LES TROPHÉES NSI

Édition 2023

DOSSIER DE CANDIDATURE
PRÉSENTATION DU PROJET

Ce document est l'un des livrables à fournir lors du dépôt de votre projet : 4 pages maximum (hors documentation).

Pour accéder à la liste complète des éléments à fournir, consultez la page Préparer votre participation.

Vous avez des questions sur le concours ? Vous souhaitez des informations complémentaires pour déposer un projet ? Contactez-nous à info@trophees-nsi.fr.

NOM DU PROJET : Flapython

> PRÉSENTATION GÉNÉRALE :

Au cours du premier trimestre, notre professeur nous a parlé du concours des Trophées NSI, et nous a expliqué le principe. Mais au début, faute d'idée, personne n'a réellement commencé.

Lors d'une discussion en cours, nous avons lancé l'idée d'un Flappy Bird en Python, que l'on appellerait ainsi "Flapython", d'abord sur le ton de la rigolade. Ce projet, partant à la base d'une blague, a commencé petit à petit à se concevoir dans nos têtes de manière beaucoup plus sérieuse. Son développement a finalement débuté à partir de janvier 2023.

D'après Google, le concepteur de Flappy Bird aurait réalisé son jeu en 2 ou 3 jours. Dû à un manque de temps (et de talent), cela nous aura pris plusieurs mois, mais le résultat en vaut le peine !

La quantité de travail et d'efforts fournis dans ce projet, malgré la petite taille de notre équipe, est ce qui nous rend le plus fier !

L'idée derrière était de créer un jeu accessible pour tous, et une adaptation de ce célèbre jeu en Python était une évidence !

Parviendrez-vous à faire un meilleur score que vos amis ?

> ORGANISATION DU TRAVAIL :

Le groupe, ou plutôt le duo qui constitue celui-ci, est composé de **Hugo VILLER** et **Esteban JUNCOSA**.

Les tâches n'ont pas réellement été réparties, nous nous sommes chacun chargé de coder les fonctionnalités nécessaires. Il n'y a pas alors eu de planification en amont, mais voici un aperçu des tâches réalisées par chacun d'entre nous deux :

- Hugo : Réalisation et implémentation de la majorité des sprites, tuyaux, boutons, icônes, arrière-plan qui défile, sons et musiques, polices d'écriture, système d'achat, animations des éléments cliquables
- Esteban : Base du jeu (mise en place de Pygame), score, stockage des données dans les fichiers csv, animations de l'oiseau, système d'argent, implémentation de la possibilité de mettre le jeu en pause

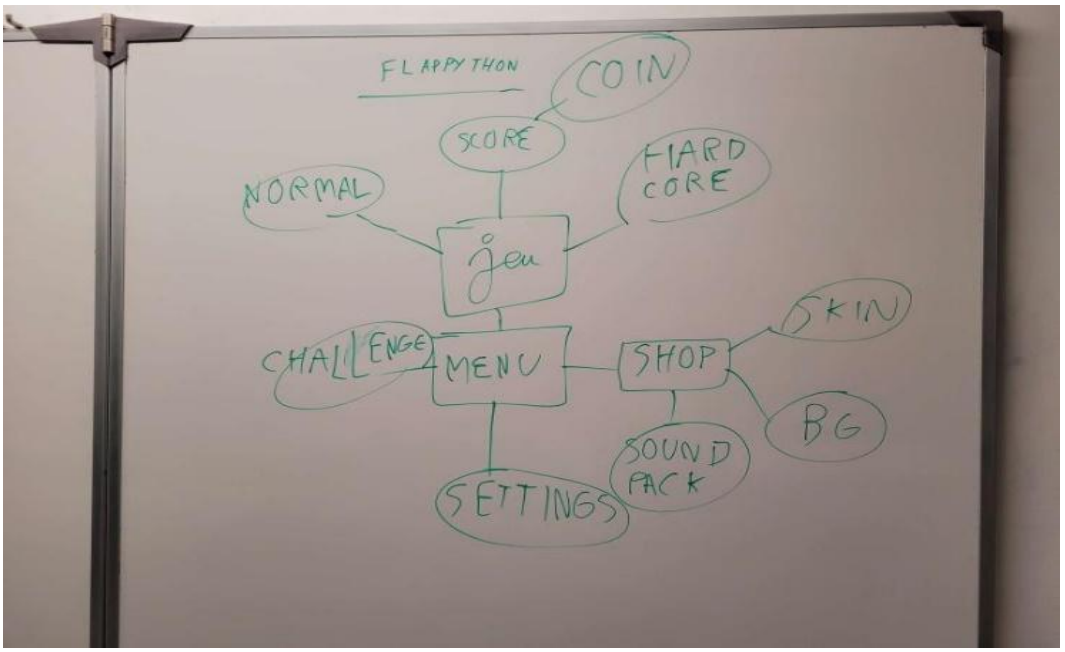
Nous avons, dès le premier jour, mis en place un **Google Drive** afin de nous partager le code et les divers fichiers (images, sons, csv, ...) pour une meilleure organisation. Nous avons aussi communiqué par le biais de **Discord,** puisque c'est ici où l'on échangeait déjà de manière générale, et car cela nous permettait de faire des partages d'écrans afin d'expliquer entre nous les changements que nous voulions apporter.

