

# ALOM - TP 1 - Setup, Maven & Tooling

## Table of Contents

1. Installation .....	1
1.1. Vérification de l'installation de Java .....	1
1.2. Configuration de Java 21 sur les postes de l'université .....	1
1.3. Vérification de l'installation de Maven .....	2
1.4. Configuration du localRepository .....	2
2. Initialisation du projet .....	3
2.1. Création de l'arborescence projet .....	3
2.2. Ajouter du code ! .....	3
2.3. Exécuter les tests .....	4

## 1. Installation

### 1.1. Vérification de l'installation de Java

```
$ java --version
openjdk 21.0.2 2024-01-16 LTS
OpenJDK Runtime Environment Temurin-21.0.2+13 (build 21.0.2+13-LTS)
OpenJDK 64-Bit Server VM Temurin-21.0.2+13 (build 21.0.2+13-LTS, mixed mode, sharing)

$ echo $JAVA_HOME
/opt/jdk-21.0.2+13
```

### 1.2. Configuration de Java 21 sur les postes de l'université



Si Java n'est pas installé, vous pouvez télécharger une distribution de Java chez Adoptium : <https://adoptium.net/>

Faites un extract d'un tar.gz contenant le jdk dans un répertoire '/local/\$USER' par exemple.



Sur mes postes Linux, j'installe mes jdk et maven dans `/opt`. Adaptez les commandes ci-dessous par rapport au répertoire où vous installez votre jdk.

Modifier le fichier `~/bashrc` pour y ajouter les lignes suivantes :

```
export JAVA_HOME=/local/$USER/jdk-21.0.4+7
export PATH=$JAVA_HOME/bin:$PATH
```

Adaptez en fonction de votre répertoire d'installation !



Java **doit** être installé et la variable d'environnement **JAVA\_HOME** **doit** être renseignée.

## 1.3. Vérification de l'installation de Maven

```
$ mvn -v
Apache Maven 3.8.4 (9b656c72d54e5bacbed989b64718c159fe39b537)
Maven home: /opt/apache-maven-3.8.4
Java version: 21.0.2, vendor: Eclipse Adoptium, runtime: /opt/jdk-21.0.2+13
Default locale: en_US, platform encoding: UTF-8
OS name: "linux", version: "6.9.3-76060903-generic", arch: "amd64", family: "unix"
```



Si Maven n'est pas installé, suivre la procédure sur <http://maven.apache.org/download.cgi> et <http://maven.apache.org/install.html>

1. Télécharger maven (prenez bien le 'Binary Archive')
2. extraire le zip ou le tar.gz
3. Ajouter le répertoire **bin** au **PATH**

## 1.4. Configuration du localRepository

La création d'un lien symbolique en remplacement du répertoire local maven permet de sauver votre quota !

```
$ mkdir -p ~/.m2 ①
$ mkdir -p /local/$USER/.m2/repository ②
$ ln -s /local/$USER/.m2/repository ~/.m2/repository ③
```

- ① Création du répertoire local maven (dans le home par défaut)
- ② Création d'un répertoire sur le disque /local (non soumis à quota)
- ③ Création du lien symbolique



Cette étape est cruciale, car elle pourrait sauver votre quota si vous travaillez sur les PC de l'université !

## 2. Initialisation du projet

### 2.1. Création de l'arborescence projet

Créer un répertoire projet :

```
$ mkdir monProjet
```

Créer les répertoires de sources java et de test

```
$ cd monProjet
$ mkdir -p src/main/java
$ mkdir -p src/test/java
```

Initialiser un fichier pom.xml à la racine du projet

```
<project>
  <modelVersion>4.0.0</modelVersion>
  <groupId>com.alom.tp</groupId>
  <artifactId>tp-maven</artifactId>
  <version>0.1.0</version>

  <properties>
    <maven.compiler.source>21</maven.compiler.source> ①
    <maven.compiler.target>21</maven.compiler.target> ②
  </properties>

  <dependencies>
    <dependency>
      <groupId>junit</groupId>
      <artifactId>junit</artifactId>
      <version>4.13.2</version>
    </dependency>
  </dependencies>

</project>
```

① On indique à maven quelle version de Java utiliser pour les sources !

② On indique à maven quelle version de JVM on cible !

### 2.2. Ajouter du code !

Créer une classe Java dans le répertoire `src/main/java`

*Hello.java*

```
public class Hello{
    public String getMessage() {
        return "Hello World";
    }
}
```

Créer une classe de tests unitaires dans le répertoire `src/test/java`

*HelloTest.java*

```
import org.junit.Test;
import static org.junit.Assert.assertEquals;

public class HelloTest{

    @Test
    public void testGetMessage(){
        assertEquals("Hello World", new Hello().getMessage());
    }

}
```

## 2.3. Exécuter les tests

Lancer la commande

```
$ mvn test

[INFO] Scanning for projects...
[INFO]
[INFO] -----< com.alom.tp:tp-maven >-----
[INFO] Building tp-maven 0.1.0
[INFO] -----[ jar ]-----
[INFO]
[INFO] --- maven-resources-plugin:2.6:resources (default-resources) @ tp-maven ---
[INFO] skip non existing resourceDirectory /home/jwittouck/workspaces/alom/tp-alom-2022-2024/src/main/resources
[INFO]
[INFO] --- maven-compiler-plugin:3.1:compile (default-compile) @ tp-maven ---
[INFO] Changes detected - recompiling the module!
[INFO] Compiling 1 source file to /home/jwittouck/workspaces/alom/tp-alom-2022-2024/target/classes
[INFO]
[INFO] --- maven-resources-plugin:2.6:testResources (default-testResources) @ tp-maven ---
[INFO] skip non existing resourceDirectory /home/jwittouck/workspaces/alom/tp-alom-2022-2024/src/test/resources
```

```
[INFO]
[INFO] --- maven-compiler-plugin:3.1:testCompile (default-testCompile) @ tp-maven ---
[INFO] Changes detected - recompiling the module!
[INFO] Compiling 1 source file to /home/jwittouck/workspaces/alom/tp-alom-2022-
2024/target/test-classes
[INFO]
[INFO] --- maven-surefire-plugin:2.12.4:test (default-test) @ tp-maven ---
[INFO] Surefire report directory: /home/jwittouck/workspaces/alom/tp-alom-2022-
2024/target/surefire-reports

-----
T E S T S
-----

Running HelloTest
Tests run: 1, Failures: 0, Errors: 0, Skipped: 0, Time elapsed: 0.041 sec

Results :

Tests run: 1, Failures: 0, Errors: 0, Skipped: 0

[INFO] -----
[INFO] BUILD SUCCESS
[INFO] -----
[INFO] Total time: 1.626 s
[INFO] Finished at: 2022-08-19T17:15:21+02:00
[INFO] -----
```