Rapport programmation concurrente

CUILLIER Edmond, L3 informatique

13 novembre 2018

Résumé

Ce rapport LaTeXà pour but de résumer notre projet de programmation concurrente. Celui-ci consiste en la création d'un programme qui affiche des balles en mouvement dans une fenêtre.

1 Introduction

A travers ce rapport, nous allons vous présenter notre programme. Celui-ci à pour but de nous habituer à l'utilisation de threads, de leur structure et des leurs fonctions.

```
notifyall(); wait();
```

. Nous commenceront par présenter les prérequis de ce programme, puis l'architecture de notre dossier et enfin le problémes que nous avons rencontrés.

2 Le programme

2.1 Les conditions de l'application

Le programme se doit de réaliser les actions suivantes :
- Si une balle touche la paroi d'une fenêtre celle-ci doit rebondir dans le sens inverse ;
- Si une balle touche une autre Balle les deux disparaissent et incrémentent un compteur comptant seulement les collisions de balles ;
- Un timer comptant le nombre de secondes écoulées depuis le début de l'animation doit être implanté ;
- Un bouton play mettant en pause l'animation et le timer lors du clic ;
- Un bouton + permettant l'ajout d'une balle dans l'animation ;
- Un bouton - permettant le retrait d'une balle dans l'animation ;

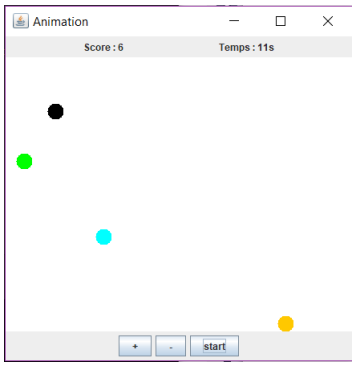

FIGURE 1 – L'interface du logiciel

2.2 Architectures

L'application est composée des classes suivantes :

1. La classe Fenetre.java génère la fenêtre et ses composants dont le Panneau pan, qui est une classe étendue de JPanel. Cette classe permet entre autre de définir la taille de la fenêtre et son agencement, nous avons opté pour un gridLayout et un Borderlayout. C'est aussi dans cette fenêtre que les fonctions d'écoutes sur les boutons sont appelées.
2. La classe Panneau.java permet de créer le dessin et de manipuler les cercles (classe cercle) appelant la fonction move. Elle retire les cercles lors de collisions.
3. La classe Cercle.java est la classe modélisant une balle, elle dispose de classes set et get pour tout ses attributs et d'une fonction move permettant de déplacer le cercle en fonction du temps.
4. La classe ThreadCollisionAffiche.java permet de gérer le nombre de collisions depuis le lancement de l'application et l'affiche dans un JLabel.
5. La classe ThreadNotPause.java permet de gérer la pause, elle appelle la fonction relaunch de l'objet pan, qui permet de notifier les threads et donc de les relancer.

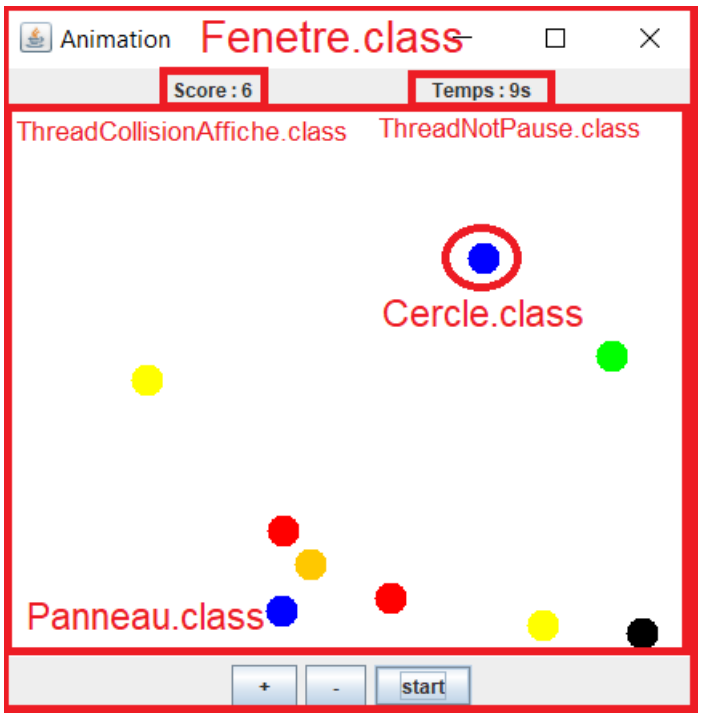

FIGURE 2 – L'interface du logiciel et les classes correspondantes

2.3 Probléme rencontré

Lors de l'arret de l'animation quand on appuis sur play, j'ai mis un

```
println(pan.pause);
```

dans la classe *ThreadNotPause.java* et cela fonctionne, l'animation s'arrête et redémarre quand j'appuie sur play. Cependant, si je retire le

```
System.out.println(pan.pause);
```

