FICHE 1

ENVIRONNEMENT DE TRAVAIL

(Fiches Java)

1 Généralités

Le **langage Java** est un langage **pseudo-compilé**. Dans un premier temps le fichier source est compilé en pseudo-code (java-code), ce pseudo-code étant interprété dans un deuxième temps (toutefois des compilateurs en code machine existent). Java fait partie de la catégorie des **langages orientés objets**.

Le pseudo-code est interprété par une **Machine Virtuelle Java (JVM)** qui dépend de l'environnement dans lequel elle doit travailler.

Un source java est constitué d'un ou plusieurs fichiers. Chacun d'entre eux définit une **classe** d'objets. Le nom de chaque fichier doit être adapté au nom de la classe qu'il définit. Ainsi, si on écrit le source d'une classe *MaClasse*, **le fichier doit impérativement s'appeler** *MaClasse.java* (attention aux majuscules et minuscules, le langage Java fait la distinction).
Pour réaliser un programme exécutable une des classes doit être écrite à cet effet et fait éventuellement appel aux autres ainsi qu'à des bibliothèques de classes.

Le langage java permet au minimum de produire deux sortes de fichiers exécutables.

- Des **programmes indépendants** qu'on exécute depuis le système d'exploitation par une JVM adhoc. La JVM prend alors le nom d'interpréteur java.
- Des **applets** destinées à être exécutées par des navigateurs www qui implémentent une JVM.

1.1 Programmes indépendants

L'environnement de travail doit au minimum comprendre

- **Un éditeur** pour éditer un programme. Le fichier source du programme aura alors un nom du type *toto.java*.
- **Le compilateur** en pseudo code **javac** qui à partir du source *toto.java* produit le fichier *toto.class*. La commande est en général du type *javac toto.java*.
- **L'interpréteur** du pseudo code **java** qui à partir du pseudo-code *toto.class* exécute le programme. La commande est de la forme *java toto*.

Remarque: En fait la situation est un peu plus compliquée, car un programme est souvent constitué de plusieurs fichiers dont l'un d'entre eux est le point d'entrée du programme. Nous verrons plus loin comment tout ceci s'articule.

1.2 Applets

L'écriture et l'utilisation d'une **applet** demande

- Comme dans le cas précédent, un éditeur et le compilateur en pseudo-code pour écrire l'applet Java.
- L'écriture d'une portion de **code HTML** qui appelle l'applet.
- **Un navigateur**, muni d'un plugin qui lance une JVM sur le code java appelé par le code HTML.

2 Les packages

Java a de nombreuses bibliothèques standards, distribuées sous forme de packages. Ces packages (en général compressés sous forme de fichiers **.jar*) ont une localisation déterminée par l'installation. Lorsque on écrit soi-même des packages, afin qu'ils soient accessibles par les programmes qui les utilisent, on définit une variable d'environnement *CLASSPATH* qui indique les répertoires racines de ces packages. Par exemple sous linux avec le shell tcsh

setenv CLASSPATH /home/robert/Travail/Prog/lang_java/packages :.

A partir de ce répertoire racine on déclare les classes à utiliser sous la forme

import rr.modulog.syntaxe.ArbreDeFonction

ce qui veut dire que la classe *ArbreDeFonction* se trouve en fait dans le répertoire

/home/robert/Travail/Prog/lang_java/packages/rr/modulog/syntaxe

Cette façon de nommer les packages garantit qu'il n'y aura pas de collision de noms. On peut dans un programme utiliser une classe avec son nom court,

par exemple ici la classe *ArbreDeFonction* si dans le préambule on a importé par exemple

import rr.modulog.syntaxe.$*$

Remarquer l'utilisation de $*$ pour dénoter toutes les classes qui sont dans *rr/modulog/syntaxe*. A remarquer aussi qu'il convient de mettre le répertoire courant "." dans la liste des répertoires racines (sinon on ne peut plus appeler une classe du répertoire de travail).