Apache Maven

Introduction

Outil pour la gestion et automatisation de production

Écrit en Java

Très utilisé dans l’industrie

Introduction

Facilite le processus de build

Système de build uniforme

Information de qualité du projet

Simplification des migrations

Introduction

Un projet a un cycle de vie ➜ clean compile package test install deploy

Un projet a des dépendances

Un projet a un output

Installation

Télécharger maven

Dézippez

Ajouter $M2_HOME / Modifier $PATH

generate

Il existe de nombreux archetypes standards pour créer un projet standard.

```
mvn generate:archetype
```

A vous de renseigner les valeurs désirées (artifactId, ...)

S’occupe de créer l’arborescence, le pom.xml préconfiguré, d’ajouter des dépendances ...

Convention > Configuration

- `pom.xml` : fichier déclaratif du projet
- `src/main/java` : code source
- `src/main/resources` : fichiers de ressources (images, fichiers annexes etc.)
- `src/main/webapp` : webapp du projet
- `src/test/java` : code source de test
- `src/test/resources` : fichiers de ressources de test
- `target` : fichiers résultat, les binaires (du code et des tests), les packages générés et les résultats des tests

Projet Maven

Le pom.xml contient :

- artifactId, groupId, version (**obligatoire**)
- informations de projet (licence, développeurs, ...)
- dépendances (bibliothèques tierces)

pom.xml

<groupId>	➜	Le groupe du projet
<artifactId>	➜	Le nom de l'artefact
<packaging>	➜	Type de sortie (jar, war ...)
<version>	➜	La version du projet
<url>	➜	L'URL du projet
<properties>	➜	Variables globales (encodage ...)
<dependency>	➜	Bibliothèque
<plugin>	➜	Plugin

Plugins

Plugins standards :

- maven-compiler-plugin : Compilation du projet
- maven-jar-plugin : Création d’un jar configurable
- maven-surefire-plugin : Lancement de tests unitaires
- maven-dependency-plugin : Export des bibliothèques
- maven-resources-plugin : Export des ressources du projet
- maven-war-plugin : Création d’un war configurable
- maven-clean-plugin : Nettoyage du projet

Dépendances

Les dépendances se trouvent dans un repository ➜ http://mvnrepository.com/

Une dépendance a un scope : compile, test, provided ...

Automatiquement téléchargées et mis en cache par maven lors du premier build

Nouvelle version d'une lib ? Il suffit de changer la version dans la pom.xml

Cycle de vie

Lors d’un build, le projet passe par des étapes successives. La réussite de toutes ces étapes définit la réussite d’un build.

clean

compile

test

package

integration-test

install

Utilisation

Tous les IDE facilitent l'utilisation de Maven via des plugins déjà intégrés

En ligne de commande :

```
mvn clean install
```

```
mvn clean install -DskipTests
```

```
mvn -T8 clean package
```

pom.xml

Exemples de pom.xml

- projet Maven ➜ lien
- projet Jongo ➜ lien
- project Hadoop ➜ lien

Conclusion

Maven est un outil très puissant, ayant une forte communauté et universel

Peut s’avérer verbeux dans la déclaration

Difficile à utiliser dès lors que l’on sort du cadre prédéfini