Haute école d'ingénierie et d'architecture Fribourg
Hochschule für Technik und Architektur Freiburg

Filière Informatique 2020-2021

Intégration des nombres complexes et des Unum en Java avec COJAC

Classe I3

Rapport

Travail de bachelor

Version : 0.2

Student : Cédric Tâche

Professor : Frédéric Bapst

Expert : Baptiste Wicht

Table des versions

Version	Date	Author	Description
0.1	03.06.2021	Cédric Tâche	Création de la structure
0.2	07.06.2021	Cédric Tâche	Description de Maven et du problème de compilation

Table des matières

Chapitre 1

Introduction

Chapitre 2

COJAC

2.1 Instrumentation

2.2 Maven

COJAC utilise Maven [1] pour créer le JAR. Maven [1] est un outil d'automatisation pour gérer les dépendances et produire une application. Cet outil peut télécharger les dépendances, compiler le projet, exécuter les tests unitaires et d'intégration et déployer le projet. Maven et Graddle sont les deux outils les plus souvent utilisés pour gérer les projets Java. Toute la configuration se trouve dans le fichier *pom.xml*.

2.2.1 Debug

Lorsqu'il y a un problème avec Maven, il y a plusieurs paramètres qui permettent d'obtenir plus d'informations. Cette section montre comment configurer ces paramètres sous IntelliJ et donne l'argument équivalent pour appeler Maven en ligne de commande.

Ouvrir la fenêtre Maven

Toutes les configurations effectuées ci-dessous seront effectuées depuis la fenêtre Maven dans IntelliJ. Le bouton pour l'ouvrir se situe sous **View** → **Tool Windows** → **Maven** comme montré sur la figure 2.1.

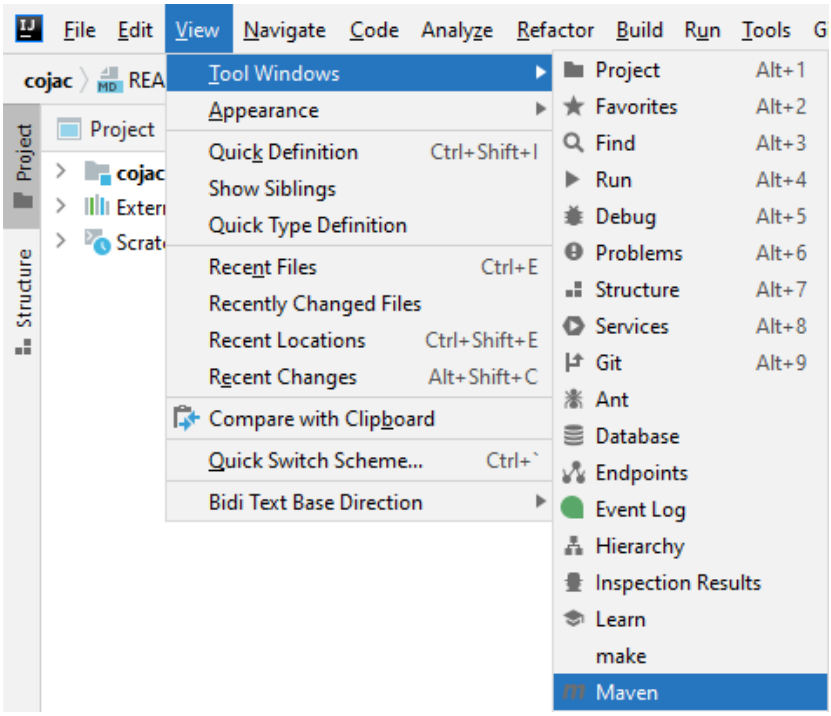

FIGURE 2.1 – Bouton d'ouverture de la fenêtre Maven

Fenêtre Maven

La fenêtre Maven montre le contenu du projet avec les étapes du cycle de vie de la production de l'application comme le montre la figure 2.2. Le bouton encadré sur l'image permet d'ouvrir les paramètres de Maven qui seront utilisés plus tard. On peut également voir les plugins et les dépendances. Cette fenêtre permet aussi d'exécuter les étapes pour produire l'application. L'étape *package* est suffisante pour générer le JAR.

