

CS359 – Web Programming

**Web Programming**  
**Netbeans 12.1 – Apache Tomcat 9**

# OVERVIEW

- Installation & Requirements
- **Open & Start** Apache Tomcat Server
- **Create** a Maven Web Application Project
- **Show** Project Structure
- Modification of Projects's/IDE's **On Save** Changes
- **Create & Run** a Servlet
- Add Dependencies to pom.xml
- Maven Online Repository

# Requirements (1/5)

- **If you already have a JDK installed you can skip this step**
- In case you get an error “No compatible JDK was found.” or something similar you have to install a JDK to solve it.
- Any JDK may work, but in our case we are using OpenJDK <https://adoptopenjdk.net/> version 11.0.9\_11
  - Windows
    - run the installer
  - Ubuntu
    - cd Downloads
    - tar xf OpenJDK11U-jdk\_x64\_linux\_hotspot\_11.0.9\_11.tar.gz
    - A new folder is now created at /home/**username**/Downloads/jdk-11.0.9+11

# Requirements (2/5)

- Download Netbeans 12.1

- <https://netbeans.apache.org/download/nb121/nb121.html>

## Downloading Apache NetBeans 12.1

Apache NetBeans 12.1 was released on September 1, 2020. See [Apache NetBeans 12.1 Features](#) for a full list of features.

Apache NetBeans 12.1 is available for download from your closest Apache mirror.

- Binaries: [netbeans-12.1-bin.zip](#) ( SHA-512, PGP ASC)
- Installers:

- [Apache-NetBeans-12.1-bin-windows-x64.exe](#) ( SHA-512, PGP ASC)
- [Apache-NetBeans-12.1-bin-linux-x64.sh](#) ( SHA-512, PGP ASC)
- [Apache-NetBeans-12.1-bin-macosx.dmg](#) ( SHA-512, PGP ASC)

**Pick your OS installer**



# Requirements (3/5)

- Windows
  - Run Apache-NetBeans-12.1-bin-windows-x64.exe
- Linux
  - Navigate to Downloads folder:
    - `cd ~/Downloads`
  - Make the binary executable
    - `chmod +x Apache-NetBeans-12.1-bin-linux-x64.sh`
  - Execute the Installer
    - `./Apache-NetBeans-12.1-bin-linux-x64.sh`

# Requirements (4/5)

Install the Apache NetBeans IDE to:  
C:\Program Files\NetBeans-12.1 Browse...

JDK™ for the Apache NetBeans IDE:  
 Browse...

Insert the path to the OpenJDK

Install the Apache NetBeans IDE to:  
C:\Program Files\NetBeans-12.1 Browse...

JDK™ for the Apache NetBeans IDE:  
C:\Program Files\AdoptOpenJDK\jdk-11.0.9.11-hotspot Browse...

Probably for Windows this will be the path

For Linux  
/home/**username**/Downloads/jdk-11.0.9+11

**And keep clicking next**

# Requirements (5/5)

- Download the latest Tomcat version
  - <https://tomcat.apache.org/download-90.cgi>

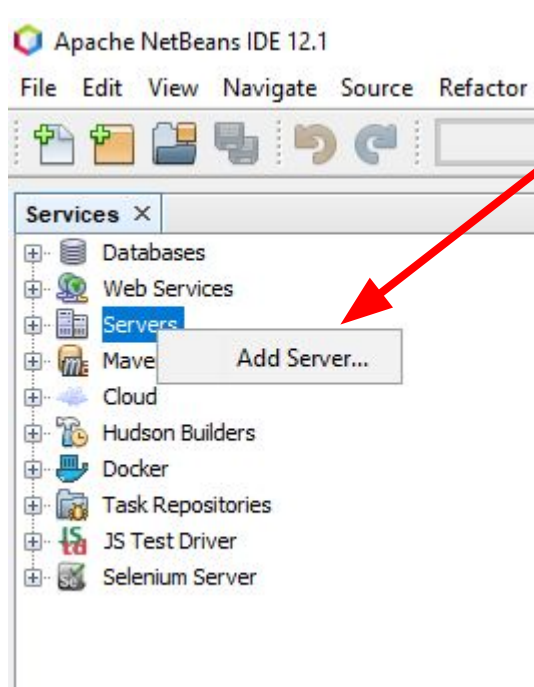
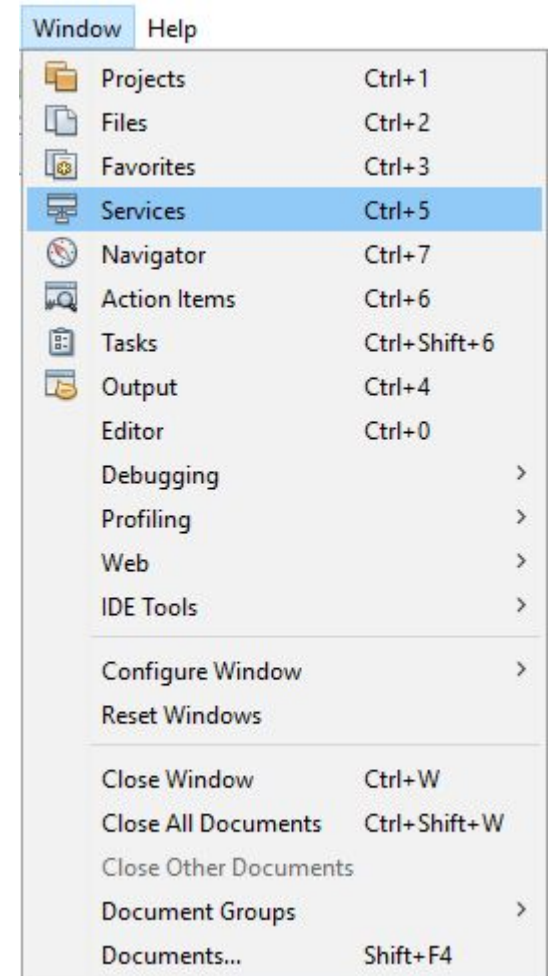
- Core:

- [zip](#) (pgp, sha512)
- [tar.gz](#) (pgp, sha512)
- [32-bit Windows zip](#) (pgp, sha512)
- [64-bit Windows zip](#) (pgp, sha512)
- [32-bit/64-bit Windows Service Installer](#) (pgp, sha512)

- Download the zip version (works for all the OSes)
- Extract it at your given path
  - On windows extract it on Desktop
  - On Linux
    - cd Downloads
    - unzip apache-tomcat-9.0.39.zip
- On both cases a new folder is created called **apache-tomcat-9.0.39**
- On Linux, cd apache-tomcat-9.0.39/bin/
- chmod +x \*.sh

# Install Tomcat (1/3)

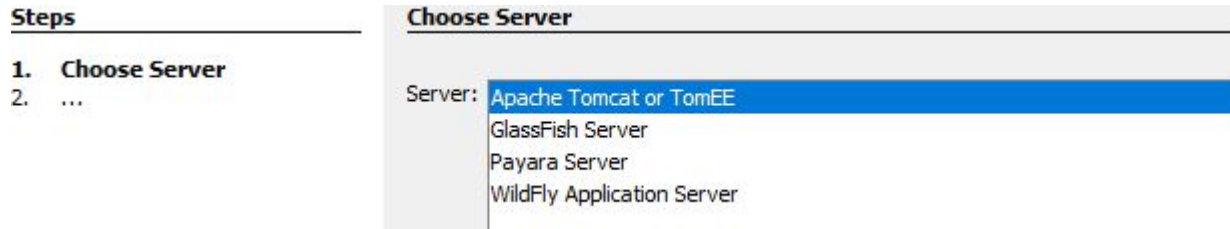
- Start Netbeans
- Click on Tab **Window**
  - Click on **Services**
- On the left, right click on Servers
  - Add Server



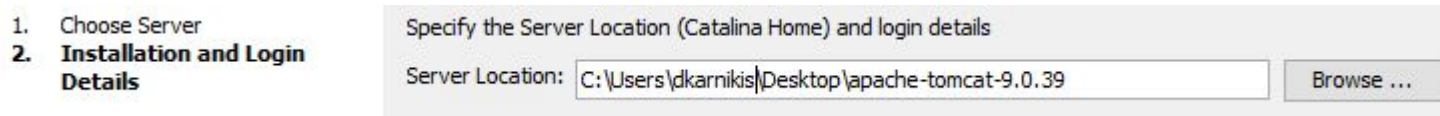


## Install Tomcat (2/3)

- On the new Window select Apache Tomcat or TomEE



- On Server Location load the folder of tomcat we previously downloaded



# Install Tomcat (3/3)

1. Choose Server
2. **Installation and Login Details**

Specify the Server Location (Catalina Home) and login details

Server Location:

☒ Use Private Configuration Folder (Catalina Base)

Catalina Base:

Enter the credentials of an existing user in the manager or manager-script role

Username:

Password:

☒ Create user if it does not exist

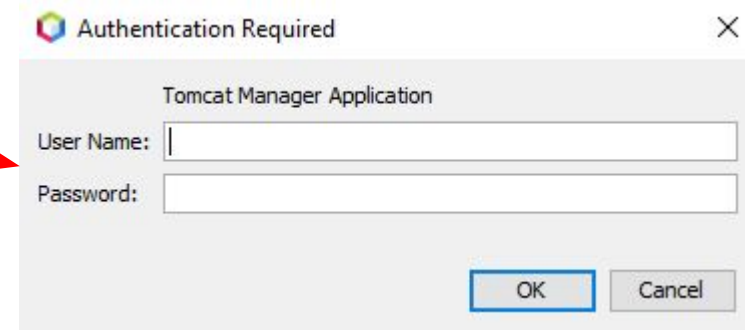
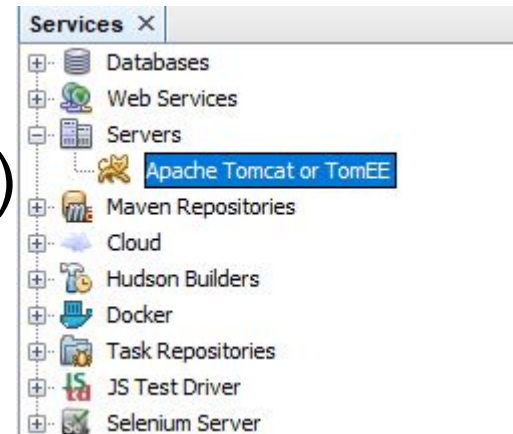
Check this and select  
as Cataline Base an  
empty folder

