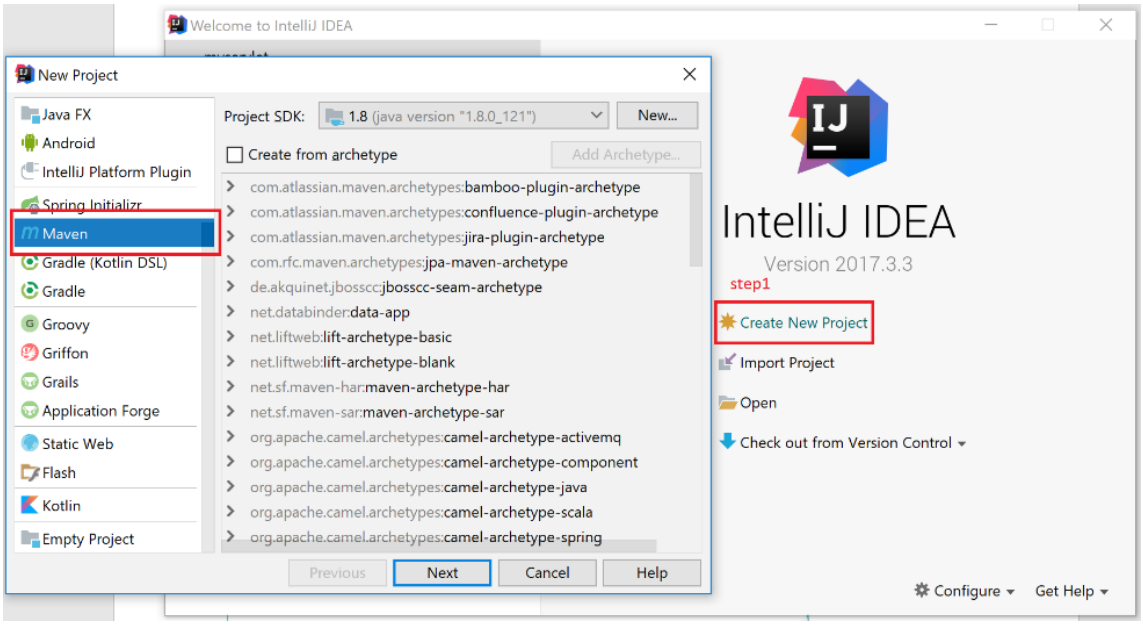
DEMO 1: First Servlet

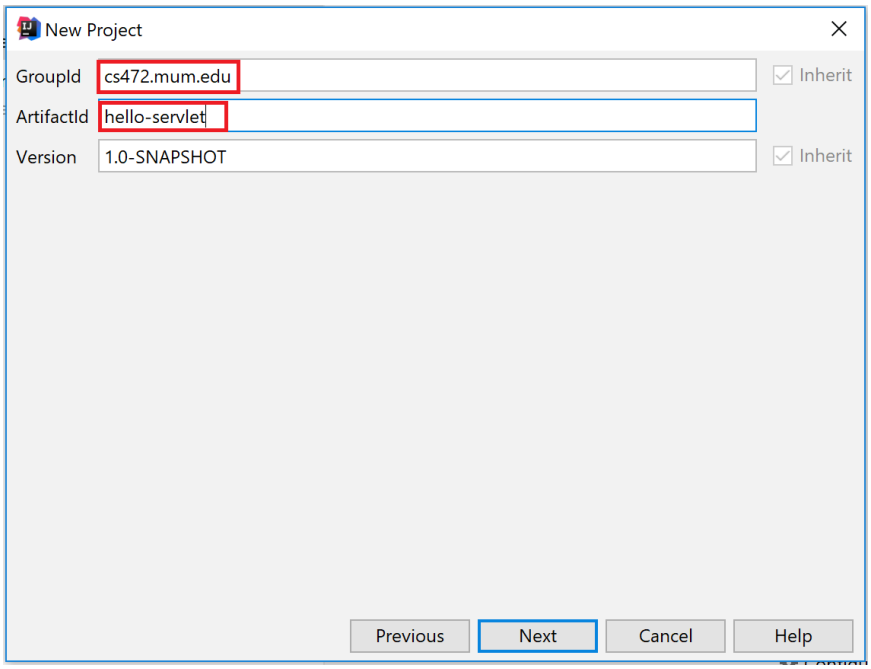
(instructions for IntelliJ IDEA using Tomcat)

