18/04/2023
Version 2

MAVEN

JÉRÉMY PERROUAULT

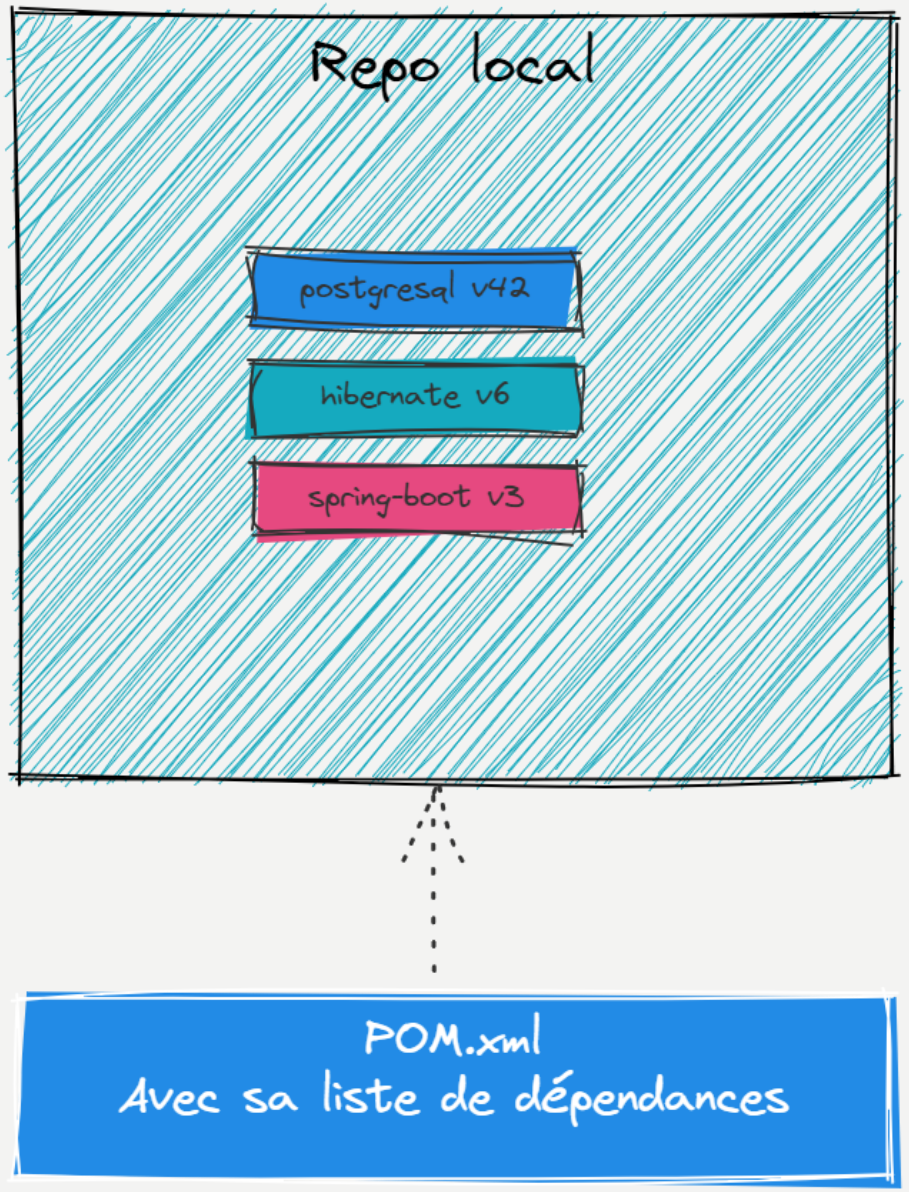
MAVEN

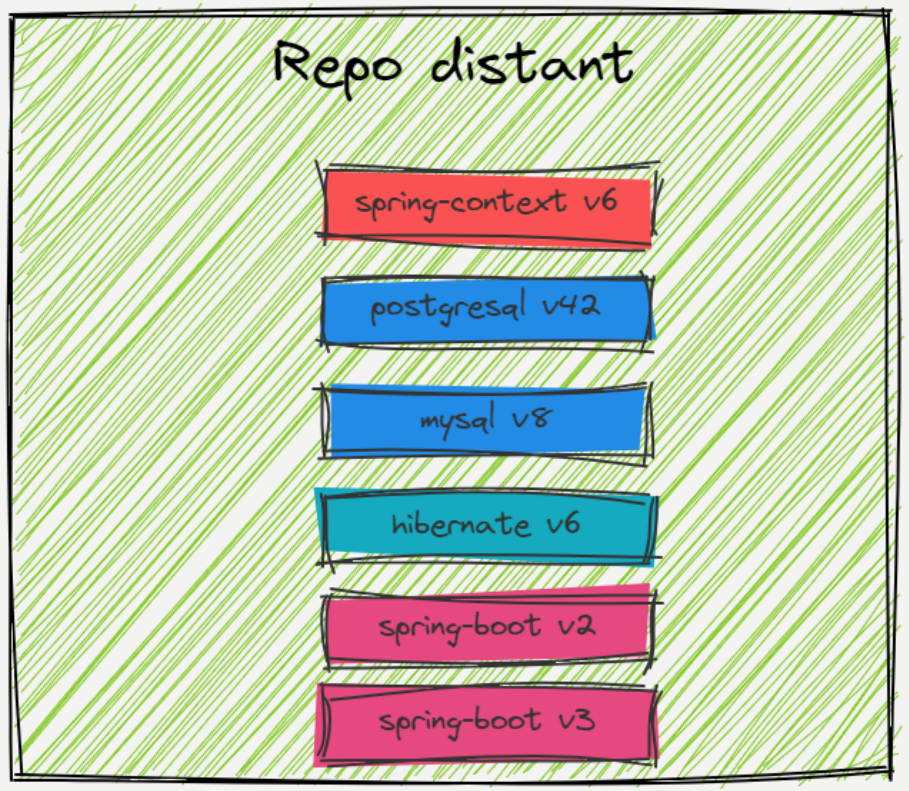
INTRODUCTION À MAVEN

INTRODUCTION

- Ensemble de standards
- Fournit un cycle de vie standard
  - Compiler (build)
  - Tester – Intégrer
  - Déployer
- S'articule autour d'une description de projet (**POM** – **P**roject **O**bject **M**odel)
- Organise les projets en composants interdépendants
  - Permet une gestion fine des dépendances
- Apporte cohérence, réutilisabilité, agilité et maintenabilité

INTRODUCTION





INTRODUCTION

- Le repository local est situé dans ce répertoire
  - *%USER_HOME%/.m2/repository*
- Vous pouvez la modifier dans la configuration de **Maven**, le fichier *settings.xml*
  - *maven/conf/settings.xml*

CONFIGURATION

- Chaque projet **Maven** est défini dans un fichier *pom.xml*
  - Format **XML**
- Contient l'identité du projet, le *GAV*
  - GroupId
  - ArtifactId
  - Version
- La liste des dépendances
- La liste des tâches (au travers de plugin) à réaliser pendant les phases
  - De compilation
  - De test
  - D'assemblage
  - D'installation

CONFIGURATION

```
<project>
  <modelVersion>4.0.0</modelVersion>
  <groupId>fr.formation</groupId>