(pan.pause correspondant à l'état de l'animation, si elle est en pause ou si elle ne l'est pas), la pause fonctionne mais désactiver la pause ne fonctionne plus, je n'ai strictement aucune idée de pourquoi cela se passe de cette façon.

FIGURE 3 – Probléme du prinln

On obtient de ce fait un retour dans la console ce qui nous empêche d'y avoir accès pendant la pause et donc de tester d'autres variables avec println() (gênant pour le débuggage).

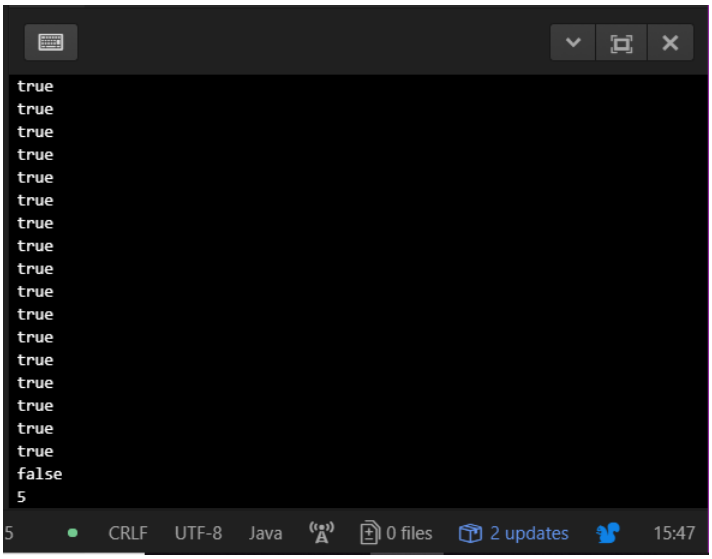

FIGURE 4 – probléme console

Un autre soucis rencontré est l'apparition de balles et la disparition de balles pendant la pause si l'on manipule les boutons + et -. Il n'est cependant pas possible de voir le compteur de collision augmenter, car le programme vérifiant les collisions est en pause. De ce fait même si deux balles apparaissent l'une sur l'autre cela ne fausse pas la vérification des collisions.

Le dernier soucis est du au fait que la fenêtre n'est pas responsive, en effet si l'on réduit la taille de celle-ci, les balles apparaitrons en hors-champs, et glisserons vers l'intérieur de la fenêtre pour rebondir contre les bords. De la même façon, si la fenêtre est trop grosse, les balles apparaitrons en haut à gauche dans tout les cas. Cela est du au fait que l'apparition des balles n'est pas liée la taille de la fenêtre.

FIGURE 5 – zone d'apparition des balles

2.4 Conclusion

Pour résumer, ce programme nous a permis d'approfondir la notion de thread d'événement et de fenêtre en java. J'ai choisi le langage Java pour réaliser ce projet car c'est celui que nous avons le plus étudié, et que je maitrise le mieux contrairement à Python que nous avons étudié il y a deux ans et revu légèrement l'année dernière. Si j'avais eu plus de temps, je pense que j'aurais aimé corriger les quelques bugs qui subsistent comme par exemple la zone d'apparition des balles qui se serais adaptée à la taille de la fenêtre et surtout le problème du println() dans la classe ThreadNotPause.java.

3 Références

3.1 Tutoriels

1. https://openclassrooms.com/fr/courses/26832-apprenez-a-programmer-en-java/23108-notre-premiere-fenetre
2. http://denis.payet.top/ (précisément le tp sur les fenêtres et les formes géométriques, cependant vous ne trouverez pas le pdf sur le site, le cour date de l'année dernière. Je peu cependant vous l'envoyer si nécessaire)
3. https://openclassrooms.com/fr/courses/1617396-redigez-des-documents-de-qualite-avec-latex

3.2 Sujet de forums

1. https://www.developpez.net/forums/d1547635/applications/developpement-2d-3d-jeux/physique/collision-entre-boules-glissade-non-rebonds/ pour la collision