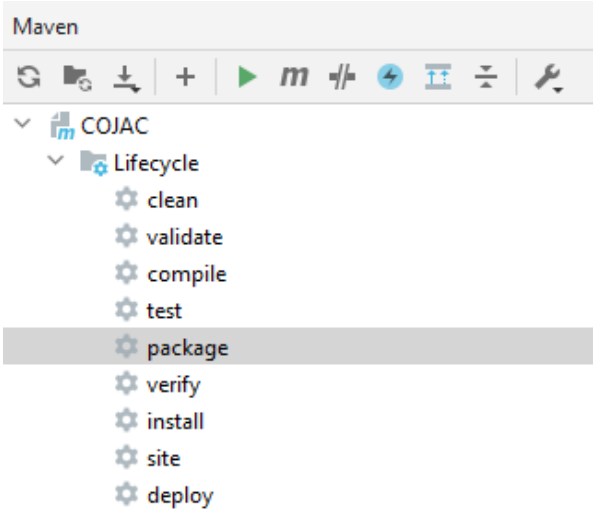

FIGURE 2.2 – Fenêtre Maven

Une option peut être ajoutée pour afficher les messages de debug. Cette option est plus souvent utilisée lors du développement de Maven ou d'un plugin Maven. Cependant, elle peut également être utile pour trouver la source d'un problème difficile. **TODO: image** . L'option en ligne de commande équivalente se nomme *--debug*.

Chapitre 3

Problèmes rencontrés

3.1 Surefire provoque une exception lors de l'exécution des tests par Maven

Surefire provoquait une erreur lors de l'exécution des tests lorsqu'un espace était présent dans le chemin d'accès.

3.1.1 Problème

Lors de la création du JAR, l'erreur suivante se produit lors de l'exécution des tests :

```
ExecutionException The forked VM terminated without properly saying
↪  goodbye. VM crash or System.exit called?
```

Le message suivant présent dans les logs indique la localisation des fichiers qui contiennent plus de détails.

```
[ERROR] Please refer to D:\Documents\Documents\HEIA\Semestre 6 -
↪  2021\Bachelor\Projet\cojac\target\surefire-reports for the
↪  individual test results.
```

3.1.2 Solution temporaire : ne pas exécuter les tests

Comme les problèmes ont uniquement lieu durant les tests, il est possible de générer le JAR sans les exécuter. **TODO: lien section Maven** Pour ce faire, il faut cliquer sur le bouton dédié dans la fenêtre Maven. Ce bouton est visible sur la figure 3.1.

Cependant, cette solution ne doit être utilisée que s'il y a besoin du JAR pour d'autres raisons. Il est nécessaire de corriger ce problème pour garantir la qualité du code.

FIGURE 3.1 – Bouton pour ignorer les tests

3.1.3 Cause

Dans un des fichiers de log produits, qui sont mentionnées dans la section 3.1.1, on peut trouver l'erreur suivante :

```
# Created at 2021-06-03T14:09:01.042
Error opening zip file or JAR manifest missing :
↪  D:\Documents\Documents\HEIA\Semestre
```

Ce lien est incomplet par rapport au chemin mentionné dans la section 3.1.1. Le chemin d'accès s'arrête lors du premier espace trouvé. Si le projet est déplacé dans un dossier ne contenant aucun espace, le JAR est correctement généré.

3.1.4 Solution

Il faut modifier une ligne de configuration du plugin Surefire dans le *pom.xml*. Des guillemets ont été ajoutés autour du chemin d'accès du JAR pour obtenir la ligne suivante.

```
1  <argLine>
2      -javaagent:"${basedir}/src/test/resources/cojac-agent-test.jar"
3  </argLine>
```

Ce changement corrige le problème.

Chapitre 4

Références

[1] Brett Porter, Jason van Zyl, and Olivier Lamy. Welcome to apache maven. URL `https://maven.apache.org/`.

Annexes