  <artifactId>tp-maven</artifactId>
  <version>1.0</version>
  <packaging>jar</packaging>
  <properties>...</properties>
  <dependencies>...</dependencies>
  <build>...</build>
</project>
```

CONFIGURATION

- project
  - Racine du fichier *pom.xml*
- modelVersion
  - Version de **POM** utilisée
- groupId
  - Identifiant du groupe du projet
  - Suit les mêmes règles de nommage que les packages
- artifactId
  - Identifiant du projet dans le groupe
  - Utilisé par défaut pour construire l'artefact final
    - artefictid-version
- version
  - La version du projet
  - SNAPSHOT
    - Version-SNAPSHOT est la version en cours de développement

packaging
- Type de packaging du projet (**JAR**, **WAR** ou **POM**)

dependencies
- Liste des dépendances

build
- Liste des plugins pour le processus de construction

properties
- Propriétés, paramètres **Maven**

CONFIGURATION – PACKAGING

- **JAR**
  - Projet de bibliothèque **Java**
  - Projet d'éxécutable **Java**
- **WAR**
  - Projet WEB
- **POM**
  - Projet **Maven** générique
  - Pourra être utilisé par d'autres projet **Maven** (Héritage)

```
<parent>
  <groupId>fr.formation</groupId>
  <artifactId>maven-parent</artifactId>
  <version>0.0.1-SNAPSHOT</version>
</parent>
```

