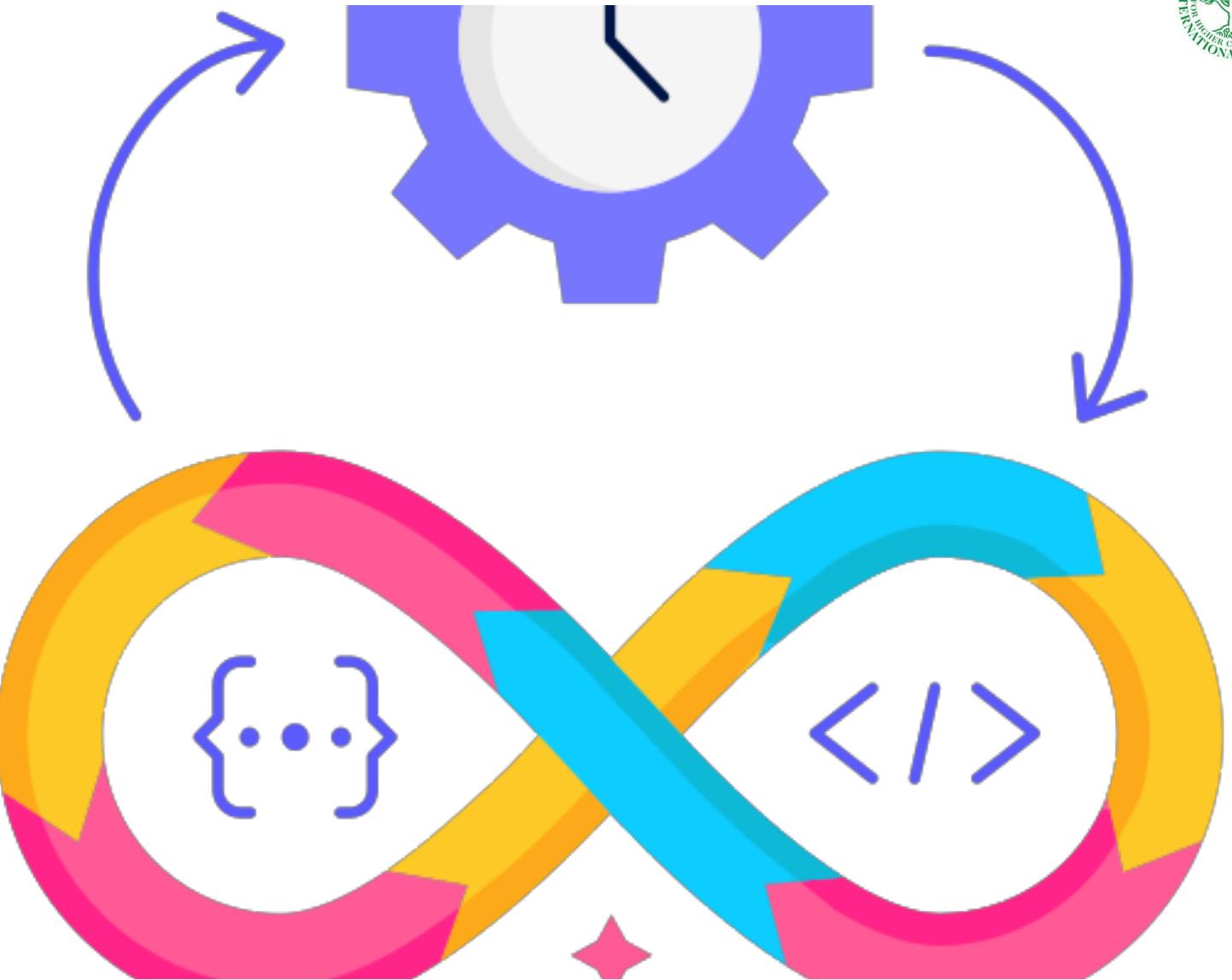


## Day #2

Software Build Automation



# Wholeness

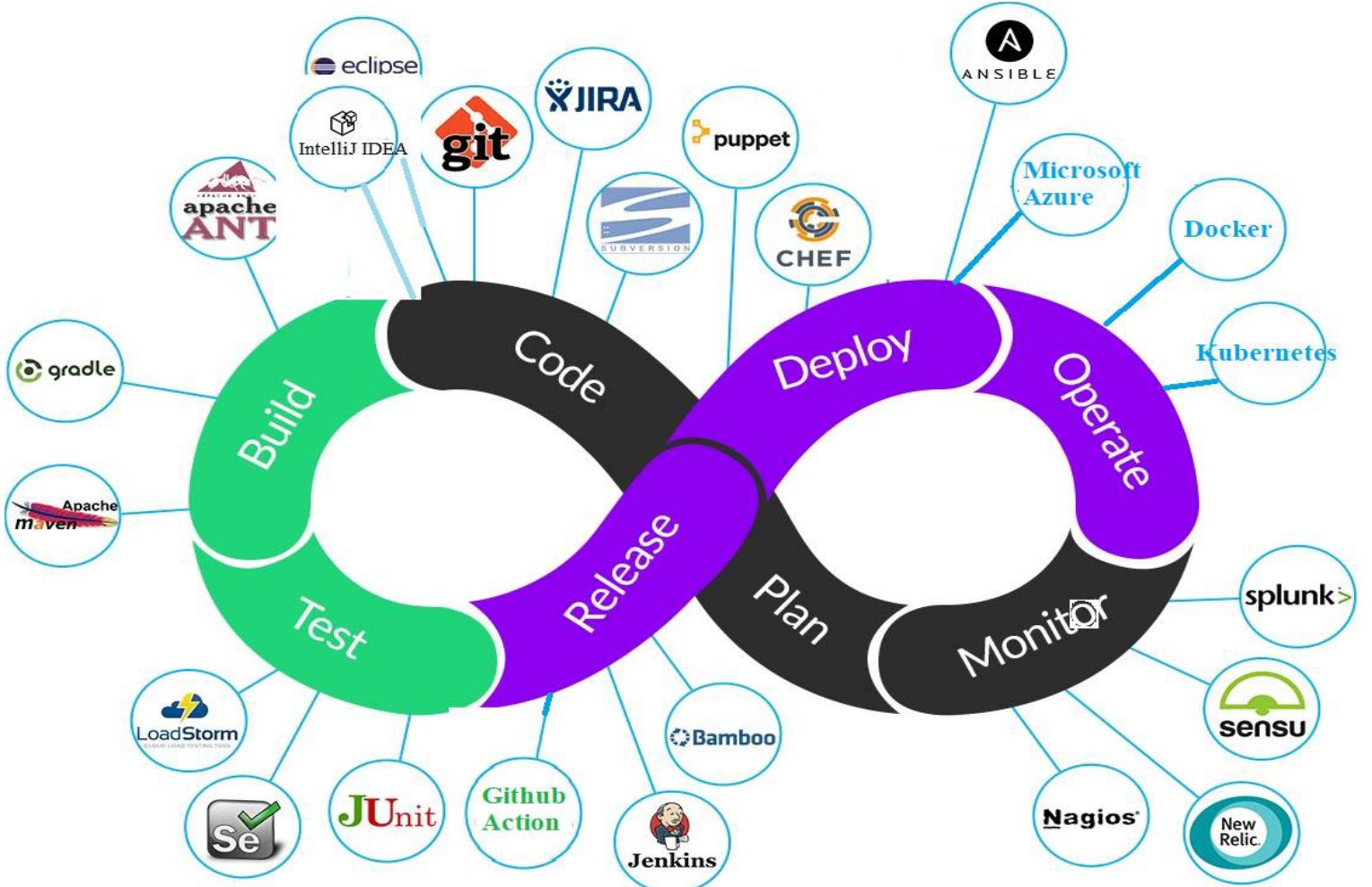
- Automation is the use of technology to perform tasks with reduced or no human assistance/intervention.
- Software Build constitutes the process of compiling source code, executing tests and creating a deployable unit or package/artifact.
- Science of Consciousness: Thought leads to Action. Action leads to Achievement. Achievement leads to Fulfillment.

