

1 Introduction to Maven and Gradle: Overview of Build Automation Tools, Key Differences Between Maven and Gradle, Installation and Setup.

Maven

Maven is an open-source project management tool that helps us to create different software in the lifecycle used by this tool. This tool focuses on the standardization (i.e.) development of the software in a standard layout within a short duration of time. With this, we can create Java projects but is compatible to be used for other languages too. Maven uses Extensible Markup language(XML) for the structuring of the application.

Advantages of Maven:

- The process of project building is simplified and well organized.
- Maven automatically executes the task of downloading Jar files and the other dependencies.
- Maven can easily incorporate new dependencies by formulating the dependency code in the POM file.
- It facilitates easy access to all the essential information.
- It's extensible and plug-ins can be easily written using scripting languages or Java.

Disadvantages of Maven:

- Installation in the working system is needed.
- We cannot implement a dependency using Maven if the Maven code for existing dependency is not found.
- In terms of execution of project, Maven is quite slow.

Gradle

Gradle is an open-source tool that helps us to create software with mechanization. This tool is widely used for the creation of different kinds of software due to its high performance. It works on Java and a Groovy-based Domain-Specific Language (DSL) for developing the project structure. Gradle supports the creation of mobile and web applications with testing and deploying on various platforms. With its functionality, it is preferred as an official tool for developing Android applications.

Advantages of Gradle:

- Its highly customizable property. The tool can be modified under various technologies for diverse projects.
- The performance of Gradle is very fast and efficient. It is around 2x in speed to that of Maven.
- Gradle is a tool used for creating plug-ins and is a flexible instrument.
- It provides a wide variety of IDEs for an enhanced user experience.

Disadvantages of Gradle:

- Great technical expertise is required to build tasks with Gradle.
- It does not come with inbuilt ant project structure.
- Documentation of Gradle is somewhat extensive.
- Ant build scripts are to be drafted with the help of XML. Moreover , to automate a difficult project , a lot of logic need to be written in XML files.

Key Differences Between Maven and Gradle

| Basis | Gradle | Maven |
|-----------------------|---|---|
| Based on | Gradle is based on developing domain-specific language projects. | Maven is based on developing pure Java language-based software. |
| Configuration | It uses a Groovy-based Domain-specific language (DSL) for creating project structure. | It uses Extensible Markup Language (XML) for creating project structure. |
| Focuses on | Developing applications by adding new features to them. | Developing applications in a given time limit. |
| Performance | It performs better than maven as it optimized for tracking only current running task. | It does not create local temporary files during software creation and is hence – slower. |
| Java Compilation | It avoids compilation. | It is necessary to compile. |
| Usability | It is a new tool, which requires users to spend a lot of time to get used to it. | This tool is a known tool for many users and is easily available. |
| Customization | This tool is highly customizable as it supports a variety of IDE's. | This tool serves a limited number of developers and is not that customizable. |
| Languages supported | It supports software development in Java, C, C++, and Groovy. | It supports software development in Java, Scala, C#, and Ruby and it does not natively support C and C++ but can support through plugins like “maven-native-plugin” or we can integrate other build systems like CMake or Makefile. |
| Project Configuration | For declaring the project configuration, it does not use the XML files. | For declaring the project configuration, it uses the XML files. |
| Based on | Graph of task dependencies that do the work. | On the phases of the fixed and linear model. |
| Goal | To add functionality in the project is the main goal of the Gradle. | To finish the project in the given timeline is the main goal of the Maven. |

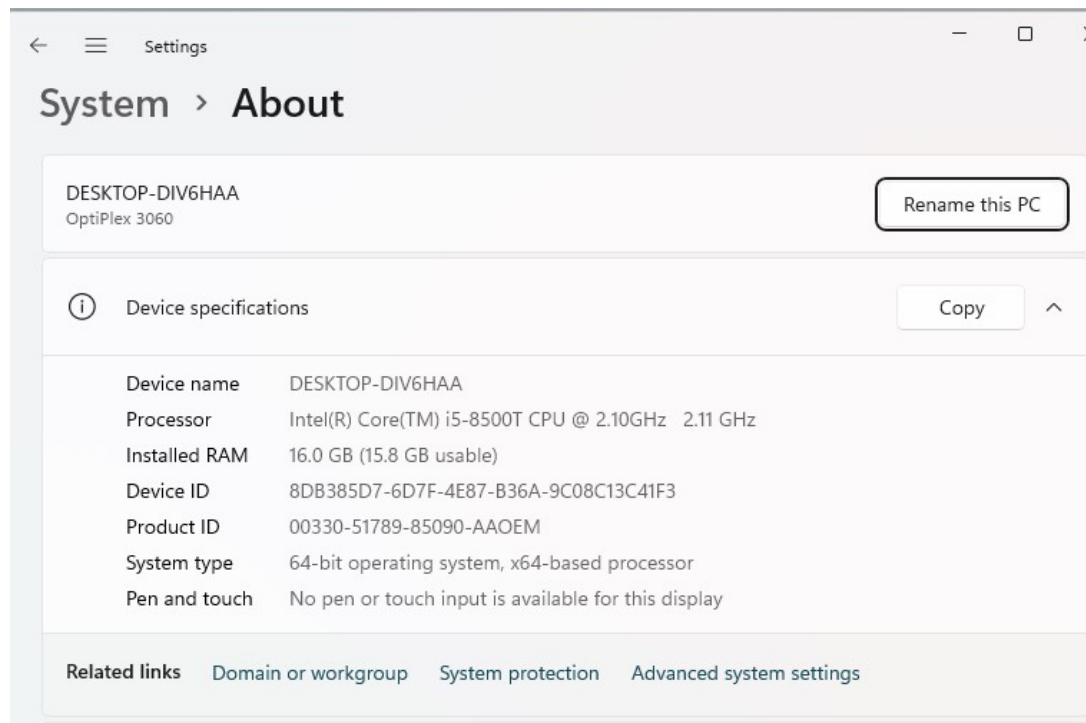
Installation of Maven:

Step1 : Download apache-maven-3.9.9.zip

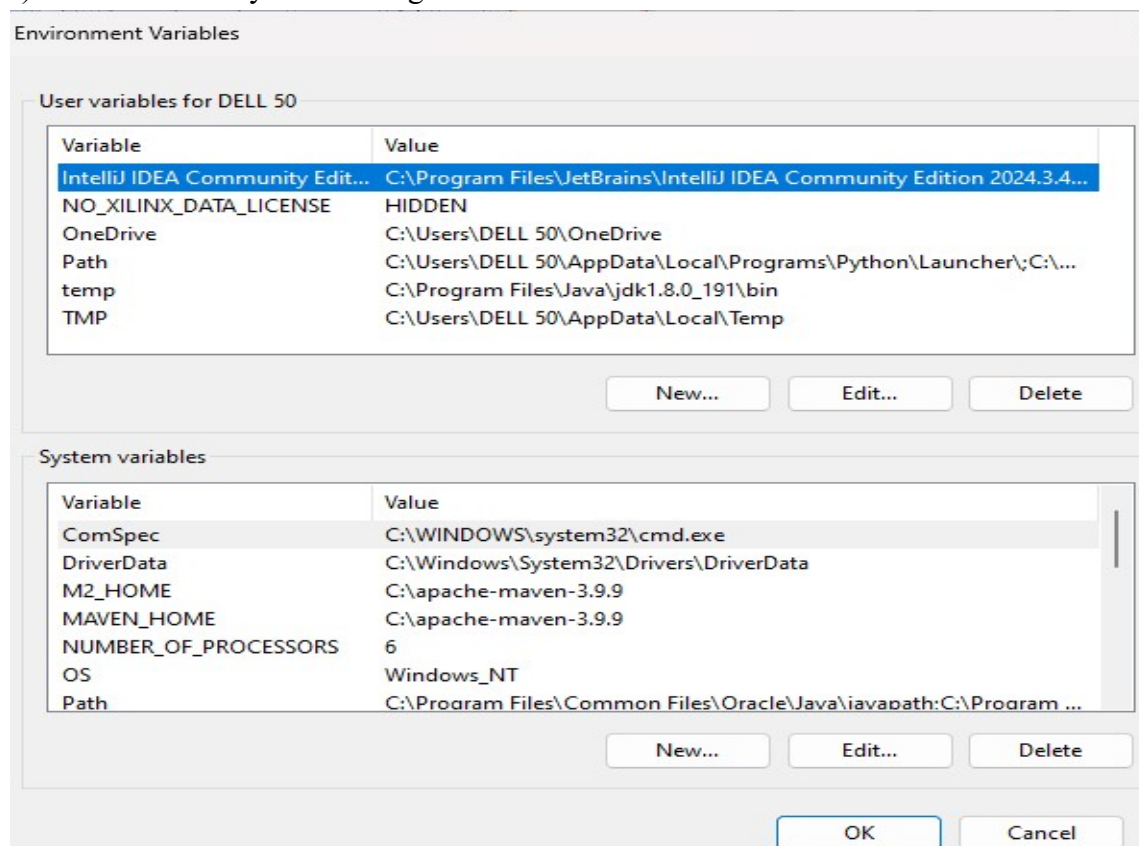
Step 2: Extract to a folder(C:\ apache-maven-3.9.9).

Step 3: Configure the environmental variables

A)Go to myPC-→ right click→Select properties

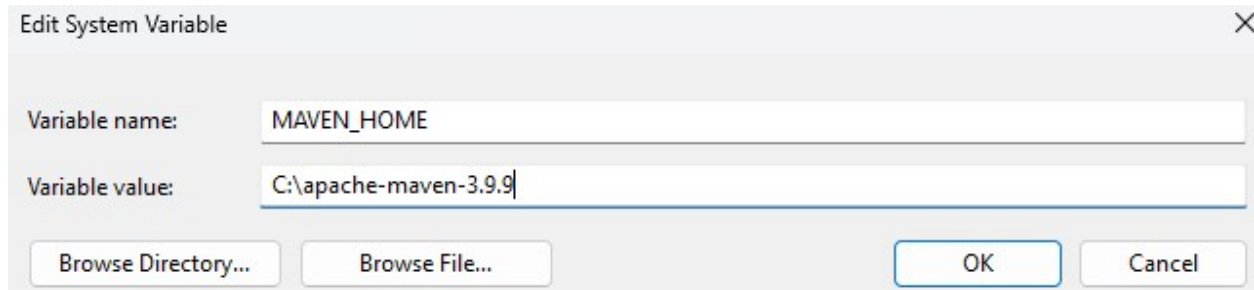


b)Click Advance System Settings -→select Environmental Variables



c) Click on new button and type

- Variable name : MAVEN_HOME
- Variable Value: C:\apache-maven-3.9.9



Edit System Variable

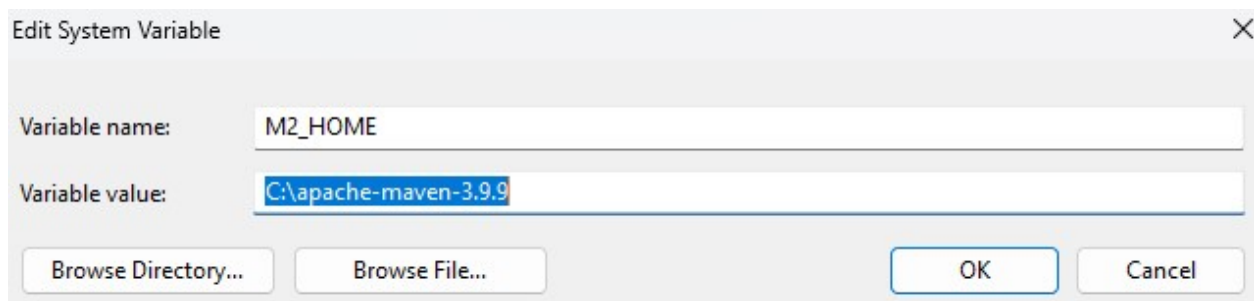
Variable name: MAVEN_HOME

Variable value: C:\apache-maven-3.9.9

Browse Directory... Browse File... OK Cancel

d) Click on new button and type

- Variable name : M2_HOME
- Variable Value: C:\apache-maven-3.9.9



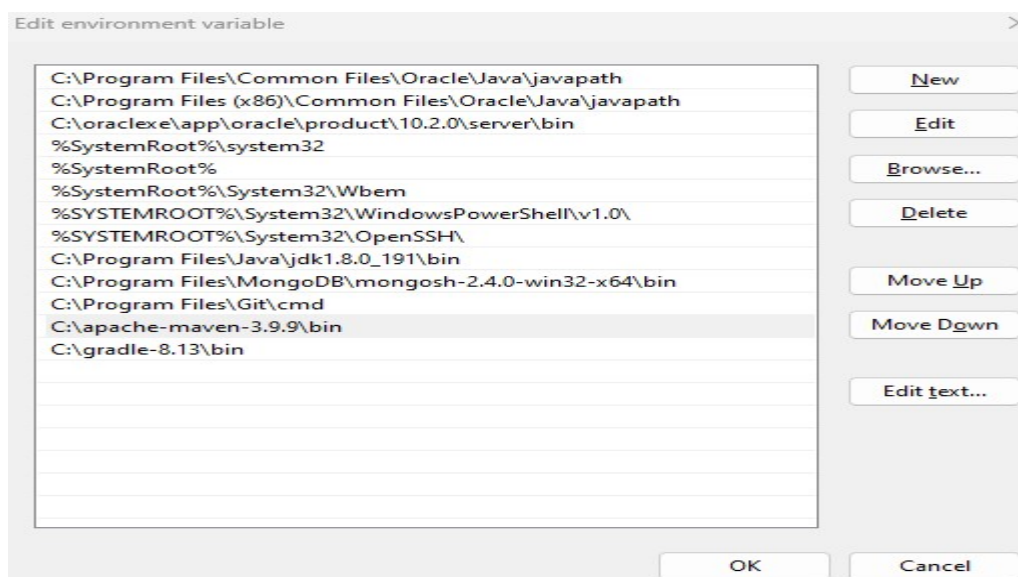
Edit System Variable

Variable name: M2_HOME

Variable value: C:\apache-maven-3.9.9

Browse Directory... Browse File... OK Cancel

e) Click on system variable → Path → new → C:\apache-maven-3.9.9\bin



Edit environment variable

C:\Program Files\Common Files\Oracle\Java\javapath
C:\Program Files (x86)\Common Files\Oracle\Java\javapath
C:\oracle\app\oracle\product\10.2.0\server\bin
%SystemRoot%\system32
%SystemRoot%
%SystemRoot%\System32\Wbem
%SYSTEMROOT%\System32\WindowsPowerShell\v1.0\
%SYSTEMROOT%\System32\OpenSSH\
C:\Program Files\Java\jdk1.8.0_191\bin
C:\Program Files\MongoDB\mongosh-2.4.0-win32-x64\bin
C:\Program Files\Git\cmd
C:\apache-maven-3.9.9\bin
C:\gradle-8.13\bin

