

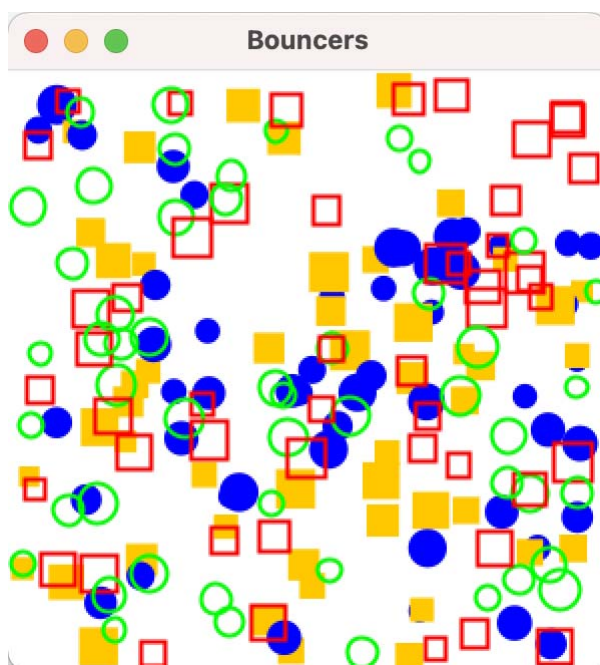
## Laboratoire 2b: Cercles et carrés

*Durée du laboratoire: 4 périodes. A rendre pour le mercredi 11 avril 2024.*

### 1. Objectif

Concevoir une application utilisant les modèles singleton, fabrication et fabrique abstraite.

### 2. Indications



Définir une application graphique qui permette d'instancier des cercles et des carrés et de les déplacer dans un espace d'affichage graphique commun.

- Deux familles de formes géométriques doivent pouvoir être générées :
  - des cercles et carrés pleins de couleur respectivement bleue et orange,
  - des cercles et carrés possédant une bordure de 2 pixels (c.f. classe `BasicStroke` d'AWT) de couleur respectivement verte et rouge.
- Les formes géométriques devront être initialisés aléatoirement (taille, position, vecteur de déplacement).
- Si une forme géométrique rencontre un bord, elle doit rebondir.
- Ces formes doivent implémenter l'interface `Bouncable` et leur affichage (déclenché par l'appel à `draw()`) doit être délégué à un objet implémentant l'interface `Renderer`.
- La classe gérant l'affichage doit mettre en oeuvre un singleton et implémenter l'interface `Displayer` donnée en annexe.

Pour gérer l'affichage, il peut être utile de passer par une `Image` obtenue depuis la méthode `createImage` (de la classe `JPanel`), afin d'utiliser son `Graphics2D` obtenu depuis la méthode `getGraphics` (de la classe

Image). Ensuite les formes peuvent être dessinées sur cet objet `Graphics2D`, qui pourra être réaffiché sur le `JPanel` au moyen des méthodes `drawImage` et `fill` (de la classe `Graphics2D`).

- Gestion du clavier (c.f. classe `KeyAdapter`) :
  - Touche 'e' : effacer l'affichage.
  - Touche 'b' : générer 10 cercles et 10 carrés possédant une bordure.
  - Touche 'f' : générer 10 cercles et 10 carrés pleins.
  - Touche 'q' : quitter le programme.
- La fenêtre contenant l'affichage doit pouvoir être redimensionnée et son affichage initial doit être vide .

Veiller à soigner l'implémentation et à ce que le code produit soit le plus élégant possible selon les standards de programmation OO. Veiller en particulier à ce qu'un nombre minimal d'objets soit instancié.

### 3. Travail à rendre

A rendre sur cyberlearn et en version imprimée:

- Diagramme des classes.
- Sources du programme.

### Annexes

Les classes `Graphics2D`, `KeyAdapter`, `Color` et `Shape` sont celles d'AWT.

```
public interface Displayer {
    int getWidth();
    int getHeight();
    Graphics2D getGraphics();
    void repaint();
    void setTitle(String s);
    void addKeyListener(KeyAdapter ka);
}

public interface Bouncable {
    void draw();
    void move();
    Color getColor();
    Shape getShape();
}

public interface Renderer {
    void display(Graphics2D g, Bouncable b);
}

public class Bouncers {
    private LinkedList<Bouncable> bouncers;
    /* ... */

    public Bouncers() {
        /* ... */
    }

    public void run() {
        /* ... */
    }

    public static void main(String ... args) {
        new Bouncers().run();
    }
}
```