# Software Build Automation

- Automation of the Software build process is an essential requirement for a quick and efficient release of software product and subsequent updates.
- For modern business software products the Build -> Test -> Release cycle needs to be fast, frequent and error-free.

# DevOps Life-cycle:

## Plan -> Code -> Build -> Test -> Release



# Build Automation Tools

Commonly used Software build automation tools:

- Apache Maven
- Gradle
- Github Action
- Jenkins

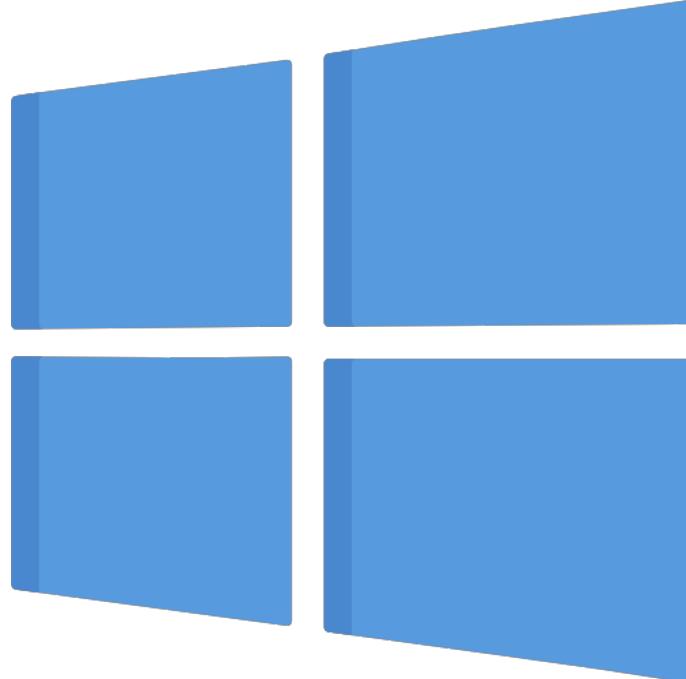
# Apache Maven

- Apache Maven is an open-source software build automation and dependency management tool, preeminent within the Java application development space.
- Maven uses the concept of a Project Object Model (POM) to manage the software build operation
- Every Maven project uses a pom.xml file which contains build configuration data



# Download and Install Maven (Windows)

<https://maven.apache.org/download.cgi>



Apache / Maven / Download Apache Maven

Welcome  
License  
  
ABOUT MAVEN  
What is Maven?  
Features  
**Download**  
Use  
Release Notes  
  
DOCUMENTATION  
Maven Plugins  
Maven Extensions  
Index (category)  
User Centre  
Plugin Developer Centre  
Maven Repository Centre  
Maven Developer Centre  
Books and Resources  
Security  
  
COMMUNITY  
Community Overview  
Project Roles  
How to Contribute  
Getting Help  
Issue Management

## Downloading Apache Maven 3.9.9

Apache Maven 3.9.9 is the latest release: it is the recommended version for all users.

## System Requirements

<b>Java Development Kit (JDK)</b>	Maven 3.9+ requires JDK 8 or above to execute. It still allows you to use Java 7 but it is not recommended.
<b>Memory</b>	No minimum requirement
<b>Disk</b>	Approximately 10MB is required for the Maven installation itself. In addition to this there will be temporary files created depending on usage but expect at least 500MB.
<b>Operating System</b>	No minimum requirement. Start up scripts are included as shell scripts.

## Files

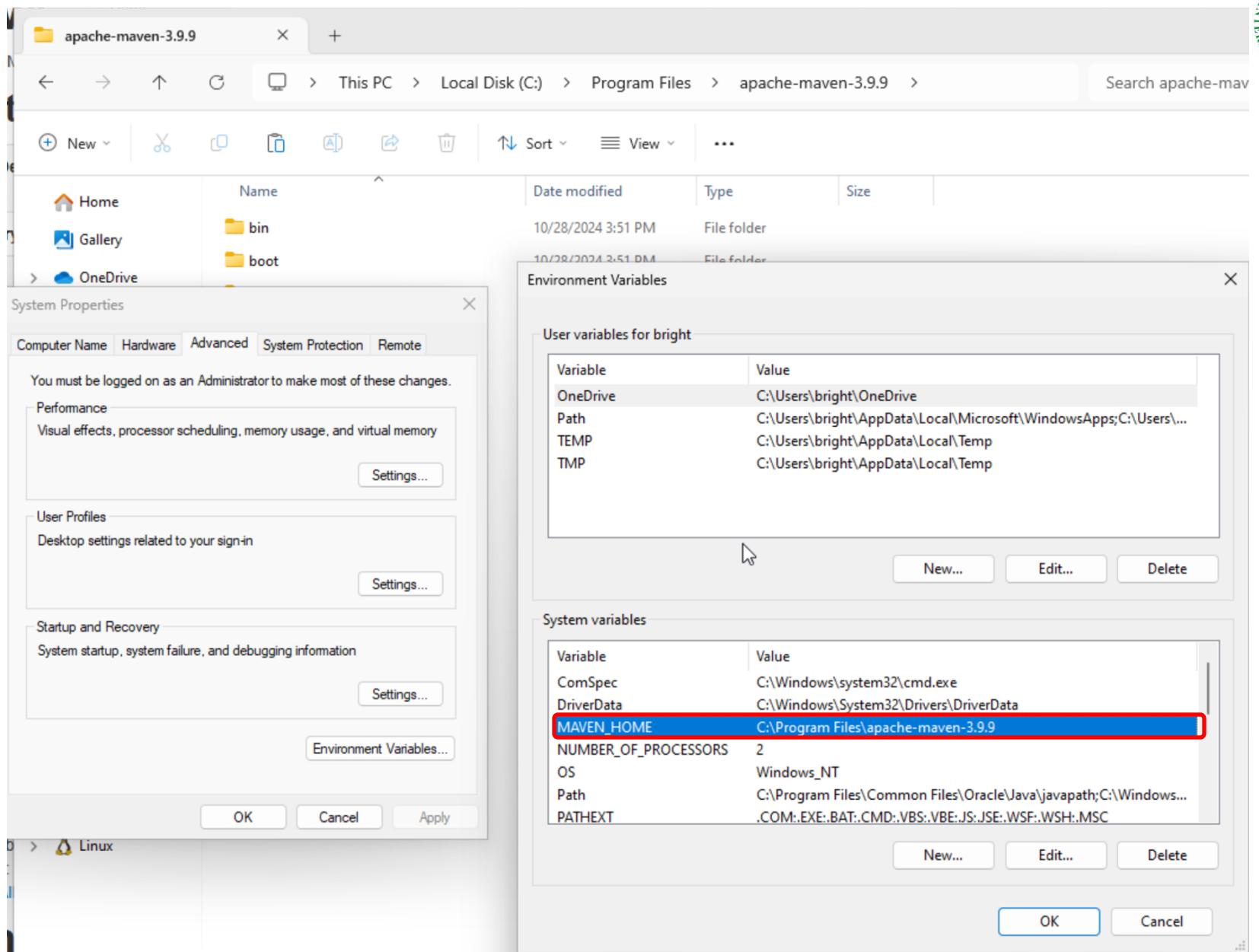
Maven is distributed in several formats for your convenience. Simply pick a ready-made binary distribution or source code archive.

