Compte Rendu de la séance 2

I. Introduction

Le but de ce projet est de réaliser une version plus simpliste du célèbre jeu *Flappy Bird*. En effet, il s'agira d'une version beaucoup plus simpliste que l'originale, nous allons matérialiser l'oiseau par un ovale et le chemin sera matérialisé par une ligne, une ligne que l'ovale ne devra pas quitter. Le but du joueur est de faire avancer l'ovale tout en faisant en sorte qu'il ne quitte pas ladite ligne. Pour cela, le joueur devra cliquer dans l'espace de la fenêtre pour faire sauter l'ovale, qui redescendra tout seul, et ainsi le faire progresser, il ne devra pas le laisser aller en dehors de la ligne ou alors la partie sera perdue et il devra recommencer. La difficulté du jeu est avant tout de réussir à appréhender le saut et la chute de l'ovale afin de ne pas quitter la ligne et espérer arriver à la fin de celle-ci.

Figure 1 : Aspect final espéré du jeu

Sur la figure 1, on peut apercevoir l'ovale ainsi que la ligne rouge qu'il devra suivre. Pour gagner, joueur devra se mouvoir pour atteindre la fin de la ligne.

II. Analyse Globale

La semaine passée, nous avons implémenté plusieurs fonctionnalités comme une interface graphique (création d'une fenêtre et d'un ovale), une interaction entre l'utilisateur et l'ovale grâce au *MouseListener*, lors que cette séance ci nous devons implémenter la chute constante de l'ovale afin de donner un effet que l'ovale doit être maintenu en l'air, la génération de la ligne brisée ainsi que son déplacement et enfin la création et la suppression de points de la ligne brisée. Pour faire cela, l'utilisation des *Thread* nous sera très utile afin de réussir à tout gérer en même temps.

III. Plan de Développement

L'avancement de ce projet s'est déroulé de façon très méthodique, voici chacune des tâches qui ont été réalisées. Tout d'abord, une lecture et une analyse du problème (environ 15 minutes) ensuite une étape de conception, de développement et de test d'une sorte de gravité pour que l'ovale descende en permanence (environ 30 minutes) par la suite, une étape de conception, développement et de test de la ligne brisée (environ 30 minutes), il s'en est suivis de la création d'une ligne brisée infinie (environ 45 minutes) et enfin, la réalisation de la documentation du projet (environ 60 minutes)

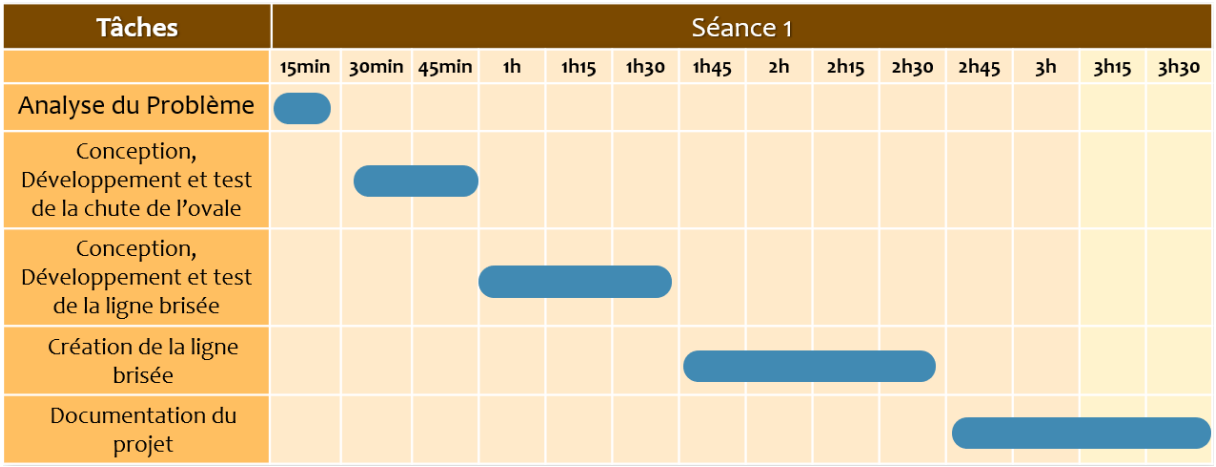

Figure 2: Diagramme de Gantt du déroulé de la séance

IV. Conception Générale

Pour la réalisation de notre projet nous avons déjà réalisé le modèle MVC, il ne restait plus qu'à continuer à le respecter. Dans le *Model*, il suffisait d'implémenter la méthode *moveDown* et la création du parcours que l'ovale devrait parcourir, dans la *View*, il y a eu quelques modifications à apporter afin d'afficher la ligne brisée et enfin il n'y avait rien à changer dans la partir *Controler* du projet.

