Maven Lite

See the english documentation

- Version : 2.0.0
- Auteur : Floris Robart

Table des matières

Compatibilité

- Windows 10 (Non testé)
- Windows 11 (Non testé)
- Linux possèdant `bash` (Testé sur Ubuntu 23.10 Mantic Minotaur)
- MacOS Mojave 10.14.6 et inférieur (Non testé)
- MacOS Catalina 10.15 et supérieur à condition d'avoir installé `bash` (Non testé)

Description

Maven Lite est un système de gestion de projet java qui permet de compiler et lancer des projets java très simplement et rapidement.

Le but est de pouvoir compiler et/ou lancer un projet java en une seule courte commande, comme maven mais en plus simple.

Maven Lite est plus simple que Maven car il ne gère pas les dépendances hors local, les plugins, les phases de build, les déployements automatique, etc. Il est donc adapté pour les projets simples qui ne nécessitent pas de gérer des dépendances externes, cependant Maven Lite gère les dépendance local sous forme de fichier jar. Pour des projets plus complexes, il est conseillé d'utiliser Maven.

Maven lite permet de compiler le projet dans un dossier target, gèrer les dépendances locales (fichier jar) et éxecuter le projet.

L'intéret principale de maven lite dans la gestion des dépendances est de fournir un dossier qui contient toutes les dépendances, le script ajoutera automatiquement toutes les dépendances dans le classpath.

Il est possible que cette deuxième version de Maven Lite contienne des bugs. Si vous en trouvez, n'hésitez pas à les signaler.

Un fichier de configuration est disponible pour configurer les options de Maven Lite. Il est possible de se passer de fichier configuration et de mettre les options dans la ligne de commande mais c'est moins pratique.

L'ordre des options n'a pas d'importance.

Chaque option peut être utilisée plusieurs fois.

Chaque argument doit être précédé de son option, vous ne pouvez pas mettre d'argument sans option ni mettre toutes les options au début puis tous les arguments à la fin. Par exemple, `mvnl --source src/main/java --target target` est valide mais `mvnl --source --target src/main/java target` ne l'est pas.

Il est recommandé de laisser les options par défaut parce que Maven Lite est basé sur les conventions simplifier de Maven qui sont des conventions de nommage et d'arborescence connues de beaucoup de développeurs java.

Prérequis

- Disposer des droits administrateurs si vous voulez utiliser les installations automatiques.
- Avoir installé `bash`
- Avoir installé `java` 17 ou supérieur

Si vous ne disposez pas des droits administrateurs, vous pouvez installer maven lite manuellement en suivant les instructions de la partie 'Installation manuelle'.

Utilisation de Maven Lite

- placer vous dans le dossier de votre projet java et exécuter la commande suivante :

```
mvnl [options] [arguments]
```

Ligne de commande

La ligne de commande est la méthode classique pour utiliser Maven Lite bien qu'elle soit moins pratique que l'utilisation d'un fichier de configuration.

Vous pouvez mettre autant d'options que vous le souhaitez dans n'importe quel ordre.

Possibilités de la ligne de commande

- Elle ne prend pas en charge les commentaires.
- Il est possible de mettre des options en utilisant le format `--option` ou `-o`. Vous pouvez mettre n'inporte quel option cité dans la liste des options.
- Il est possible de passer des arguments avec des espaces à la class main de votre projet en utilisant des cotes simples plus des guillemets, par exemple `mvnl --args '"mon argument"'`.
- Il est possible de mettre plusieurs options sur la même ligne en les séparant par un espace, par exemple `mvnl --quiet --verbose` ou `mvnl -q -v`.
- Il est possible d'échapper les caractères spéciaux avec un antislash `\`, par exemple `--args '"exemp\"le'"`. Il est fortement recommandé d'utiliser le système côte simple + guillemets pour passer des arguments à la class main de votre projet avec l'option `-args` et `--arguments` pour éviter tous problèmes d'échappement.

    - Les caractères spéciaux dans la ligne de commande sont : `\`, `"`
        - `--args '"exemp\"le"'` devient `exemp"le`
        - `--args '"-exemple"'` devient `-exemple`
        - `--args '"--exemple"'` devient `--exemple`
        - `--args '"exemp\\le"'` devient `exemp\le`
        - `--args '"exemp\\\"le"'` devient `exemp\"le`.

Fichier de configuration

L'utilisation d'un fichier de configuration se fait avec l'option `--file` ou `-f`.