CONFIGURATION – PROPERTIES

- Propriétés, paramètres

```
<properties>
  <spring.version>6.0.8</spring.version>
</properties>
```

```
<dependency>
  <groupId>org.springframework</groupId>
  <artifactId>spring-context</artifactId>
  <version>${spring.version}</version>
</dependency>
```

- Modifier des propriétés préexistantes
  - Indiquer à **Maven** quelle version de **Java** utiliser

```
<properties>
  <maven.compiler.source>17</maven.compiler.source>
  <maven.compiler.target>17</maven.compiler.target>
</properties>
```

CONFIGURATION – DEPENDENCIES

- Dépendances
  - Référence vers un artefact spécifique contenu dans un repository
  - On y fait référence avec son *GAV*
  - On peut préciser sa portée (scope)
    - `compile` disponible dans toutes les phases, valeur par défaut
    - `provided` utilisée lors de la compilation, mais pas déployée
    - `runtime` déployée, mais pas nécessaire à la compilation
    - `test` compiler et exécuter des tests

```
<dependency>
  <groupId>org.springframework</groupId>
  <artifactId>spring-context</artifactId>
  <version>${spring.version}</version>
  <scope>compile</scope>
</dependency>
```

Pour rechercher une dépendance, vous pouvez utiliser
http://mvnrepository.com/

CONFIGURATION – DEPENDENCIES

Scope	Compilation	Exécution	Distribution	Déploiement
Compile	X	X	X	X
Provided	X	X		
Runtime		X	X	X
Test				

CONFIGURATION – BUILD

- Processus build

```
<plugin>
  <artifactId>maven-compiler-plugin</artifactId>
  <version>3.5.1</version>
  <configuration>
    <source>17</source>
    <target>17</target>
  </configuration>
</plugin>
```

```
<build>
  <plugins>
    <plugin>
      <!-- ... -->
    </plugin>

    <plugin>
      <!-- ... -->
    </plugin>
  </plugins>
</build>
```

INITIALISATION D'UN PROJET

- **Maven** peut utiliser des archetypes (un pattern – un modèle), constitué
  - D'un descripteur (archetype.xml)
  - Des templates (fichiers et répertoires) qui seront copiés pour le nouveau projet
  - D'un fichier POM
- Il en existe déjà
  - maven-archetype-archetype
  - maven-archetype-quickstart
  - maven-archetype-j2ee-simple
  - maven-archetype-plugin
  - maven-archetype-simple
  - maven-archetype-site
  - maven-archetype-webapp

COMMANDES UTILES

- Compile le projet

```
mvn compile
```

- Exécute les tests unitaires du projet

```
mvn test
```

- Package le projet (**JAR** ou **WAR**)

```
mvn package
```

- Supprime le répertoire *target*

```
mvn clean
```

- Test et compile le projet

```
mvn test compile
```

- Purge les dépendances locales

```
mvn dependency:purge-local-repository
```

- Nettoie et package le projet

```
mvn clean package
```

- Génère la documentation **JAVA**

```
mvn javadoc:javadoc
```

EXERCICE

- Télécharger **Maven**
  - Dézipper l'archive dans C:\Maven (par exemple)
  - Ajouter « C:\Maven\bin » dans votre variable d'environnement Path
  - Vérifier que la variable système **JAVA_HOME** existe et qu'elle pointe sur la **JDK**

```
mvn -version
```

- Créer un nouveau projet **Maven** depuis Eclipse
  - Associer la version de la **JDK** 17

ECLIPE

- Pour un projet **Maven** sous Eclipse, possible de faire un Maven-Update
  - Clique droit > **Maven** > Update Project
  - Force **Maven** à
    - Télécharger les dépendances et à mettre à jour les SNAPSHOTS
    - Nettoyer le projet