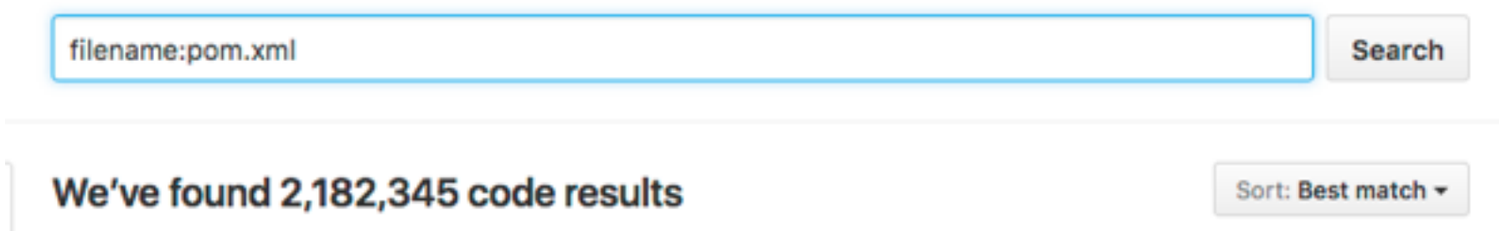
Let’s play with Maven

Cyril Cecchinel - cecchine@i3s.unice.fr
IUT Nice Côte d’Azur - Dept. Info S3A 2016/2017

basé en partie sur les slides de Sid Anand « Hands On With Maven »
un peu modifié par M. Blay-Fornarino

Maven en quelques mots

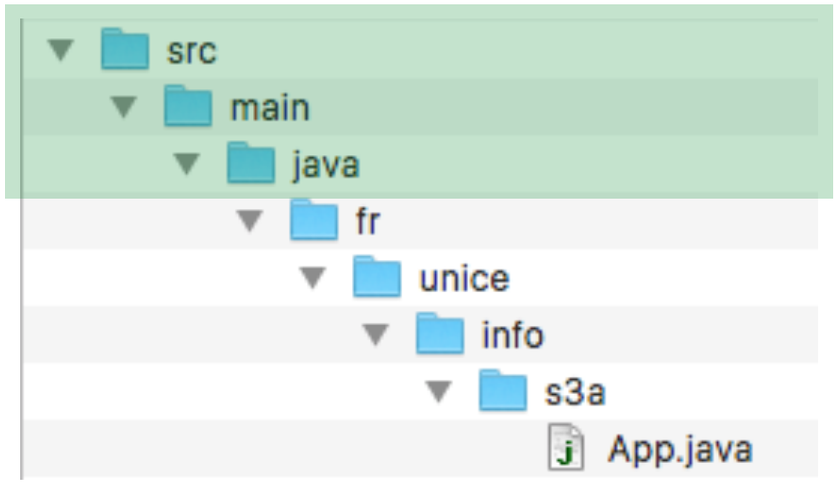
- Structure de répertoires standardisée
- Cycle de vie standardisé
- Gestion des dépendances
- Nombreux plugins
- Incontournable

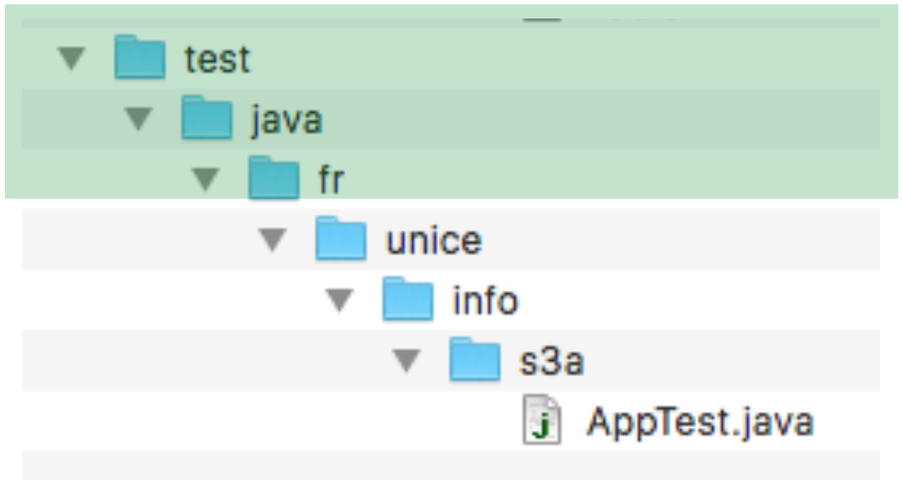


Structure de répertoire

Structure de répertoire

- Tout comme les conventions de nommage Java, les repertoires doivent suivre une hiérarchie précise
- Les sources Java vont dans **/src/main/java**
- Les tests Java vont dans **/src/test/java**