Le fichier de configuration est un fichier unique à chaque projet qui permet de configurer les options que Maven Lite devra utiliser.

Le nom par défaut du fichier de configuration est `LPOM.conf` et doit être à la racine du projet. Il est possible de le renommer mais dans ce cas il faudra préciser son nom l'ors de l'utilisation de l'option, par exemple `mvnl -f monFichier`.

si vous voulez ajouté votre CLASSPATH système au CLASSPATH de Maven Lite dans le fichier de configuration, il faut utiliser le terme `CLASSPATH` en majuscule.

Possibilités du fichier de configuration

- Il est possible de mettre des commentaires en utilisant le caractère `#` au début de la ligne.
- Il est possible de mettre des options en utilisant le même format que dans la ligne de commande, par exemple `--option` ou `-o`. Vous pouvez mettre n'inporte quel option cité dans la liste des options.
- Il est possible de passé des arguments avec des espaces en utilisant des guillemets, par exemple `mvnl -args "mon argument"`.
- Il est possible de mettre plusieurs options sur la même ligne en les séparant par un espace, par exemple `mvnl --quiet --verbose` ou `mvnl -q -v`.
- Il est possible d'échapper les caractères spéciaux avec un antislash `\`, par exemple `--args exemp\"le`.
    - Les caractères spéciaux dans le fichier de configuration sont : `\`, `"` et `-` uniquement s'il sont au début de l'argumentet que ce dernier n'a pas de guillement. Par exemple :
        - `--args exemp\"le` devient `exemp"le`
        - `--args \-exemple` devient `-exemple`
        - `--args "--exemple"` devient `--exemple`
        - `--args \--exemple` devient `--exemple`
        - `--args exemp\\le` devient `exemp\le`
        - `--args exemp\\\"le` devient `exemp\"le`.
- Une option et ses arguments peuvent être sur plusieurs lignes, par exemple

```
--args
"mon argument

    sur plusieurs lignes"
```

- Un exemple type de fichier de configuration est le suivant :

```
# Source du projet
--source src/main/java

# Dossier de sortie des fichiers compilés
--target target

# Liste des libraries à ajouter au classpath
--libraries src/main/resources/lib
```

```
# Affiche les commandes éxécutées
--verbose

# Ajoute le classpath système au classpath de Maven Lite
--classpath CLASSPATH
```

create

L'utilisation d'un fichier de configuration se fait avec l'option `--create` ou `-cr`.

Il est impossible de créer un projet avec un espace dans le nom. Il est impossible de créer un projet avec un nom qui est déjà utilisé par un fichier ou un dossier.

Vous pouvez créer un projet dans le dossier courant grâce à la commande suivante :

```
mvnl --create ./
```

mvc

L'utilisation de cette option se fait avec l'option `--model-view-controller` ou `-mvc`.

Permet de spécifier à l'option create de créer l'arborescence d'un projet MVC.

compilation

L'utilisation de cette option se fait avec l'option `--compilation` ou `-c` ou `--compile-launch` ou `-cl` ou `--launch-compile` ou `-lc`.

Cette option permet de compiler le projet.

launch

L'utilisation de cette option se fait avec l'option `--launch` ou `-l` ou `--compile-launch` ou `-cl` ou `--launch-compile` ou `-lc`.

Cette option permet de lancer le projet.

quiet

L'utilisation de cette option se fait avec l'option `--quiet` ou `-q`.

Cette option permet de supprimer l'affichage de java dans le terminal lors de l'exécution du projet.

verbose

L'utilisation de cette option se fait avec l'option `--verbose` ou `-v`.

Cette option permet d'afficher les commandes Java exécutées par Maven Lite.

exclude

L'utilisation de cette option se fait avec l'option `--exclude` ou `-ex`.

Cette option permet d'exclure des fichiers java et des dossiers de la compilation.

compile jar

L'utilisation de cette option se fait avec l'option `--compile-jar` ou `-cj`.

Cette option permet de créer un fichier jar exécutable de votre projet. Le convention de nommage du fichier jar est la suivante : `<nom>-<M>.<m>.<b>.jar`.

- `<nom>` : Nom du projet.
- `<M>` : Numéro de la version majeur. Il commence à 1 et est incrémenté à chaque nouvelle version non compatible avec la précédente et avec de grosses modifications.
- `<m>` : Numéro de la version mineur. Il commence à 0 et est incrémenté à chaque nouvelle version compatible avec la précédente et avec de petites modifications et retourne à 0 à chaque changement de version majeur.
- `<b>` : Numéro de la version de build. Il commence à 0 et est incrémenté à chaque correction de bug et retourne à 0 à chaque changement de version mineur ou majeur.

launch jar

L'utilisation de cette option se fait avec l'option `--launch-jar` ou `-lj`.