L'avancée était rapide au début, mais a connu un fort ralentissement dû notamment aux épreuves du baccalauréat, mais aussi à cause des difficultés que nous heurtions (notamment les problèmes de collisions).

Les différents commentaires et la docstring ont été réalisés et modifiés à chaque fois qu'un grand changement fonctionnel était implémenté.

LES ÉTAPES DU PROJET :

Au tout début, la première étape fut la réalisation d'un **schéma** afin de fixer nos objectifs :



Après avoir eu les idées claires dans nos têtes, nous nous sommes dirigés vers les ordinateurs. Nous avons d'abord commencé par faire la grande **base du programme**, c'est-à-dire une fenêtre vide avec un carré au milieu symbolisant l'oiseau. Puis nous avons implémenté la "**gravité**" qui fait tomber l'oiseau en continu sauf lorsque l'on appuie sur la touche "espace". Nous avons, après coup, remplacé le carré par l'image de l'**oiseau** et avons implémenté l'**arrière-plan**, avant de mettre en place le système en 3 temps (les **fonctions** gameStart, gameLoop et gameOver représentant respectivement le début du jeu, le déroulement de la partie et l'écran de fin proposant au joueur de recommencer) qui se répètent indéfiniment sauf en cas d'interruption de l'utilisateur.

Nous avons ensuite implémenté les **tuyaux**, les **collisions** avec ceux-ci, et nous avions notre base.

> FONCTIONNEMENT ET OPÉRATIONNALITÉ :

Le plus gros du jeu a été fait, mais de nombreuses fonctionnalités importantes et qui nous tenaient à cœur n'ont pas encore été implémentées.

Nous nous sommes concentré sur le **score** du jeu ainsi que sur un système de stockage du **record**, et c'est ainsi que nous avons commencé à utiliser les fichiers csv. Ensuite, nous avons fait en sorte qu'à chaque paire de tuyaux atteinte, nous gagnions une **pièce**, qui sera stockée par la suite dans un fichier csv pour avoir le nombre total de pièces récupérées. Une fois cela correctement implémenté, nous avons voulu rendre le jeu un peu plus attrayant pour nos oreilles. Ainsi, nous avons mis en place un bouton sur l'écran principal rendant la possibilité à l'utilisateur de changer la **musique** lors de sa session de jeu, tout ceci gratuitement (on est gentil quand même). Enfin, nous voulions absolument apporter une touche de personnalisation dans notre jeu, c'est pourquoi nous avons codé un **système d'achat** permettant de modifier l'apparence de l'oiseau ainsi que de l'arrière-plan. Cependant, pour effectuer cette action, vous devez verser des pièces que vous gagnerez au fur et à mesure de vos parties (contrairement aux musiques, cela n'est pas gratuit désolé). Un mode "**clavier**" et un mode "**souris**" est également disponible, permettant comme son nom l'indique de jouer soit avec le clavier avec la touche "espace" soit avec la souris avec le clic gauche.