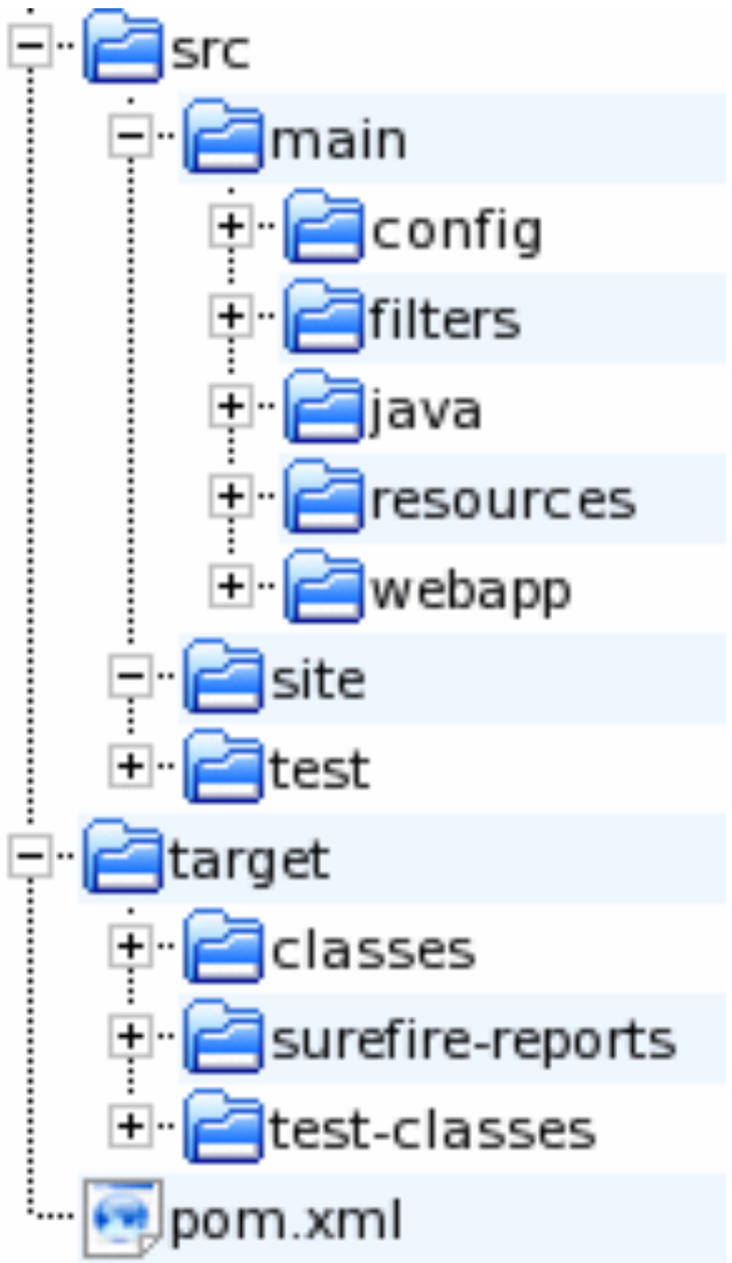


Structure de répertoire

- Si vous utilisez des ressources (e.g. fichiers de configuration, images) :
  - **/src/main/resources**
  - **/src/test/resources**
- Lors du build, **/target** est créé automatiquement et contiendra les .class et les fichiers jar éventuellement générés