admin as username  
and password

check this and finish

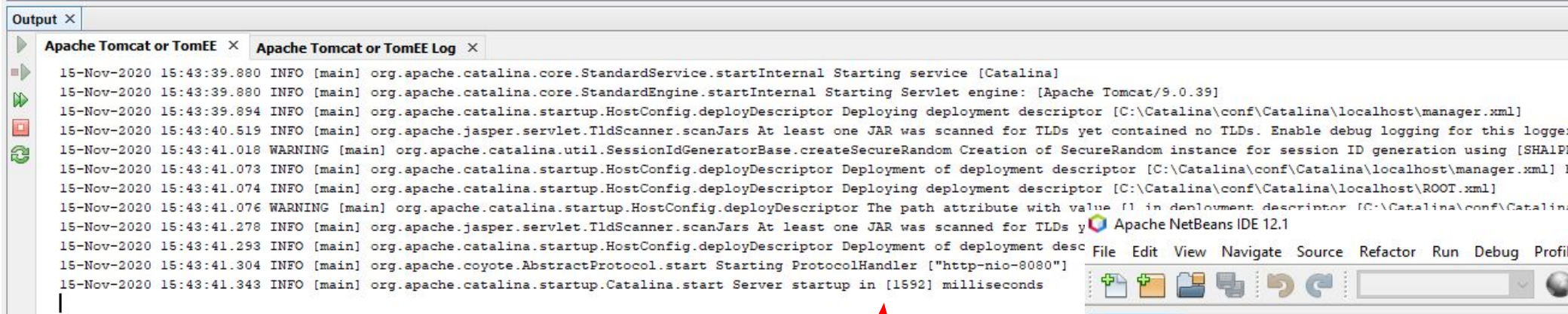
# Start Apache Tomcat Server (1/2)

- From **Windows** => **Services** tab
- Click on Servers on the left (on the + sign)
- Right click on Apache -> start
- Click Allow access to any firewall (Windows)
- enter admin admin
- Click on Window -> Output to show Tomcat's log



# Start Apache Tomcat Server (2/2)

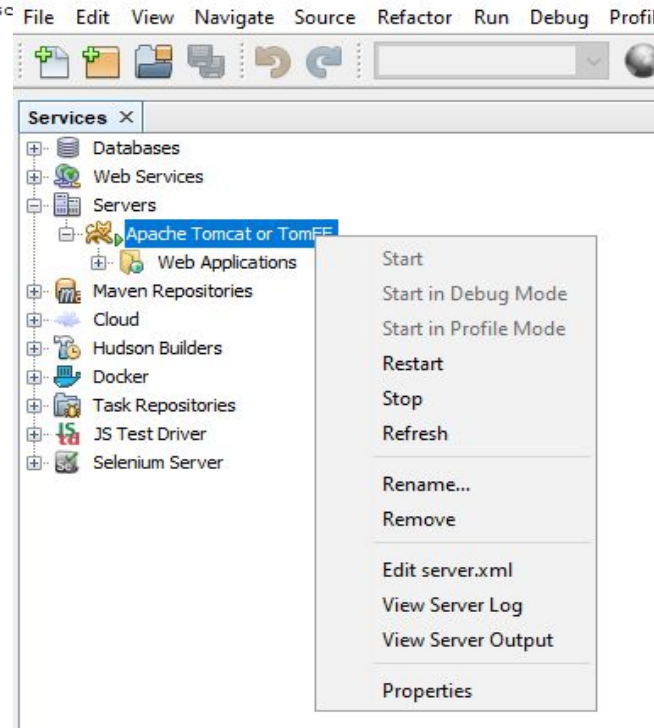
- On the bottom of your screen you should see something like this



The screenshot shows the Apache NetBeans IDE 12.1 interface. The 'Output' window is open, displaying logs for 'Apache Tomcat or TomEE'. The logs show the server starting successfully, including messages like 'Starting service [Catalina]', 'Starting Servlet engine: [Apache Tomcat/9.0.39]', and 'Server startup in [1592] milliseconds'. A red arrow points from the 'Services' window to the 'View Server Output' option in the context menu.