La principale difficulté que nous avons rencontrée a été un **problème de collision**. En effet, nous utilisions des rectangles afin de vérifier si l'oiseau touchait un autre élément. Mais les Sprites n'étant pas rectangulaires, cela causait des problèmes de hitbox très frustrants qui rendaient le jeu tout simplement injouable. Afin de résoudre ce problème, Hugo eut la bonne idée de se servir de 'mask' à la place. En effet, il permet d'obtenir des collisions quasi-parfaites en détectant chaque pixel qui "entoure" l'image. Nous l'avons alors utilisé sur les objets "Surface" (dans notre cas, c'était l'oiseau et les tuyaux) et ainsi, les collisions étaient bien meilleures qu'auparavant. Une autre difficulté à laquelle nous avons fait face était l'apparence globale des **polices d'écriture**. Nous les avions pourtant bien choisies, mais elles ne possédaient aucun contour, ce qui les rendait moins intéressantes pour le style "arcade" et "rétro" de notre jeu. Au fil des recherches sur Internet, nous avons trouvé une fonction permettant d'ajouter un contour autour de chaque caractère d'un texte sur Pygame. Nous l'avons par conséquent utilisé pour notre jeu, et les polices d'écriture étaient devenues nettement plus jolies et pertinentes.

> OUVERTURE :

Malgré tout, il y a beaucoup de choses que nous aurions pu rajouter avec un peu plus de temps, qui auraient pu rendre le jeu plus “propre”.

Tout d’abord, il y a le système de **boutique**, qui a dû être hâté à cause des contraintes temporelles. En effet, le système actuel comporte plein de défauts, notamment le fait que l’on ne puisse pas conserver les éléments achetés, mais que l’on puisse juste payer pour changer d’élément. Nous pensions à la base à un menu qui pourrait apparaître et où il aurait été possible de choisir l’apparence que l’on veut.

Le système de **changement de difficulté** aurait lui aussi pu être intéressant à mettre en place, puisque le défaut qui est ressorti le plus souvent dans les tests que nous avons fait réaliser à nos cobayes (des membres de notre famille, des amis ou tout bonnement le reste de la classe de NSI) est la difficulté mal équilibrée. Ainsi, il y aurait eu un mode normal et un mode difficile/challenge/hardcore (le nom n’était pas encore décidé) où l’on obtiendrait par exemple 2 pièces au lieu d’une à chaque point, afin de récompenser le joueur qui prend plus de risques.

Une autre chose que nous aurions aimé implémenter, mais qui est bien moins prioritaire, est un **système de trophées**, qui donnerait aux joueurs un certain nombre de pièces dès qu’ils franchissent un palier important en thème de score (par exemple obtenir 20 pièces en plus des pièces que l’on obtient normalement avec le score dès que le score du joueur serait supérieur ou égal à 10 pour la première fois).

Plusieurs stratégies peuvent être viables pour populariser notre jeu, mais de nos jours, le plus intéressant serait les **réseaux sociaux**. En effet, ouvrir un compte sur plusieurs réseaux sociaux (TikTok, Instagram, YouTube, etc) permettrait de faire connaître le jeu et d’être en contact avec notre public de manière directe, puisqu’il pourrait nous faire des suggestions en se servant des commentaires.

Mais ce qui aurait le plus aidé à convaincre les gens de jouer à ce jeu aurait très probablement été une **adaptation sur mobile**, ~~du jamais vu~~. Mais, ce jeu étant déjà très populaire sur mobile, il aurait fallu une manière de se démarquer de la concurrence, et pour cela, nous avons pensé à implémenter plusieurs modes de jeu (un mode où, en plus de devoir éviter les tuyaux, il fallait tirer sur d’autres oiseaux, un mode où il n’aurait que fallu se battre contre un oiseau en face en esquivant ses tirs et en lui tirant dessus à notre tour, etc). Les idées et les possibilités n’étaient pas ce qui manquait, mais plutôt le temps, et peut être aussi un manque de personnel.

