Construire son JDK en 10 étapes

Speaker : José Paumard
Format : Tools in action
Date : 17 avril 2019

Dans ce Tools in Action, José nous explique étape par étape comment recompiler un JDK à la main. Lors de la préparation du talk, il s'est rendu compte qu'il n'y a pas exactement 10 étapes. Ce n'est pas le nombre d'étapes qui compte, mais le temps qui incombe : télécharger du code pendant 10mn et le builder pendant 20mn serait trop long.

Pourquoi construire son propre JDK ?

- Le côté fun : challenge technique
- La totalité du JDK est Open Source. On peut construire le JDK qui n'existe pas encore. Choisir la branche qui nous intéresse. Construire un JDK qui n'existera peut-être jamais. Tester certaines fonctionnalités.
- Est-ce réellement un JDK ? Non. Le TCK n'est pas distribué gratuitement (vendu par Oracle). Qui plus est, le TCK n'existe pas pour les versions du JDK qui n'existent pas encore.

Le repository https://github.com/forax/java-next de Rémo Forax contient un .travis.yml buildant des JDK pour Linux et MacOS.
Le code source d'OpenJDK est hébergé sur un repository Mercurial. Sur GitHub, il y'a bien un miroir d'OpenJDK. Et à termes, le repository Mercurial pourrait être migré sur GitHub (discussion en cours). Cela dit, les branches (ex : Valhalla) ne sont pas présentes sur le miroir. Il est donc nécessaire de récupérer le repository Mercurial.

Le lien de départ : repository mercurial https://hg.openjdk.java.net
65 projets référencés :

- Amber, valhalla, loom, panama
- Graal
- Duke : images de la mascotte Duke de Java

Le LondonJug (LJC) a lancé Adopt OpenJDK : créer des distributions du JDK gratuites et libres de droit : https://adoptopenjdk.net

Adopt OpenJDK supporte 10 plateformes, 5 versions de Java et 2 JVM (HotSpot et J9) , ce qui fait un total de 100 versions disponibles.

Prérequis :

- Disposer d'un Linux (plus facile que sur Windows) : Distribution Ubuntu Desktop 18.04. Avec 50 Go de disque minimum.
- Installer Mercurial : sudo apt-get install mercurial

Ensuite il faut choisir le JDK à construire. Par exemple, le repository valhalla. Pour savoir ce qu'il y'a précisément dans la branche, il faut aller checker la mailing-list. A ce stade, il n'existe pas encore de documentation ou de tutoriel.
Les changeset Mercurial sont l'équivalent des commits Git.

Commande : hg clone http://hp.openjdk.java.net/loom/loom/
Attendre 20mn de téléchargement (le code source pèse 2,5 Go). Les serveurs Mercurial de l'OpenJDK ne sont pas très rapides.

Se synchroniser avec le repo :
Commande : hg pull (git fetch)

Ensuite, il faut se mettre sur la bonne branche (par défaut, on est sur la branche default).
Commande : hg amber-demo-II

Le fichier configure nécessite d'installer autoconf avec un apt-get et d'installer make. Plus simple : installer build-essential (117 Mo) : installe make, autoconf, le compilateur GCC …

Il y'a ensuite toutes les libx à installer : libx11-dev, …
Il faut également installer libcups2-dev pour les imprimantes, libasound2-dev pour le son.

La commande : ./configure peut être lancée.
Commande suivante : make images
Attendre 20 à 40 minutes de compilation. Le ventilateur s'emballe.
Le répertoire build/ est créé.
Et voilà : on peut utiliser le JDK ☺

La classe java.lang.AbstractRecord est visible dans le projet Amber.
La classe java.lang.Continuation est visible dans le projet Loom.

Dans Eclipse, le code ne va pas fonctionner.
En live, José utilise javac pour compiler la ligne de code *public record Point(int x, int y) ;*

En conclusion :

- Très long, beaucoup d'attente,
  L'IDE ne supporte pas les nouveaux mots clés (ex : record)
- Permet de suivre en temps réel ce qui est fait dans le JDK et de participer aux discussions des mailings list.