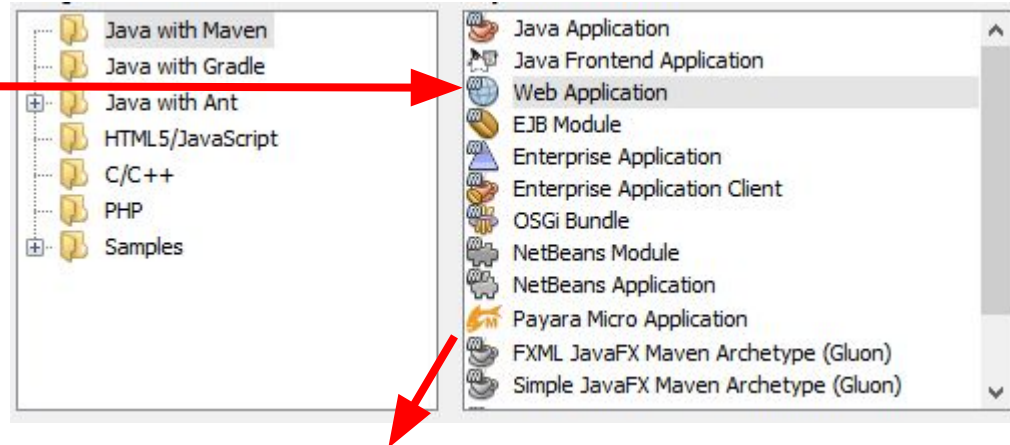
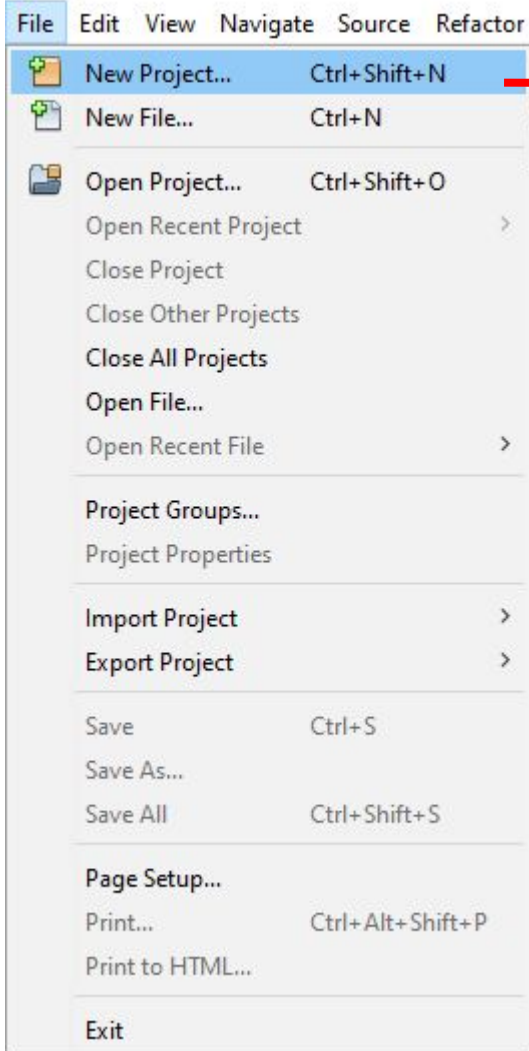
```
15-Nov-2020 15:43:39.880 INFO [main] org.apache.catalina.core.StandardService.startInternal Starting service [Catalina]
15-Nov-2020 15:43:39.880 INFO [main] org.apache.catalina.core.StandardEngine.startInternal Starting Servlet engine: [Apache Tomcat/9.0.39]
15-Nov-2020 15:43:39.894 INFO [main] org.apache.catalina.startup.HostConfig.deployDescriptor Deploying deployment descriptor [C:\Catalina\conf\Catalina\localhost\manager.xml]
15-Nov-2020 15:43:40.519 INFO [main] org.apache.jasper.servlet.TldScanner.scanJars At least one JAR was scanned for TLDs yet contained no TLDs. Enable debug logging for this logger...
15-Nov-2020 15:43:41.018 WARNING [main] org.apache.catalina.util.SessionIdGeneratorBase.createSecureRandom Creation of SecureRandom instance for session ID generation using [SHA1PRNG]
15-Nov-2020 15:43:41.073 INFO [main] org.apache.catalina.startup.HostConfig.deployDescriptor Deployment of deployment descriptor [C:\Catalina\conf\Catalina\localhost\manager.xml]
15-Nov-2020 15:43:41.074 INFO [main] org.apache.catalina.startup.HostConfig.deployDescriptor Deploying deployment descriptor [C:\Catalina\conf\Catalina\localhost\ROOT.xml]
15-Nov-2020 15:43:41.076 WARNING [main] org.apache.catalina.startup.HostConfig.deployDescriptor The path attribute with value {} in deployment descriptor [C:\Catalina\conf\Catalina\localhost\ROOT.xml]
15-Nov-2020 15:43:41.278 INFO [main] org.apache.jasper.servlet.TldScanner.scanJars At least one JAR was scanned for TLDs yet contained no TLDs. Enable debug logging for this logger...
15-Nov-2020 15:43:41.293 INFO [main] org.apache.catalina.startup.HostConfig.deployDescriptor Deployment of deployment descriptor [C:\Catalina\conf\Catalina\localhost\ROOT.xml]
15-Nov-2020 15:43:41.304 INFO [main] org.apache.coyote.AbstractProtocol.start Starting ProtocolHandler ["http-nio-8080"]
15-Nov-2020 15:43:41.343 INFO [main] org.apache.catalina.startup.Catalina.start Server startup in [1592] milliseconds
```

- If The output is empty
  - Right click on Apache Tomcat or TomEE
    - View Server Output
- This indicates that the server is running!



# Create a Maven Web Application Project

Apache NetBeans IDE 12.1



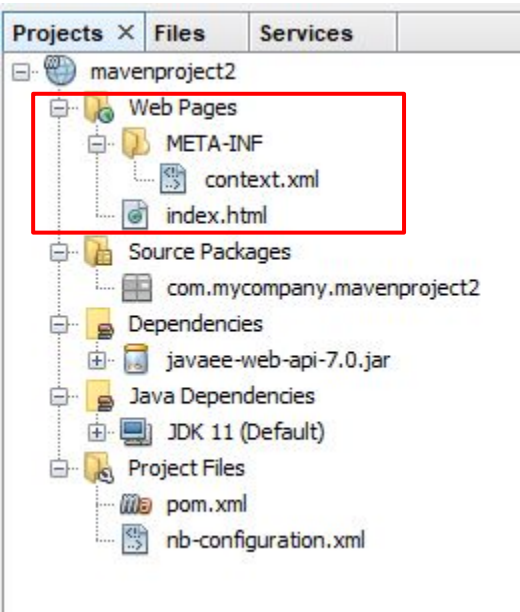
Project Name:	mavenproject2
Project Location:	C:\Users\dkarnikis\Documents\NetBeansProjects
Project Folder:	C:\Users\dkarnikis\Documents\NetBeansProjects\mavenproject2
Artifact Id:	mavenproject2
Group Id:	com.mycompany
Version:	1.0-SNAPSHOT
Package:	com.mycompany.mavenproject2

