Maven

Qu'est-ce que Maven ?

- `Apache Maven` est un outil de gestion et de comprehension de projets logiciels basés sur `Java`.
- Basé sur le modèle Objet de projet(`POM`).
- `Maven` peut gérer la construction, le reporting et la documentation d'un projet à partir d'une information centrale.
- `Maven` fournit un cadre complet de cycle de vie de build pour la gestion de projet.
- L'équipe de développement peut automatiser l'infrastructure de build pour un projet en un rien de temps, car `Maven` utilise une disposition de repertoire standard et un cycle de vie de build par défaut.
- En résumé, `Maven` simplifie et standardise le processus de construction de projet.
- Il gère de manière transparente la compilation, la distribution, la collaboration en équipe et d'autres tâches.
- `Maven` augmente la réutilisation et prend en charge la plupart des tâches liées à la construction.

1) Processus de Construction

- Le `Build`est processus qui couvre toutes les étapes nécessaires pour céer un produit livrable de votre logiciel en préproduction et en production. Dans le monde Java, cela inclut généralement :
  - 1) La génération de la source
  - 2) Compilation des sources
  - 3) Exécuter des tests (tests unitaires, tests d'intégration,etc).
  - 4) Emaballage (en pot, war, ejb-jar, ear).
  - 5) Exécution de contrôle de santé (analyseurs statiques comme `Checkstyle`, `Findbugs`, `PMD`, couverture de test, etc).
  - 6) Générer des rapports
- Un `processus de build` défini une partie essentielle de tout cycle de développement car il permet de combler l'écart entre les environnements de développement, d'intégration, de test de production.
- Un `processus de build` à lui seul accélérera la migration des logiciels d'un environnement à un autre.
- Il supprime également de nombreux problèmes liés à la compilation, au chemin de classe ou aux propriétés qui côutent du temps et de l'argent à de nombreux projets.

2) Qu'est-ce qu'un outil de Construction ?

- Un `outil de build` est outil qui automatise tout ce qui concerne la construction du projet logiciel.
- La construction du projet logiciel comprend généralement une ou plusieurs de ces activités :
  - Générer du code source (si du code généré automatiquement est utilisé dans le projet).
  - Générer de la documentation du code source.
  - Compilation du code source.
  - Conditionnement du code compilé dans les fichiers `JAR ou ZIP`.
  - Installation du code packagé sur un serveur, dans un référentiel ou ailleurs.
- Tout projet logiciel donné peut comporter plus d'activités que celles nécessaires à la création du logiciel fini.
- De telles activités peuvent normalement être connectées à un outil de construction, de sorte que ces activités peuvent également être automatisés.
- L'avantage de l'automatisation de du processus de création est que vous `minimisez le rique que des humains commettent des erreurs lors de la création du logiciel`.
- De plus un outil de création automatisé est généralement plus rapide qu'un humain effectuant les mêmes étapes manuellement.

3) Comprendre le problème Courant Sans Maven

- Nous sommes confrontés à de nombreux problèmes lors du développement du projet. Ils sont discutés ci-dessous :
  - Ajout d'un ensemble de fichiers `JAR` et de dépendances dans chaque projet : Dans le cas des frameworks `Struts, Spring, Hibernate`, nous devons ajouter un ensemble de fichiers `JAR` dans chaque projet. Il doit également être dans toutes les dépendances des `jar`.
  - Créer et maintenir la bonne structure de projet. Nous devons créer la bonne structure de projet dans le `servlets, struts, etc`, sinon elle ne sera pas exécutée.
  - Construire et déployer le projet : Nous devons construire le projet pour qu'il puisse fonctionner.

Compilation Et Exécution

- `La compilation et l'éxécution d'un programme Java` se déroulent en deux étapes.
- Pendant la phase de compilation `Java` compile le code source et génère du `bytecode`.
- Ce `bytecode` intermédiaire est enregistré sous la forme d'un fichier `.class`.

- Dans la deuxième phase, la `machine virtuelle Java (JVM)`, également appelée `interpréteur Java`, prend le `.class` et génère une sortie en exécutant le `bytecode`.

