**Structure d’un projet Maven 2 / Le répertoire « SRC » :**

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/filters:** Les filtres de ressources, sous forme de fichier de propriétés, qui peuvent être utilisés pour définir des variables connues uniquement au moment du build.
* **src/main/config:** Les fichiers de configuration
* **src/main/webapp:** Le répertoire d'application web pour les projets WAR.
* **src/test/java:** Les tests unitaires
* **src/test/resources:** Les ressources nécessaires aux tests unitaires, qui ne seront pas déployées
* **src/test/filters:** Les filtres nécessaires aux tests unitaires, qui ne seront pas déployées
* **src/site**: Les fichiers utilisés pour générer le site web du projet Maven

**Les phases les plus utiles du cycle de vie Maven 2 :**

* generate-sources: Génère le code source supplémentaire nécessité par l'application, ce qui est généralement accompli par les plug-ins appropriés.
* 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.
* deploy: Réalisé dans un environnement d'intégration ou de production, copie le produit final dans un entrepôt distant pour être partagé avec d'autres développeurs ou projets.

**Porté des dépendences :**

Dans une application d'entreprise du monde réel, vous n'avez pas nécessairement besoin d'inclure toutes les dépendances dans l'application déployée. Certains des JARs sont nécessaires uniquement pour les tests unitaires, alors que d'autres seront fournis à l'exécution par le serveur d'application. En utilisant la technique de la portée de dépendances, Maven 2 vous permet d'utiliser certain JAR uniquement quand vous en avez besoin et de les exclure du classpath quand vous n'en avez pas besoin.

Maven 2 mets à disposition quatre portés de dépendances :

* compile: Une dépendance de portée compile est disponible dans toutes les phases. C'est la valeur par défaut.
* provided: Une dépendance de portée provided est utilisée pour compiler l'application, mais ne sera pas déployée. Vous utiliserez cette portée quand vous attendez du JDK ou du serveur d'application qu'il vous mette le JAR à disposition. L'API servlet est un bon exemple.
* runtime: Les dépendances de portées runtime ne sont pas nécessaires pour la compilation, uniquement pour l'exécution, comme les drivers JDBC (Java Database Connectivity).
* test: Les dépendances de portées test sont uniquement nécessaires pour compiler et exécuter les tests (par exemple Junit).