Choose any path you like

Server:	Apache Tomcat or TomEE
Java EE Version:	Java EE 7 Web

then click Finish

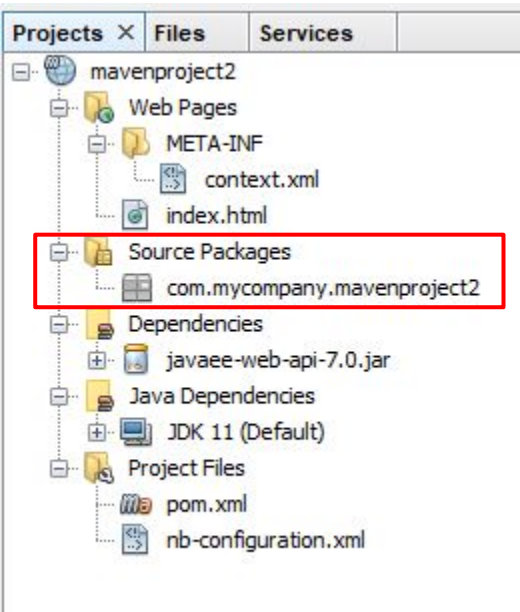
# Show Project Structure



The project folder that html,  
jss, css will be stored



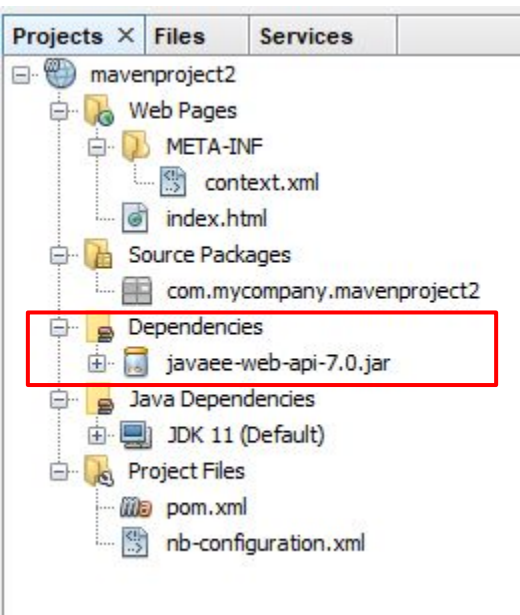
# Show Project Structure



Contains all the server side files (servlets, javabeans)



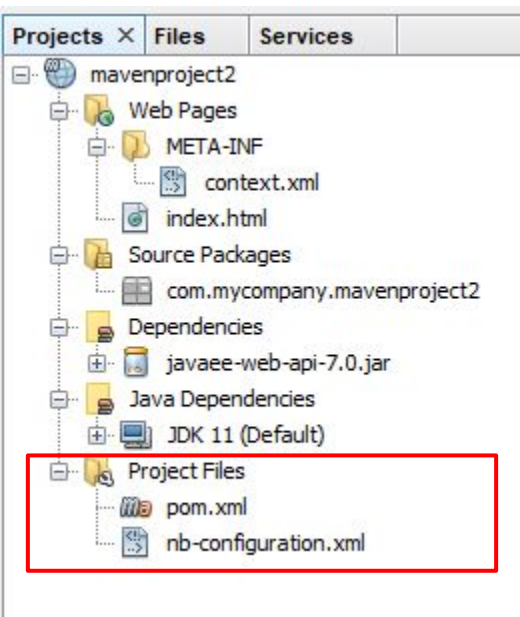
# Show Project Structure



Dependencies contains all the dependency artifacts of the project (jar)



# Show Project Structure



pom.xml is a single configuration file that contains the majority of information required to build a project in just the way you want

# Modification of Project's On Save Changes (1/2)

The image shows the 'Run' configuration dialog for a Maven project in an IDE. The 'Categories' list on the left has 'Run' selected. The right pane shows the configuration for running the project, including the server, Java EE version, context path, and browser. The 'Display Browser on Run' checkbox is checked, and the 'Always perform build before running application' checkbox is unchecked. The 'OK' button is highlighted at the bottom.

**Categories:**

- General
- Sources
- Configurations
- Frameworks
- JavaScript Libraries
  - npm
  - Bower
  - CDNJS
- CSS Preprocessors
- Build
  - Compile
- Run**
- Actions
- JavaScript
  - RequireJs
  - Orade JET
- License Headers
- Formatting
  - CheckStyle Formatting
- Hints