New
Edit
Browse...
Delete
Move Up
Move Down
Edit text...

OK Cancel

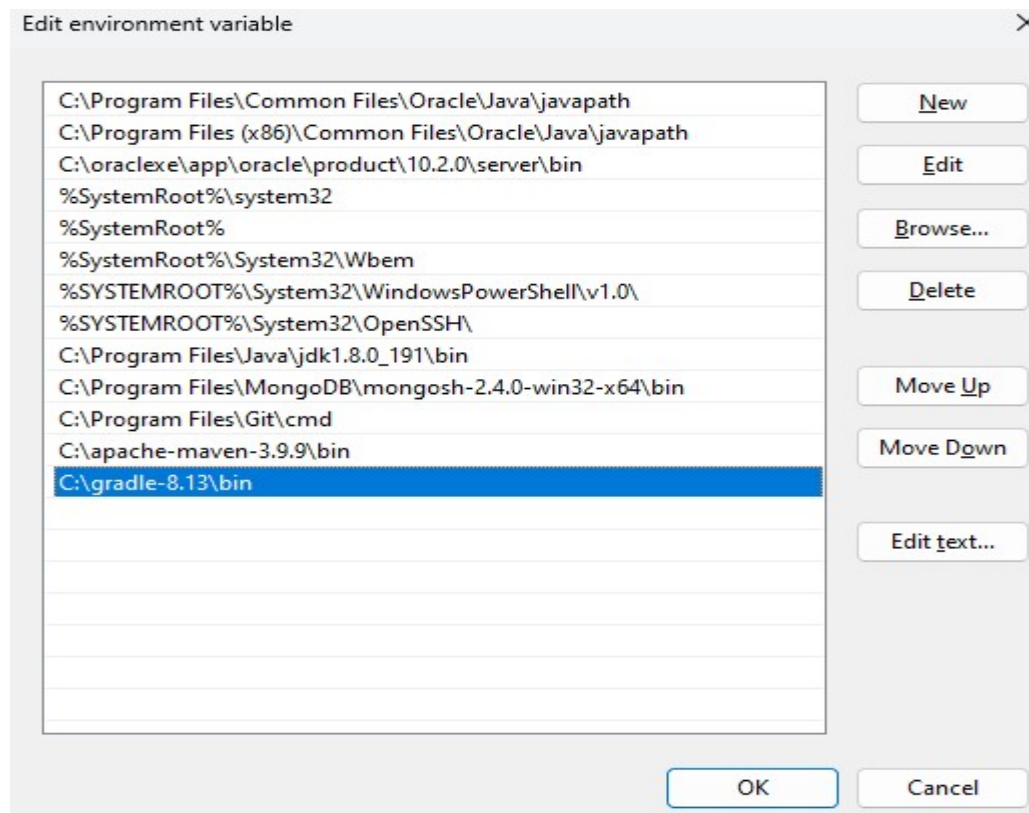
Installation of Gradle:

Step1 : Download gradle-8.13-bin

Step 2: Extract to a folder(C:\ gradle-8.13).

Step 3: Configure the environmental variables

A)Go to myPC-→ right click→Select properties→ system variable → Path →new → C:\ gradle-8.13\bin



To Verify : Goto Command prompt and type

A)mvn --version

```
C:\Users\DELL 50>mvn --version
Apache Maven 3.9.9 (8e8579a9e76f7d015ee5ec7bfcdc97d260186937)
Maven home: C:\apache-maven-3.9.9
Java version: 17.0.12, vendor: Oracle Corporation, runtime: C:\Program Files\Java\jdk-17
Default locale: en_IN, platform encoding: Cp1252
OS name: "windows 11", version: "10.0", arch: "amd64", family: "windows"
```

B)gradle -v

```
C:\Users\DELL 50>mvn --version
Apache Maven 3.9.9 (8e8579a9e76f7d015ee5ec7bfc97d260186937)
Maven home: C:\apache-maven-3.9.9
Java version: 17.0.12, vendor: Oracle Corporation, runtime: C:\Program Files\Java\jdk-17
Default locale: en_IN, platform encoding: Cp1252
OS name: "windows 11", version: "10.0", arch: "amd64", family: "windows"

C:\Users\DELL 50>gradle -v

-----
Gradle 8.13
-----

Build time:    2025-02-25 09:22:14 UTC
Revision:     073314332697ba45c16c0a0ce1891fa6794179ff

Kotlin:       2.0.21
Groovy:       3.0.22
Ant:          Apache Ant(TM) version 1.10.15 compiled on August 25 2024
Launcher JVM: 17.0.12 (Oracle Corporation 17.0.12+8-LTS-286)
Daemon JVM:   C:\Program Files\Java\jdk-17 (no JDK specified, using current Java home)
OS:           Windows 11 10.0 amd64
```

