Documentation Projet Flappy Bird

A. Introduction

L'objectif de ce TP était de nous initier à la création d'une interface interactive dans le langage JAVA ainsi qu'à la programmation concurrentielle.

Nous avons pour consigne de créer un jeu dans lequel le joueur, à l'aide de control, peut faire remonter sur l'axe verticale un cercle. L'objectif est d'empêcher les extrémités du cercle d'entrer en contact avec la courbe qu'il parcourt.

Le joueur gagne un point à chaque saut effectué. Son score final sera donc le nombre total de saut réalisé sans toucher la courbe.

B. Analyse

Le projet est structuré en différents package qui suivent le modèle MVC regroupant nos différents class selon leurs fonctionnalités. Quelques exemples de fonctionnalité qui ont été intégré :

Déplacer une image uniquement de manière verticale

Création aléatoire d'une courbe à partir de Points(x,y)

Manipulation de différents threads qui nous permette de manipuler séparément les différents états des objets du jeu.

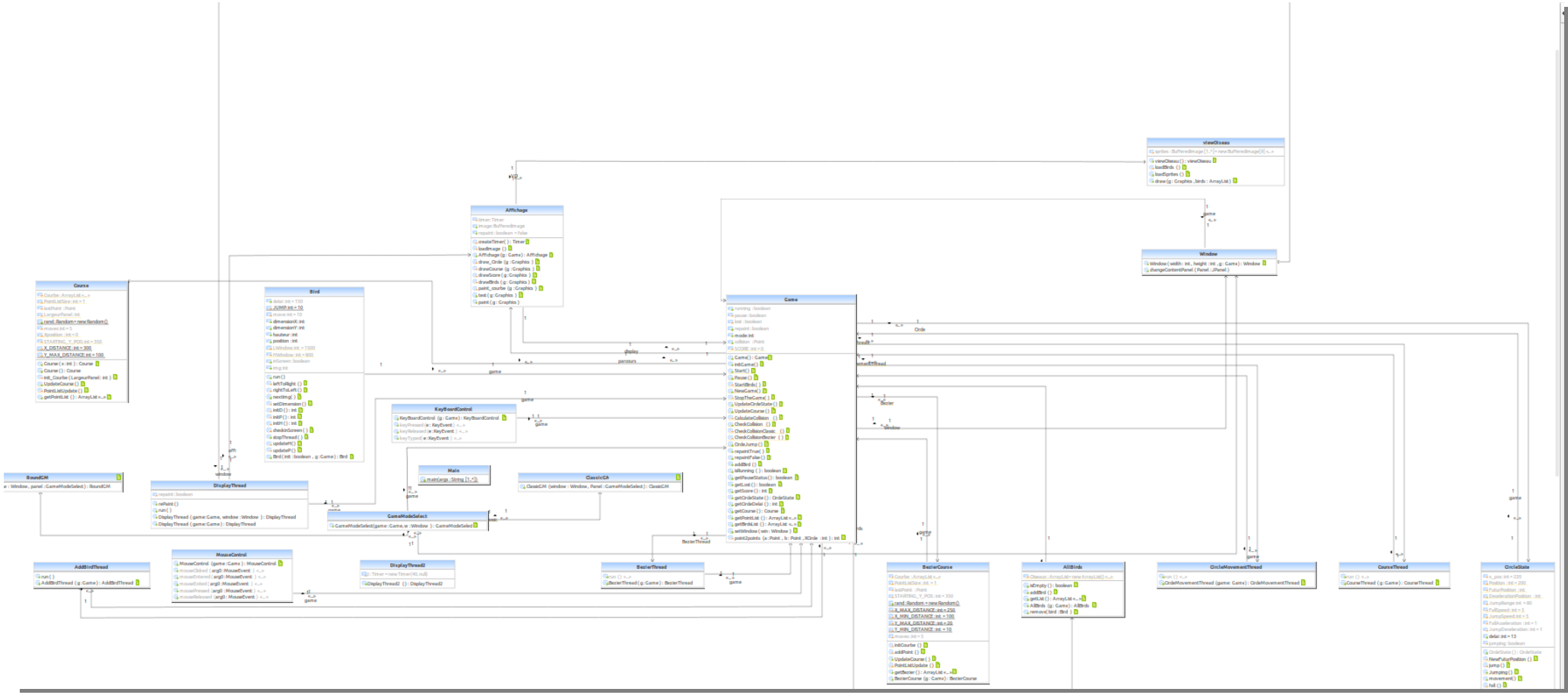
C. Organisation du travail

- Analyse du problème (30 minutes)
- Conception (1 heure)
- Implémentation (10 heure)
- Documentation (30 min)
- Acquisition de compétences (1 heure)