In order to guard against corrupted downloads/installations, it is highly recommended to [verify the checksums](#).

	Link	Checksum
Binary tar.gz archive	<a href="#">apache-maven-3.9.9-bin.tar.gz</a>	apache-maven-3.9.9-bin.tar.gz.asc
<b>Binary zip archive</b>	<b><a href="#">apache-maven-3.9.9-bin.zip</a></b>	<b>apache-maven-3.9.9-bin.zip.asc</b>
Source tar.gz archive	<a href="#">apache-maven-3.9.9-src.tar.gz</a>	apache-maven-3.9.9-src.tar.gz.asc
Source zip archive	<a href="#">apache-maven-3.9.9-src.zip</a>	apache-maven-3.9.9-src.zip.asc

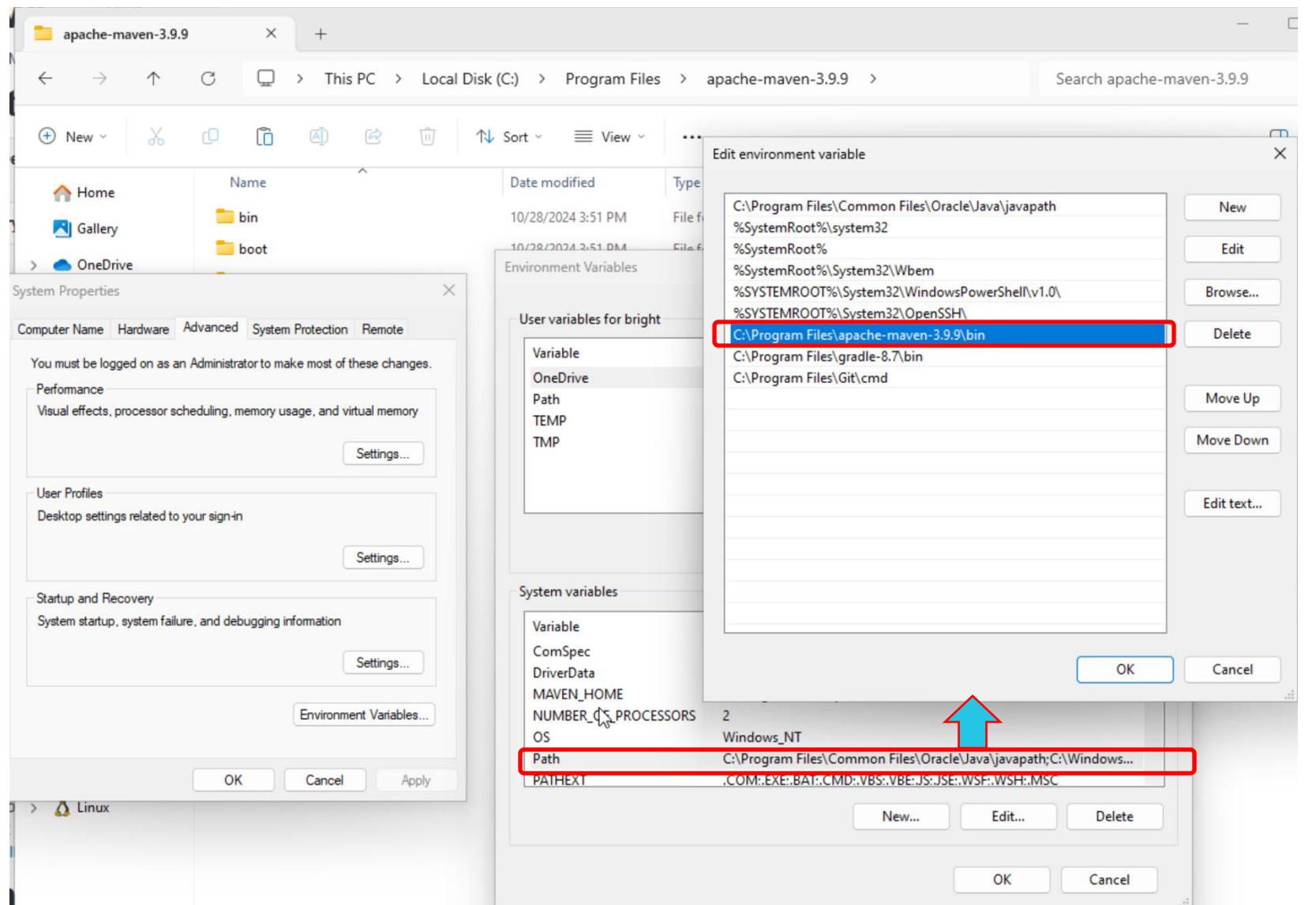
# Set MAVEN\_HOME

## Environment Variable



# Set Path

## Environment Variable



# Check mvn

mvn -v

```
Command Prompt + ▾  
Microsoft Windows [Version 10.0.22631.4317]  
(c) Microsoft Corporation. All rights reserved.  
  
C:\Users\bright>mvn -v  
Apache Maven 3.9.9 (8e8579a9e76f7d015ee5ec7bfcdc97d260186937)  
Maven home: C:\Program Files\apache-maven-3.9.9  
Java version: 23, vendor: Oracle Corporation, runtime: C:\Program Files\Java\jdk-23  
Default locale: en_US, platform encoding: UTF-8  
OS name: "windows 11", version: "10.0", arch: "amd64", family: "windows"  
  
C:\Users\bright>
```