2 Working with Maven: Creating a Maven Project, Understanding the POM File, Dependency Management and Plugins

A)Select Project→new Project→Java

Name: MyApp

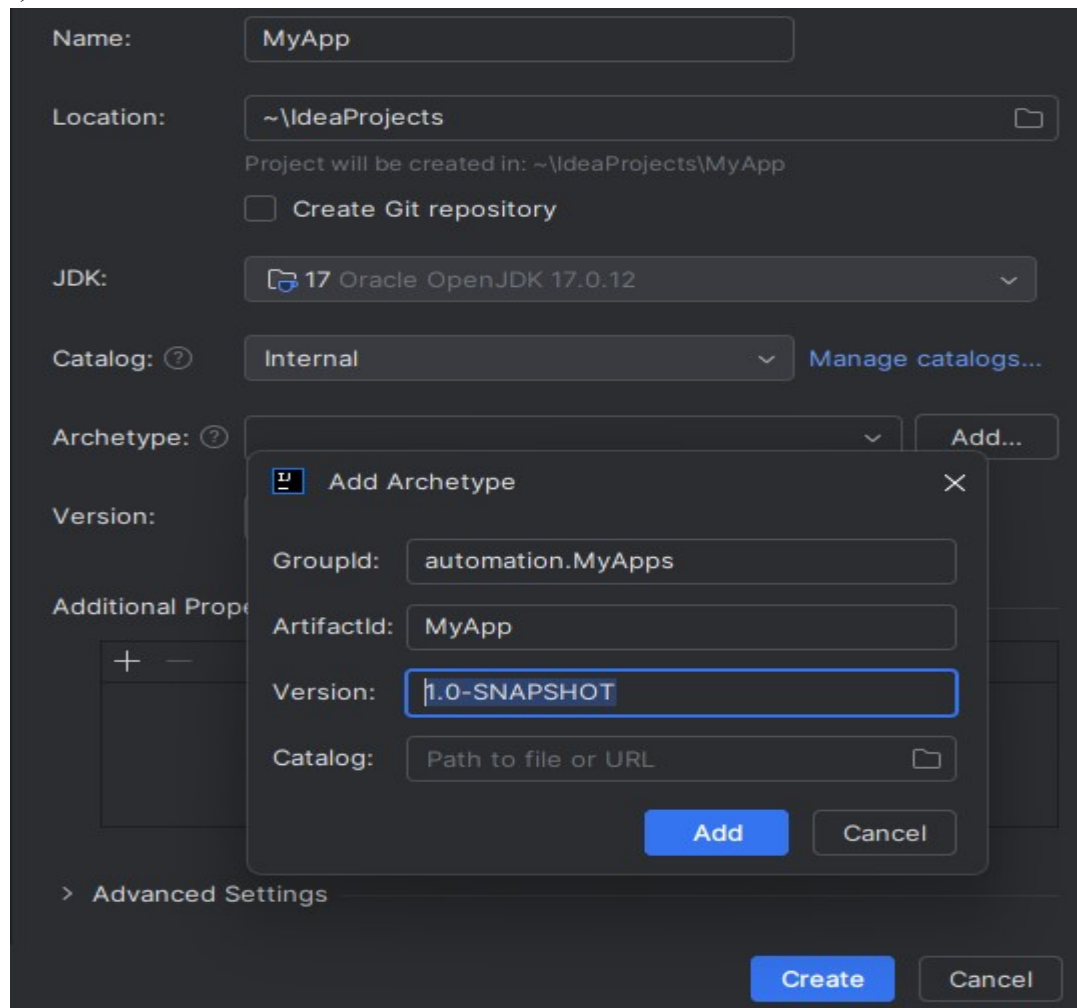
B)Select Archtype → Add

Groupid: automation.MyApps

Artifactid: MyApp

Version: 1.0-SNAPSHOT

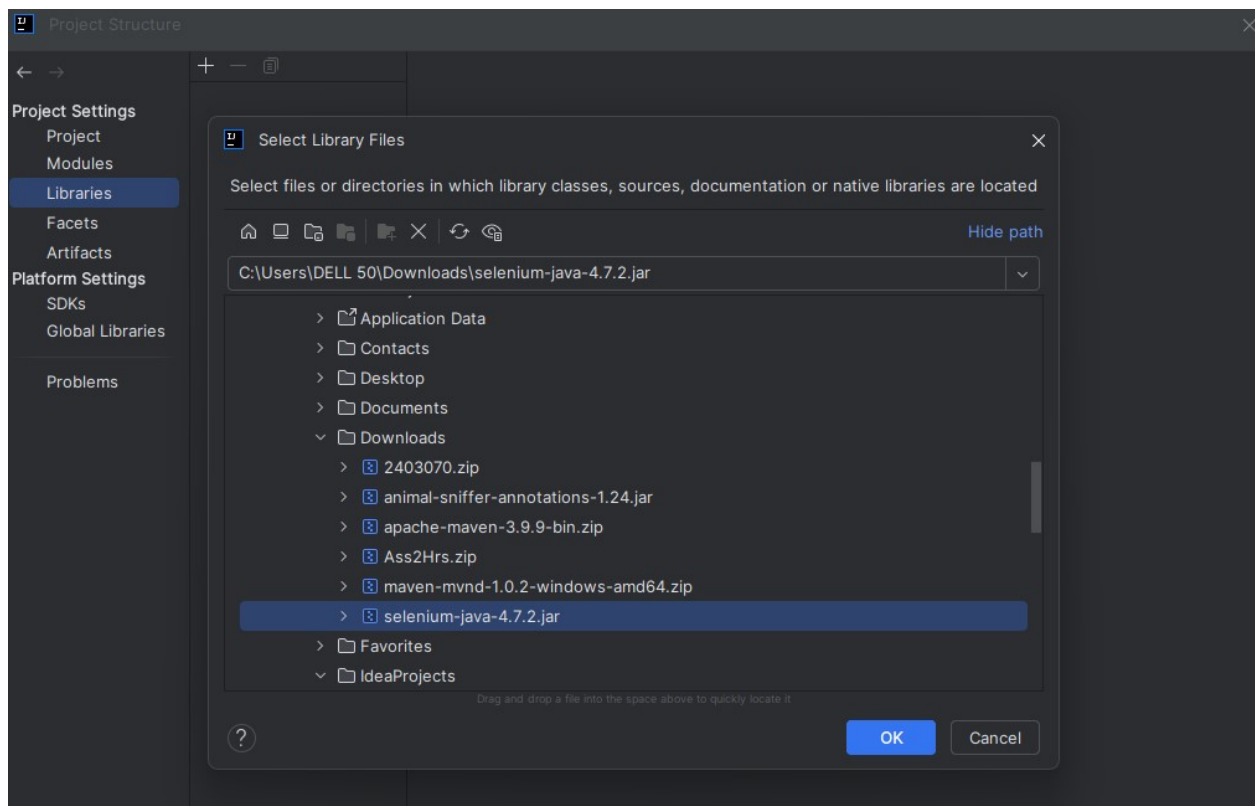
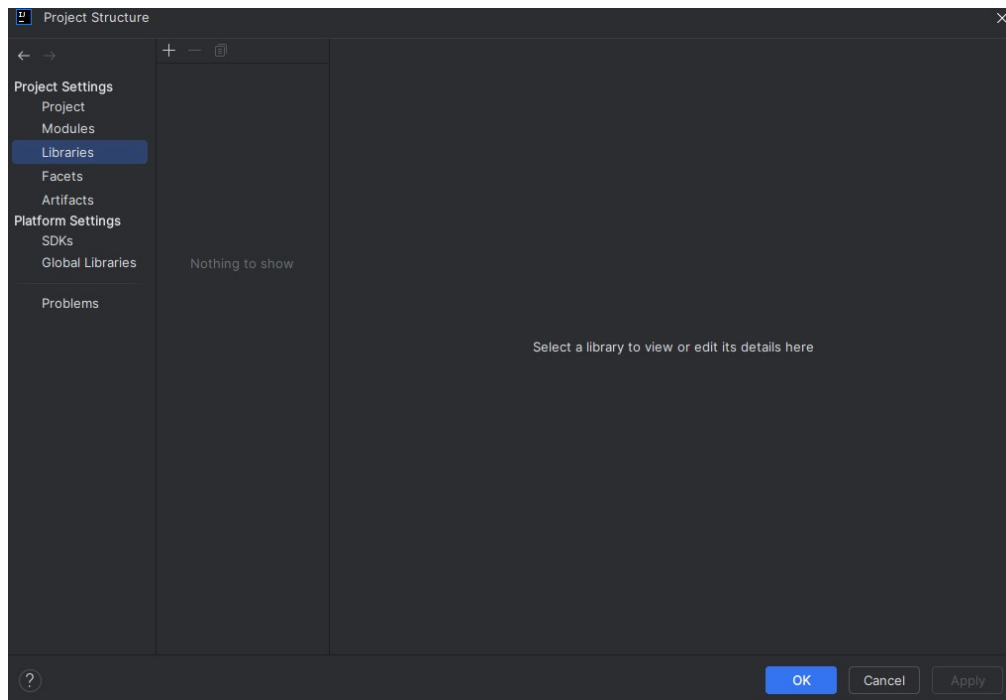
c) **Click Create** button



Pom.xml file will open.

