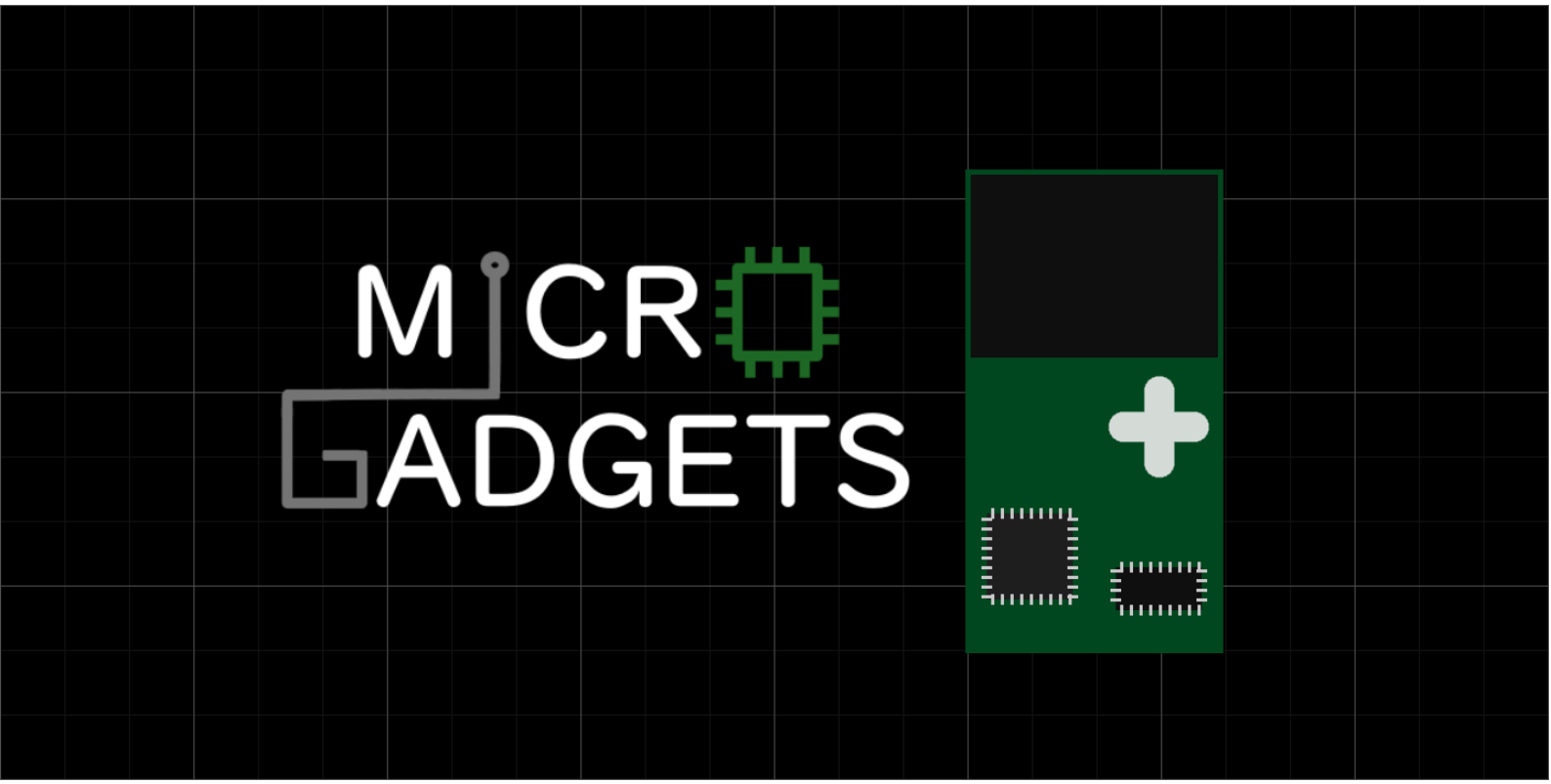
LES TROPHÉES NSI

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 : MICRO GADGETS

>PRÉSENTATION GÉNÉRALE :