D. Conception

- Control
  - KeyBoardControl.java — Contrôle au clavier
  - MouseControl.java — Contrôle a la souris
- Model
  - AddBirdsThread.java — Ajoute des oiseaux de façon aléatoire au cours de la partie
  - AllBirds.java — Liste des Oiseaux de la partie
  - BezierCourse.java — Etat du parcours avec des courbes de Bézier
  - BezierThread.java — Thread pour déplacer le parcours
  - Bird.java — Thread qui gère le déplacement d'un oiseau
  - CircleMovement.java — Thread qui gère le déplacement du cercle
  - CircleState.java — Etat du cercle
  - Course.java — Etat du parcours
  - CourseThread.java — Thread pour déplacer le parcours
- theGame
  - Game.java — Gere l'état des différents model du jeu
  - Main.java — Lance le jeu

- View
  - Affichage.java — Affiche les états du jeu dans une fenêtre
  - GameModeSelect.java — Panel de choix de sélection du mode de jeu
  - viewOiseau.java — Gere l'affichage d'un oiseau
  - Window.java — Créer une fenetre et y ajoute du contenu


viewOiseau
Affichage
Window
Course
Bird
Game
KeyBoardControl
Main
ClassicGA
RoundGM
DisplayThread
GameModeSelect
MouseControl
AddBirdThread
DisplayThread2
BezierThread
BezierCourse
AllBirds
CircleMovementThread
CourseThread
CircleState