Case1: Add Jar.files

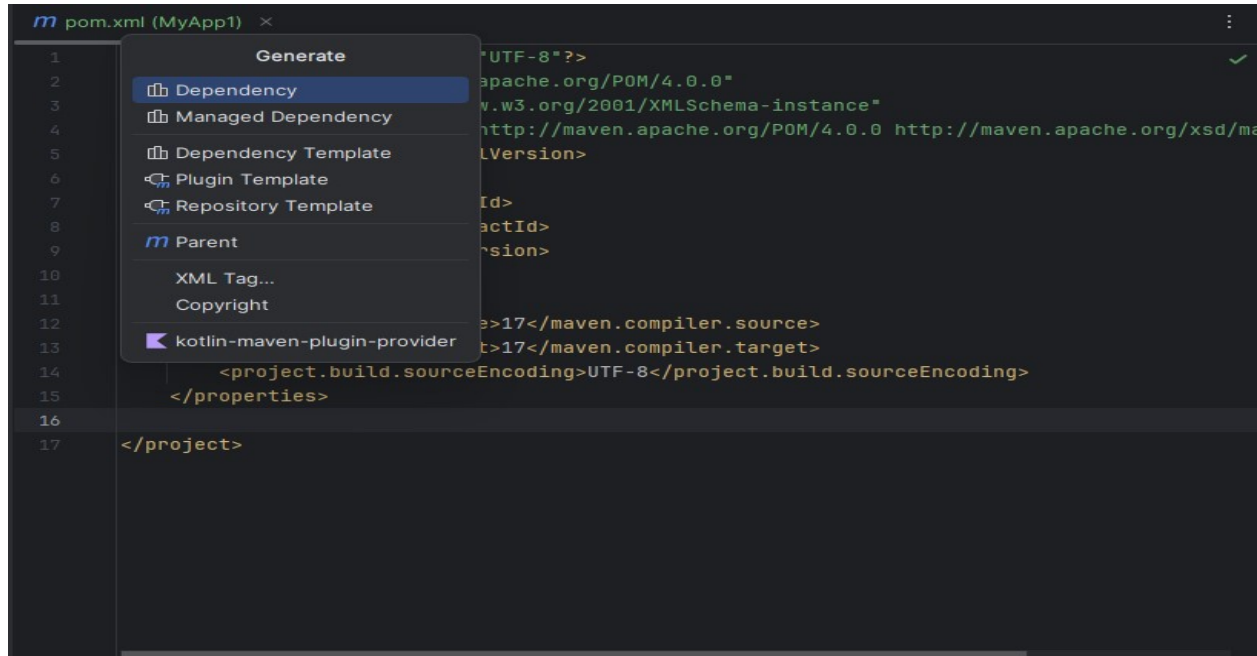
File→ProjectStructure→Libraries→ +→java→Selenium-java-4.72.jar