Creating new maven based dynamic web project

Step 1



Step 2



Step 3

Next -> Finish

Step4

a. Add following properties in file `pom.xml`

```
<properties>
    <maven.compiler.source>1.8</maven.compiler.source>
    <maven.compiler.target>1.8</maven.compiler.target>
</properties>
```

Notes:

- Copy above snipped between before project closing`</project>`

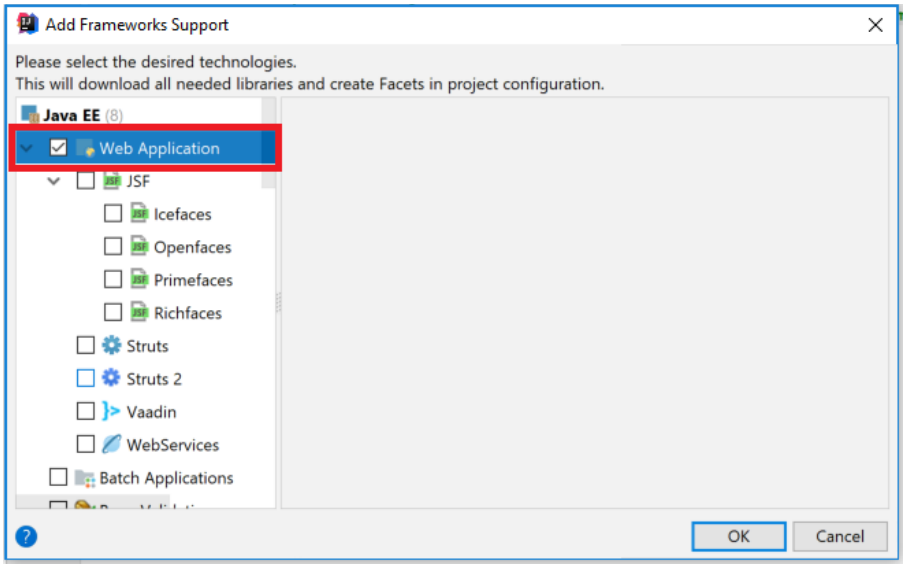
b. Next, copy the following maven dependencies after the properties

```
        <dependencies>
                <dependency>
                        <groupId>javax.servlet</groupId>
                        <artifactId>javax.servlet-api</artifactId>
                        <version>3.1.0</version>
                        <scope>provided</scope>
                </dependency>
                <dependency>
                        <groupId>javax.el</groupId>
                        <artifactId>javax.el-api</artifactId>
                        <version>3.0.0</version>
                        <scope>provided</scope>
                </dependency>

                <!-- https://mvnrepository.com/artifact/jstl/jstl -->
                <dependency>
                        <groupId>jstl</groupId>
                        <artifactId>jstl</artifactId>
                        <version>1.2</version>
                </dependency>

                <!--Required for Custom Tags -->
                <dependency>
                    <groupId>javax.servlet.jsp</groupId>
                    <artifactId>javax.servlet.jsp-api</artifactId>
                    <version>2.3.1</version>
                    <scope>provided</scope>
                </dependency>

        </dependencies>
```