Lorsque nous avons vu les bases de données SQL en cours, nous avons directement eu l’idée d’utiliser un système de **classement global en ligne** (contrairement au système de record local dont nous disposons actuellement), mais nous aurions eu besoin de payer

pour garder ce système de classement totalement en ligne, ce qui rend donc cette implémentation irréalisable.

- ~~Raytracing~~
- ~~Transition vers Unreal Engine 5~~
- ~~Système de monétisation pour les développeurs (nous) avec des publicités obligatoires de minimum 1min30 entre chaque parties~~

Si l'on avait la possibilité de refaire ce projet, nous aurions probablement commencé le projet plus tôt, afin d'avoir assez de temps pour implémenter toutes ces fonctionnalités qui auraient été géniales à mettre en place.

DOCUMENTATION

Avant de pouvoir jouer, il est nécessaire de disposer de **Python3** ainsi que de posséder le module Python **Pygame**. Dans l'invite de commande, écrire la commande :
"pip install pygame" (sans les guillemets)

Le jeu fonctionne en trois temps :

GameStart

GameStart correspond à ce qu'il se passe dès lors que l'on lance le jeu, ou après que l'on appuie sur "recommencer" à la fin d'une partie, lorsque l'on accède. Dans celui-ci, nous pouvons accéder au nombre de pièces, changer ou désactiver la musique, changer le type de contrôle, voir le record, changer d'apparence et d'arrière-plan, et bien évidemment commencer une partie en faisant sauter l'oiseau.

GameLoop

GameLoop correspond à ce qu'il se passe pendant la partie, lorsque le joueur la commence et sort donc de la fonction "gameStart". On a donc l'oiseau qui tombe de plus en plus vite sauf lorsque l'utilisateur le fait sauter en appuyant sur "espace" ou "clic gauche" (selon la touche définie plus tôt). Le but est de passer entre les tuyaux en faisant sauter l'oiseau afin qu'il reste à la bonne hauteur. Le joueur a également la possibilité de mettre en pause le jeu en appuyant sur un bouton.

GameOver

GameOver correspond à la fin du jeu, lorsque l'oiseau touche un tuyau où l'un des bords de la fenêtre. Le joueur a la possibilité de recommencer la partie ou d'y mettre fin. On a également le score de la partie qui s'affiche, ainsi qu'un message qui apparaît lorsque l'on bat notre record.

Les fonctions gameStart, gameLoop et gameOver sont appelées en boucle, sauf interruption de l'utilisateur (si l'on appuie sur la croix, sur echap où si l'on décide de quitter à la fin de la partie.)

Le jeu, entièrement codé en python, repose très fortement sur la **POO** (Programmation Orientée Objet) étant donné l'utilisation massive de classes dans notre programme et sur la **modularité** (utilisation des modules pygame, os, sys, math, random et csv).