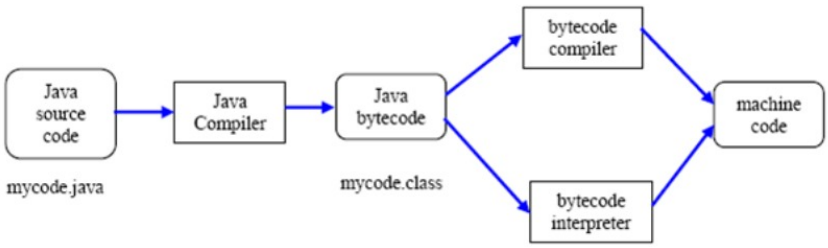

Figure 1: Alt Text

Divers Outils de Construction Disponibles :

- Pour `Java` :
  - `Ant`
  - `Maven`
  - `Gradle`
- Pour le framework `.NET` :
  - `NAnt`
- Pour `C#` :
  - `MsBuild`

Comparaisons Entre Quelques Outils De Construction Basé Sur Java :

Ant Contre Maven Conte Gradle

Ant :

- `Apache Ant` est une bibliothèque `Java` et un outil de ligne de commande dont la mission est de piloter les processus décrits dans les fichiers de construction en tant que cibles et points d'extension les uns des autres.
- La principale utilisation connu d' `Ant` est la création d'applications `Java`.
- `Ant` fourni un certain nombre de tâches intégrées permettant de compiler, d'assembler, de tester et d'exécuter des applications `Java`.
- `Ant` peut également être utilisé efficacement pour créer des applications `non Java`, par exemple des applications `C ou C++`.
- Plus généralement, `Ant` peut être utilisé pour piloter tout type de processus pouvant être décrit en termes de cibles et de tâches.
- Sa courbe d'apprentissage est très faible, permettant ainsi à quiconque de commencer à l'utiliser sans aucune préparation particulière.
- Il est basé sur l'idée de programmation procédurale. Après sa version initiale, il a été amélioré avec la possibilité d'accepter des `plug-ins`.

- L'inconvénient majeur était XML comme format pour écrire des scripts de construction XML, étant de nature hiérarchique, n'est pas adapté à l'approche de programmation procédurale utilisé par Ant.
- Un autre problème avec Ant est que son XML a tendance à devenir ingérable lorsqu'il est utilisé avec tous les projets, sauf de très petite taille.

Maven

- Apache Maven est un outil de gestion et de compréhension de projets logiciels.
- Basé sur le concept de modèle objet de projet (POM), Maven peut gérer la construction, le reporting et la documentation d'un projet à partir d'une information centrale.
- Son objectif était d'améliorer certains des problèmes rencontrés par les développeurs lors de l'utilisation d'Ant.
- Maven continue d'utiliser XML comme format pour écrire les spécifications de construction. Mais la structure est diamétralement différente.
- Alors qu'Ant exige que le programmeur écrive toutes les commandes qui conduisent à l'exécution réussie d'une tâche.
- Maven s'appuie sur des conventions et fournit les cibles (objectifs) disponibles qui peuvent être invoquées.
- Comme ajout supplémentaire, et probablement le plus important, Maven a introduit la possibilité de télécharger des dépendances sur le réseau (plus tard adoptée par Ant via Ivy).
- Alors qu'Ant exige que le programmeur écrive toutes les commandes qui conduisent à l'exécution réussie d'une tâche.
- Maven s'appuie sur des conventions et fournit les cibles (objectifs) disponibles qui peuvent être invoquées.
- Maven s'appuie sur des conventions et fournit les cibles (objectifs) disponibles qui peuvent être invoquées.
- Comme ajout supplémentaire, et probablement le plus important, Maven a introduit la possibilité de télécharger des dépendances sur le réseau (plus tard adopté par Ant via Ivy).
- Le principal avantage de Maven est son cycle de vie. Tant que le projet est basé sur certaines normes, avec Maven, on peut parcourir tout le cycle de vie avec une vie relative facilité.
- Cela a un coût en termes de flexibilité.
- Aujourd'hui, l'intérêt pour les DSL (Domain Specific Languages) ne cesse de croître. L'idée est de disposer de langages conçus pour résoudre des problèmes appartenant à un domaine spécifique.
- Dans le cas des builds, l'un des résultats de l'application de DSL est Gradle et, par exemple, gradle est utilisé dans Android pour la construction et l'empaquetage.