V. Conception Détaillée

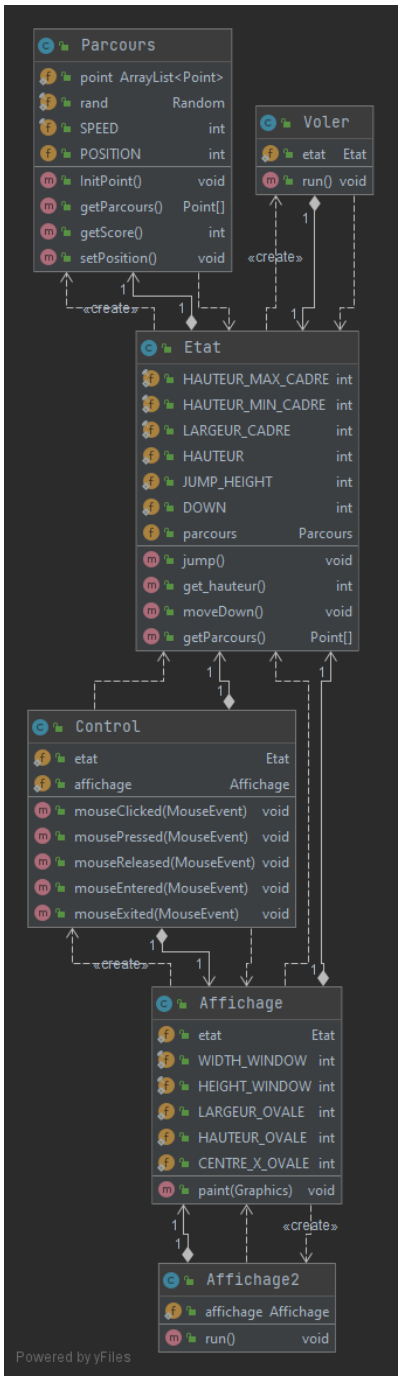

Figure 3 : Diagramme de classe

Afin de simuler la chute de notre ovale sans utiliser la méthode *repaint()* vu que nous sommes dans la partie *Model* de la MVC, on va créer deux *Thread*, un premier dans la classe *Etat* qui va modifier la hauteur de l'ovale et un deuxième dans la classe *Affichage* qui va redessiner la fenêtre ou bout d'un lapse de temps défini au préalable.

VI. Résultats

Ci-dessous les premiers résultats obtenus lors de cette deuxième séance, on peut y voir la ligne brisée ainsi que l'ovale qui descend si l'utilisateur ne le maintien pas en l'air. Nous pouvons remarquer que ce visuel se rapproche de celui présenté en *figure 1*.

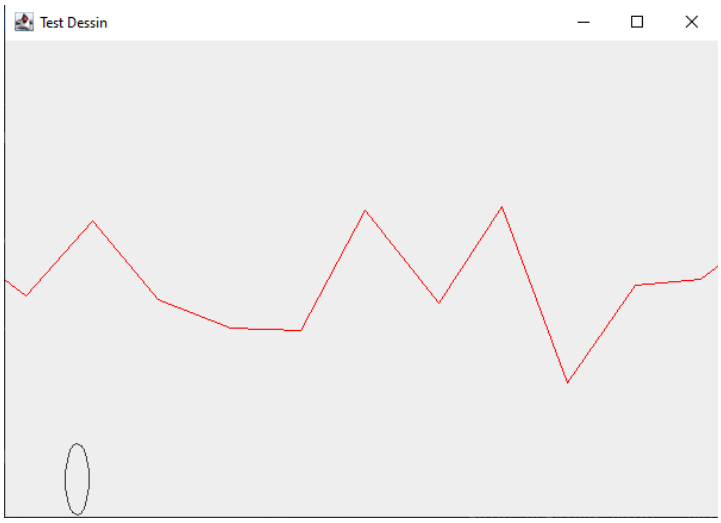

Figure 4 : Visuel du jeu à la fin de la séance 2

VII. Documentation Utilisateur

Pour pouvoir jouer au jeu il y a quelques petites choses à savoir. Tout d'abord, un IDE Java est nécessaire (ou Java seul si jamais vous avez fait un export *.jar* en exécutable). Si vous êtes dans le premier cas et que vous avez un IDE Java, il vous faut importer le projet dans votre IDE, sélectionner la classe *Main* et enfin lancer l'application '*Run as Java Application*'. Vous pouvez cliquer sur la fenêtre pour faire monter l'ovale. Si vous être dans le second cas, que vous possédez un exécutable *.jar*, il vous suffira de double cliquer sur le fichier exécutable et de cliquer dans la fenêtre pour faire monter l'ovale.

VIII. Documentation Développeur

Il n'y a que peu de choses à dire pour le moment, la fonctionnalité la plus difficile à implémenter est la génération ainsi que la suppression de la ligne brisée. Toutefois, les classes *Etat* et *Affichage* sont des classes qui sont tout autant intéressantes, nous avons implémenté des sous classes permettant de faire tomber l'ovale ainsi que de redessiner la fenêtre après un certain lapse de temps. Cependant, certaines des fonctionnalités à développer cette séance n'ont pas été faites et devront donc être implémentées pour la séance suivante.

IX. Conclusion et Perspectives

En cette fin de deuxième séance, nous n'avons pas pu implémenter toutes les fonctionnalités que l'on aurait aimé. En effet, la suppression et la génération de la ligne brisée ne se pas encore implémentées, il reste également beaucoup de choses à améliorer ou à créer comme par exemple la fameuse suppression de la ligne brisée mais aussi rendre le jeu plus jouable vis-à-vis des pentes de la ligne brisée qui pour le moment sont en aléatoire et nous pourrons peut-être entrevoir de profiter de ce jeu comme il se doit. La séance prochaine nous devrons finir d'implémenter les fonctionnalités que nous n'avons pas pu faire cette semaine ainsi que de rajouter la collision ainsi qu'un system de score.