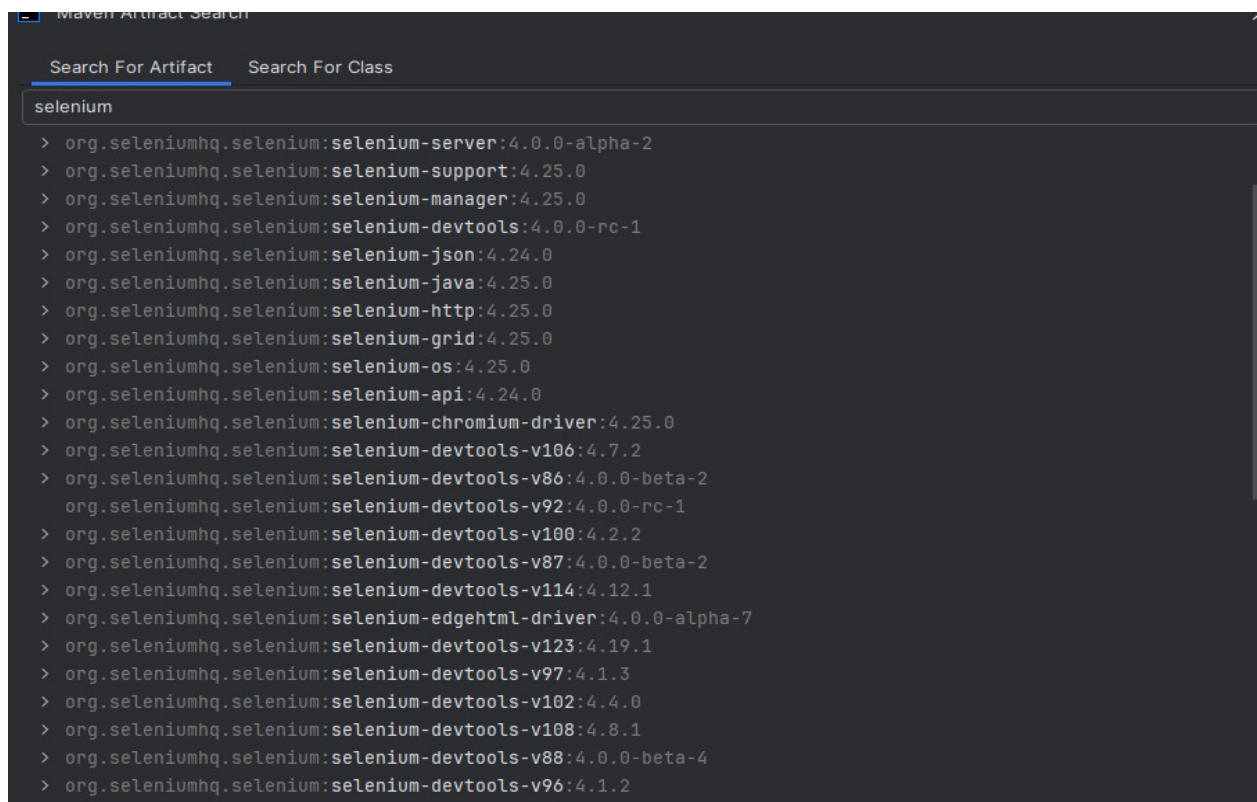
Ok→apply→ok

Case 2: Add Dependency Files

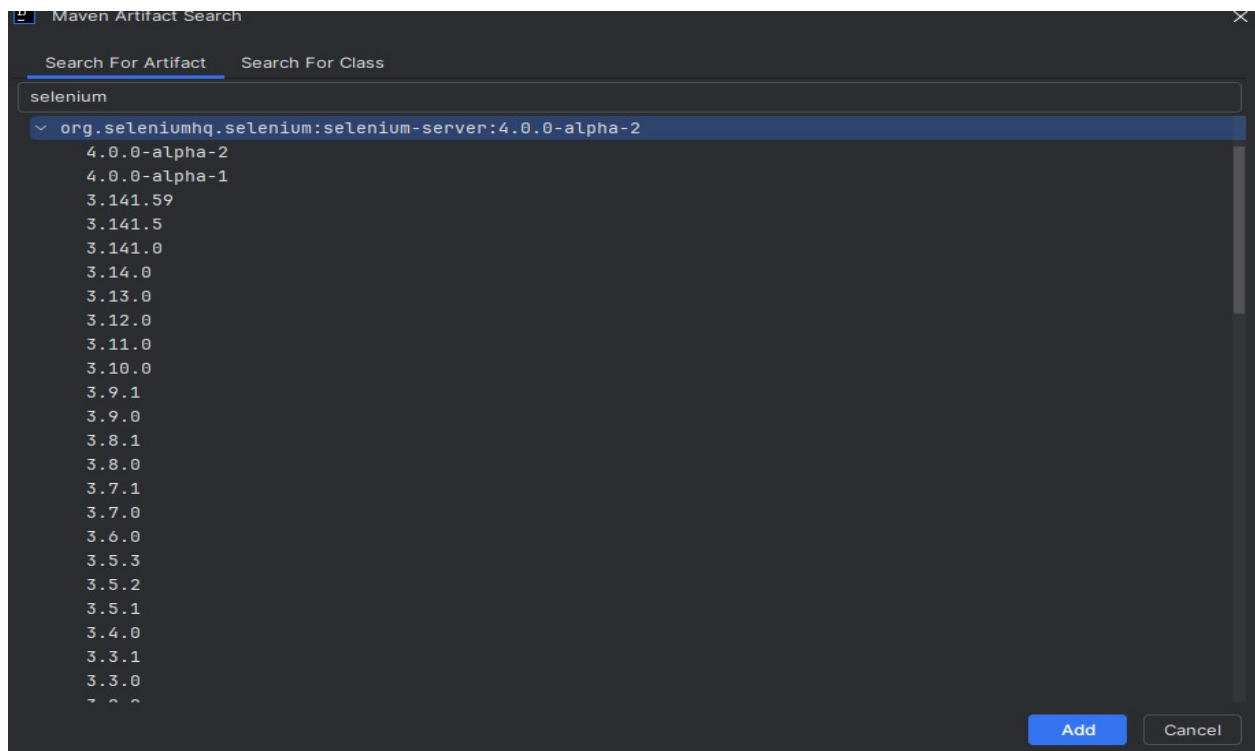
Pom.xml → right click → select → Generate → dependency