E. Résultat

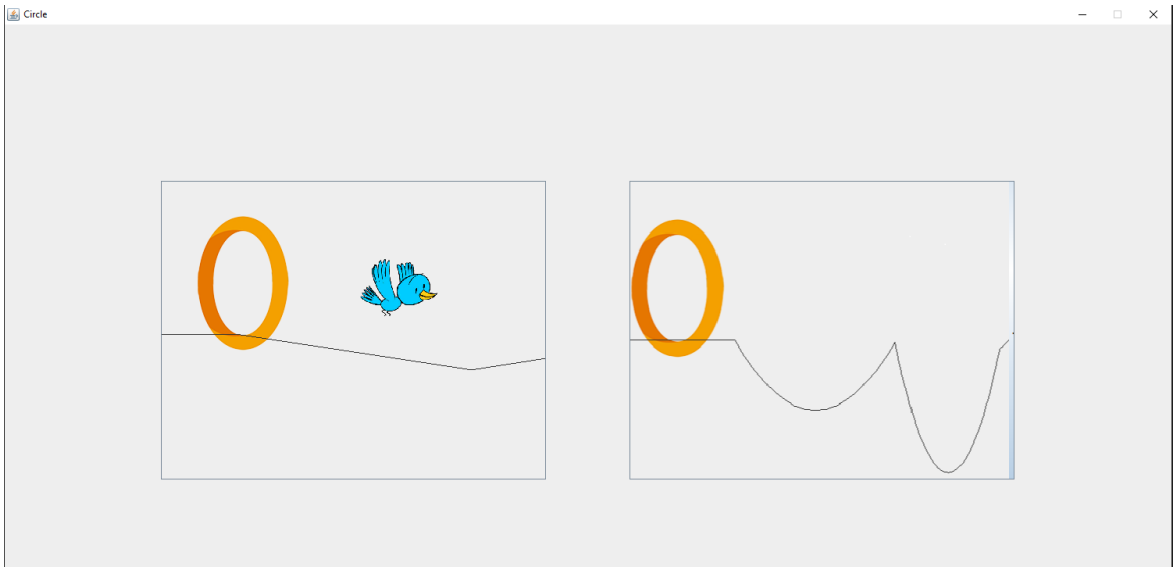


Figure 1 Mode Select

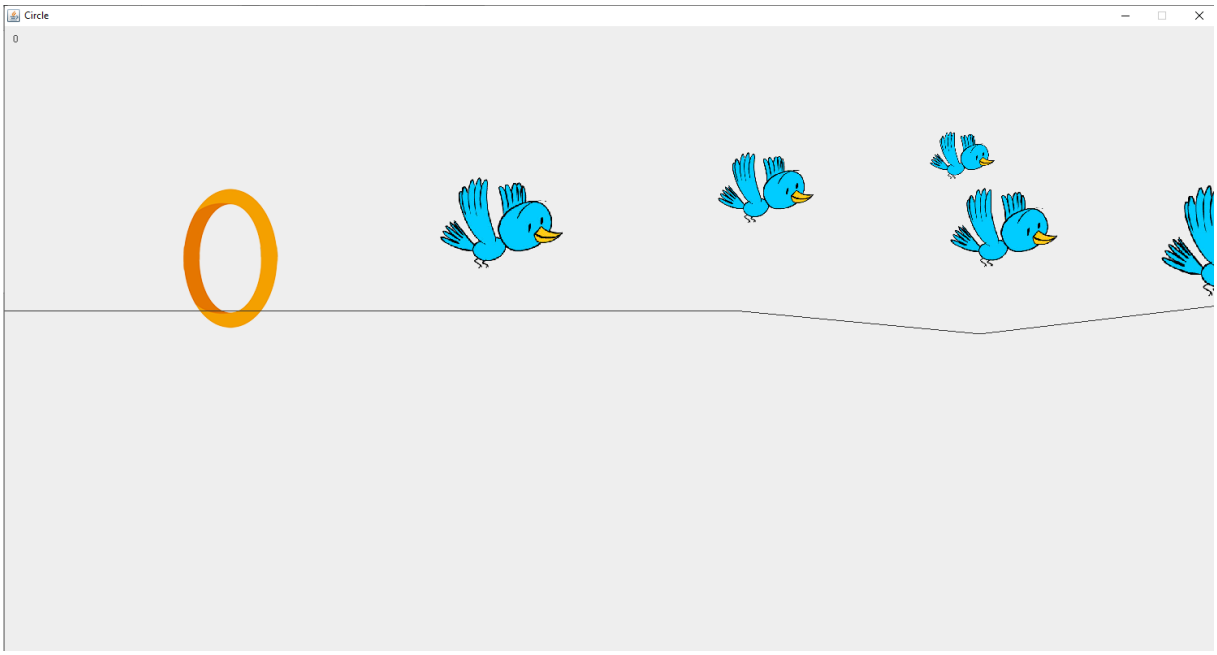


Figure 2 Classique Mode

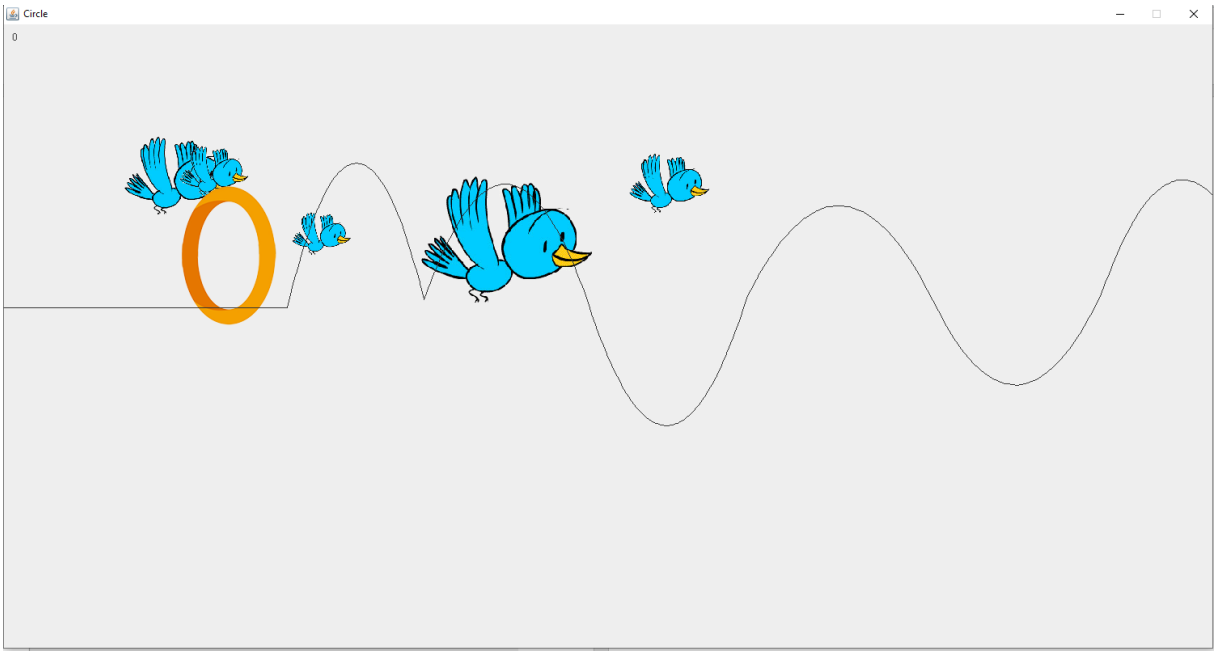


Figure 3 Bézier Mode

F. Documentation utilisateur

- Lancer l'exécutable java
- Touches de contrôles :
  - **Monter**
    - Barre espace
    - Clic gauche souris
  - Pause
    - Touche clavier « p »
  - Start en début de partie
    - Touche clavier « s »

- Quitter en fermant la fenêtre.

G. Documentation développeur

Choses modifiables :

- Couleur et épaisseur de la courbe dans la classe ***Affichage*** méthode *drawLine*
- Possibilité de changer la vitesse de saut du cercle dans la classe **CircleState** en modifiant la variable *delai*
- La distance horizontale entre deux points du parcours classique dans la classe **Course** en modifiant la variable *X_DISTANCE*
- La distance verticale max entre deux points du parcours classique dans la classe **Course** en modifiant la variable *Y_MAX_DISTANCE*
- La distance horizontale entre deux points du parcours Bézier dans la classe **BezierCourse** en modifiant les variables *X_MAX_DISTANCE* et *X_MIN_DISTANCE*
- La distance verticale max entre deux points du parcours Bézier dans la classe **BezierCourse** en modifiant les variables *Y_MAX_DISTANCE* et *Y_MIN_DISTANCE*
- Le nombre d'oiseaux à générer en début de partie en modifiant le constructeur de la classe **AllBirds**
- Temps de rafraichissement pour déterminer si oui ou non on ajoute un nouvel oiseau à la partie en modifiant le delai du Thread dans la methode run() de la classe **AddBirdThread**

H. Conclusion et perspectives

Evolution futures :

- Ajouter du décor
- Trouver une manière de gérer les collisions dans le cas des courbes de Bézier car la méthode *contains* de la classe *Shape* ne correspond pas à ce que l'on souhaite faire.