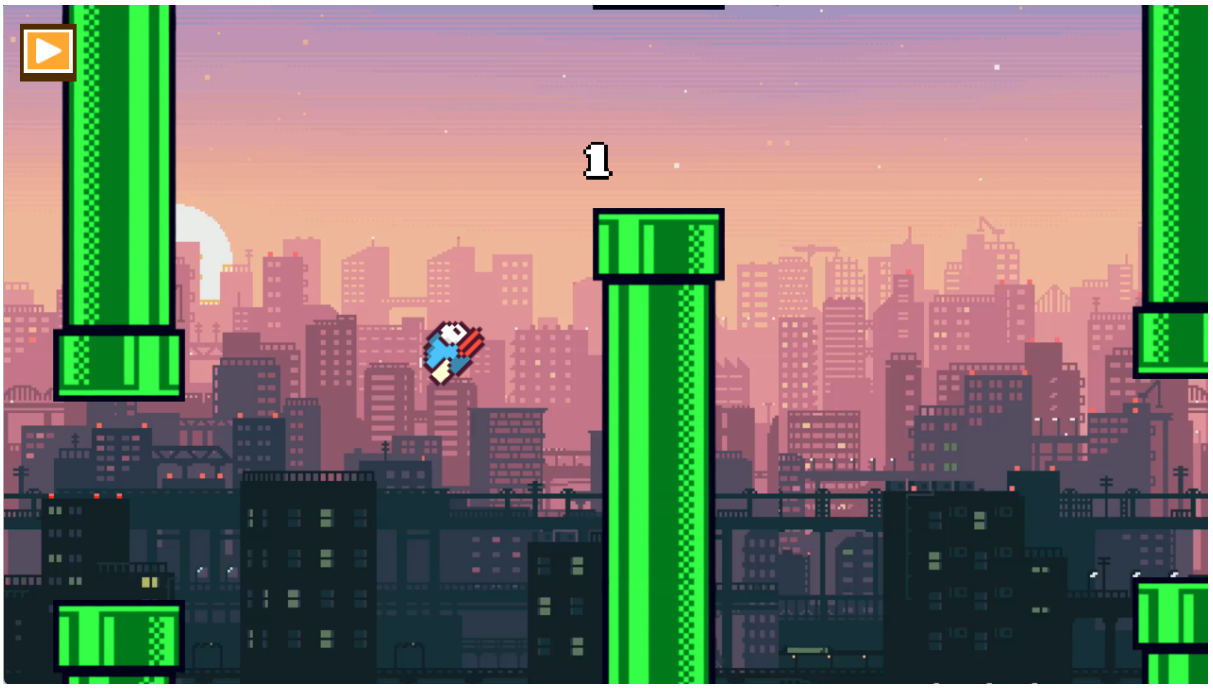
Le **stockage des données en local** se fait par le biais d'un fichier de type CSV

IMAGES DU JEU

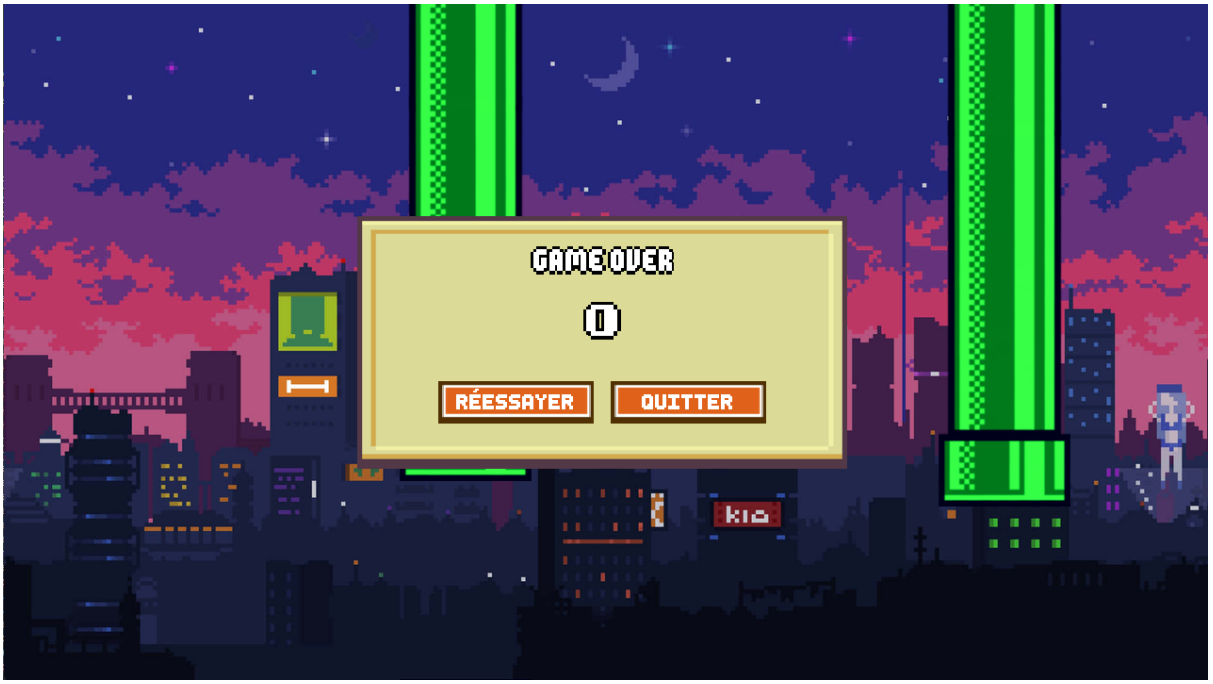
Menu principal (gameStart) :