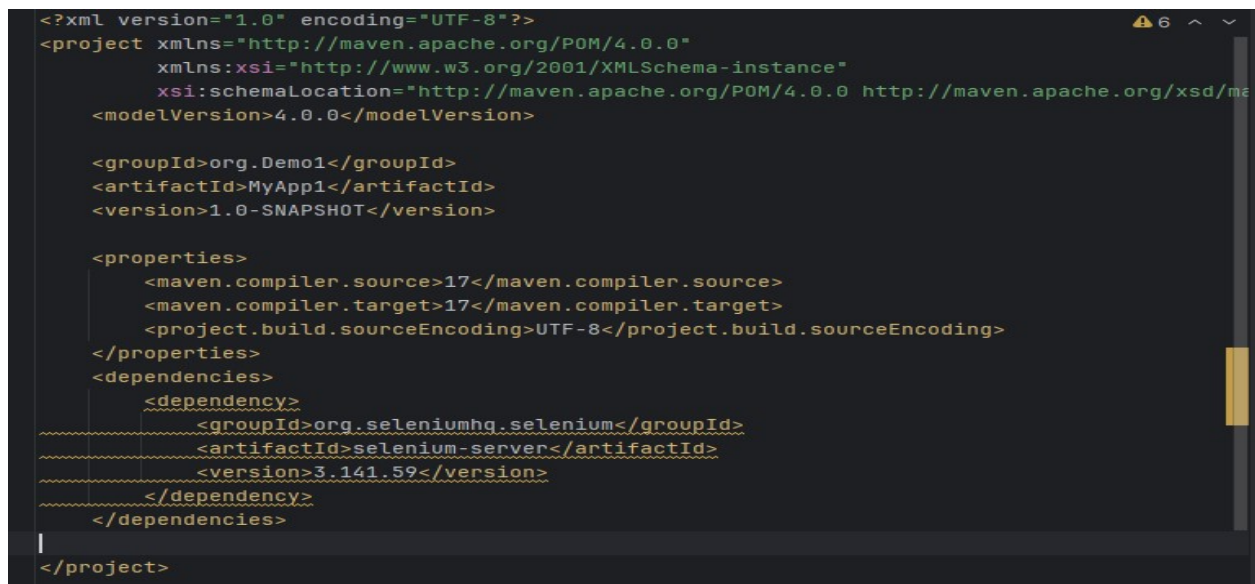
→ Search → selenium



→selenium→org.seleniumhq.selenium-server:4.0.0-alpha-2→3.141.59

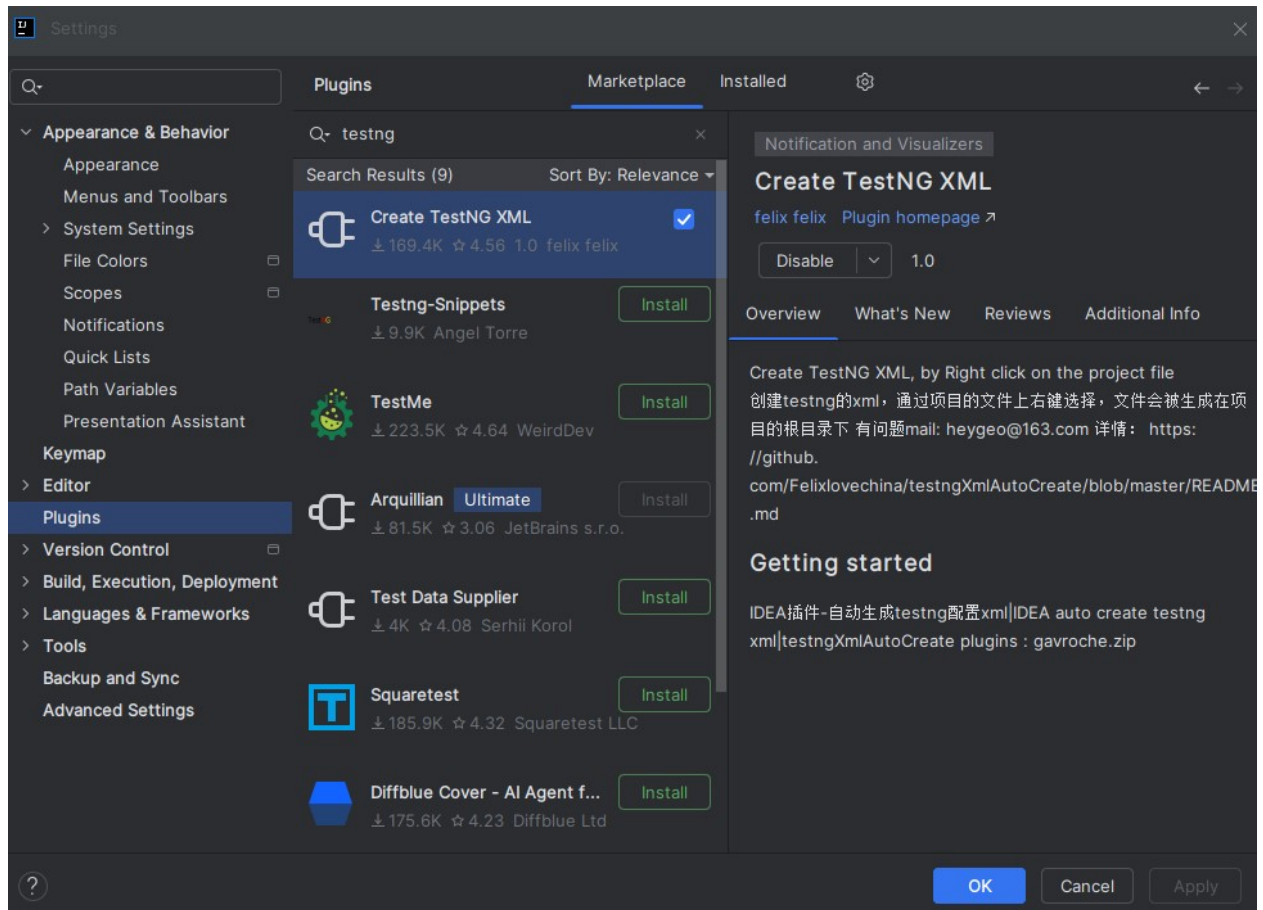


Auto-generated dependency code

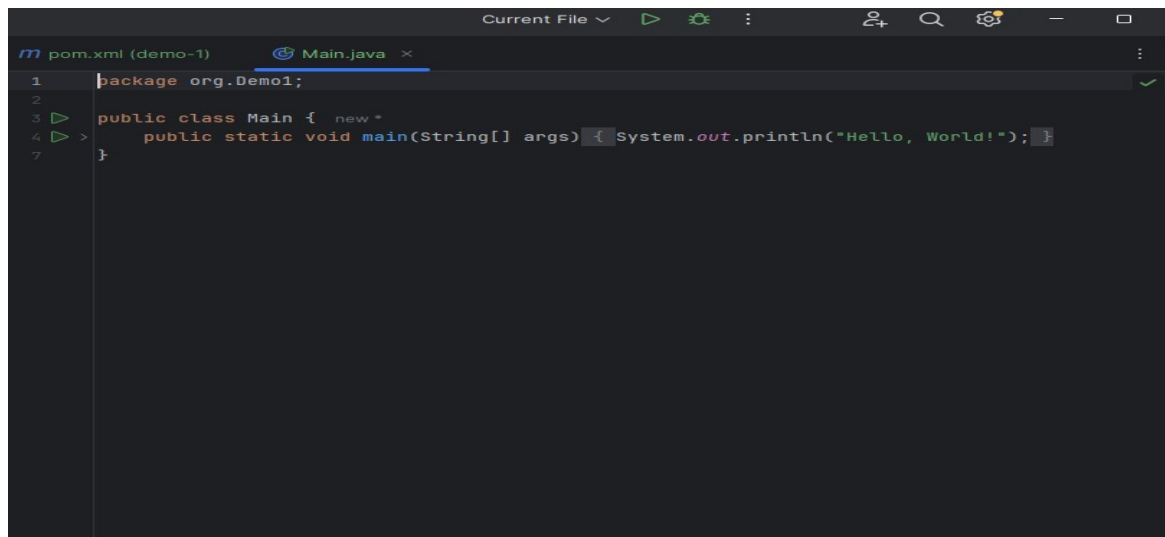


Case 3: Plugins

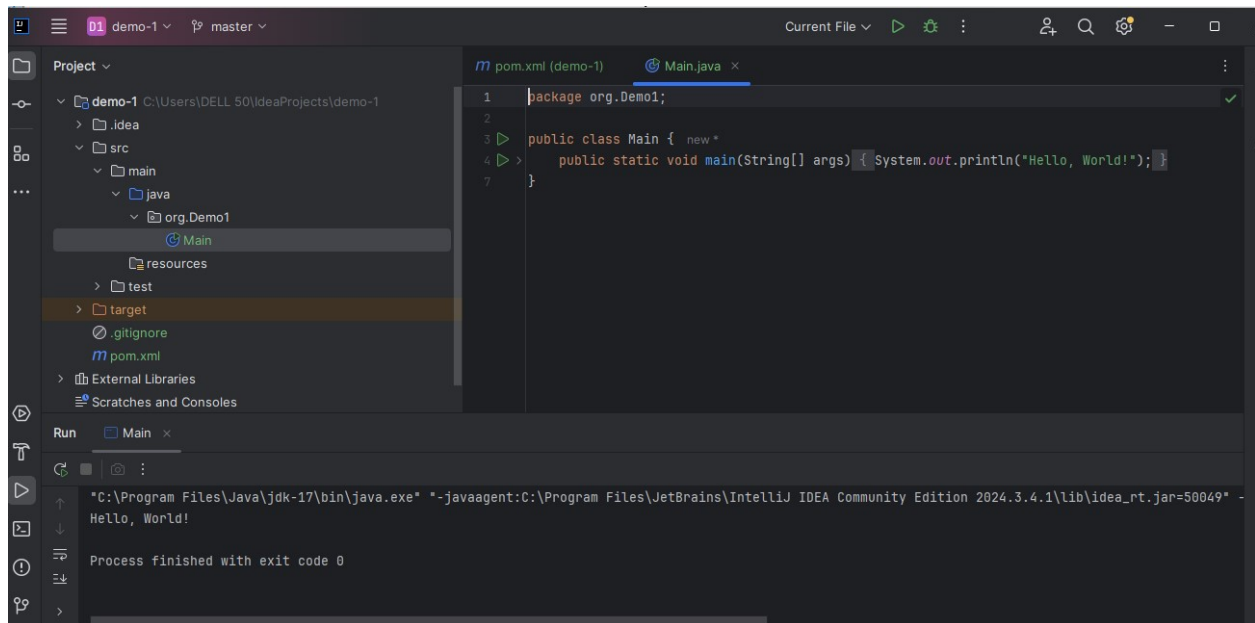
File→settings→plugins→testng→ok



Case4: Run java program in main method.(hello world)