L'idée de base de notre projet est de simuler, avec python, la fabrication de petites machines programmables.
Quand notre professeur nous a parlé des trophées NSI en classe, plusieurs d'entre nous étaient partants ; il ne nous manquait plus que l'idée, et quelqu'un venait justement de l'avoir. On la doit à Ahmad-Amine, un des membres de notre groupe, qui décida de sortir des sentiers battus et penser différemment : et si, au lieu de programmer un simple jeu, on programmait un simulateur d'appareils programmables? Une sorte de tout-en-un dans lequel on pourrait coder n'importe quoi, comme un Pong, un démineur ou un Snake ? C'est un concept assez large, certes, mais il permet d'offrir une expérience 100% personnalisable et des possibilités infinies. On peut citer comme inspiration principale *Retro Gadgets,* un environnement de création de "gadgets" disponible sur Windows, l'aspect limité de l'expérience développeur de PICO-8 a aussi été une source d'inspiration.
Ainsi, notre objectif pour ce simulateur était d'offrir à l'utilisateur la possibilité de créer tout et n'importe quoi ; l'utilisateur a une liberté créative totale et il définit lui-même son expérience, à la manière d'un jeu type bac-à-sable (autrement dit, si le jeu ne vous plaît pas, c'est de votre faute). Cette idée a charmé plusieurs élèves de la classe, qui ont décidé de former notre groupe et commencer à travailler ensemble sur ce nouveau projet.
L'interface du logiciel est simple : on peut ajouter des composants, les déplacer, les lier entre eux pour former des objets, puis ouvrir une console et coder. Ainsi, le logiciel est facilement accessible et son utilisation est intuitive, ce qui était un autre de nos objectifs.
L'intérêt de ce projet est de créer une première approche simple et sans prise de tête au monde du hardware, afin de cultiver le goût de l'électronique chez l'utilisateur qui s'y intéresse, mais qui ne sait pas vraiment comment s'y prendre.
Le projet s'intègre bien dans le cadre de la spécialité NSI car il met en lien les notions de *programmation orientée objet* et de *System On a Chip.*

> ORGANISATION DU TRAVAIL :

Notre groupe est composé de quatre membres:

Ahmad-Amine	Implémentation de l'IDE intégré, structuration du projet, intégration du pseudo-langage Python
Meriem	Graphismes et logo, conception du menu, montage vidéo
Axel	Système de gestion des langues/traduction, documentation
Étienne	Implémentation des composants et de tout ce qui est en lien avec leur manipulation

Nous avons créé un serveur Discord en ligne afin de pouvoir discuter du projet en dehors des heures de cours. C'est sur ce serveur que nous discutons des tâches à effectuer, de leur répartition et des éventuels changements ou améliorations que l'on pourrait faire. Il nous a permis de nous entraider: si l'un de nous avait des difficultés il pouvait demander de l'aide au reste du groupe. Aussi, si l'un de nous avait une nouvelle idée, il la partageait avec le reste du groupe et on discutait pour décider de l'implémenter ou non selon les capacités de chacun, le temps et les ressources disponibles.

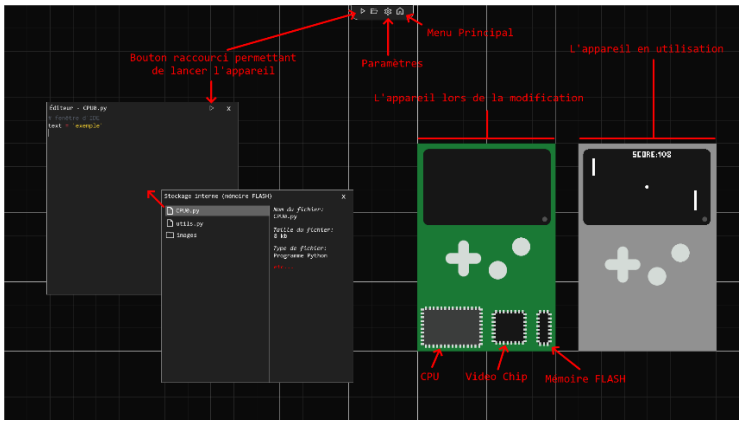
C'est aussi là-bas que nous reportions les bugs rencontrés et travaillions sur leur correction, à laquelle tous les membres du groupe ont participé.

Nous avons choisi de mettre en place un dépôt sur Github. Cela nous a permis de travailler sur le projet en même temps et mettre notre travail en commun plus facilement. Et nous ne le regrettons pas : tout développeur moderne qui se respecte doit savoir utiliser Github comme outil de développement et de communication.