c. Right click **project**
    I. Maven -> Reimport

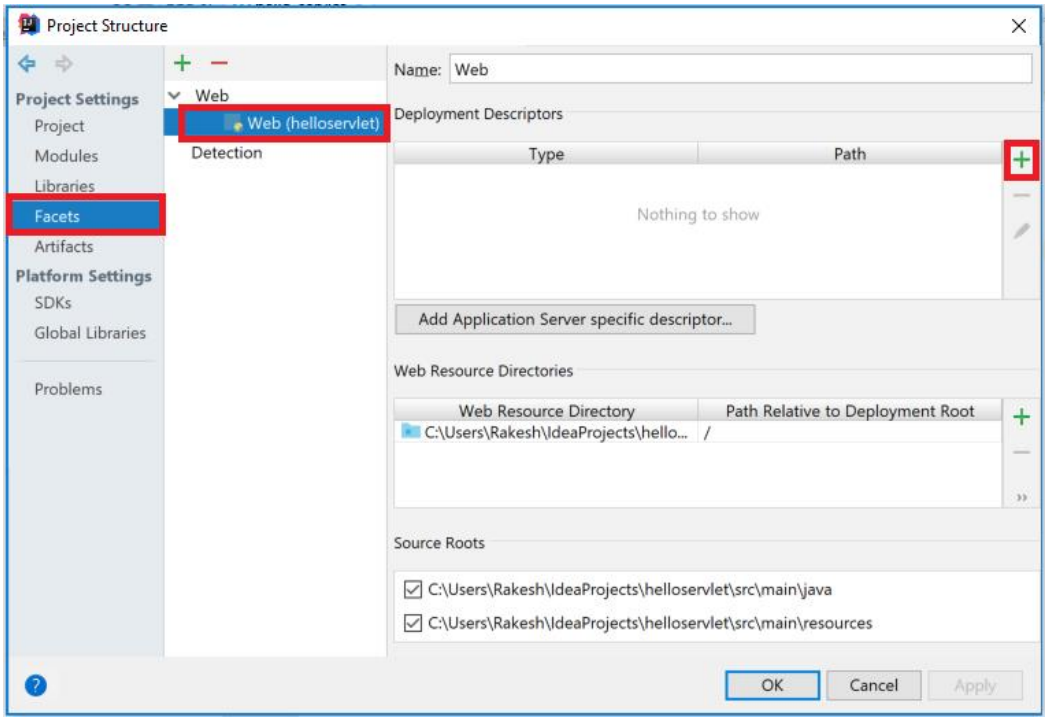
Step 5

Right click project -> Add Framework Support…



Step 6

a. File -> Project Structure -> Facets



b. Click `WEB-INF/web.xml` to auto create deployment descriptor.

Creating first servlet

1. Right click **`src/main/java`** folder in the Java Resources folder create a new Java Class '`HelloServlet`'

2. Extend `HttpServlet` class

3. Override `doGet()` method

3. Replace the stub code in doGet with following one line of code:

```
resp.getWriter().println("Hello, World!");
```

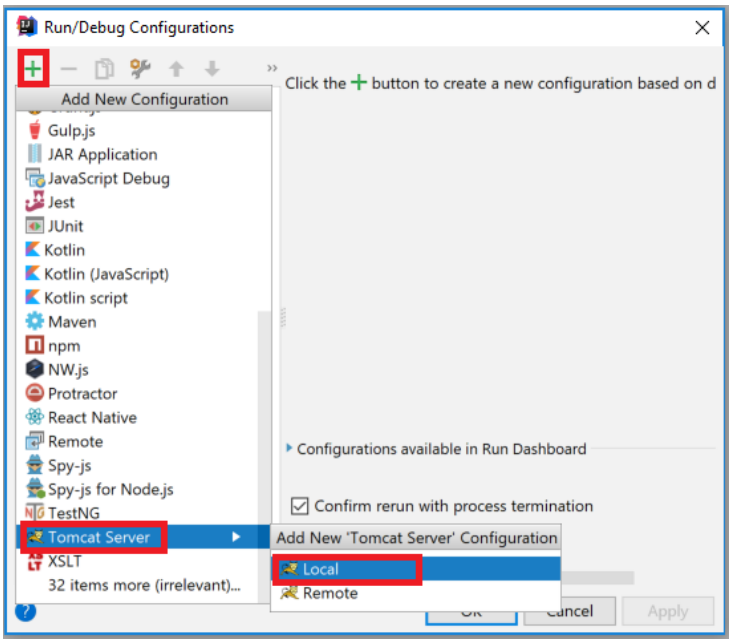
4. Add url-sevlet mapping in web.xml as below

