

TD1 POO / Java

bases du langage, encapsulation et abstraction

Objectif : Découvrir les bases du langage Java.

Matériel / Logiciel : Environnement de développement sur PC (sous Linux) : JDK1.7 (ou supérieur) : compilateur java (javac) et JVM (java) éditeur gedit et commandes en mode console.

Acquisition : Maîtrise des concepts objets fondamentaux : classe, objet, attribut, méthode, paquetages, abstraction et encapsulation (niveaux de visibilité « private » et « public »), éléments de syntaxe (class, this, this(), main, toString(), accesseurs et mutateurs)

Préparation de votre environnement

1. Depuis la racine de votre compte utilisateur (dossier personnel ou « *home directory* »).
 - 1.1. Afficher les fichiers cachés du répertoire personnel. (sous linux, les fichiers cachés commencent par un point).
 - 1.2. Éditer (ou créer-le si il n'existe pas) le fichier « .bashrc » et ajouter les lignes suivantes à la fin du fichier :


```
export CLASSPATH=$HOME/classes
alias javac="javac -d $HOME/classes "
```
 - 1.3. Créer un répertoire « classes ». N.B. : ce dossier « classes » accueillera les fichiers des classes compilées.
2. Placez-vous dans le dossier « Documents ».
 - 2.1. Créer un dossier de travail pour vos fichiers sources (« .java »), nommé « tdjava ».

Remarque : le dossier « classes » sera utilisé par le compilateur, vos programmes (sources) d'extension « .java » sont à placer dans le sous-dossier de « Documents » nommé « tdjava ».

Vérification de votre configuration (étape 1) :

Éditer, compiler et exécuter le programme « Hello World » (fichier Hello.java) :

```
public class Hello {
    public static void main(String[] args) {
        System.out.println("Hello Wordl !!") ;
    }
}
```

Remarque : vous utiliserez l'éditeur gedit pour ce premier programme simple.

```
javac Hello.java
java Hello
```

Vérification de votre configuration (étape 2) :

Ajouter un paquetage « test » (ligne « package test ; » au début du fichier) et vérifier en mode console (terminal) les commandes javac et java sur ce premier programme :

```
javac Hello.java
```

```
java test.Hello
```

La classe Point

Reprendre la classe Point vue en CM (écrire également le petit programme de test) et vérifier les étapes de compilation et d'exécution. Tester avec les commandes javac et java dans la console.

Ajouter les paquetages « geometrie » et « test » à vos classes et vérifier le bon fonctionnement du programme.

La classe Etudiant

On souhaite à présent définir une classe « Etudiant » permettant de représenter les caractéristiques principales d'un étudiant dans une application informatique (comme celle qu'utilise le service scolarité de l'université). On ne considérera que les informations suivantes : nom, prénom, numéro d'étudiant, téléphone et mail (adresse courriel de l'université).

- a) Définir le modèle d'une telle classe en utilisant la représentation UML¹ (diagramme de classes).
- b) Implémenter le classe « Etudiant » ainsi qu'une classe de test séparée.
- c) Vérifier les étapes de compilation et d'exécution.
- d) Ajouter les paquetages.
- e) Vous veillerez à spécifier les accesseurs et mutateurs utiles ainsi que les différentes variantes de constructeurs.
- f) Apporter les améliorations utiles à votre code (méthode « toString() », usage de « this. » et « this() » et « this », non redondance de code, contrôle des entrées au niveau des mutateurs).

Important : vous veillerez à bien respecter le principe de l'encapsulation (niveaux d'accessibilité aux attributs et méthodes) et la présentation de vos programmes (indentation et commentaires).

¹ UML : *Unified Modeling Language*.