Haute école d'ingénierie et d'architecture Fribourg
Hochschule für Technik und Architektur Freiburg

Prolog avec Truffle et GraalVM
Manuel d'installation et d'utilisation

Projet de semestre 5
I-3
Automne 2019/2020

Martin Spoto
martin.spoto@edu.hefr.ch

Responsable :
Frédéric Bapst
Frederic.Bapst@hefr.ch

30 janvier 2020

Table des matières

1 Introduction

Ce document décrit l'installation et l'utilisation du projet "Prolog avec Truffle et GraalVM", pour l'environnement Linux. Le projet fonctionne également sous Windows, mais les étapes d'installation ne sont pas détaillées.
Ce manuel assume que vous disposez localement d'une copie du projet, et que vous vous trouvez dans le répertoire "code" de celui-ci.

2 Installation des dépendances

Cette section décrit l'installation des différents pré-requis au fonctionnement du projet.

2.1 GraalVM

Le projet a été développé avec GraalVM Community Edition 19.2.0.1.
On peut télécharger cette version ici : https://github.com/oracle/graal/releases/tag/vm-19.2.0.1. Prenez la version qui correspond à votre système d'exploitation, et décompressez l'archive, par exemple dans `/usr/lib/graalvm`. Prenez note du chemin d'accès de l'archive décompressée.

2.2 Maven

Maven est utilisé comme gestionnaire de version. La version 3.6.3 a été utilisée durant le projet, mais les versions proches et futures devraient également fonctionner.
On peut télécharger Maven ici : https://maven.apache.org/download.cgi. Comme pour GraalVM, décompressez l'archive, par exemple dans `/usr/lib/maven`.

3 Préparation de l'environnement

Pour préparer les variable d'environnement nécessaires, un script `envsetup` est fourni. Ce script assume que les chemins de GraalVM et de Maven sont ceux proposés plus haut. Si ce n'est pas le cas, éditez le script avec vos chemins d'installation avant le lancement.
Ce script définit GraalVM comme environnement Java en réglant la variable JAVA_HOME. Il ajoute également les répertoires bin de GraalVM et de Maven au PATH. Les changements de ce script sont temporaires. En tapant `exit`, on peut à tout moment sortir de l'environnement (pratique si vous avez plusieurs installations de Java par exemple).
Le script définit également quelques alias qui sont surtout utiles au développement. Ces alias sont consultables dans le fichier `envaliases`. Toujours pour le développement, le script récupère une copie de la librairie ANTLR utilisée pour générer le parser/lexer (vous aurez besoin pour cela de la commande `wget`). (D'ailleurs, si un jour vous souhaitez faire exactement ça, un script `generate_parser` est également fourni.)
Si tout s'est bien passé, la commande `mvn --version` devrait s'effectuer avec succès, et la version Java devrait être celle de GraalVM.

4 Compilation du projet

Si l'environnement est correctement configuré, alors la compilation s'effectue simplement avec la commande `mvn package`. Cette opération produit un fichier JAR dans launcher/target et dans language/target.

5 Lancement du projet

Si la compilation du projet s'est passée sans encombres, alors on peut lancer le projet à l'aide du script fourni `prolograal`, en lui donnant comme argument un fichier Prolog, par exemple : `./prolograal language/tests/02_variables.pl`, et voilà ! Le projet tourne.

6 Notes pour Windows

Pour Windows, à part que les scripts ne sont pas utilisables, l'installation et l'utilisation se déroulent exactement de la même manière. Il faut donc simplement configurer la variable JAVA_HOME et le PATH correctement.