LES ÉTAPES DU PROJET :

Une fois que le groupe fut rassemblé autour de l'idée d'Ahmad-Amine, nous nous sommes mis d'accord sur la structure et les grandes étapes de la conception de Micro Gadgets. Très vite, on s'est mis d'accord sur un prototype.

Maquette de départ :



La première chose à faire fut d'ajouter des panels permettant d'écrire du code, la fameuse console, puis des composants. Dans les stades de développement initial, les composants n'étaient que des formes géométriques basiques qu'on pouvait déplacer sur le plan de travail, mais pas lier entre elles.

En parlant de déplacement, on a ensuite ajouté une grille d'arrière-plan sur laquelle les composants s'alignaient ; tout de suite, le plan de travail était bien mieux organisé.

Un des premiers problèmes qu'on a rencontré était lié à Pygame, une des bibliothèques que nous avons utilisées. Ses anciennes versions faisaient planter le programme, on a donc créé un fichier scripts/version.py dans lequel on pouvait récupérer la version actuelle de Pygame et vérifier si la version actuelle était compatible.

On a également rencontré des difficultés quant à la gestion des langues dans notre code ; il nous fallait un moyen de stocker efficacement les textes anglais et français. C'est pour cela que nous avons abandonné le système initial qui utilisait le format .txt et opté pour le langage YAML, solution plus moderne et appropriée.

Une fois que le prototype fonctionnait bien, on a commencé à étoffer le projet en ajoutant de plus en plus de fonctionnalités: un menu, un système de sauvegarde, une traduction anglaise, des sons... Alors que le projet ressemblait de plus en plus à un vrai logiciel, certaines difficultés apparurent à ce moment pour nous freiner dans notre élan (bonjour le spaghetti code et les listes mutables à grande échelle). Mais à force de réflexion et d'entraide (et beaucoup de discussions sur Discord), la majorité des bugs furent corrigés rapidement.

À ce stade, le projet avançait à un bon rythme et la plupart des bugs étaient corrigés. Mais la date limite du dépôt du projet approchait à grands pas; on s'est donc attaqués à l'étape suivante, la documentation et la vidéo de présentation. Tout le groupe était impliqué et les compétences de chacun mises à l'œuvre. C'est aussi à ce moment qu'on a nommé le projet "Micro Gadgets", nom trouvé par Axel, et qu'on a conçu son logo. Il s'agissait d'avoir un moyen de désigner et illustrer

le projet, afin de faciliter sa promotion. Jusqu'ici, on l'appelait "le simulateur pour le concours de NSI là", ce qui devint vite un peu long à dire.
Enfin, nous étoffâmes le projet en intégrant au simulateur un appareil déjà préfait, permettant de jouer à Snake. Cet appareil servira à la fois de démonstration pour le jury et d'exemple pour les développeurs (et peut être les jurés) qui oseront se plonger dans la création d'appareils avec Micro Gadgets.

> FONCTIONNEMENT ET OPÉRATIONNALITÉ :

Le projet est bien avancé. On a réussi à y mettre la grande majorité des fonctionnalités qu'on voulait implémenter. Afin de contrôler la présence de bugs, nous jouions régulièrement le rôle de testeurs; à chaque fois que l'un d'entre nous ajoutait de nouvelles fonctions ou modifiait une partie du code, les autres contrôlaient son travail en testant les nouvelles fonctionnalités et reportaient les bugs rencontrés.
Pour le débuggage, nous avons créé un outil développeur (pouvant être activé en appuyant sur F3), nous permettant d'afficher un graphe de performance absolument nécessaire pour qu'Ahmad-Amine optimise la console, qui était sans doute la chose la plus difficile à implémenter, car elle nécessitait une reproduction totale dans Pygame des entrées de textes que l'on utilise quotidiennement.
Ce projet nous a posé plus d'un problème lorsqu'il s'agissait d'implémenter du nouveau contenu (près de 5500 lignes de codes, c'est pas si facile à organiser). La console, la sauvegarde des appareils, la manipulation des composants, tous on été difficiles à implémenter car nécessitant une très bonne optimisation du code (il fallait oublier les algorithmes en $O(n^2)$...). Mais nous avons finalement réussi à les implémenter en passant du temps et en ne négligeant pas la communication. Le projet étant devenu très conséquent, nous avons plusieurs fois du refactoriser notre code pour éviter les dépendances circulaires. C'est pour cela aussi que nous avons fracturé le code en une soixantaine de fichiers différents pour augmenter la lisibilité et facilier la navigation dans le code.