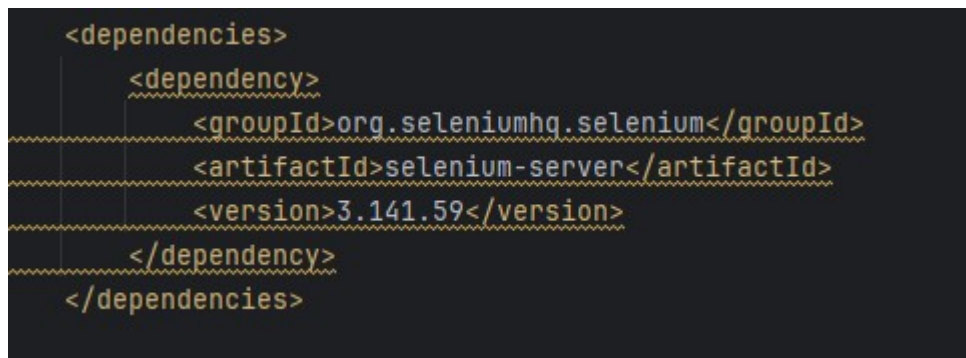


a)Run java file



b)Go to pom.xml

a) add dependency files



b)add plugins

```

<build>
  <plugins>
    <plugin><groupId>org.apache.maven.plugins</groupId>
      <artifactId>maven-site-plugin</artifactId>
      <version>3.12.1</version>
    </plugin>
    <plugin>
      <groupId>org.codehaus.mojo</groupId>
      <artifactId>exec-maven-plugin</artifactId>
      <version>3.4.1</version>
      <executions>
        <execution>
          <goals>
            <goal>java</goal>
          </goals>
        </execution>
      </executions>
      <configuration>
        <mainClass>org.Demo1.Main</mainClass>
      </configuration>
    </plugin>
  </plugins>
</build>

```

c) go to terminal → mvn test

```

Copyright (C) Microsoft Corporation. All Rights Reserved.

Install the latest PowerShell for new features and improvements! https://aka.ms/PSWindows

PS C:\Users\DELL 50\IdeaProjects\test1> mvn test
[INFO] Scanning for projects...
[INFO]

```

d) go to right side → Maven → test → plugins → exec → exec:java

The screenshot shows the IntelliJ IDEA IDE with the Maven tool window on the right. The 'test1' project is selected, and the 'exec' plugin is expanded, showing the 'exec:java' goal. The 'Run' button is clicked, and the output console at the bottom shows the execution of the 'java' goal, which prints 'Hello, World!' and reports 'BUILD SUCCESS'.

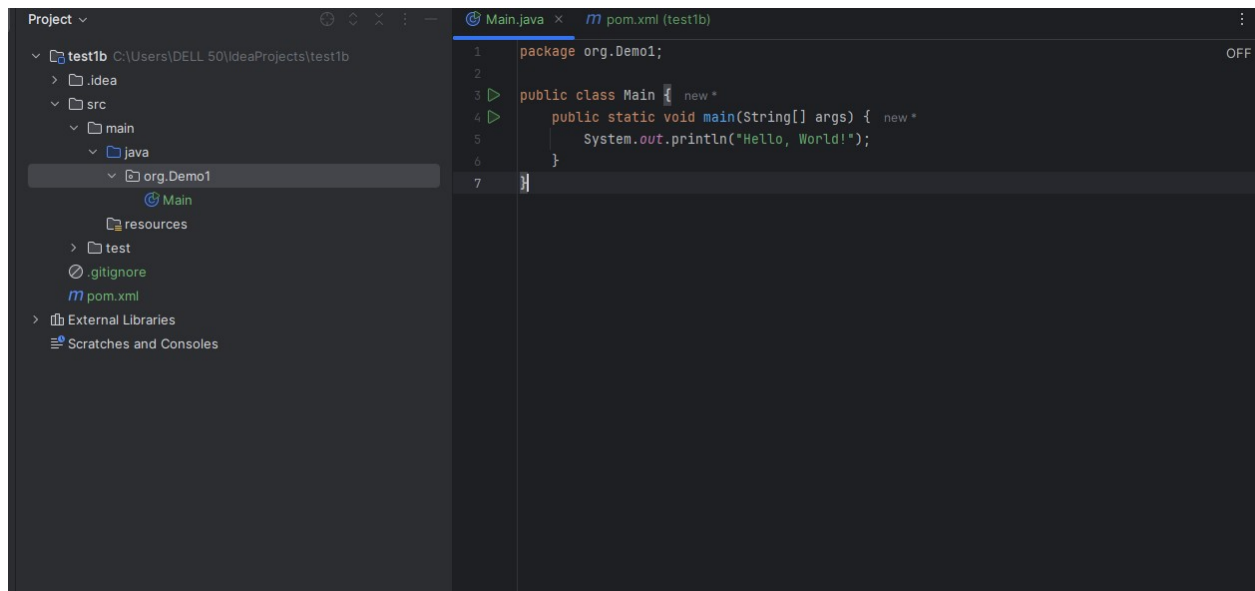
```

[INFO] -----[ jar
[INFO]
[INFO] --- exec:3.4.1:java (default-cli) @ t
Hello, World!
[INFO]
[INFO] BUILD SUCCESS

```


Case 5: write java class to testing the title of website using selenium, java and testng

File→new→project→test1b



Pom.xml→right click→select→Generate→dependency→ org.seleniumhq.selenium-server:4.0.0-alpha-2→3.141.59

Pom.xml→right click→select→Generate→dependency→ org.seleniumhq:selenium-java:4.25.0

Pom.xml→right click→select→Generate→dependency→org.testng:testng:7.10.2

```
<dependencies>
  <dependency>
    <groupId>org.seleniumhq.selenium</groupId>
    <artifactId>selenium-server</artifactId>
    <version>4.0.0-alpha-2</version>
  </dependency>
  <dependency>
    <groupId>org.seleniumhq.selenium</groupId>
    <artifactId>selenium-java</artifactId>
    <version>4.25.0</version>
  </dependency>
  <dependency>
    <groupId>org.testng</groupId>
    <artifactId>testng</artifactId>
    <version>7.10.2</version>
  </dependency>
</dependencies>
```

b) go to Main. java file and type the code

```
package org.Demo1;
import org.openqa.selenium.By;
import org.openqa.selenium.WebDriverException;
import org.openqa.selenium.WebDriver;
import org.openqa.selenium.WebElement;
import org.openqa.selenium.chrome.ChromeDriver;

public class Main {
    public static void main(String[] args) {
        // System.setProperty("webdriver.chrome.driver","C:\\Users\\DELL
50\\Documents\\chromedriver-win64\\chromedriver.exe");
        WebDriver driver=new ChromeDriver();
        driver.get("https://www.facebook.com");

        driver.findElement(By.id("email")).sendKeys("lingaraj.superstar@gmail.com");
        driver.findElement(By.id("pass")).sendKeys("point");
        driver.findElement(By.name("login")).click();
    }
}
```

run

