Polytech Lille IMA2A4
Conception Modélisation Objets (CMO)
TP1

1 Prise en main du JDK (Java Development Kit)

— Ajouter à votre `PATH` (dans votre fichier `.bashrc`) le chemin du jdk1.7 :
`export PATH=/usr/local/jdk7/bin:$PATH`
Atention : ne utiliser pas l'installation par défaut (`gcj`), les autres versions de JDK sont disponibles sur : `/usr/local/jdkXX.`

— La documentation (javadoc) est sur :
`https://docs.oracle.com/javase/7/docs/api/`
conserver le lien dans vos bookmarks.
La documentation est entièrement navigable. A gauche, vous pouvez choisir une navigation alphabétique de "All Classes" ou d'un package particulier. A droite apparaissent les informations sur la sélection, notez les options :
- "Tree" : hiérarchie des packages et des classes de la sélection
- "Index" : index alphabétique de tous les symboles associés (classes, variables, méthodes, constructeurs, ...)

— Observer comment sont décrites quelques classes vues en cours : `Object, String, Applet, System, Scanner ...`

— Programmer l'applet `Salut` du cours :
- la tester d'abord sur l'outil "appletviewer" standard du JDK en lançant la commande :
`appletviewer salut.html &`
- puis dans un browser (firefox, konqueror, ...).
Remarque pour konqueror :
- allez dans "Settings/Configure Konqueror/Web Browsing/Java & Javascript/Java"
- cocher "Enable Java globally"
- ajouter en bas le "Path to Java executable" :
`/usr/local/jdk7/bin/java` Remarque pour firefox : - allez à l'URL about :plugins et activer le plugin Java.

— Essayez d'autres applets (il n'est pas demandé de comprendre le code...) :
`http://weppes.studserv.deule.net/~bcarre/anemometre`
`http://www.polytech-lille.fr/~bcarre/ppo/euro`
`http://www.oracle.com/technetwork/java/index-135948.html`

— Applications en ligne :
Ecrire le programme `Echo` du cours. Compiler (commande `javac`), exécuter (commande `java`).

2 Rectangles

Travailler dans un répertoire ‘‘`rectangles`’’.

2.1 `java.awt.geom`

Programmer une classe `Rectangle` (dans un fichier `Rectangle.java`) telle que celle vue en cours :

— un rectangle est représenté par un couple de points “origine” (point supérieur gauche) et “corner” (point inférieur droit). On utilisera la classe `Point2D.Double` du package `java.awt.geom` (voir la javadoc).
— programmer les méthodes : `largeur(), longueur(), surface(), perimetre()`.
— programmer un constructeur paramétré par les coordonnées des 2 points origine et corner.
— tester la classe `Rectangle` en programmant une classe principale `TestRectangle` (fichier `TestRectangle.java` dans le même répertoire) munie d'une méthode `main` qui :
- instancie un rectangle de coordonnées fixées, par exemple : `10.0 10.0 40.0 50.0`
- affiche ses caractéristiques : largeur, longueur, surface, périmètre.

2.2 `toString()`

Programmer une méthode `toString()` dans la classe Rectangle qui renvoie sous forme de chaine de caractères son couple de points caractéristiques : ‘‘`(<origine> , <corner>)`’’. `<origine>` et `<corner>` correspondent à la représentation sous forme de chaine de caractères des points renvoyée par leur propre méthode `toString()` (voir la javadoc), laissez les s'afficher comme ils veulent !

Ajouter dans la classe `TestRectangle` l'affichage du rectangle par appel automatique à `toString()`.

2.3 Méthodes `static` et utilisation de `Scanner` sur `System.in`

Modifier la classe `TestRectangle` en ajoutant une méthode `creerRectangle()` qui demande à l'utilisateur les coordonnées de 2 points origine et corner, instancie le rectangle correspondant et le retourne en résultat. Cette méthode devant être appelée par le `main` qui est `static` doit elle-même être `static` (même niveau) de la façon suivante :

```
public class TestRectangle {
 static Rectangle creerRectangle() {
  //saisie utilisateur et creation du rectangle
  ...
 }
 public static void main(String[] args) {
  Rectangle r;
  r = creerRectangle();
  ...
 }
}
```

2.4 Paramètres du `main`

Dans la classe `TestRectangle`, utiliser les paramètres du main pour récupérer un quadruplet de coordonnées, instancier le rectangle correspondant et le tester.

Comme toujours, les paramètres du `main` sont des chaînes de caractères, encodant ici des doubles (''`10.0`'' par exemple) qu'il faut transformer en valeurs (`10.0`). Pour cela utiliser la méthode static `Double.parseDouble(String s)` de la classe `Double`, wrapper de `double`.

2.5 Tableau de rectangles

Modifier la classe `TestRectangle` comme suit :

- Créer un tableau de rectangles dans la méthode `main` en demandant à l'utilisateur le nombre de rectangles à créer (taille du tableau) et en le remplissant par appel itéré à la méthode `creerRectangle()`.
- Afficher le tableau de rectangles ainsi créés (par appel à leur méthode `toString()`).
- Ajouter une méthode :
  `static Rectangle max(Rectangle[] t)`
  qui renvoie le rectangle de plus grande surface du tableau t.
- Tester cette méthode dans le main.

2.6 Exercice supplémentaire : Rectangle dans une applet

Programmer une applet (classe `RectApplet` et fichier `html` correspondant) dont la méthode `paint` crée une instance de `Rectangle` et l'affiche sur son support `Graphics`. Pour cela :

- voir dans la doc la méthode `drawRect(x,y,width,height)` de `Graphics`
- ajouter à `Rectangle` une méthode `display(Graphics g)` qui l'affiche par `drawRect` sur `g`. Le support g est fourni par l'applet dans cet exemple mais pourrait venir de tout autre environnement d'affichage graphique, ce qui rend Rectangle réutilisable.
- quels sont les objets en jeu et leurs interactions dans ce petit programme ?