Structure de répertoire

- Le répertoire src contient plusieurs sous-repertoires, chacun avec une utilité précise :
- src/main/java: Votre code java va ici (étonnamment)
- src/main/resources: Les autres ressources dont votre application a besoin
- src/main/config: Les fichiers de configuration
- src/test/java: Les tests unitaires
- src/test/resources: Les ressources nécessaires aux tests unitaires, qui ne seront pas déployées



Cycle de vie Maven

Cycle de vie par défaut

Phase	Description
Validate	Validation de la configuration Maven
Compile	Compilation des sources après résolution et téléchargement des dépendances
Test	Exécution des tests unitaires
Package	Création du fichier JAR
Verify	Vérification des critères de qualité
Install	Installation de l'application dans le repository local

- L'exécution de la commande « mvn <phase> » appelle **toutes les étapes antérieures à <phase> + la phase souhaitée**
  eg. « **mvn test** » appelle « **validate** », « **compile** » et enfin « **test** »

Plugins

- Maven est intrinsèquement constitué de plugins
  - ➡ Chacune des phases est un plugin

- L’ajout de plugins permet de ajouter/remplacer/ complémenter des phases Maven

Cycle de vie d’un projet

- compile: Compile le code source du projet
- test-compile: Compile les tests unitaires du projet
- test: Exécute les tests unitaires (typiquement avec Junit) dans le répertoire src/test
- package: Mets en forme le code compilé dans son format de diffusion (JAR, WAR, etc.)
- integration-test: Réalise et déploie le package si nécessaire dans un environnement dans lequel les tests d'intégration peuvent être effectués.
- install: Installe les produits dans l'entrepôt local, pour être utilisé comme dépendance des autres projets sur votre machine locale.

L’exécution de la commande « mvn <phase> » appelle **toutes les étapes antérieures à <phase> + la phase souhaitée**
eg. « **mvn test** » appelle « **compile** » et enfin « **test** »

POM.xml

POM.xml

Il s’agit du fichier de spécifications Maven

Le modèle objet projet

- Project Object Model = POM

```xml
<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/maven-v4_0_0.xsd">
    <modelVersion>4.0.0</modelVersion>
    <groupId>fr.unice.iut.info.methodo</groupId>
    <artifactId>members</artifactId>
    <packaging>jar</packaging>
    <version>1.0-SNAPSHOT</version>
    <name>Members registration</name>
    <url>http://mbf-iut.i3s.unice.fr/doku.php?id=2016_2017:s3:methodo:start</url>

    <developers>
        <developer>
            <id>blay</id>
            <name>Mireille Blay-Fornarino</name>
            <email>blay@i3s.unice.fr</email>
            <url>http://mireilleblayfornarino.i3s.unice.fr/</url>
            <timezone>+2</timezone>
            <roles>
                <role>Developer</role>
            </roles>
        </developer>
        <developer>
            <id>cecchinel</id>
            <name>Cyril Cecchinel</name>
            <email>cecchine@3s.unice.fr</email>
            <url>http://www.i3s.unice.fr/~cecchine</url>
            <timezone>+2</timezone>
            <roles>
                <role>Developer</role>
            </roles>
        </developer>
    </developers>                </plugin>
```

Dépendances

```xml
<dependencies>
        <dependency>
            <groupId>junit</groupId>
            <artifactId>junit</artifactId>
            <version>4.12</version>
            <scope>test</scope>
        </dependency>
        <dependency>
            <groupId>org.hamcrest</groupId>
            <artifactId>hamcrest-all</artifactId>
            <version>1.3</version>
            <scope>test</scope>
        </dependency>
        <dependency>
            <groupId>com.googlecode.json-simple</groupId>
            <artifactId>json-simple</artifactId>
            <version>1.1.1</version>
        </dependency>
    </dependencies>
```

POM.xml

- Exemple de dépendance:

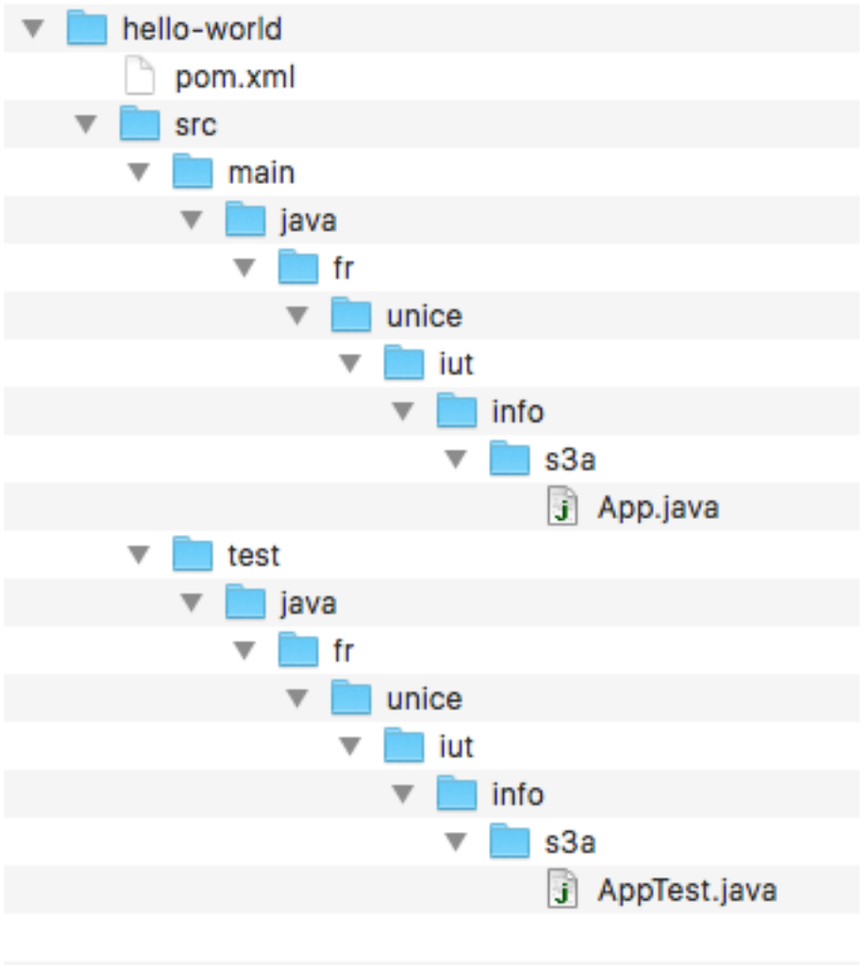
```
<dependency>
    <groupId>org.json</groupId>
    <artifactId>json</artifactId>
    <version>20090211</version>
</dependency>
```

- Moteur de recherche: http://mvnrepository.com/

Créer un projet Java Maven automatiquement

```
mvn archetype:generate
 -DgroupId={project-packaging}
 -DartifactId={project-name}
 -DarchetypeArtifactId=maven-archetype-quickstart
 -DinteractiveMode=false
```