> OUVERTURE :

Nous pourrions améliorer notre projet en ajoutant de meilleurs graphismes et développer l'environnement sonore. Limités par le temps, nous avons aussi dû limiter le nombre de composants et de machines ; à l'avenir, on pourrait donc ajouter de plus grandes quantités de composants et de machines préconstruites afin de créer une expérience plus réaliste, diversifiée et donc plus enrichissante pour l'utilisateur. Il serait préférable d'implémenter aussi la documentation technique au logiciel, étant donné qu'il requiert une documentation conséquente pour que l'utilisateur puisse développer ses propres appareils. Une autre idée que nous avions eu sans avoir le temps de l'implémenter était d'ajouter la possibilité de faire un retour arrière (Ctrl+Z) et avant (Ctrl+Y). Si le projet était à refaire, nous pourrions revoir notre répartition des tâches afin de travailler plus efficacement dès le début. En effet, le projet eut un départ assez lent, même si on a pu se rattraper par la suite. Ainsi, nous aurions peut-être pu mettre en œuvre les fonctionnalités citées plus haut.
En plus de parler de Micro Gadgets autour de nous, nous prévoyons de diffuser le projet sur des réseaux sociaux comme Discord sur des serveurs dédiés (communautés de développeurs Python), et Instagram grâce au compte de notre lycée.

DOCUMENTATION

Spécifications techniques

Notre projet est organisé en cinq dossiers. Le dossier *fonts* contient toutes les polices d'écriture utilisées dans le jeu, *lang* comporte le texte et ses traductions au format *.yaml* ainsi que la table de caractères utilisable. Dans le dossier *modules* se trouve les fichiers du module pyperclip. *script* contient tous les fichiers Python nécessaires au fonctionnement du jeu. *sprites* est le dossiers des images, et *data* contient les appareils sauvegardés.

Guide d'utilisation

Le projet est programmé en Python et utilise principalement la bibliothèque Pygame (www.pygame.org) qu'il faut préalablement installer. La bibliothèque yaml est aussi indispensable.

> pip install pygame
> pip install pyyaml

Pour lancer le projet il faut exécuter le ficher main.py qui se trouve dans le dossier sources du dossier technique ou exécuter le fichier .exe au même endroit. Lorsque l'on lance le programme on arrive sur le menu. Il contient cinq options : charger un appareil, nouvel appareil, options, crédits et quitter.
Les options permettent de changer de langue (le français et l'anglais sont disponible) et de disposition de clavier (les claviers AZERTY et QWERTY sont pris en charge).
Le bouton charger un appareil permet de charger les appareils sauvegardés par l'utilisateur ou l'appareil pré-fait nommé Snake.pkl, servant à la fois de démonstration pour le jury et d'exemple pour les développeurs.
Le bouton nouvel appareil envoie l'utilisateur sur un plan de travail vide, prêt à accueillir la nouvelle création du siècle.
Une fois sur le plan de travail, il y a quatre boutons de navigation du haut de l'écran :
- le premier en partant de la gauche permet de sauvegarder l'appareil ;
- le deuxième sert à ouvrir une fenêtre qui permet d'ajouter les composants ;
- le troisième sert à ouvrir un éditeur Python intégré pour programmer l'appareil (code stocké dans le CPU de ce dernier) ;
- le dernier permet d'accéder au menu.

Pour le jury

Nous vous conseillons de tester notre logiciel en chargant dans un premier tant l'appareil Snake.pkl. Ensuite, ouvrez l'éditeur de code (3e bouton en haut de l'écran) et appuyez sur le bouton play de la fenêtre de la console. Vous pourrez donc voir en direct ce à quoi peut ressembler un appareil fonctionnel fait sur Micro Gadgets.

Pour aller plus loin

Micro Gadgets étant un projet particulier car se basant sur l'utilisation d'un pseudo-language Python customisé dans un environnement inhabituel, nous avons jugé nécessaire de créer un guide d'utilisation du développeur pour comprendre plus en détail comment créer ses propres appareils. Vous pouvez accéder à ce guide en cliquant ici.