**Server :** Apache Tomcat or TomEE

**Java EE Version :** Java EE 7 Web

**Context Path :** /mavenproject2  
(e.g. /admin/login.jsp)

**Relative URL :**

**Browser:** IDE's default browser

☒ Display Browser on Run

☒ Copy Static Resources on Save

☒ Deploy on Save

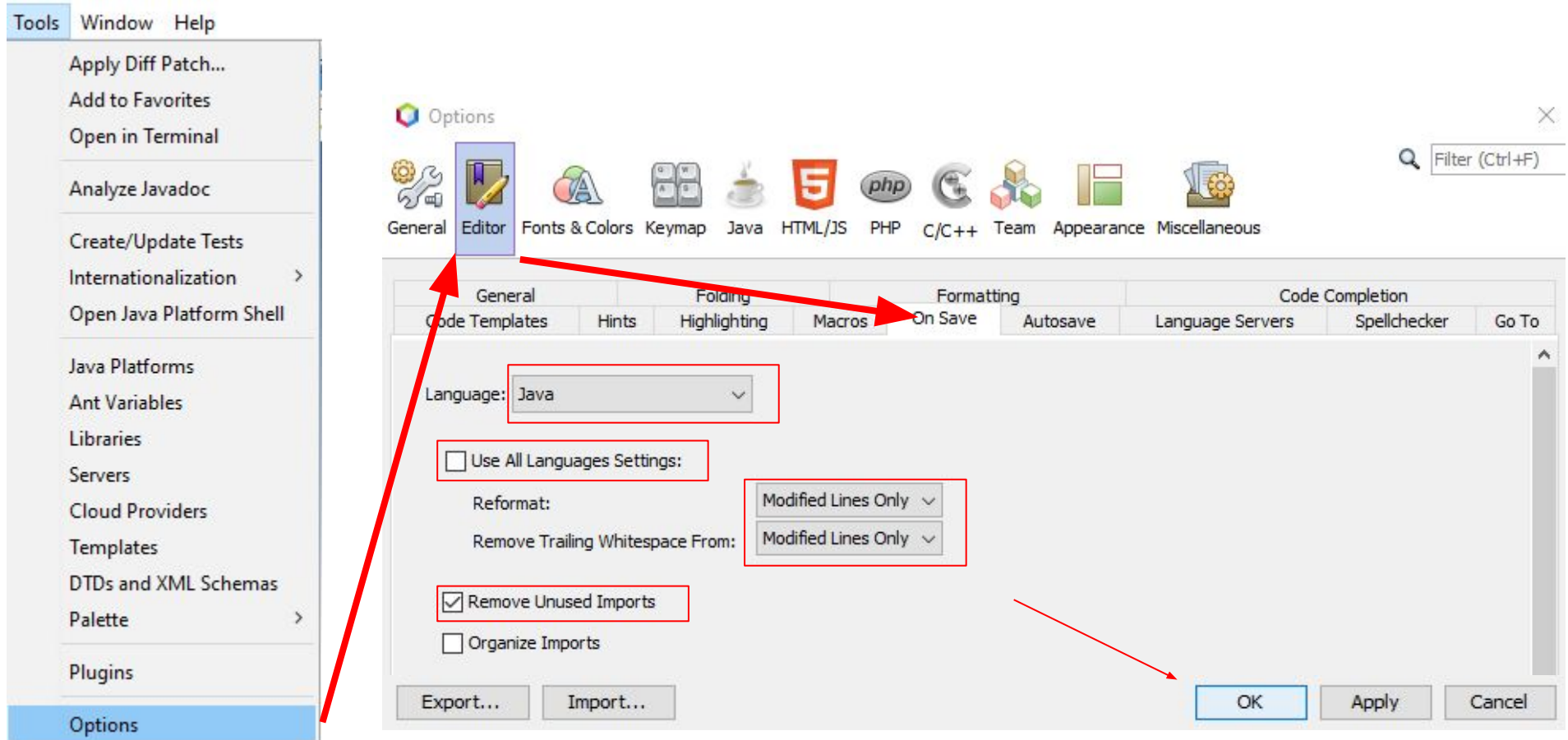
If selected, files are compiled and deployed when you save them.  
This option saves you time when you run or debug your application in the IDE.

☐ Always perform build before running application

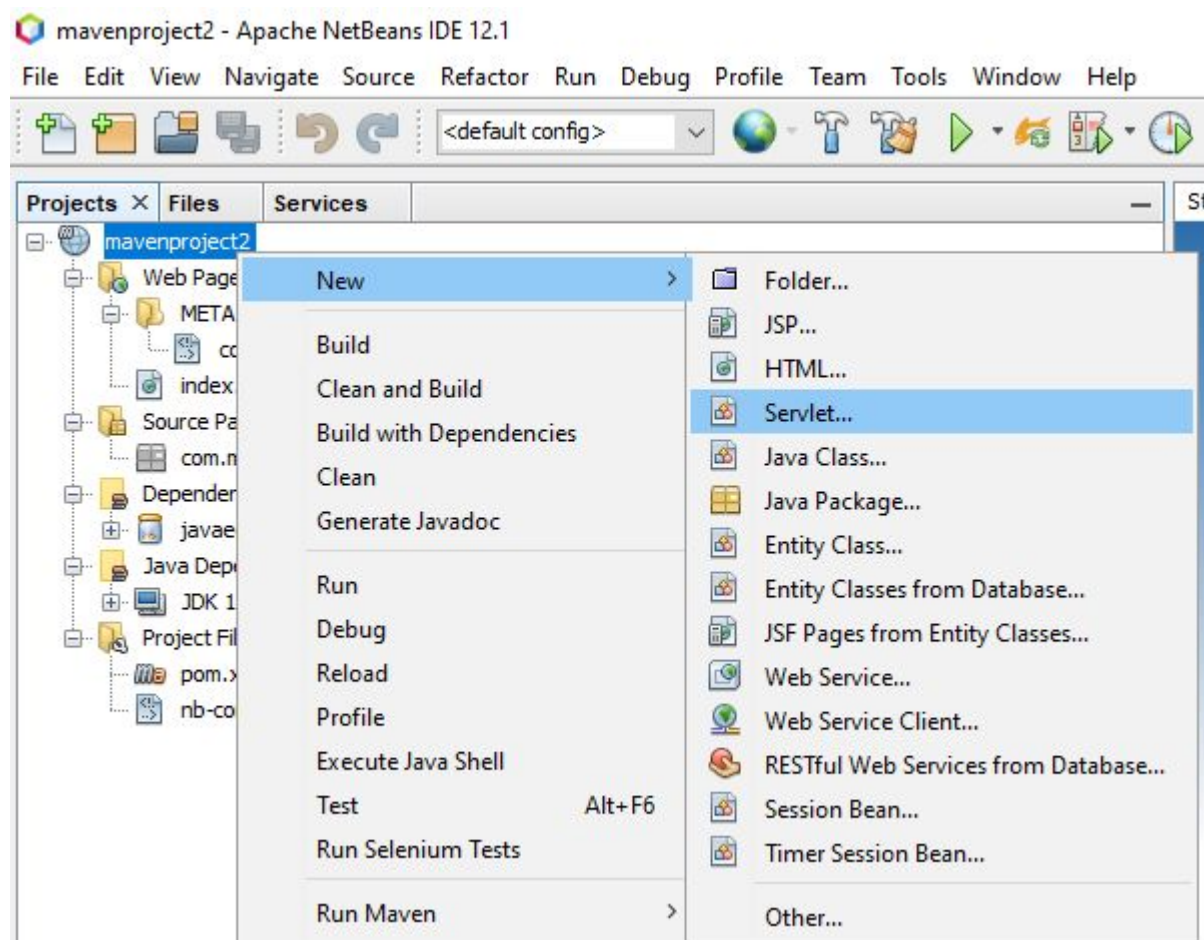
[Learn More about Deploy On Save feature in Maven projects](#)

**Buttons:** OK, Cancel, Help

# Modification of Project's On Save Changes (2/2)



# Create a Servlet (1/2)



# Create a Servlet (2/2)

## Steps

1. Choose File Type
2. **Name and Location**
3. Configure Servlet Deployment

**Name and Location**

Class Name:

Project:

Location:

Package:

Created File:

Register the Servlet with the application by giving the Servlet an internal name (Servlet Name). Then specify patterns that identify the URLs that invoke the Servlet. Separate multiple patterns with commas.

