UNIVERSITÉ DE ROUEN

Compilateur LLVM

Langage jouet Kawa

Manuel d'utilisation 0.1

29 juin 2015

Auteur(s): Pierre-Luc BLOT

Version	Date	Changelog
0.1	19/04/2015	Version initiale.

Table des matières

1 Description du compilateur kawac **2**
1.1 Installation . . . 2
1.2 Désinstallation . . . 2
1.3 compilation . . . 2
1.4 afficher l'aide du compilateur . . . 3
1.5 afficher la version du compilateur . . . 3
1.6 afficher la sortie en couleur . . . 3
1.7 définir le nom de l'exécutable . . . 3
1.8 mode autorun . . . 3

2 Dépendances de kawac **3**
2.1 Installation des dépendances . . . 3

3 Fonctionnement de la chaine de compilation **4**

1 Description du compilateur kawac

L'outil **kawac** lit des définitions de classes et des interfaces écrites dans le langage de programmation Kawa et les compiles en executable ELF. Il est aussi possible de les compiler en bytecode.

1.1 Installation

Situez-vous dans le répertoire src/KAWACompiler puis lancer la commande :

```
:$ sudo make install
```

Cette action va :
— compiler le frontend **kawap**
— créer le répertoire kawa dans **/usr/bin**
— copier **kawac** et **kawap** dans **/usr/bin/kawa**
— créer deux liens symboliques **kawac** et **kawap** dans **/usr/bin/** qui pointent vers les fichiers **kawac** et **kawap**, respectivement, du répertoire **/usr/bin/kawa**

1.2 Désinstallation

Situez-vous dans le répertoire src/KAWACompiler puis lancer la commande :

```
:$ sudo make uninstall
```

Cette action va :
— supprimer le répertoire **/usr/bin/kawa** et son contenu
— supprimer les deux liens symboliques **kawac** et **kawap** dans **/usr/bin/**

1.3 compilation

Il y a deux manières de passer des noms de fichiers sources à **kawac**.
— L'utilisateur introduit un ensemble de modules (classe/interface) kawa afin de les pré/compiler et de générer une application sous forme d'un seul exécutable.

Afin de permettre la compilation dans ce mode monolithique il faut :
— Indiquer l'emplacement des fichiers sources de l'application.
— Introduire le commutateur -m en premier paramètre dans la ligne de commande permettant la compilation.
— L'absence des deux commutateurs (**-h**) et (**-v**) en premier paramètre.
Afin de pouvoir lancer la compilation dans ce mode l'utilisateur doit :
— Indiquez au moins un module (une classe avec une méthode main qui constitue le point d'entrée d'une application).
— Tapez la ligne de commande suivante afin de compiler les sources kawa :

```
:$ kawac -m filessources
```

Cela déclanche la production d'un unique fichier exécutable sous le format d'ELF qui ne dépend d'aucune bibliothèque kawa ainsi que le nom du fichier exécutable correspond au nom du module contenant la méthode main.

La compilation peut être interrompue pour divers raisons, on cite :
— Le fichier de sortie n'a pu être créé.
— Le code source d'un des modules de l'application comporte une erreur syntaxique, l'erreur rencontrée pendant cette étape d'analyse est renvoyée sur la sortie standard du compilateur.
— Le code source d'un des modules (classe/interface) dont dépend le programme est introuvable. Un message renvoyé par le compilateur indique le nom du (ou des) module(s) manquants sur la sortie standard.
— Le code source d'un des modules de l'application comporte une erreur sémantique, exemple : l'incompatibilité de type statique lors d'une opération d'affectation.
— Aucun des modules indiqués ne comporte le point d'entrée (main).

— Plusieurs méthodes main ont été trouvées parmi les classes indiquées dans le source, le compilateur renvoie la liste des points d'entrée trouvés sur la sortie standard.
— Notez qu'il est important que vos définitions de **package** soient bien situées dans des fichiers se trouvant dans des répertoires ayant le nom de ce **paquetage**.

1.4 afficher l'aide du compilateur

Il est possible d'afficher l'aide du compilateur avec le commutateur **-h** ou –**help**. Cela permet d'obtenir rapidement une aide quant à la manière d'utiliser **kawac** et ce, depuis le terminal via une ligne de commande.

```
:$ kawac --help
```

1.5 afficher la version du compilateur

Il est possible de savoir la version du compilateur avec le commutateur **-v**
— L'ordre de priorité entre le commutateur d'aide (**-h**) et le commutateur de version (**-v**) est définit par la première occurrence de l'un des deux commutateurs i-e si nous avons un **-v** avant un **-h**, kawac annule tout le reste et affiche la version

```
:$ kawac --version
```

1.6 afficher la sortie en couleur

L'affichage de la sortie en couleur est possible via le commutateur **-c**
— Afin d'afficher la sortie en couleur le commutateur devra être placé avant les commutateurs qui produise du texte en sortie. Comme **-v**, **-h**, **-m**

1.7 définir le nom de l'exécutable

Afin de définir le nom du programme compilé, il sera nécessaire d'utiliser le commutateur **-o** suivi du nom de l'exécutable à produire.
— Il n'y pas de réelle priorité avec ce commutateur. La seule existante est de l'appeler avant la compilation (**-m**).

1.8 mode autorun

Une fois le programme compilé il est possible de demander au compilateur de le lancer automatiquement. Cette action est défini via le commutateur **-r**
— Il n'y pas de réelle priorité avec ce commutateur. La seule existante est de l'appeler avant la compilation (**-m**).

2 Dépendances de kawac

Le compilateur kawa **kawac** utilise la technologie **LLVM** afin de produire du bytecode. **kawac** offre un frontend **LLVM** pour le langage Kawa ainsi qu'une partie du backend ce qui produit du bytecode. Néanmoins, pour produire un exécutable à partir du bytecode **kawac** a besoin d'**llc** dans sa version 3.4/3.5 et de **gcc** dans sa version 4.8.2 ou plus.

2.1 Installation des dépendances

Afin de compiler les sources de kawac, il est nécessaire d'installer les paquets suivants :
— bison
— flex

— llvm-3.5
— llvm-3.5-tools
— clang-3.5
— zlib1g-dev
— libedit-dev

```
:$ sudo apt-get install bison flex llvm-3.5 llvm-3.5-tools clang-3.5 zlib1g-dev libedit-dev
```

Il est possible que la commande llc ne soit pas installée automatiquement. Si tel est le cas il sera nécessaire de le faire manuellement :

```
:$ sudo ln -s /usr/bin/llc-3.5 /usr/bin/llc
```

Cela dit, kawac détecte si llc-3.5 est bien installé si llc ne l'est pas.

3 Fonctionnement de la chaine de compilation

Le frontend de kawac se nomme kawap (pour kawa bytecode production) qui utilise la technologie **LLVM** afin de produit du bytecode. Le compilateur en lui même se nomme **kawac** et encapsule le frontend ainsi que le backend. Voici comment se déroule la compilation d'un exécutable avec la commande :

```
:$ kawac -m filessources
```

1. kawap -m filessources → produit du bytecode dans un fichier .ll
2. llc → qui produit du code assembleur à partir du bytecode
3. gcc → qui crée le fichier exécutable à partir de l'assembleur

L'exécutable **kawac** est un script bash qui explicite cette chaîne de compilation et vérifie les dépedances automatiquement.