify the files that need to be invoked by the server by default.

# Building & Running the Application

- Select your **Local Tomcat Server** that you previously created (See Figure 14)
- Click on the **Play** icon to run your web application
- A web page will pop up after your app builds (See Figure 15)
- Click on the **Hello World** link
    - It will call the servlet which will send back an HTML page as a **response** (See Figure 16).

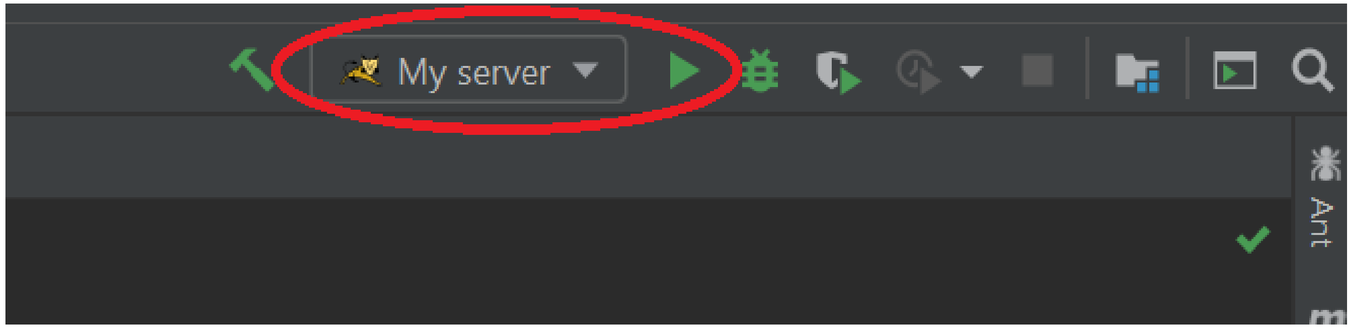



*Figure 14. Running Your Web Application*

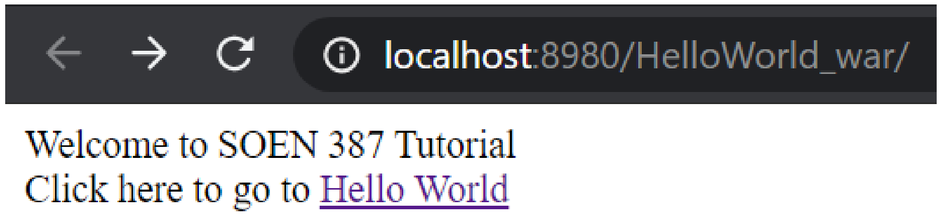



*Figure 15. index.html Displayed in the Browser*


*Figure 16. Calling the Servlet*

# Types of Java Application

## Web Application

Servlet technology is used to create **web applications** to fulfill user requests. It resides at server side and generates a dynamic web page. We have covered this in the previous slides.

## Console Application

Unlike web applications, **console applications** don't need a server to run. It is designed to be used through the command line interface.
Watch this video to see how you can create a simple console application.

## Class Library

**Java Class Library** is a collection of packages, classes and interfaces which can be used in any Java project. Many libraries already exist but you can create your own and use it in any project.

# Debugging a Servlet File

- Open the **HelloWorldServlet** file
- Place a **breakpoint** on any line in the **doGet** method by clicking on the margin (See Figure 17)