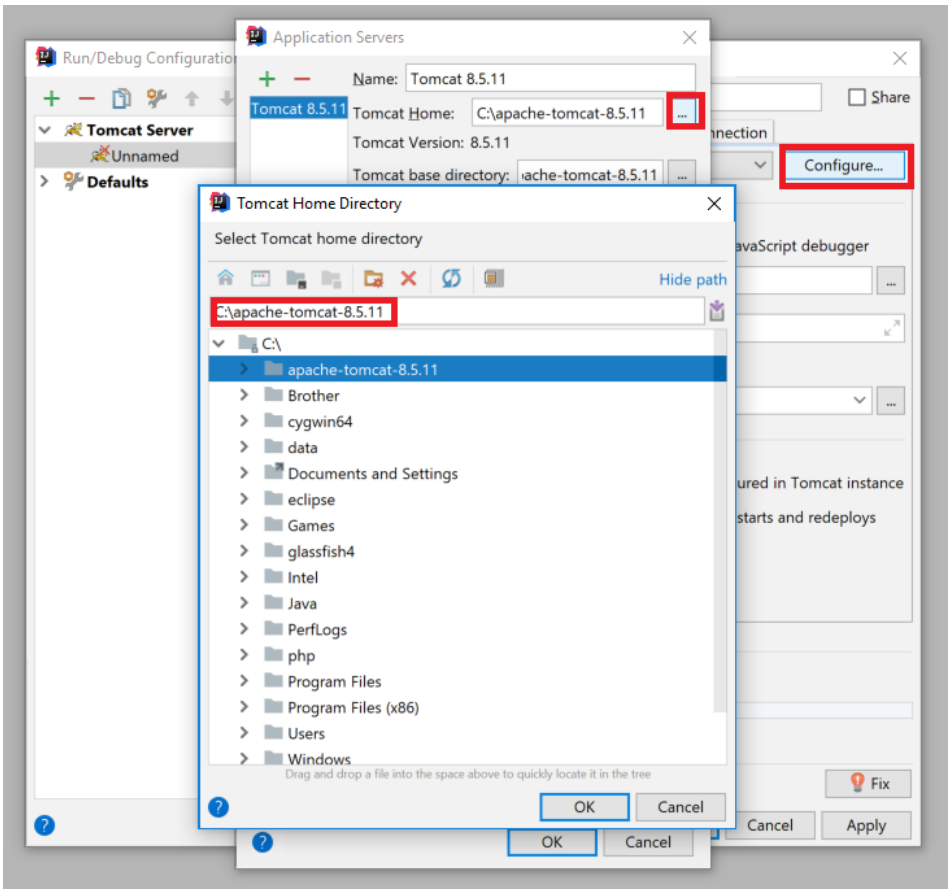
```
<servlet>
    <servlet-name>myservlet</servlet-name>
    <servlet-class>HelloServlet</servlet-class>
</servlet>
<servlet-mapping>
    <servlet-name>myservlet</servlet-name>
    <url-pattern>/hello</url-pattern>
</servlet-mapping>
```

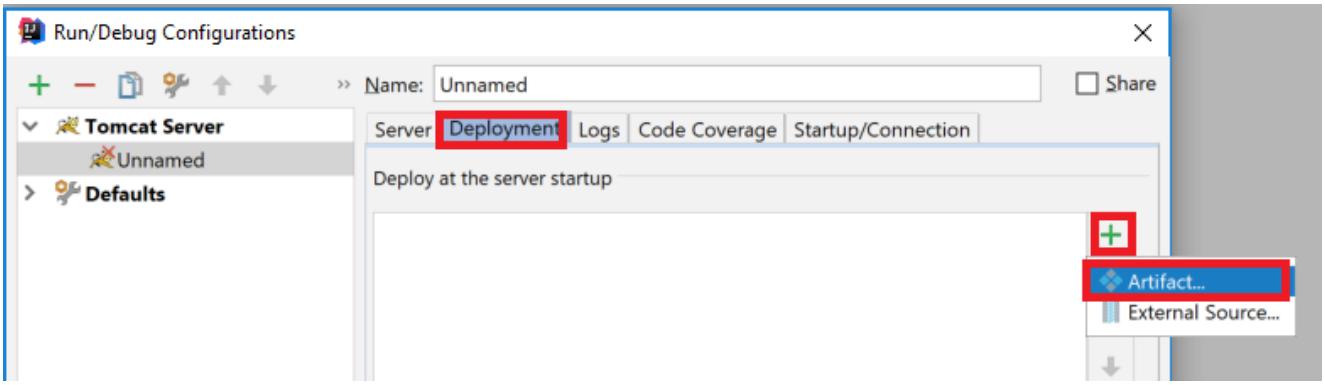
4. Now your first application is ready to run. (run it on Tomcat), but before that you have to configure Tomcat.

