

FLOPPY BIRD

<https://gitlab-cw1.centralesupelec.fr/maceo.tetelin/flappy-bird>

Wiktor PAZDRO
Arthur VIALIX
Thomas BANZET

Macéo TETELIN
Gabriel DE LENCQUESAING

Bilan de la semaine 1

Objectifs réalisés :

Minimum Viable Product réalisé

Initialisation de l'interface graphique Tkinter, mais pas entièrement implémentée

-> prise en main de GIT

-> prise en main de Tkinter

-> nécessité de répartir mieux les fonctionnalités entre les membres du groupe pour plus d'efficacité



SOMMAIRE



1

Présentation du projet

2

Structuration du Code

3

Répartition des tâches

4

Conclusion





01

Présentation du Projet





Objectif Principal



- Jeu vidéo d'obstacles en 2D inspiré de FlappyBird
- En Python, utilisation de la bibliothèque Pygame :
 - > hitboxes
 - > gestion des événements
- Menus pour gérer le lancement des parties, autres fonctionnalités



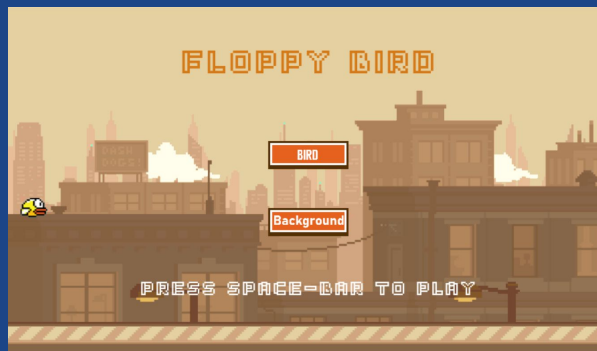
└ Le produit final: Comment l'utiliser ? ─

- Pour se procurer le jeu: suivre les instructions du fichier README.md (cf. dépôt Gitlab)
- Exécution du programme Python contenant la boucle de jeu
- Jouabilité à partir d'un clavier et d'une souris, la touche espace permet de lancer la partie et d'interagir (sauter)



▮ Différentes fonctionnalités ▮

- Menu permettant le choix du thème et du personnage

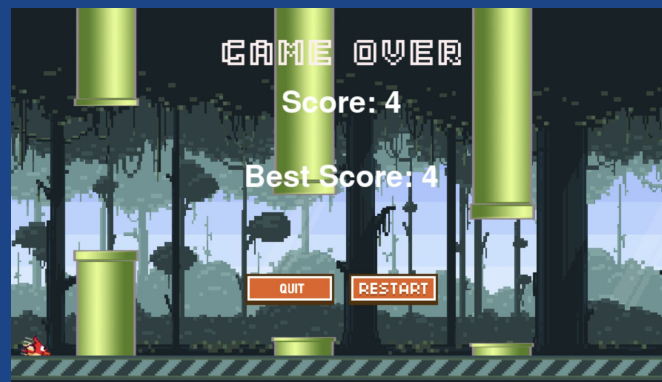


▮ Différentes fonctionnalités ▮

- Phase de jeu



- Ecran de défaite



- Système de comptage de score et implémentation du score max.
- Les fonds sont animés (défilement) et ont un effet 3D
- L'affichage des oiseaux est adapté à leur état (montée/ descente, battement d'ailes...)
- Possibilité de quitter le jeu ou de relancer une partie

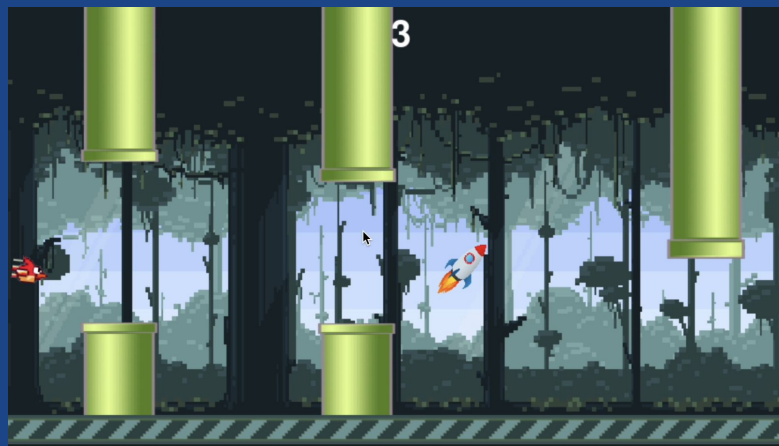
▮ Différentes fonctionnalités ▮

Power Up

- Changement de gravité



- Augmentation de vitesse

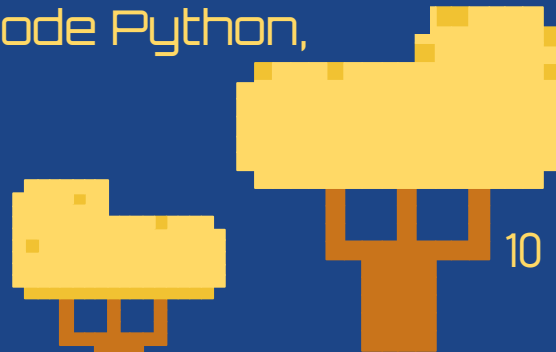


Avantages

- Prenant / Addictif
- Simple d'utilisation, menus clairs et fonctionnels

Limites

- Difficulté à étoffer encore le contenu du jeu
- L'exécution se fait uniquement via un code Python, rendant le produit final peu accessible





02

STRUCTURATION DU CODE





STRUCTURE



- Programmation Orientée Objet
- Répertoire jeu contenant les différents modules pour chaque élément du jeu où les différentes classes sont définies (bird, pipe...)