# Install Maven

Mac OS

- Install
  - brew install maven
- Upgrade
  - brew upgrade maven

```
bright~$brew upgrade maven
==> Downloading https://formulae.brew.sh/api/formula.jws.json
#####
==> Downloading https://formulae.brew.sh/api/cask.jws.json
#####
Warning: maven 3.9.9 already installed
bright~$mvn -v
Apache Maven 3.9.9 (8e8579a9e76f7d015ee5ec7bfcfdc97d260186937)
Maven home: /usr/local/Cellar/maven/3.9.9/libexec
Java version: 22, vendor: Oracle Corporation, runtime: /Library/Java/JavaVirtual
Machines/jdk-22.jdk/Contents/Home
Default locale: en_US, platform encoding: UTF-8
OS name: "mac os x", version: "15.0.1", arch: "x86_64", family: "mac"
bright~$
```



## Install Maven

Linux

- `sudo apt-get install maven`

# Maven Commands

mvn compile

mvn clean

mvn package

mvn install

# Running all tests

mvn test

# Running all tests in a class

mvn test -Dtest=MyClassName

# Running single test method

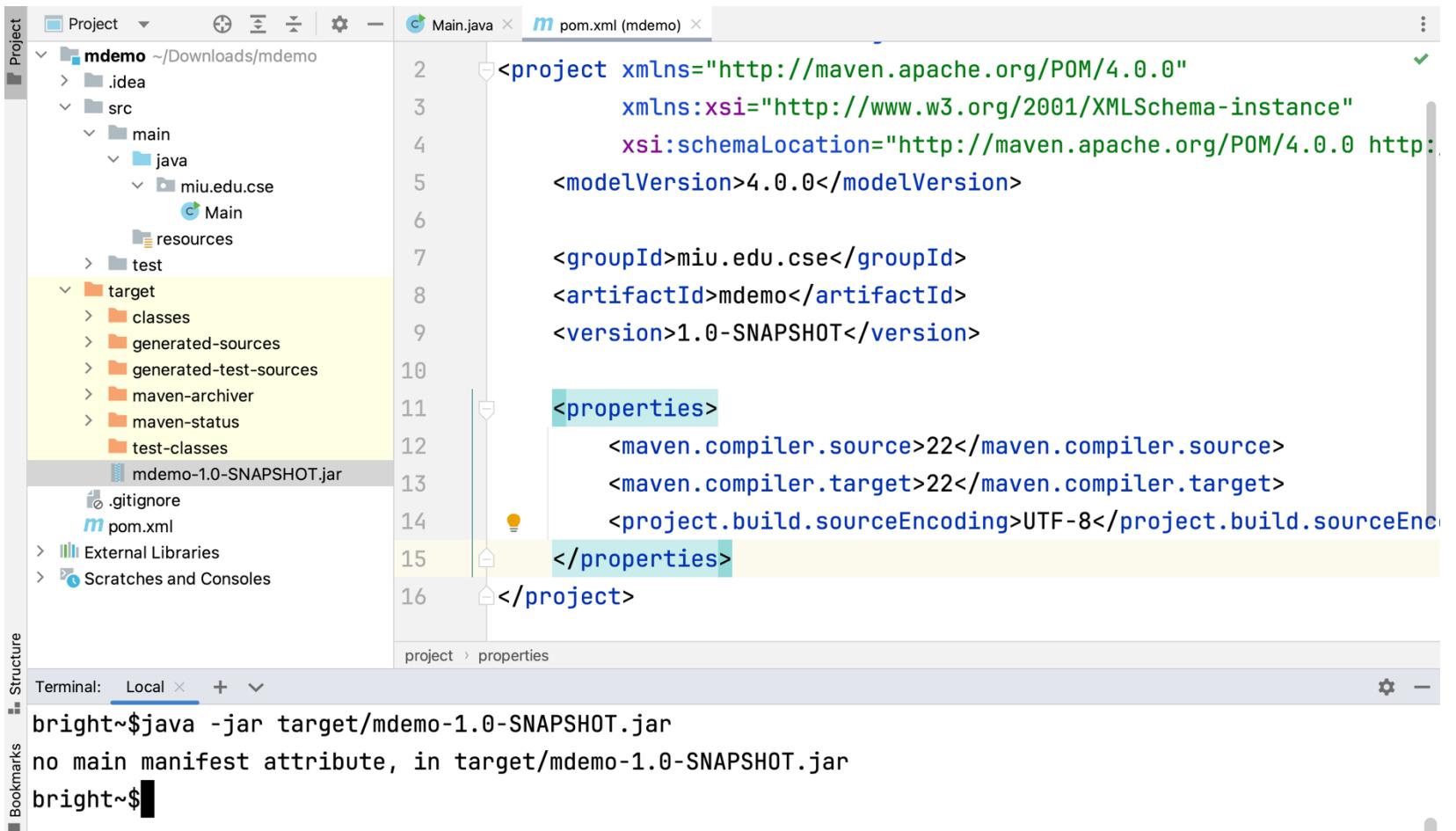
mvn test -Dtest=MyClassName#myTestMethod



```
bright~$pwd  
/Users/bright/.m2/repository/edu/miu/cs489/exdemo/0.0.1-SNAPSHOT  
bright~$ls  
_remote.repositories  
exdemo-0.0.1-SNAPSHOT.jar  
bright~$
```

exdemo-0.0.1-SNAPSHOT.pom  
maven-metadata-local.xml

# Fix this error



The screenshot shows an IntelliJ IDEA interface with the following details:

- Project View:** Shows the 'mdemo' project structure. The 'target' directory is highlighted in yellow.
- Main.java:** A Java file is shown in the editor.
- pom.xml (mdemo):** The XML configuration file is open in the editor. A warning icon is displayed next to the closing tag of the properties section on line 15.
- Terminal:** The command `bright~$java -jar target/mdemo-1.0-SNAPSHOT.jar` is run, resulting in the error message: `no main manifest attribute, in target/mdemo-1.0-SNAPSHOT.jar`.