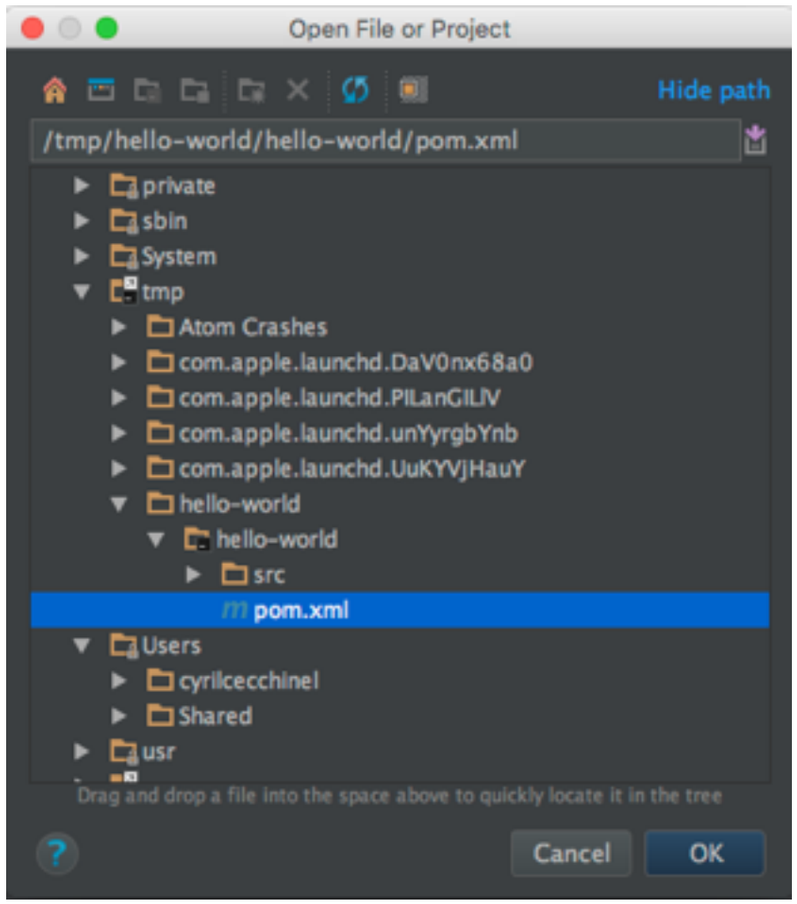
```
lyoko:hello-world cyrilcecchinel$ mvn archetype:generate -DgroupId=fr.unice.iut.info.s3a -DartifactId=hello-world -DarchetypeArtifactId=maven-archetype-quickstart -DinteractiveMode=false
[INFO] Scanning for projects...
[INFO]
[INFO] ------------------------------------------------------------------------
[INFO] Building Maven Stub Project (No POM) 1
[INFO] ------------------------------------------------------------------------
[INFO]
[INFO] >>> maven-archetype-plugin:2.2:generate (default-cli) > generate-sources @ standalone-pom >>>
[INFO]
[INFO] <<< maven-archetype-plugin:2.2:generate (default-cli) < generate-sources @ standalone-pom <<<
[INFO]
[INFO] --- maven-archetype-plugin:2.2:generate (default-cli) @ standalone-pom ---
[INFO] Generating project in Batch mode
[INFO] ----------------------------------------------------------------------------
[INFO] Using following parameters for creating project from Old (1.x) Archetype: maven-archetype-quickstart:1.0
[INFO] ----------------------------------------------------------------------------
[INFO] Parameter: basedir, Value: /private/tmp/hello-world
[INFO] Parameter: package, Value: fr.unice.iut.info.s3a
[INFO] Parameter: groupId, Value: fr.unice.iut.info.s3a
[INFO] Parameter: artifactId, Value: hello-world
[INFO] Parameter: packageName, Value: fr.unice.iut.info.s3a
[INFO] Parameter: version, Value: 1.0-SNAPSHOT
[INFO] project created from Old (1.x) Archetype in dir: /private/tmp/hello-world/hello-world
[INFO] ------------------------------------------------------------------------
[INFO] BUILD SUCCESS
[INFO] ------------------------------------------------------------------------
[INFO] Total time: 3.047 s
[INFO] Finished at: 2016-08-23T17:32:20+02:00
[INFO] Final Memory: 15M/309M
[INFO] ------------------------------------------------------------------------
```

