

TP6 : Analyse dynamique avec Spoon

Travail réalisé :

Programming 1 : Utilisez Spoon afin de compter tous les appels de méthode d'un programme donné.

J'ai créé un processeur(`CountProcessor`) qui dans chaque début de méthode rajoute du code. Ce code appelle une méthode d'une classe(`CounterMethod`) que j'ai créé. Cette classe s'occupe de compter le nombre de fois que chaque méthode est appelé. Le code ajouter par le processeur à chaque début de méthode permet d'indiquer à la classe `CounterMethod` qu'il faut incrémenter le compteur de la méthode qu'on est en train de parcourir.

Résultat : Après avoir lancé la commande

`mvn package,`

puis

`java -cp ./target/tpSpoon-1.0-SNAPSHOT-jar-with-dependencies.jar vv.spoon.MainCountMethod src/main/resources/example src/main/resources/example-instrumented`

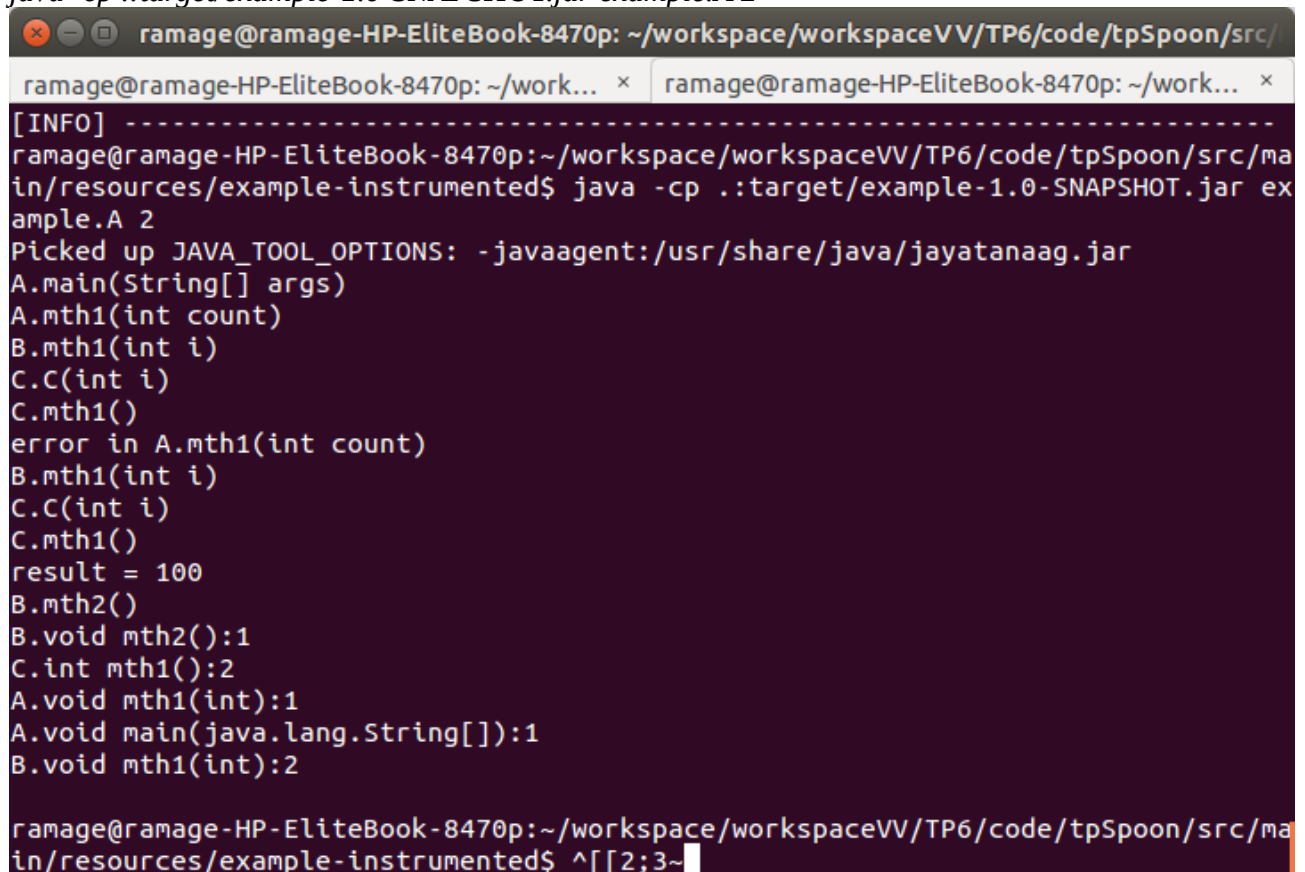
pour exécuter le processeur `CounterMethod` sur le dossier `src/main/resources/example`.

