Licence Informatique 3 année

POO & Java

TP n°5 – Intégration de Maven dans un Projet Java

Contexte

Ce TP fait suite aux **TP3/TP4**, dans lesquels vous avez manipulé des fichiers multimodaux et introduit le multithreading en Java. L'objectif de cette séance est d'intégrer **Maven** dans le projet développé lors des **TP2**, **TP3**, et **TP4 (bonus)**. Maven est un **gestionnaire de projet** qui facilite la gestion des dépendances, la compilation, les tests, et la génération d'exécutables Java. Pour cela, il est vivement recommandé de vous appuyer sur le Cours 5 - Apache Maven.

Les étapes suivantes sont à titre indicatif, et des ressources externes sont également mises à votre disposition (voir la section **Ressources**).

Énoncé du TP

1. Initialisation du Projet avec Maven

- Créez un projet Maven depuis votre IDE (Eclipse/IntelliJ) ou manuellement via la ligne de commande.
- Configurez le fichier `pom.xml` pour inclure les informations suivantes :
  - Le nom du projet, sa version et sa description.
  - La configuration d'une **dépendance Java standard** comme `JUnit` pour les tests.

Exemple de `pom.xml` minimal :

```xml
<project xmlns="http://maven.apache.org/POM/4.0.0" xmlns:xsi="http://www.w3.org/2001
    <modelVersion>4.0.0</modelVersion>
    <groupId>com.univlemans</groupId>
    <artifactId>tp5-maven</artifactId>
    <version>1.0-SNAPSHOT</version>
    <description>Projet Maven pour TP5</description>
    <dependencies>
        <!-- Exemple de dépendance -->
        <dependency>
            <groupId>junit</groupId>
            <artifactId>junit</artifactId>
            <version>4.13.2</version>
            <scope>test</scope>
        </dependency>
    </dependencies>
</project>
```

2. Organisation des Fichiers avec Maven

Maven impose une structure de projet standard :

```
src/
 main/
|    java/          # Code source principal
|    resources/     # Ressources (fichiers de configuration, etc.)
 test/
     java/          # Tests unitaires
     resources/     # Ressources de test
```

- Déplacez les fichiers des **TP2**, **TP3**, et **TP4** dans cette structure.
- Assurez-vous que le projet compile correctement avec Maven (`mvn compile`).

3. Génération d'un JAR exécutable

- Configurez Maven pour générer un fichier JAR exécutable contenant votre projet.
- Ajoutez un plugin dans le `pom.xml` pour spécifier le point d'entrée (`Main` ou une autre classe).

Exemple :

```xml
<build>
    <plugins>
        <plugin>
            <groupId>org.apache.maven.plugins</groupId>
            <artifactId>maven-compiler-plugin</artifactId>
            <version>3.8.1</version>
            <configuration>
                <source>1.8</source>
                <target>1.8</target>
            </configuration>
        </plugin>
        <plugin>
            <groupId>org.apache.maven.plugins</groupId>
            <artifactId>maven-jar-plugin</artifactId>
            <version>3.2.0</version>
            <configuration>
                <archive>
                    <manifest>
                        <mainClass>com.univlemans.Main</mainClass>
                    </manifest>
                </archive>
            </configuration>
        </plugin>
    </plugins>
</build>
```

4. Gestion des Dépendances

Ajoutez des dépendances nécessaires pour vos tests et extensions (par exemple, manipulation de fichiers ou gestion de threads).

Étapes à Réaliser

1. Créez un nouveau projet Maven ou convertissez le projet existant en projet Maven.
2. Organisez les fichiers source et les tests dans la structure Maven standard.
3. Configurez Maven pour inclure les dépendances et générer un JAR exécutable.
4. Ajoutez et exécutez des tests unitaires avec JUnit pour valider votre code.

Livrables

1. Projet Maven complet, mis à jour sur le **répertoire GitLab** avec :
   - Le fichier `pom.xml` configuré correctement.
   - La structure standard de Maven respectée.
   - Les fichiers source et tests organisés dans les dossiers appropriés.
2. Documentation (README) mise à jour pour expliquer comment exécuter le projet avec Maven.
3. Un fichier JAR exécutable dans le dossier `target/`.
4. Démonstration du projet compilé et exécuté avec Maven.

Ressources

- Introduction à Maven
- Tutoriel Maven sur GeeksforGeeks
- JUnit et Maven