# Add the following configuration in pom.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<project xmlns="http://maven.apache.org/POM/4.0.0"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:schemaLocation="http://maven.apache.org/POM/4.0.0
  http://maven.apache.org/xsd/maven-4.0.0.xsd">
  <modelVersion>4.0.0</modelVersion>

  <groupId>edu.miu.cse</groupId>
  <artifactId>demo2</artifactId>
  <version>1.0-SNAPSHOT</version>

  <properties>
    <maven.compiler.source>22</maven.compiler.source>
    <maven.compiler.target>22</maven.compiler.target>
    <project.build.sourceEncoding>UTF-8</project.build.sourceEncoding>
  </properties>

  <build>
    <plugins>
      <plugin>
        <groupId>org.apache.maven.plugins</groupId>
        <artifactId>maven-jar-plugin</artifactId>
        <version>3.4.2</version>
        <configuration>
          <archive>
            <manifest>
              <mainClass>
                edu.miu.cse.Main
              </mainClass>
            </manifest>
          </archive>
        </configuration>
      </plugin>
    </plugins>
  </build>
</project>
```



```
<build>
  <plugins>
    <plugin>
      <groupId>org.apache.maven.plugins</groupId>
      <artifactId>maven-jar-plugin</artifactId>
      <version>3.4.2</version>
      <configuration>
        <archive>
          <manifest>
            <mainClass>
              edu.miu.cse.Main
            </mainClass>
          </manifest>
        </archive>
      </configuration>
    </plugin>
  </plugins>
</build>
```

```
bright~$mvn clean package  
...  
bright~$java -jar target/mdemo-1.0-SNAPSHOT.jar  
Hello world!  
bright~$
```



DPE University is live! Learn more about Gradle Build Tool, Gradle Build Scan, Develocity and the ecosystem.

[About](#)[Docs](#)[Learn](#)[Get Help](#)

## Releases

Here you can find binaries and reference documentation for current and past versions of Gradle. You can find the next [release candidate](#) or a bleeding edge nightly build for the [snapshot](#) and [master](#) branches on their respective pages.

You can [install Gradle](#) through various other tools, or download a ZIP using the link on the [download page](#).

Command-line completion scripts for bash and zsh can be downloaded from the [gradle-completion project page](#).

# Install Gradle

Windows

## Getting Started Resources

There are many [Gradle Build Tool trainings](#) available to help you get started quickly. The new free courses are a part of the [DPE University](#) by Gradle Inc.

The Gradle team offers free [training](#)

↳ v8.10.2

📅 Sep 23, 2024

- Download: [binary-only](#) or [complete \(checksums\)](#)
- [User Manual](#)
- [API Javadoc](#)
- [Groovy DSL Reference](#)
- [Release Notes](#)

- <https://gradle.org/releases/>

# Download & Extract it to Program Files

← Extract Compressed (Zipped) Folders

### Select a Destination and Extract Files

Files will be extracted to this folder:  
C:\Program Files

Show extracted files when complete

15% complete

Copying 312 items from gradle-8.10.2-bin to Program Files

15% complete

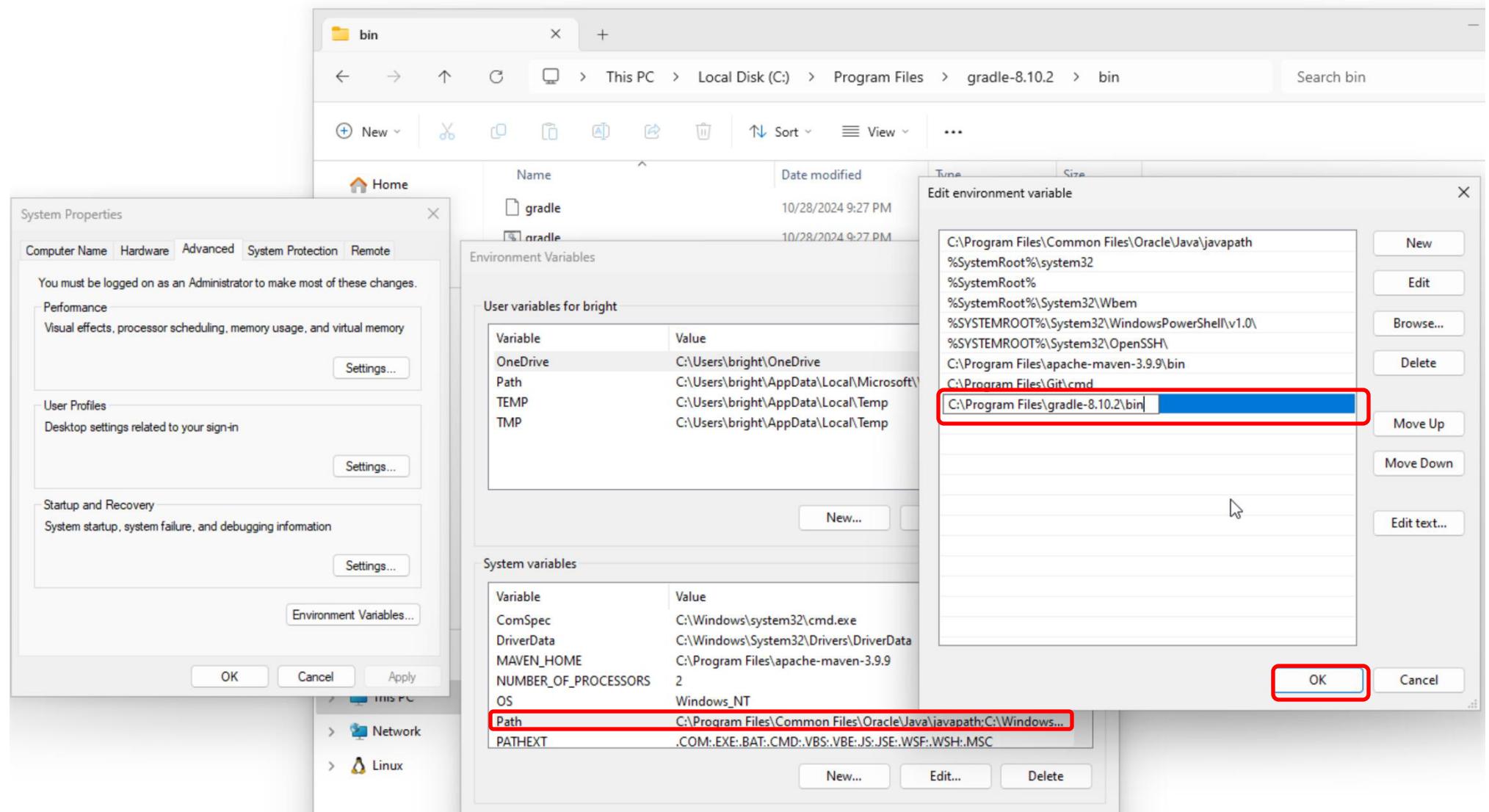


Speed: 991 KB/s

Name: guava-32.1.2-jre  
Time remaining: About 14 minutes  
Items remaining: 228 (123 MB)

^ Fewer details

# Set Environment Variables



# Check the Gradle tool

gradle -v

```
Command Prompt x + v
Microsoft Windows [Version 10.0.22631.4317]
(c) Microsoft Corporation. All rights reserved.

C:\Users\bright>gradle -v

Welcome to Gradle 8.10.2!