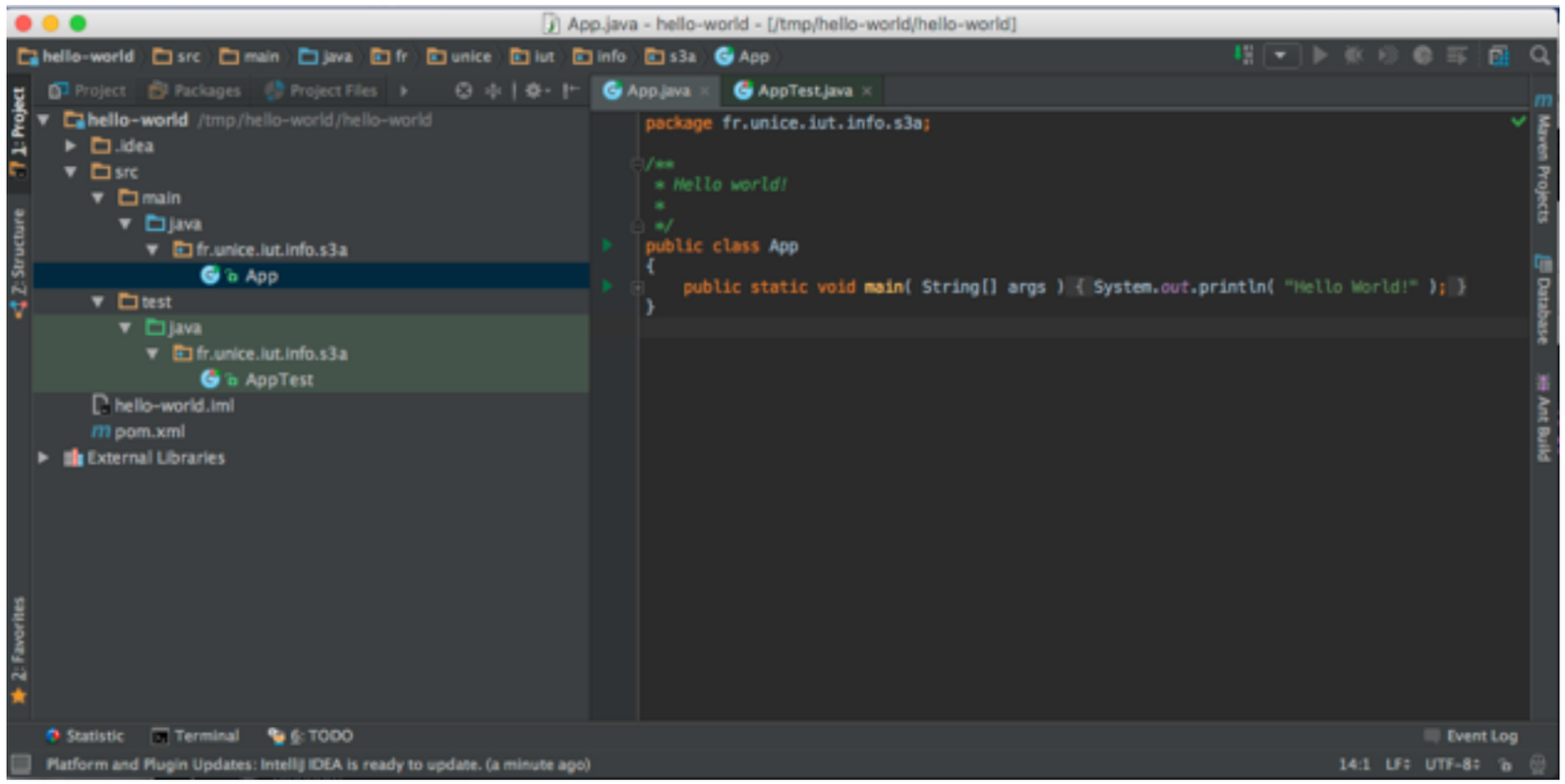


La structure de répertoire est créée automatiquement

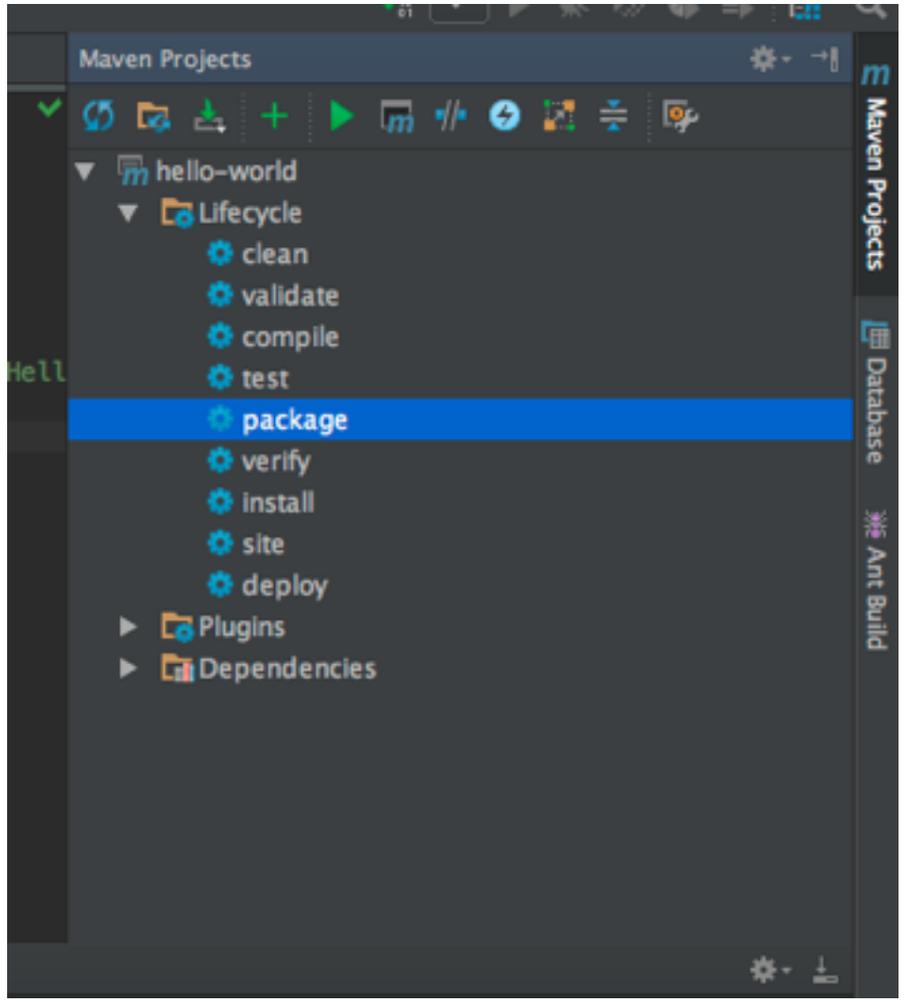
Un fichier pom.xml est créé automatiquement

- Le projet créé par Maven peut être directement ouvert dans IntelliJ grâce au **`pom.xml`**

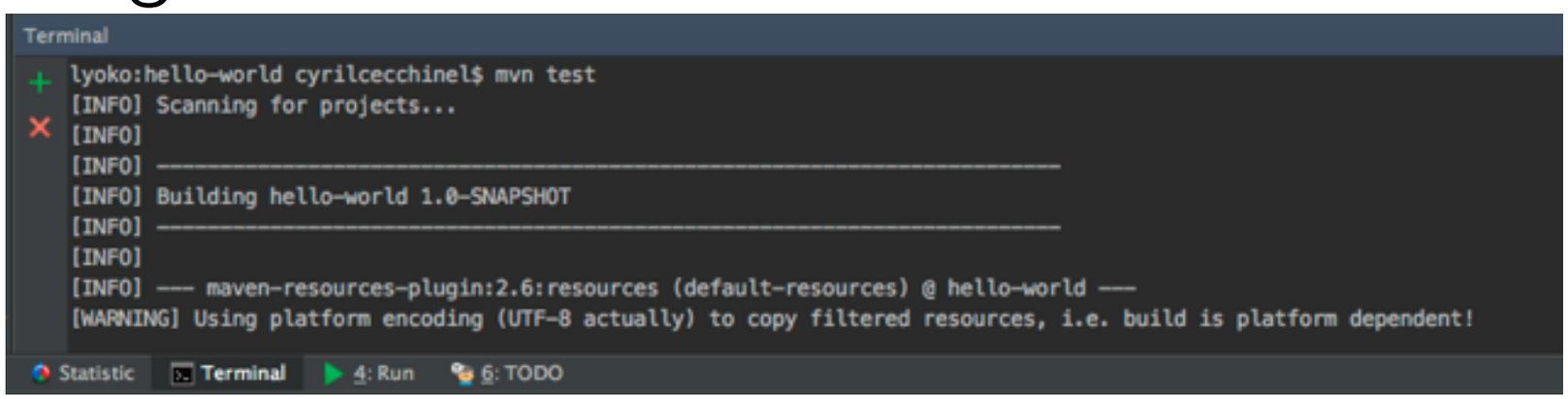




- Il est possible de déclencher les différentes étapes du cycle de vie Maven directement depuis IntelliJ …



- … ou en ligne de commande



Pratique

- Récupérer ces slides sur Jalon
- Créer un projet Maven correspondant à votre sujet de projet
  - groupId = fr.unice.iut.info.methodo.s3a
  - artifactId = <nom de votre sujet>
- Compiler le projet par défaut obtenu et vérifier la bonne exécution de JUnit