Cette option permet de lancer le fichier jar exécutable de votre projet.

source

L'utilisation de cette option se fait avec l'option `--source` ou `-s`.

Cette option permet de spécifier le dossier contenant les fichiers java à compiler.

target

L'utilisation de cette option se fait avec l'option `--target` ou `-t`.

Cette option permet de spécifier le dossier de sortie des fichiers compilés, le dossier d'entrée des fichiers à lancer, le dossier de sortie des fichiers jar ainsi que le dossier d'entrée des fichiers .class à mettre dans le jar.

Vous n'avez pas besoin de créer le dossier de sortie des fichiers compilés, Maven Lite le fera pour vous.

Vous n'avez pas besoin d'ajouter le dossier de sortie des fichiers compilés au classpath, Maven Lite le fera pour vous.

classpath

L'utilisation de cette option se fait avec l'option `--classpath` ou `-cp`.

Cette option permet de spécifier le classpath à utiliser lors de la compilation et du lancement.

libraries

L'utilisation de cette option se fait avec l'option `--libraries` ou `-lib`.

Cette option permet de spécifier le dossier contenant les fichiers jar utiliser par le programme. Tout les fichiers jar seront ajoutés au classpath lors de la compilation et du lancement.

Vous pouvez créer des sous-dossiers dans le dossier des librairies pour mieux organiser vos fichiers jar, Maven Lite les prendra en compte.

arguments

L'utilisation de cette option se fait avec l'option `--arguments` ou `-args`.

Cette option permet de spécifier tous les arguments à passer à la classe principale.

Vous pouvez passer des arguments avec des espaces en utilisant des côtes simples plus des guillemets, par exemple `mvnl -args '"mon argument"'`.

Ces arguments seront passés à la classe principale dans l'ordre où ils sont écrits.

Exemple avec un fichier de configuration :

- Fichier de configuration

```
--args "arg2"
```

- Ligne de commande à lancé

```
mvnl -args '"arg1"' -f -args '"arg3"' '"arg4"'
```

- Arguments passés à la classe principale

```
arg1 arg2 arg3 arg4
```

main

L'utilisation de cette option se fait avec l'option `--main` ou `-m`.

Cette option permet de spécifier la classe principale à lancer avec le package sous la forme `package.MainClass`. Cette option est utile uniquement si vous avez plusieurs classes main dans votre projet, si se n'est pas le cas, Maven Lite trouvera et utilisera automatiquement la classe main du projet.

encoding

L'utilisation de cette option se fait avec l'option `--encoding` ou `-e`.

Cette option permet de spécifier l'encodage des fichiers java à compiler.

export

L'utilisation de cette option se fait avec l'option `--export` ou `-exp`.

Cette option permet de créer un fichier jar exécutable permettant de lancer le projet sans avoir installer MavenLite.

maven

L'utilisation de cette option se fait avec l'option `--maven` ou `-mvn`.

Cette option permet de convertir le projet en projet maven en créant un fichier pom.xml et en déplaçant les fichiers si nécessaire.

version

L'utilisation de cette option se fait avec l'option `--version` ou `-V`.

Cette option permet d'afficher la version de Maven Lite ainsi que la localisation du fichier principale de Maven Lite, la version de java, le type de build, le runtime java utilisé, la langue utilisée par le système, la plateforme d'encodage utilisée par Maven Lite, le nom du système d'exploitation, la version du noyaux du système d'exploitation et l'architecture du système.

- Exemple

```
Maven Lite 2.0.0
Maven Lite home : /etc/maven-lite/
Java version : 17.0.9, vendor : Private Build, runtime : OpenJDK Runtime Environment
Default locale : fr_FR, platform encoding : UTF-8
OS name : "Linux", version : "6.5.0-14-generic", architecture : "amd64"
```

help

L'utilisation de cette option se fait avec l'option `--help` ou `-h`.

Cette option permet d'afficher la liste des options ainsi que leur description, le nombre d'arguments qu'elles prennent et leur valeur par défaut si elle en ont une.

clear

L'utilisation de cette option se fait avec l'option `--clear` ou `-cle`.

Cette option permet de supprimer les fichiers dans le dossier de sortie des fichiers compilés.

Exemple, fonctionnaliés et limites