Here are the highlights of this release:
- Support for Java 23
- Faster configuration cache
- Better configuration cache reports

For more details see https://docs.gradle.org/8.10.2/release-notes.html

-----
Gradle 8.10.2

-----
Build time: 2024-09-23 21:28:39 UTC
Revision: 415adb9e06a516c44b391edff552fd42139443f7

Kotlin: 1.9.24
Groovy: 3.0.22
Ant: Apache Ant(TM) version 1.10.14 compiled on August 16 2023
Launcher JVM: 23 (Oracle Corporation 23+37-2369)
Daemon JVM: C:\Program Files\Java\jdk-23 (no JDK specified, using current Java home)
OS: Windows 11 10.0 amd64

C:\Users\bright>
```

# Install Gradle on Mac

brew install gradle

If you have already installed it, upgrade it

- brew upgrade gradle

```
bright~$brew upgrade gradle
==> Downloading https://formulae.brew.sh/api/formula.jws.json
#####
# 100.0%
==> Downloading https://formulae.brew.sh/api/cask.jws.json
#####
# 100.0%
Warning: gradle 8.10.2 already installed
bright~$gradle -v
-----
Gradle 8.10.2
-----
Build time: 2024-09-23 21:28:39 UTC
Revision: 415adb9e06a516c44b391edff552fd42139443f7

Kotlin: 1.9.24
Groovy: 3.0.22
Ant: Apache Ant(TM) version 1.10.14 compiled on August 16 2023
Launcher JVM: 22 (Oracle Corporation 22+36-2370)
Daemon JVM: /Library/Java/JavaVirtualMachines/jdk-22.jdk/Contents/Home (no JD
K specified, using current Java home)
OS: Mac OS X 15.0.1 x86_64

bright~$
```

# Install Gradle on Linux

Manually

## Step 1 - [Download](#) the latest Gradle distribution

The distribution ZIP file comes in two flavors:

- Binary-only (bin)
- Complete (all) with docs and sources

Downloading the bin file.

## Step 2 - Unpack the distribution

Unzip the distribution zip file in the directory of your choosing:

```
› mkdir /opt/gradle  
› unzip -d /opt/gradle gradle-8.10.2-bin.zip  
› ls /opt/gradle/gradle-8.10.2
```

## Step 3 - Configure your system environment

To install Gradle, the path to the unpacked files needs to be in your Path.

Configure your PATH environment variable to include the bin directory of the unzipped distribution,

```
› export PATH=$PATH:/opt/gradle/gradle-8.10.2/bin
```

# Install Gradle on Linux

Installing with a package manager

```
sudo apt-get update  
sudo apt-get install gradle
```

## Run Maven

### Project

- demo3 ~/Documents/NovCS489\_P/c
  - .gradle
  - .idea
  - build
  - gradle
  - src
    - main
      - java
        - edu.miu.cse
          - Main
      - resources
    - test
  - .gitignore
  - build.gradle.kts
  - gradlew
  - gradlew.bat
  - settings.gradle.kts
  - External Libraries
  - Scratches and Consoles

```
plugins {  
    id("java")  
    application  
}  
  
group = "edu.miu.cse"  
version = "1.0-SNAPSHOT"  
  
repositories {  
    mavenCentral()  
}  
  
application{  
    mainClass = "edu.miu.cse.Main"  
}  
  
dependencies {  
    testImplementation(platform("org.junit:junit-bom:5.10.0"))  
    testImplementation("org.junit.jupiter:junit-jupiter")  
}  
  
tasks.jar{  
    manifest{  
        attributes["Main-Class"] = "edu.miu.cse.Main"  
    }  
}  
  
tasks.test {  
    useJUnitPlatform()  
}
```

# How to clean, build, and run?

`./gradlew clean build`

BUILD SUCCESSFUL in 1s

6 actionable tasks: 6 executed

`~/Documents/NovCS489_P/demo3`

`java -jar build/libs/demo3-1.0-SNAPSHOT.jar`

Hello world!

`~/Documents/NovCS489_P/demo3`

`./gradlew run`

Project  build.gradle.kts (demo3) 

```

1 plugins {
2     id("java")
3     application
4 }
5
6 group = "edu.miu.cse"
7 version = "1.0-SNAPSHOT"
  
```

Terminal Local  

`~/Documents/NovCS489_P/demo3`

`./gradlew clean build`

BUILD SUCCESSFUL in 1s

6 actionable tasks: 6 executed

`~/Documents/NovCS489_P/demo3`

`java -jar build/libs/demo3-1.0-SNAPSHOT.jar`

Hello world!

`~/Documents/NovCS489_P/demo3`

`./gradlew run`

> Task :run

Hello world!

BUILD SUCCESSFUL in 775ms

2 actionable tasks: 1 executed, 1 up-to-date

Project build.gradle.kts (gradledemo)

```
> build
  - gradle
    - wrapper
      - gradle-wrapper.jar
      - gradle-wrapper.properties
  - src
    - main
      - java
        - miu.edu.cse
          - Main
          - MyCalc
        - resources
    - test
    - .gitignore
    - build.gradle.kts
    - gradlew
    - gradlew.bat
    - settings.gradle.kts
  - External Libraries
  - < 19 > /Users/bright/Library/Java/Java
```

15 }  
16  
17 tasks.test {  
18 useJUnitPlatform()  
19 }  
20  
21 tasks.run {  
22 register<JavaExec>( name: "run" ) {  
23 mainClass = "miu.edu.cse.Main"  
24 classpath = sourceSets.main.get().runtimeClasspath  
25 }  
26 }

Terminal: Local + ▾  
bright~\$./gradlew run

