2020

Bovay Louis

DIVTEC

05/11/2020

FlappyBird – Documentation technique



**Table des matières**

[1. Description du projet 1](#_Toc55470734)

[2. Convention de nommage 1](#_Toc55470735)

[3. Explications supplémentaires 2](#_Toc55470736)

[3.1 Les « fausses scènes » 2](#_Toc55470737)

[3.2 CSS dans le projet 2](#_Toc55470738)

[4. Sources 3](#_Toc55470739)

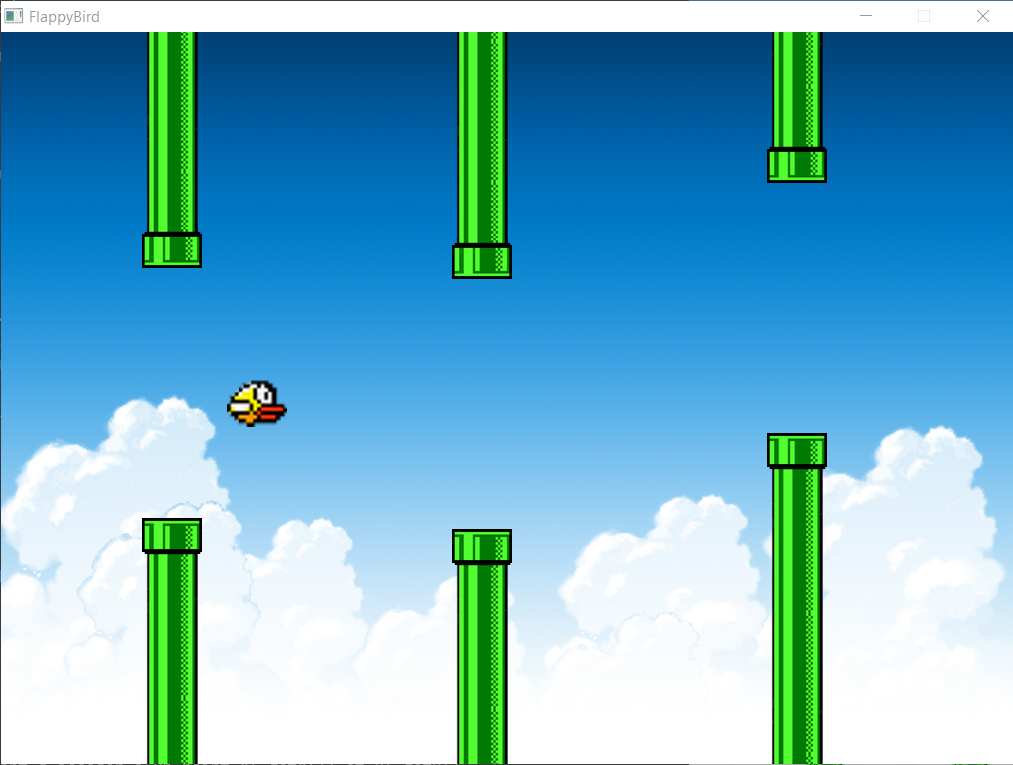
# Description du projet

Le projet à pour but de réaliser le jeu FlappyBird en programmation orientée objet.

Le but du jeu est de faire battre des ailes un oiseau pour qu’il ne se cogne pas à des tuyaux placés en haut et en bas qui foncent sur lui.

S’il l’oiseau touche un tuyau, le plafond ou le sol, il meurt.

A chaque fois qu’il esquive un couple de tuyaux, un point est marqué.



# Convention de nommage

-Tous les noms de variable sont écrits en CamelCase

-Tous les noms de fonction doivent décrire l’action faite dans la fonction et sont écrits en CamelCase.

-Tous les noms de classe sont écrits en CamelCase et commence par une majuscule

# Explications supplémentaires

Cette partie est là pour expliquer ce qui ne pourrait pas être facile à comprendre ou pour facilité l’apprentissage de méthodes utilisées au sein du projet.

Si une fonction / variable est mentionnée ou alors qu’une partie de code est montrée, il sera précisé à chaque début de sujet dans quel fichier la trouver.

## Les « fausses scènes »

Main.java

En théorie, on pourrait dire que le jeu à trois scènes :

* Le menu principal
* Le jeu
* Fin de partie

En pratique, c’est toujours la même scène qui se joue mais qui change d’état.

Il y a pour ça deux booléen :

*boolean* isGameRunning = *false*; *boolean* isGameStarted = *false*;

IsGameStarted sert juste une seule fois pour faire apparaître le menu principal, nous ne sommes as sensé pouvoir retourner au menu principal, à moins de relancer l’application

IsGameRunning sert à savoir si le jeu est en cours ou non, si isGameRunning = true alors le jeu se joue, si isGameRunning = false l’écran de score s’affiche et propose de rejouer.

Tout ça sert savoir quand lancer une de ces trois fonctions :

*public void* startGame() {…}

*public void* restartGame() {…}

*public void* endGame() {…}

Ces trois fonctions servent comme leur nom l’indique à démarrer, redémarrer et finir la partie.

## CSS dans le projet

Main.java  
style.css

Le projet contient du CSS (le background est fait en CSS), et pour en utiliser il faut donner un ID à l’élément que l’ont souhait décorer, pour ensuite assigner du CSS à l’ID.

StackPane root = *new* StackPane();

*//Attribution de l’ID*

root.setId("pane");

*//Récupération de la feuille de style css*scene.getStylesheets().add("css/style.css");

CSS :

#pane {  
 -fx-padding: 10px;  
 -fx-background-image: url("../Sprites/background.png");  
}

## Gestion des tuyaux

Dans le jeu, il y a 4 couples de tuyaux « PipeCouple » qui contiennent deux tuyaux « Pipe » chacun et qui bouge en continu de droite à gauche.

Les deux tuyaux présents dans un couple ont un espace fixe de 250, l’espace pour apparaître n’importe où sur leur axe Y

Arrivé à gauche, ils sont déplacés instantanément à droite, regénère un espace aléatoire et recommencent à avancer.

À chaque fois qu’un couple passe la position Y de l’oiseau, leur état passe de « apte à donner un point » à « non apte à donner un point » (canGivePts)

# Sources

N’oublie pas les sources, fils.