TP n°5 Java

Pour ce TP, nous allons utiliser une bibliothèque simplissime de dessin. Il faut télécharger le fichier *drawing.jar* dont la documentation se trouve sur :
http://lacl-upec.github.io/l2epotp/ext/drawing/doc/

Vous devez ajouter la bibliothèque à BlueJ (sur le menu Outils–> Preferences –> aller sur l'onglet Librairies et cliquer sur Ajouter, puis sélectionner le fichier Drawing.jar. Redémarrer BlueJ).

Ajouter également la ligne suivante dans votre programme : import fr.lacl.cpo.Drawing ;

Exercice 1

Pour se familiariser avec la bibliothèque, effectuer les dessins suivants:

- un rectangle.
- un étoile à cinq branche (on rappelle que pour obtenir les coordonnées des cinq points extrêmes, on peut utiliser, pour `i` allant de 0 à 4, les coordonnées :

```
xCentre+rayon*Math.cos(i*2*Math.PI/5)
yCentre+rayon*Math.sin(i*2*Math.PI/5)
```

- un mélange aléatoire des ces formes de différentes tailles, positions et couleurs.
- comme le point précédent mais en ajoutant une temporisation entre chaque apparition
- un étoile qui tourne sur elle-même (pour dessiner l'étoile mais ayant tourné de « `x` », il faut faire varier `i` de `x` à `4+x` par pas de 1 ; `x` est à prendre entre 0 et 1).

Exercice 2

Pour cet exercice, il faudra écrire la documentation au fur et à mesure des classes.

1. Créer une classe Point pour représenter un point. Elle contiendra ses coordonnées, sa couleur et l'épaisseur du trait.
2. Ajouter une méthode void `draw(Drawing d)` qui dessine le point.
3. Créer une classe `Triangle` pour représenter un triangle. Les trois segments peuvent être de couleurs différentes mais sont tous de la même épaisseur. Les points ont leur propre couleur et épaisseur (la même pour les trois). Y adjoindre la méthode `draw`.
4. Idem pour les rectangles.
5. Créer une classe `Maison` qui représente une maison, toujours avec une méthode `draw`.
6. En utilisant la classe précédente, faire une animation avec une maison dont la taille devient de plus en plus grande.
7. Faire un diagramme de classe de votre solution.

Exercice 3

1. Créer une classe `Forme` qui contient un tableau de points (utiliser un tableau classique dans une première solution puis dans le conteneur ArrayList dans une seconde solution). Implémenter la méthode `draw` pour qu'elle dessine la forme (d'une seule couleur et une seule épaisseur)
2. Surcharger la méthode `draw` : implémentez une méthode `void draw(Drawing d, boolean diagonales)` qui dessine aussi les « diagonales » si le deuxième paramètre est `true`. Si la forme a moins de 4 points, cette méthode doit lancer une exception de type `IllegalArgumentException`. On rappelle la double boucle `for` pour parcourir toutes les paires *(i,j)* d'un `tableau` :

   ```
   for (int i=0;i<tableau.length-1;i++) {
      for(int j=i+1;j<tableau.length;j++) {
   ```

3. Créer une classe `Mondrian` qui contient un tableau de formes, toujours avec une méthode `draw`
4. Surcharger la méthode `draw` de la classe `Mondrian` pour qu'elle prenne en paramètre aussi un `double diag` et qui dessine les formes avec la diagonale avec probabilité `diag`. Si l'exception `IllegalArgumentException` est lancée, la méthode dessinera un X rouge de largeur 50 en haut à gauche de la fenêtre.
5. Faire un diagramme de classe de votre solution.