```
> Task :run  
Hello and welcome!i = 1  
i = 2  
i = 3  
i = 4  
i = 5
```

# GitHub Action

Experimenting is fun 😊

## Get started with GitHub Actions

Build, test, and deploy your code. Make code reviews, branch management, and issue triaging work the way you want. Select a workflow to get started.

Skip this and [set up a workflow yourself →](#)

Search workflows

### Suggested for this repository

#### Publish Java Package with Maven

By GitHub Actions

Build a Java Package using Maven and publish to GitHub Packages.

Configure

#### Java with Maven

By GitHub Actions

Build and test a Java project with Apache Maven.

Configure

#### Java with Ant

By GitHub Actions

Build and test a Java project with Apache Ant.

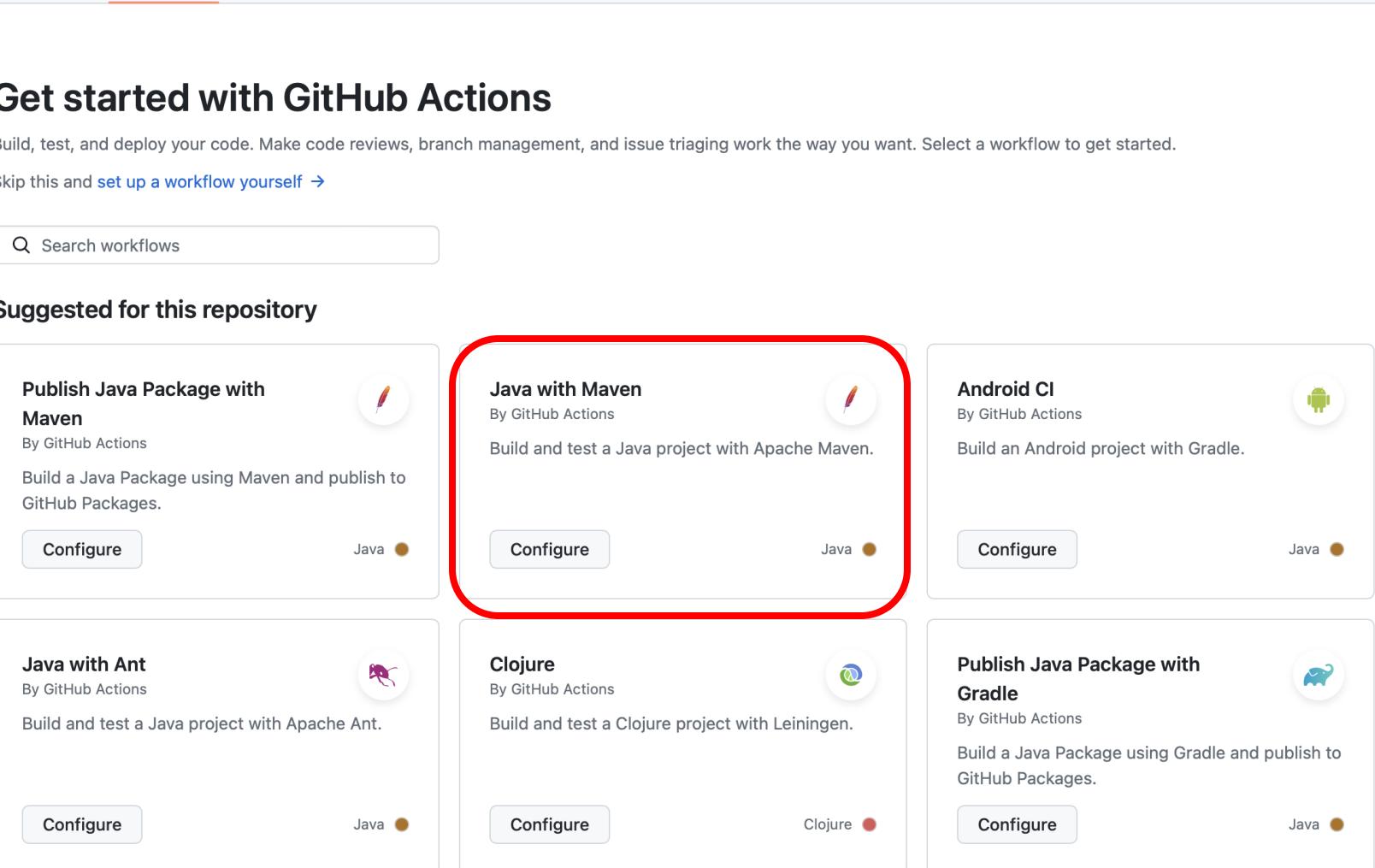
Configure

#### Clojure

By GitHub Actions

Build and test a Clojure project with Leiningen.

Configure



[Edit](#)[Preview](#)

Code 55% faster with GitHub Copilot

Spaces

2

No w

# GitHub Action

Experimenting is fun 😊

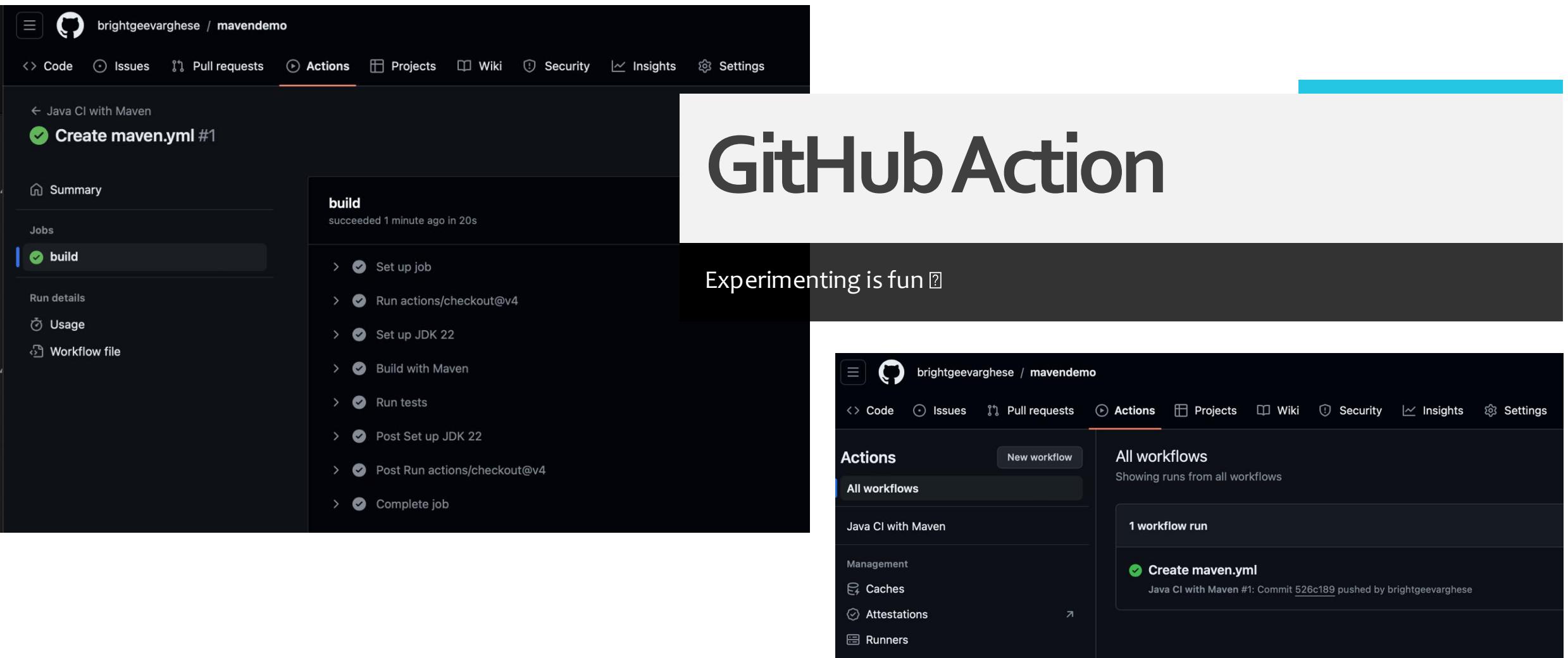
```
1  # This workflow will build a Java project with Maven, and cache/restore any dependencies to improve the workflow execution time
2  # For more information see: https://docs.github.com/en/actions/automating-builds-and-tests/building-and-testing-java-with-maven
3
4  # This workflow uses actions that are not certified by GitHub.
5  # They are provided by a third-party and are governed by
6  # separate terms of service, privacy policy, and support
7  # documentation.
8
9  name: Java CI with Maven
10
11 on:
12   push:
13     branches: [ "main" ]
14   pull_request:
15     branches: [ "main" ]
16
17 jobs:
18   build:
19
20     runs-on: ubuntu-latest
21
22     steps:
23       - uses: actions/checkout@v4
24       - name: Set up JDK 22
25         uses: actions/setup-java@v3
26         with:
27           java-version: '22'
28           distribution: 'temurin'
29           cache: maven
30       - name: Build with Maven
31         run: mvn -B package --file pom.xml
32
33       - name: Run tests
34         run: mvn test
35
36       # Optional: Uploads the full dependency graph to GitHub to improve the quality of Dependabot alerts this repository can receive
37       # - name: Update dependency graph
38       #   uses: advanced-security/maven-dependency-submission-action@571e99aab1055c2e71a1e2309b9691de18d6b7d6
```

This action retrieves the project's code repository from GitHub and makes it available to the runner machine that executes the workflow.

