

RAPPORT DU PROJET JAVA

The Péchoux Interpreter



07 janvier 2015

UFR MATHEMATIQUES ET INFORMATIQUE

Licence L3 MIASHS parcours MIAGE

Table des matières

[Introduction 1](#_Toc408428714)

[Les choix 1](#_Toc408428715)

[Pour la construction de l’AST 1](#_Toc408428716)

[Pour l’interprétation de l’AST 1](#_Toc408428717)

[Pour l’interface graphique 1](#_Toc408428718)

[Les difficultés rencontrées 1](#_Toc408428719)

[Conclusion 1](#_Toc408428720)

# Introduction

# Les choix

La date du début de l’écriture du code a été relativement tard par rapport aux autres groupes. Et la raison est pourtant simple, nous avons pris beaucoup de temps pour analyser le sujet, et essayer de comprendre la meilleure manière de construire un AST (Abstract Syntaxic Tree) et de l’interpréter, étant donné le fait que nous avons décidé de ne pas utiliser d’outils existants, c’est-à-dire que nous avons tout réalisé « à la main » à l’aide de nos connaissances (et de l’API Java très riche).

## Pour la construction de l’AST

Le souci, c’est que la longueur d’une commande est indéfinie. Par exemple, pour une commande d’assignation, la longueur peut différer :

* a := 3
* a := 3 + 4 / 5 – 2 \* 3 + 10

Et on peut faire très long. Par conséquent, il faut maîtriser une notion avancée d’algorithmique qui s’appelle : **la récursivité**. Il est nécessaire d’avoir une méthode qui construit un arbre en fonction d’une ligne, et il faut qu’elle soit récursive. C’est ce qui va nous permettre de mieux visualiser les choses.

Reprenons les deux exemples ci-dessus. Le premier cas est assez simple à comprendre, nous détectons le signe « := », et donc nous allons créer un Nœud pour l’assignation. Ensuite nous détectons le 3 comme valeur, pas besoin de récursivité. En revanche, pour le second exemple, la récursivité est alors fondamentale ! Pour la valeur, on va obtenir une chaîne de caractère relativement complexe, et par conséquent, on va appeler à nouveau notre méthode de création d’arbre. On va détecter un par un les signes de calcul (+, / …) et à chaque étape, un nouveau appel récursif.

Nous allons adopter ce principe pour l’ensemble des commandes de ce mini langage impératif. Cependant, il va falloir faire face à certains désagréments.

En fait, tous les opérateurs de calcul seront traités de la même manière dans la méthode récursive, mais il ne faut pas oublier qu’il faut gérer la priorité des opérateurs (les multiplications et divisions prioritaires par rapport aux additions et aux soustractions). Pour pallier à ce souci, il suffit simplement de séparer tous les opérateurs en deux types d’opérateurs (prioritaires et non prioritaires) et de traiter tous les prioritaires si existants, puis les non prioritaires.

Un autre exemple de désagréments se situe au niveau de l’ordre de traitement. En effet, si on prend la commande suivante : if a < b then return a. Dans tous les cas, pour créer l’arbre entier, il faudra traiter le « if » et le signe « < ». Mais, on est bien d’accord, on traite d’abord le « if », c’est-à-dire que le nœud « if » sera la racine de l’arbre, et le signe « < » sera un Nœud intermédiaire.

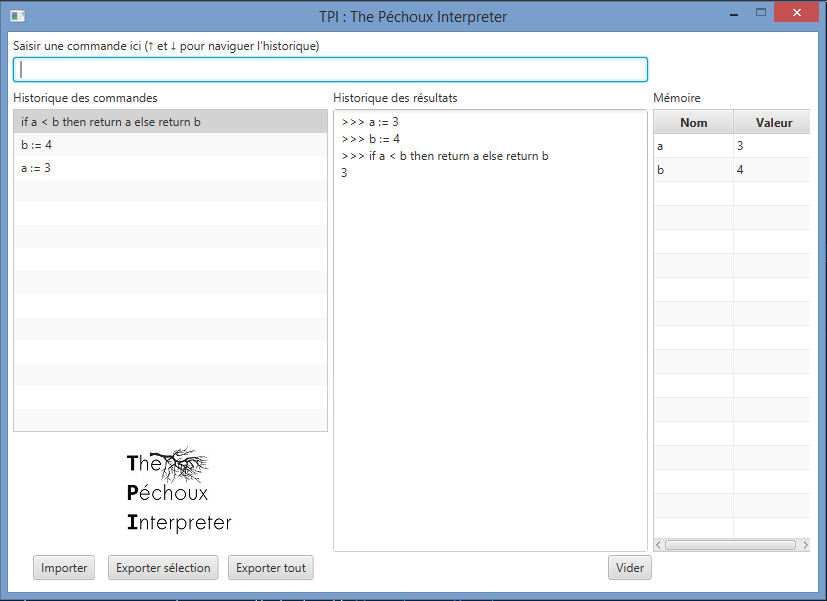
## Pour l’interprétation de l’AST

## Pour l’interface graphique

Le responsable de l’interface graphique aime bien l’API Java FX. C’est la raison pour laquelle toute l’interface fonctionne avec cette bibliothèque. Le seul éventuel problème est le fait qu’il est préférable d’utiliser le JDK 1.8. Sinon, il faut installer la bibliothèque Java FX nous-mêmes. (Petit rappel : Java FX n’est pas disponible dans openjdk).

Ensuite, étant donné que tous les algorithmes de création d’arbres syntaxiques et d’interprétation de ceux-ci ont été réalisés nous-mêmes, il en est de même pour la conception et l’écriture du code de l’interface. Le placement des éléments sur la fenêtre a été calculé à la main, en fonction des variables de classe défini dans un fichier Java nommé « Config.java ». Ceci permet de dimensionner les éléments de la fenêtre en fonction de sa taille, pratique pour changer la taille de la fenêtre.

Sans attendre, voici une illustration de l’interface (sa dernière version) :



# Les difficultés rencontrées

# Conclusion