Menu principal avec autre apparence / arrière plan :

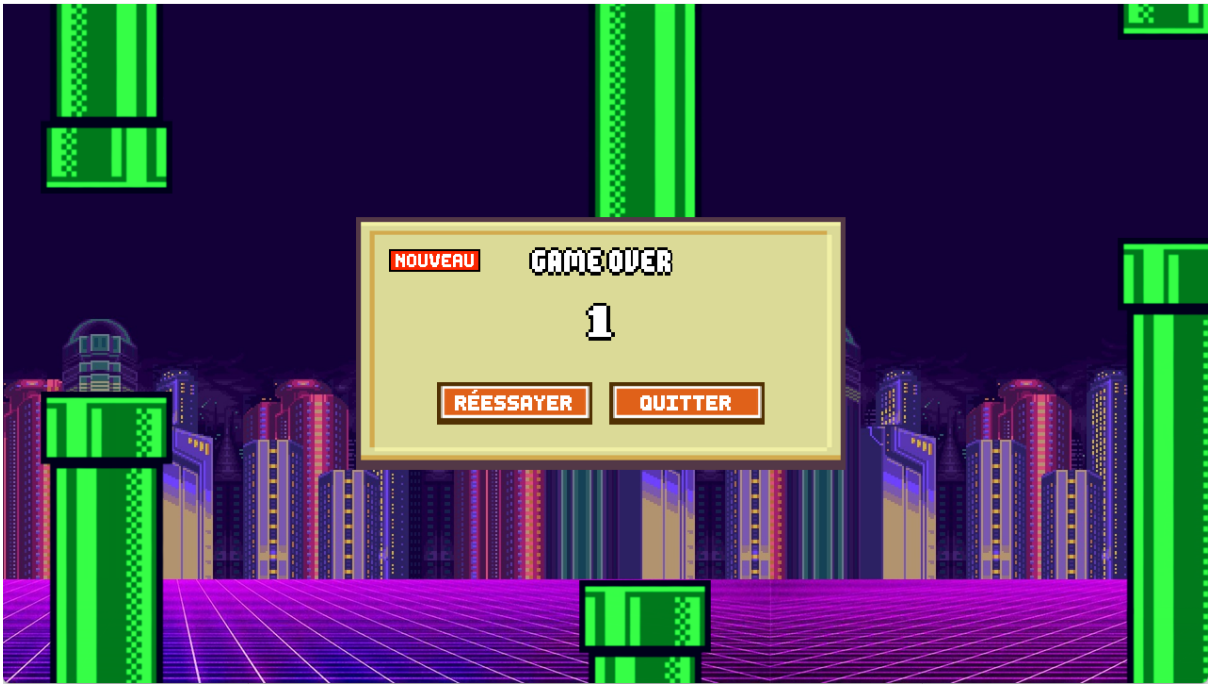
Déroulement d'une partie (gameLoop) :



Écran de fin de jeu (gameOver) :



Écran de fin de jeu avec nouveau record :



Tout début de la conception du jeu (le fond d'écran bleu représente l'arrière plan et le carré noir représente l'oiseau) :