☒ Add information to deployment descriptor (web.xml)

Class Name:

Servlet Name:

URL Pattern(s):

Initialization Parameters:

Name	Value

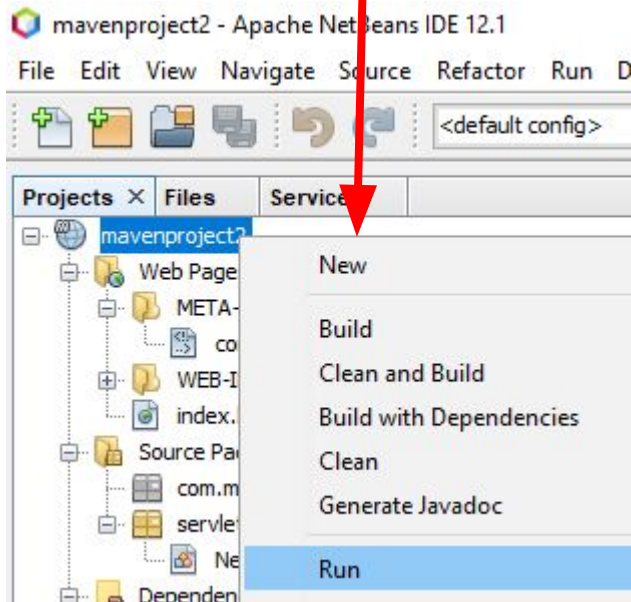
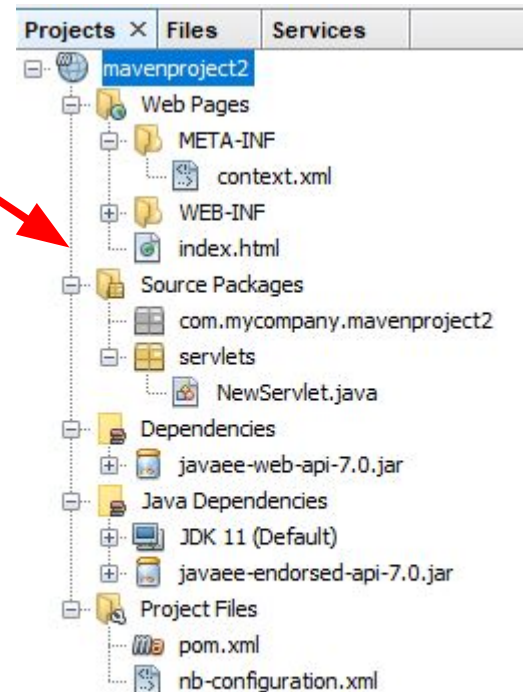
New

Edit...

Delete

# Run Servlet

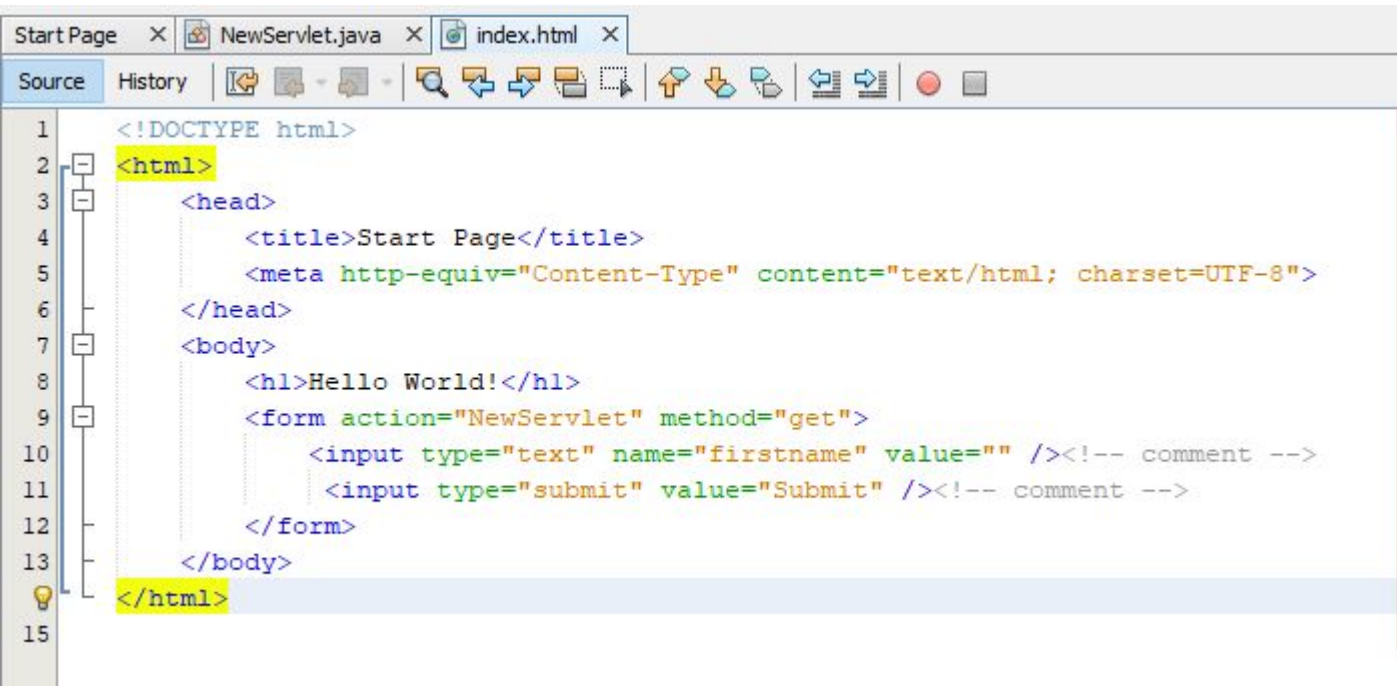
- index.html is the file that is loaded and runs on the server
- Right click on your project
  - Run