- Dans chaque module, on définit une classe. Ces dernières sont importées dans la boucle de jeu
- Répertoire contenant les sprites/son/police

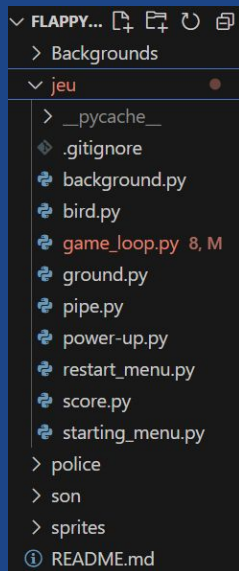




STRUCTURE



- Répertoire jeu contenant les différents modules pour chaque élément du jeu où les différentes classes sont définies (bird, pipe...)



```
class Button():
    def __init__(self,x,y,image,image_hovered,screen):
        self.image = image
        self.image_hovered = image_hovered
        self.rect = self.image.get_rect()
        self.rect.topleft = (x,y)
        self.screen = screen
        self.clicked = False

    def draw(self):
        action = False

        # Récupère la position de la souris
        pos = pygame.mouse.get_pos()

        # Vérifie si la souris est sur le bouton
        if self.rect.collidepoint(pos):
            if pygame.mouse.get_pressed()[0] == 1 and self.clicked == False:
                action = True
                self.clicked = True
            if pygame.mouse.get_pressed()[0] == 0:
                self.clicked = False

        # Fait apparaître le bouton
        if action == True:
            self.screen.blit(self.image_hovered, (self.rect.x,self.rect.y))
        else:
            self.screen.blit(self.image, (self.rect.x,self.rect.y))

        return action
```



Les différentes étapes

MVP

Première boucle de jeu fonctionnelle, le jeu se ferme à chaque défaite et se lance directement à l'exécution



Ajout des Menus

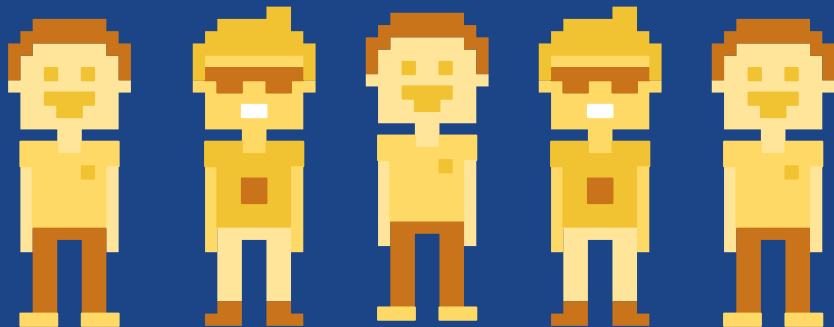


Ajout des
Textures/Musique

Ajout des PowerUp
(incomplet)

03

RÉPARTITION DES TÂCHES



RÉPARTITION DES TÂCHES






- Utilisation de Gitlab pour faciliter le travail en commun
- Identification des fonctionnalités à établir, changeant au fur et à mesure de l'avancée du projet.
- Répartition par une ou deux personnes sur une fonctionnalité, travail sur des branches distinctes puis merging avec le main une fois le bon fonctionnement validé
- Entraide entre les membres du groupe. Gabriel avait rôle de coordinateur pour assurer la bonne répartition des tâches





Exemple



- Création de nouveaux fonds animés et implémentation: Macéo 
- Création du système de score et implémentation: Wiktor
- implémentation d'une solution pour gérer l'apparition des tuyaux: Thomas
-  • Création menu de début de partie: Arthur
- Création menu fin de partie: Gabriel et Thomas 
- Mise en place d'un power-up permettant d'inverser la gravité du jeu: Gabriel et Macéo

T



04

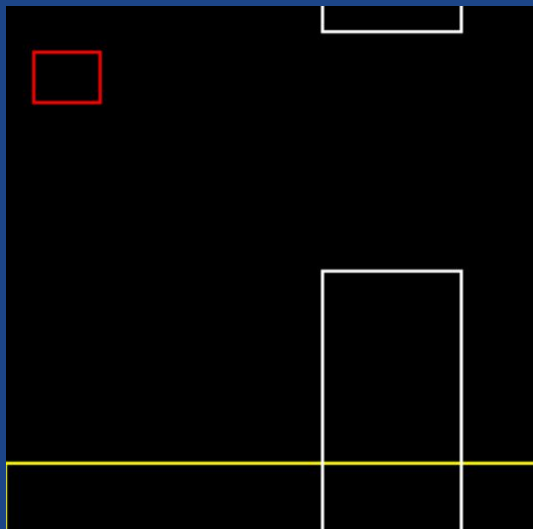
CONCLUSION



PISTES D'AMÉLIORATION



- Meilleure gestion des hitboxes



✚ PISTES D'AMÉLIORATION ✚



- Création d'une monnaie ou d'achievements pour ajouter une initiative de jeu permettant éventuellement de débloquent les différentes maps et skins de l'oiseau
- Ajout d'autres power-ups pour plus de diversité lors de la phase de jeu