```
...
- name: Login to Docker registry (using secrets)
  env:
    DOCKER_USERNAME: ${{ secrets.DOCKER_USERNAME }}
    DOCKER_PASSWORD: ${{ secrets.DOCKER_PASSWORD }}
  run: docker login -u $DOCKER_USERNAME -p $DOCKER_PASSWORD

- name: Build Docker image
  run: docker build -t my-java-app:latest . # Replace "my-java-app" with your desired image name

- name: Push Docker image to registry (Optional)
  run: docker push my-java-app:latest
```



The image shows two screenshots of the GitHub Actions interface.

**Left Screenshot:** A detailed view of a workflow run titled "Create maven.yml #1". The "build" job is highlighted, showing its steps: Set up job, Run actions/checkout@v4, Set up JDK 22, Build with Maven, Run tests, Post Set up JDK 22, Post Run actions/checkout@v4, and Complete job. The build succeeded 1 minute ago in 20s.

**Right Screenshot:** The "All workflows" page for the repository "brightgeevarghese / mavendemo". It displays 1 workflow run for "Java CI with Maven" triggered by commit 526c189. The workflow run is labeled "Create maven.yml" and was pushed by brightgeevarghese.

# GitHub Action

Experimenting is fun ☀



brightgeevarghese Create gradle.yml ✓

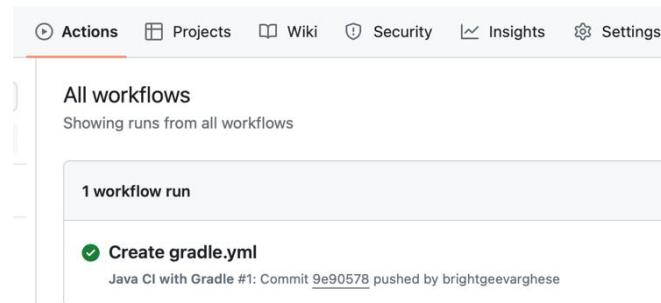
Code

Blame

71 lines (57 loc) · 2.37 KB

Code 55% faster with GitHub Copilot

```
1  # This workflow uses actions that are not certified by GitHub.
2  # They are provided by a third-party and are governed by
3  # separate terms of service, privacy policy, and support
4  # documentation.
5  # This workflow will build a Java project with Gradle and cache/restore any dependencies to improve the wor
6  # For more information see: https://docs.github.com/en/actions/automating-builds-and-tests/building-and-testin
7
8  name: Java CI with Gradle
9
10 on:
11   push:
12     branches: [ "main" ]
13   pull_request:
14     branches: [ "main" ]
15
16 jobs:
17   build:
18
19     runs-on: ubuntu-latest
20     permissions:
21       contents: read
22
23     steps:
24       - uses: actions/checkout@v4
25       - name: Set up JDK 19
26         uses: actions/setup-java@v4
27         with:
28           java-version: '19'
29           distribution: 'temurin'
30
31     # Configure Gradle for optimal use in GitHub Actions, including caching of downloaded dependencies.
32     # See: https://github.com/gradle/actions/blob/main/setup-gradle/README.md
33     - name: Setup Gradle
34       uses: gradle/actions/setup-gradle@417ae3ccd767c252f5661f1ace9f835f9654f2b5 # v3.1.0
35
36     - name: Build with Gradle Wrapper
37       run: ./gradlew build
38     - name: Test with Gradle Wrapper
39       run: ./gradlew test
```



All workflows  
Showing runs from all workflows

1 workflow run

Create gradle.yml  
Java CI with Gradle #1: Commit 9e90578 pushed by brightgeevarghese

# Requirements for self-hosted runner machines

<https://docs.github.com/en/actions/hosting-your-own-runners/managing-self-hosted-runners/about-self-hosted-runners>

```
<?xml version="1.0" encoding="UTF-8"?>
<project xmlns="http://maven.apache.org/POM/4.0.0"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:schemaLocation="http://maven.apache.org/POM/4.0.0
http://maven.apache.org/xsd/maven-4.0.0.xsd">
    <modelVersion>4.0.0</modelVersion>

    <groupId>miu.edu.cse</groupId>
    <artifactId>demo</artifactId>
    <version>1.0-SNAPSHOT</version>

    <properties>
        <maven.compiler.source>22</maven.compiler.source>
        <maven.compiler.target>22</maven.compiler.target>
        <project.build.sourceEncoding>UTF-8</project.build.sourceEncoding>
    </properties>
```

```
<dependencies>
```

```
  <dependency>
    <groupId>org.junit.jupiter</groupId>
    <artifactId>junit-jupiter</artifactId>
    <version>5.8.1</version>
    <scope>test</scope>
  </dependency>
</dependencies>
```

```
<build>
  <plugins>
    <!-- compiling Java source code within a Maven project-->
    <plugin>
      <groupId>org.apache.maven.plugins</groupId>
      <artifactId>maven-compiler-plugin</artifactId>
      <version>3.11.0</version>
      <configuration>
        <enablePreview>true</enablePreview>
      </configuration>
    </plugin>
    <!-- for executing the JUnit tests during the Maven build process.-->
    <plugin>
      <groupId>org.apache.maven.plugins</groupId>
      <artifactId>maven-surefire-plugin</artifactId>
      <version>3.2.5</version>
    </plugin>
  </plugins>
</build>
</project>
```