Aide-mémoire pour les TPs

Informatique Fondamentale (IF121)

5 octobre 2004

Les sujets et les corrections des TPs sont accessibles sur ma page `http://www.pps.jussieu.fr/~lengrand/Work/Teaching/` que j'essaierai de mettre à jour régulièrement.

Pour écrire un programme Java[1], il vous faut :

- un éditeur de texte (kwrite, xemacs, pico, vi, ...)
- la plateforme Java contenant notamment le compilateur Java (commande `javac`) et la machine virtuelle Java (commande `java`).
- la classe `fr.jussieu.script.Deug`, dont la documentation se trouve à l'adresse suivante : `http://www.liafa.jussieu.fr/~yunes/deug/Deug/`

1 Les fichiers sources

Les fichiers sources d'un programme contiennent le code source de ce dernier et sont lisibles et éditables par le programmeur. Ils ont en général le suffixe de fichier `.java`

Que contiennent-ils ?

- (éventuellement) des instructions pour importer des packages, de la forme
  `import Monpackage;`,
  comme `import fr.jussieu.script.Deug;`
  (que l'on utilisera toujours en TP)
- des définitions de classes, de la forme
  ```
  class MaClasse{
        ...
  }
  ```
  Il est d'usage que les noms de classes commencent par une majuscule.
- (éventuellement) des commentaires, qui peuvent apparaître à n'importe quel endroit du fichier.
  Le compilateur ignorera tout ce qui suit `//` jusqu'à la fin de la ligne. Exemple dans la ligne suivante :
  `bla bla // MonCommentaire`
  le compilateur compile `bla bla` mais ignore `MonCommentaire`
  Le compilateur ignorera aussi tout ce qui suit `/*` jusqu'à ce qu'il rencontre `*/`. Exemple ci-après.

Il est d'usage de décrire le contenu de chaque fichier source par un en-tête (sous la forme d'un commentaire) indiquant le nom de la classe, les auteurs, la date et une notice explicative, sous la forme :

```
/* MaSuperClasse
 *
 * auteurs : Gustave Martin et Antoinette Dupond
 *
 * date : le 25 septembre 2003
 *
 * Affiche le message ``Bonjour''.
 */
```

Que contiennent les définitions de classes ?

- des attributs (à voir plus tard dans le cours)

[1] http ://java.sun.com/docs/books/tutorial/getStarted/cupojava/unix.html#2

- des méthodes
  Ce sont plus ou moins des fonctions qui prennent en entrée un certain nombre d'arguments (éventuellement 0), exécutent des instructions, puis retournent éventuellement un résultat.
  (Parmi les instructions, il peut y avoir l'affichage de quelque chose dans le terminal, la saisie de quelque chose par l'utilisateur dans le terminal... ceux-ci ne sont pas considérées comme le résultat du programme au sens strict.)
  La syntaxe d'une méthode est la suivante :

```
TypeDuResultat NomDeMaMethode (TypeArgument1,...,TypeArgumentN){
    ... //instructions, chacune terminée par ;
    return MonResultat //objet envoyé, EVENTUELLEMENT
}
```

  Attention, une telle définition peut parfois être précédée de mots-clef come `public` ou `static`

On appelle **classe exécutable** une classe contenant la définition d'une méthode intitulée `main`. Celle-ci **doit** avoir la syntaxe suivante :

```
public static void main(String[] args) {
    ...
}
```

Très brièvement,

- `public` veut dire, comme son nom l'indique, que la méthode peut être accessible par n'importe qui, en particulier par celui qui veut l'éxécuter
- `static` veut dire que la définition de la méthode ne pourra pas évoluer au cours de l'exécution du programme
- `void` est le type de l'objet renvoyé comme résultat de la méthode `main`. Comme on ne veut pas qu'elle retourne d'objet comme résultat, le type "neutre" qui ne contient pas d'objet est appelé `void`
- `String[] args` est le type des arguments que l'utilisateur peut vouloir donner (à tort ou a raison) à l'éxecution du programme. Que le programme se serve de ces arguments **ou non**, il faut que ce type apparaisse dans la définition de la méthode `main`, sinon on ne saurait pas quoi faire de ces arguments.

Bref, comme un fichier source est un fichier contenant du texte, vous pouvez les créer, les éditer et les sauvegarder dans un éditeur de texte, comme ceux sus-mentionnés.

Il est d'usage de ne définir qu'une seule classe par fichier source, et de donner au fichier le nom de la classe avec le sufffixe `.java`

2 Compilation et Exécution

On compile un fichier source (par exemple intitulé MonFichier.java) par la commande à saisir dans un terminal
`javac MonFichier.java`

Le compilateur crée alors un fichier Blop.class **pour chaque** classe `Blop` définie dans le fichier MonFichier.java

Si une classe est exécutable (par exemple, `MaClasseExecutable`), alors on peut l'exécuter, une fois compilée (en un fichier `MaClasseExecutable.class`), à partir d'un terminal avec la commande
`java MaClasseExecutable`
Ne pas mettre l'extension .class dans cette commande

Et je ne le redirai jamais assez, toutes ces manipulations sont sensibles aux Majuscules/Minuscules. Pensez à vérifier qu'à chaque parenthèse (ou accolade) ouvrante correspond une parenthèse (ou accolade) fermante, et à finir chaque instruction par un ;

3 Divers

Copie sur disquette :
`mcopy -r *.java a:`

Pour lancer un programme à partir du terminal sans perdre la main dans le terminal, rajouter `&` à la fin de la commande. Exemple :
`kwrite &`