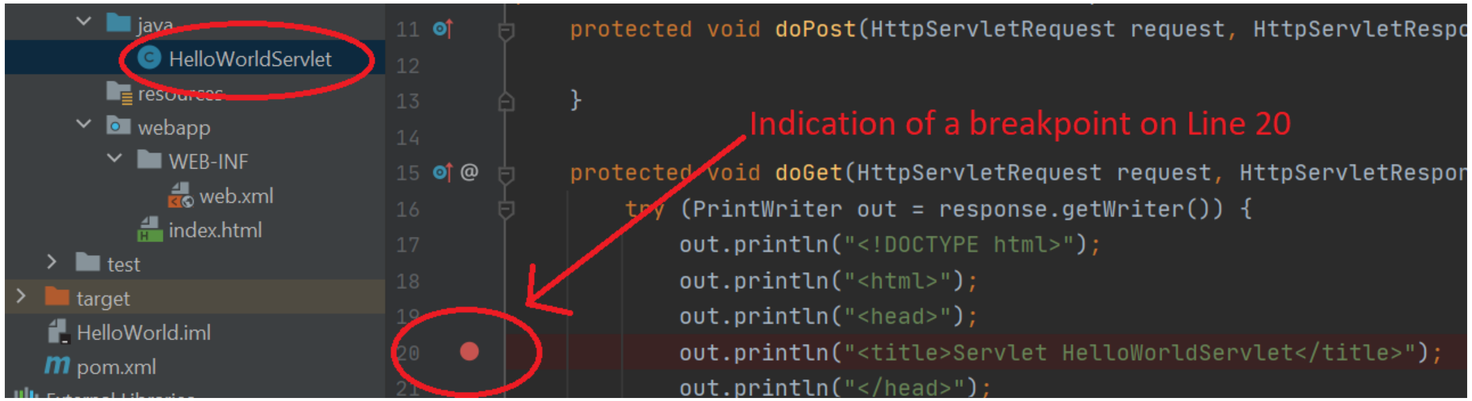



*Figure 17. Breakpoint Indication*

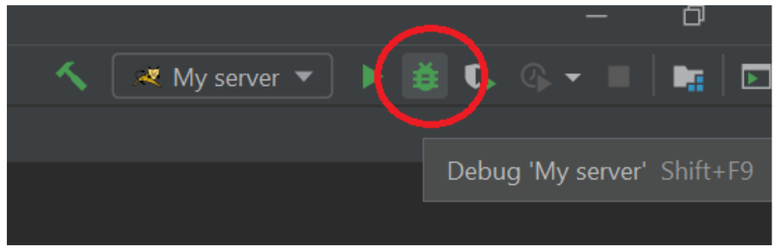



*Figure 18. Debug Button*

- Click on the **Debug** button to run the web application in **Debug Mode** (See Figure 18)
- Click the **Hello World** link on the web page and return on IntelliJ IDEA
- Note: The application will **run normally** until you click on the **Hello World** link which will then execute the **doGet** method in the **HelloWorldServlet**

Notes:

- When opening IntelliJ IDEA, you will see the debug tool window in the bottom half (See Figure 19).
- Under the **Variables Section**, you can view all the variables that currently exist in your application.
- At the top right in Figure 19, there are buttons to help you navigate the code during execution. You can find the description of their functionalities here.

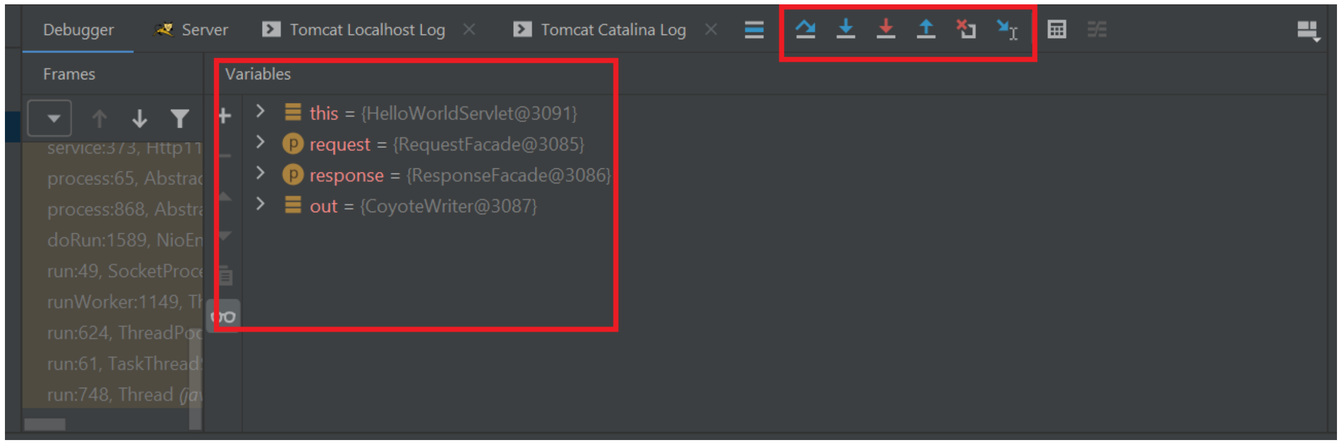



*Figure 19. Debug Tool Window*

# Thank You!

# References


- https://www.google.com/search?q=time+to+code+meme&tbm=isch&ved=2ahUKEwikx8KYktXrAhURe6wKHTQcD18Q2-cCegQIABAA&oq=time+to+code+meme&gs_lcp=CgNpbWcQAzoECAAQHlDyCljXGGCHGmgDcAB4AIABWogBiwWSAQE4mAEAoAEBqgELZ3dzLXdpei1pbWfAAQE&sclient=img&ei=GChVX6TbIZH2sQW0uLz4BQ&bih=678&biw=1422&rlz=1C1GCEA_enCA910CA910&hl=en#imgrc=jnSAnNnmgqBUrM
- https://www.javatpoint.com/servlet-tutorial
- https://i.redd.it/se6s384mdna41.jpg