Lorsque l'on va dans le dossier `src/main/resources/example-instrumented` et qu'on lance les commandes :

`mvn package,`

puis

`java -cp ./target/example-1.0-SNAPSHOT.jar example.A 2`



```

ramage@ramage-HP-EliteBook-8470p: ~/workspace/workspaceVV/TP6/code/tpSpoon/src/
ramage@ramage-HP-EliteBook-8470p: ~/work... x ramage@ramage-HP-EliteBook-8470p: ~/work... x
[INFO] -----
ramage@ramage-HP-EliteBook-8470p:~/workspace/workspaceVV/TP6/code/tpSpoon/src/main/resources/example-instrumented$ java -cp ./target/example-1.0-SNAPSHOT.jar example.A 2
Picked up JAVA_TOOL_OPTIONS: -javaagent:/usr/share/java/jayatanaag.jar
A.main(String[] args)
A.mth1(int count)
B.mth1(int i)
C.C(int i)
C.mth1()
error in A.mth1(int count)
B.mth1(int i)
C.C(int i)
C.mth1()
result = 100
B.mth2()
B.void mth2():1
C.int mth1():2
A.void mth1(int):1
A.void main(java.lang.String[]):1
B.void mth1(int):2

ramage@ramage-HP-EliteBook-8470p:~/workspace/workspaceVV/TP6/code/tpSpoon/src/main/resources/example-instrumented$ ^[[2;3~

```

A la fin du terminal, on peut voir le nombre d'appel de chaque méthode.

Programming 2 : Utilisez Spoon afin de construire l'arbre d'appel des méthodes d'un programme donné.

J'ai choisi de représenter l'arbre d'appel des méthodes dans un arbre (`TreeCall`) où chaque nœud de cette arbre est une méthode. Dans cette classe, j'ai un attribut nœud courant qui me permet à chaque fois de savoir où je suis rendu dans l'arbre.

J'ai créé un processeur (`CallProcessor`) qui à chaque début de méthode rajoute du code. Ce code rajoute un nœud enfant au nœud courant dans la classe `TreeCall`, le nœud courant devient ensuite le nœud que je viens de rajouter. A chaque fin de méthode mon processeur rajoute aussi du code, pour m'indiquer qu'il faut remonter le nœud courant de l'arbre. Le nœud courant devient le parent du nœud courant.

Une fois que mon processeur a fini de parcourir toutes les méthodes, toute la structure de l'arbre se trouve dans la classe `TreeCall`, il ne reste plus qu'à l'afficher.

J'utilise aussi un processeur (`ExceptionHandler`) qui va parcourir toutes les block catch, ce processeur va me rajouter du code dans ce block, ce code appelle une fonction de mon arbre, pour remonter au bon parent dans ma classe `TreeCall`.

Résultat : Après avoir lancé la commande

```
mvn package,
```

```
puis
```

```
java -cp .:target/tpSpoon-1.0-SNAPSHOT-jar-with-dependencies.jar vv.spoon.MainCallMethod  
src/main/resources/example src/main/resources/example-instrumented
```

pour exécuter le processeur `CounterMethod` sur le dossier `src/main/resources/example`.

Lorsque l'on va dans le dossier `src/main/resources/example-instrumented` et qu'on lance les commandes :

```
mvn package,
```

```
puis
```

```
java -cp .:target/example-1.0-SNAPSHOT.jar example.A 2
```

```

ramage@ramage-HP-EliteBook-8470p: ~/workspace/workspaceVV/TP6/code/tpSpoon/src/
ramage@ramage-HP-EliteBook-8470p: ~/work... x ramage@ramage-HP-EliteBook-8470p: ~/work... x
ramage@ramage-HP-EliteBook-8470p:~/workspace/workspaceVV/TP6/code/tpSpoon/src/main/resources/example-instrumented$ java -cp ./target/example-1.0-SNAPSHOT.jar example.A 2
Picked up JAVA_TOOL_OPTIONS: -javaagent:/usr/share/java/jayatanaag.jar
A.main(String[] args)
A.mth1(int count)
B.mth1(int i)
C.C(int i)
C.mth1()
error in A.mth1(int count)
B.mth1(int i)
C.C(int i)
C.mth1()
result = 100
B.mth2()
A.void main(java.lang.String[])
|      A.void mth1(int)
|      |      B.void mth1(int)
|      |      |      C.int mth1()
|      |      |      B.void mth1(int)
|      |      |      |      C.int mth1()
|      |      |      |      B.void mth2()
ramage@ramage-HP-EliteBook-8470p:~/workspace/workspaceVV/TP6/code/tpSpoon/src/main/resources/example-instrumented$

```

A la fin du terminal, on peut voir l'arbre d'appel des méthodes de mon programme