Gradle

- `Gradle` vise à aider les organisations à fournir de meilleurs logiciels, plus rapidement. Des constructions plus rapides sont l'un des moyens les plus directs d'y parvenir; `Gradle` combine de bonnes parties des deux outils et s'appuie sur eux avec le `DSL` et d'autres améliorations.
- Il possède la puissance et la flexibilité d'`Ant` avec le cycle de vie et la facilité d'utilisation de `Maven`.
- Par exemple, `Google` a adopté `Gradle` comme outil de création par défaut pour le système d'exploitation d'Android.
- `Gradle` n'utilise pas `XML`. Au lieu de cela, il disposait de son propre `DSL` basé sur `Groovy (l'un des langages JVM)`.
- En conséquence, les scripts de `build Gradle` ont tendance à être beaucoup plus courts et plus clairs que ceux écrits pour `Ant` ou `Maven`.
- La quantité de code passe-partout est beaucoup plus petite avec `Gradle` puisque son `DSL` est conçu pour résoudre un problème spécifique :
  - déplacer le logiciel tout au long de son cycle de vie, de la compilation à l'analyse statique et aux tests jusqu'à l'empaquetage et au déploiement.
- `Maven` simplifie et apporte une solution aux problèmes mentionnés ci-dessus. Il effectue principalement les tâches suivantes :
  - Il facilite la construction d'un projet.
  - Il fournit un processus de construction uniforme (le projet maven peut être partagé par tous les projets maven)
  - Il fournit des informations sur le projet (document de journal, sources de références croisés, liste de diffision, liste de dépendances, test unitaire rapports, etc.)
  - Il est facile de migrer pour les nouvelles fonctionnalités de `Maven`.

Utilisation d'Apache Maven

- 1) Utilisation comme outil de construction.
- 2) Utiliser comme pour gérer la structure du projet
- 3) Construction, publication et déploiement
- 4) Documentation
- 5) Rapports
- 6) Versions
- 7) Distributions

Configuration et installation pour Maven

Site Web Maven :

- Le site `Maven` se trouve ici :
  - http://maven.apache.org

- Depuis ce site, vous pouvez télécharger la dernière version `Maven` et suivre le projet en général.
- Vous pouvez télécharger et installer `maven` sur les plateformes `Windows, Linux et MAC OS`.
- **Remarque :** `Maven` est un outil basé sur `Java`, la toute première condition est donc d'avoir installé `JDK` sur votre machine.

Téléchargez les archives Maven

- Maven archives

OS	Archive name
Windows	apache-maven-3.5.0-bin.zip
Linux	apache-maven-3.5.0-bin.tar.gz **or sudo apt-get install maven**
Mac	apache-maven-3.5.0-bin.tar.gz

Figure 2: Alt Text

Extrayez l'archive Maven

- Extrayez l'archive dans le repertoire dans lequel vous souhaitez installer `Maven 3.5.0`.
- Le sous-répertoire `apache-maven-3.5.0` sera créé à partir de l'archive.

OS	Location (can be different based on your installation)
Windows	C:\Program Files\Apache Software Foundation\apache-maven-3.5.0
Linux	/usr/local/apache-maven
Mac	/usr/local/apache-maven

Figure 3: Alt Text

Définir les variables d'environnement Maven

- Ajoutez `M2_HOME, MAVEN_OPTS` aux variables d'environnements.

Comment fonctionne Maven ?& Concepts de base

- “

Set Maven Environment Variables

Add M2_HOME, M2, MAVEN_OPTS to environment variables.

OS	Output
Windows	Set the environment variables using system properties. M2_HOME=C:\Program Files\Apache Software Foundation\apache-maven-3.5.0 M2=%M2_HOME%\bin MAVEN_OPTS=-Xms256m -Xmx512m
Linux	Open command terminal and set environment variables. export M2_HOME=/usr/local/apache-maven/apache-maven-3.5.0 export M2=$M2_HOME/bin export MAVEN_OPTS='-Xms256m -Xmx512m
Mac	Open command terminal and set environment variables. export M2_HOME=/usr/local/apache-maven/apache-maven-3.5.0 export M2=$M2_HOME/bin export MAVEN_OPTS=-Xms256m -Xmx512m

Add Maven Bin Directory Location To System Path

OS	Output
Windows	Append the string ;%M2% to the end of the system variable, Path.
Linux	export PATH=$M2:$PATH
Mac	export PATH=$M2:$PATH

Verify Maven Installation

Now open console, execute the following mvn command.

OS	Task	Command
Windows	Open Command Console	c:\> mvn –version
Linux	Open Command Terminal	$ mvn –version
Mac	Open Terminal	machine:~ joseph$ mvn --version

Figure 4: Alt Text