# Run Servlet - Example (1/2)

- index.html

A screenshot of an IDE window showing the source code of a file named index.html. The window has tabs for 'Start Page', 'NewServlet.java', and 'index.html'. The 'index.html' tab is active, and the 'Source' view is selected. The code is as follows:

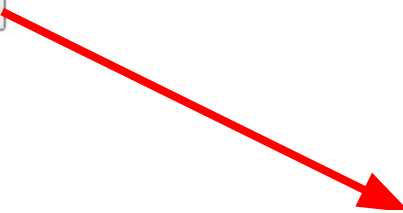
```
1 <!DOCTYPE html>
2 <html>
3   <head>
4     <title>Start Page</title>
5     <meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
6   </head>
7   <body>
8     <h1>Hello World!</h1>
9     <form action="NewServlet" method="get">
10       <input type="text" name="firstname" value="" /><!-- comment -->
11       <input type="submit" value="Submit" /><!-- comment -->
12     </form>
13   </body>
14 </html>
```

- run the project

# Run Servlet - Example (2/2)

---

**Hello World!**



**Servlet NewServlet at /mavenproject2**



# Add Dependencies to pom.xml

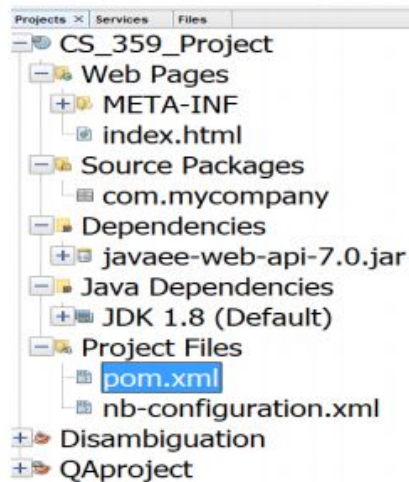
- Open pom.xml file and observe:
  - Within the dependencies tag, you can see the dependencies that are already added in your project (e.g. “.jar” files).
  - By adding a new dependency as a child element of dependencies you can let maven know which jars to automatically download after building the project.

```
<dependencies>
  <dependency>
    <groupId>javax</groupId>
    <artifactId>javaee-web-api</artifactId>
    <version>7.0</version>
    <scope>provided</scope>
  </dependency>
  <dependency>
    <groupId>junit</groupId>
    <artifactId>junit</artifactId>
    <version>4.12</version>
    <scope>test</scope>
  </dependency>
</dependencies>
```

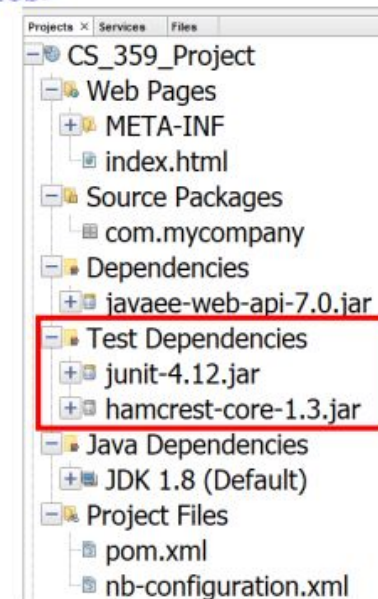
# Add Dependencies to pom.xml

Before and after adding the dependency in the pom.xml

```
<dependencies>
  <dependency>
    <groupId>javax</groupId>
    <artifactId>javaee-web-api</artifactId>
    <version>7.0</version>
    <scope>provided</scope>
  </dependency>
</dependencies>
```



```
<dependencies>
  <dependency>
    <groupId>javax</groupId>
    <artifactId>javaee-web-api</artifactId>
    <version>7.0</version>
    <scope>provided</scope>
  </dependency>
  <dependency>
    <groupId>junit</groupId>
    <artifactId>junit</artifactId>
    <version>4.12</version>
    <scope>test</scope>
  </dependency>
</dependencies>
```



# Add Dependencies to pom.xml

- You can search for dependencies at maven's online repository:

<https://mvnrepository.com/>

The screenshot shows the Maven Repository search results for 'junit'. The search bar at the top contains 'junit' (annotated with a red box and a green circle with '1'). Below the search bar, it says 'Found 1245 results'. The results are sorted by 'relevance'. The first result is '1. JUnit' (annotated with a red box and a green circle with '2'). It shows the JUnit logo, the text 'junit » junit', and a description: 'JUnit is a unit testing framework for Java, created by Erich Gamma and Kent Beck.' Below this, it says 'Last Release on Dec 4, 2014'. A table of versions is shown, with '4.12' highlighted (annotated with a red box and a green circle with '3'). To the right of the version table, there are tabs for different build systems: Maven, Gradle, SBT, Ivy, Grape, Leiningen, and Buildr. The 'Maven' tab is selected, showing the XML snippet for the dependency:

```
<!-- https://mvnrepository.com/artifact/junit/junit -->
<dependency>
  <groupId>junit</groupId>
  <artifactId>junit</artifactId>
  <version>4.12</version>
  <scope>test</scope>
</dependency>
```