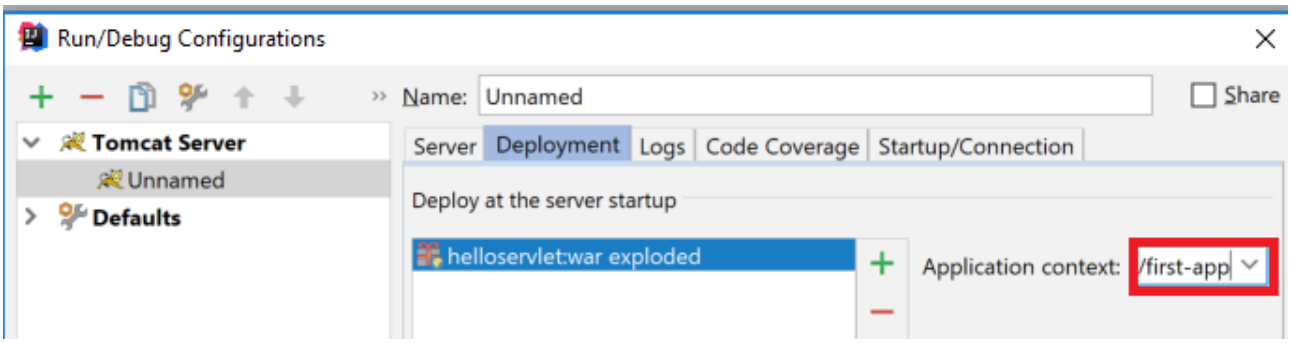
a. Download and extract Tomcat to your machine (e.g. on C:\ drive)

b. Click drop-down at top right corner of IntelliJ IDEA and click on edit configuration




Run/Debug Configuration
Application Servers
Name: Tomcat 8.5.11
Tomcat 8.5.11
Tomcat Home: C:\apache-tomcat-8.5.11
Tomcat Version: 8.5.11
Tomcat base directory:
Tomcat Server
Unnamed
Defaults
Share
Configure...
Tomcat Home Directory
Select Tomcat home directory
Hide path
C:\apache-tomcat-8.5.11
C:\
apache-tomcat-8.5.11
Brother
cygwin64
data
Documents and Settings
eclipse
Games
glassfish4
Intel
Java
PerfLogs
php
Program Files
Program Files (x86)
Users
Windows
Drag and drop a file into the space above to quickly locate it in the tree
OK
Cancel
Fix
Apply


Run/Debug Configurations
Name: Unnamed
Share
Tomcat Server
Unnamed
Defaults
Server
Deployment
Logs
Code Coverage
Startup/Connection
Deploy at the server startup
Artifact...
External Source...


Run/Debug Configurations
Name: Unnamed
Share
Tomcat Server
Unnamed
Defaults
Server
Deployment
Logs
Code Coverage
Startup/Connection
Deploy at the server startup
helloservlet:war exploded
Application context: /first-app

Create second servlet

1. Put the following in the index.jsp (html) file

```
<a href='simple'>Start the simplest servlet app</a>
```

2. Map this servlet to url `/simple`

3. **Replace** the doGet method contents with following lines of code:

```
PrintWriter out = resp.getWriter();
out.print("<html><head><title>Test</title></head><body>");
out.print("<form method='post'>");
out.print("<p>Please click the button</p>");
out.print("<input type='submit' value='Click me'/>");
out.print("</form>");
out.print("</body></html>");
```

4. **Replace** the doPost method contents with following lines of code:

```
PrintWriter out = resp.getWriter();
out.print("<html><head><title>Test</title></head><body>");
out.print("<p>Postback received</p>");
out.print("</body></html>");
```

5. Run application on Tomacat