INSA INSTITUT NATIONAL DES SCIENCES APPLIQUÉES TOULOUSE

Belafki Ayoub
Pizzinat Dimitri
4AE-SE

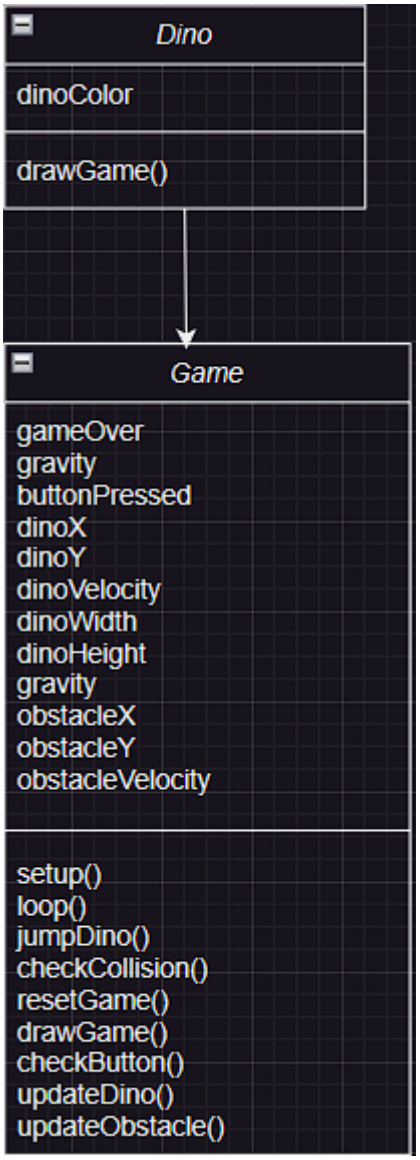
Rapport BE C++

Dino Game

Introduction:

Le projet consiste à développer une implémentation du jeu "Dino" sur un microcontrôleur compatible Arduino, spécifiquement avec un écran OLED pour l'affichage et un bouton physique pour les interactions. L'objectif principal était de mettre en œuvre nos compétences en programmation orientée objet et en développement de systèmes embarqués, tout en offrant une expérience de jeu attrayante. Ce jeu comprend un personnage (Dino) qui saute par-dessus des obstacles se déplaçant à vitesse constante. Le "Dino", aussi connu sous le nom de "T-Rex Runner", est un jeu de course sans fin très simple qui consiste à parcourir la plus grande distance possible en évitant les obstacles. Il est souvent vu dans le navigateur Google Chrome lorsque l'utilisateur est hors ligne.

Diagramme de classe :



Un seul objet principal, de type Game, est instancié dans le système. Il est initialisé avec une référence à un objet U8G2_SH1107_SEEED_128X128_1_SW_I2C, qui gère la communication avec l'écran OLED.

La classe Game comporte les attributs représentant les différentes entités du jeu (le dino, l'obstacle) ainsi que l'état du bouton et du jeu lui-même (par exemple, si le jeu est terminé). Elle comprend aussi plusieurs méthodes pour gérer la logique du jeu (comme la détection des collisions, la mise à jour de l'état du dino et de l'obstacle, la gestion de la fin du jeu, etc.)

Schéma de fonctionnement matériel et logiciel :

Le microcontrôleur est au cœur de ce système. Il communique avec l'écran OLED par le biais du protocole de communication I2C, qui nécessite deux lignes de données : SCL (horloge) et SDA (données). Les données affichées à l'écran sont générées par le microcontrôleur, qui exécute le programme du jeu Dino.

En plus de l'écran, un bouton physique est connecté au microcontrôleur. L'état de ce bouton est vérifié régulièrement par le microcontrôleur pour déterminer si l'utilisateur a demandé au Dino de sauter. Le bouton est connecté à une broche d'entrée numérique du microcontrôleur et est configuré pour déclencher une interruption lorsque son état change, ce qui permet d'économiser de l'énergie en évitant un balayage constant de l'état du bouton.

Le logiciel, écrit en C++, suit une architecture orientée objet. Le diagramme de classe détaillé ci-dessus illustre la structure du logiciel. La classe Game est le centre de la logique du jeu. Elle comporte divers attributs pour représenter l'état actuel du jeu, tels que la position et la vitesse du dino et de l'obstacle, l'état du bouton, et si le jeu est terminé. De plus, elle comporte plusieurs méthodes pour mettre à jour l'état du jeu, vérifier les collisions, gérer les sauts du dino, etc.

Le cycle de vie du logiciel est simple : lors de l'initialisation (setup()), les différents composants du jeu sont initialisés. Puis, dans une boucle infinie (loop()), l'état du bouton est vérifié, la logique du jeu est mise à jour, et l'état actuel du jeu est dessiné sur l'écran.

En résumé, le fonctionnement du système peut être représenté comme suit :

1. Initialisation des composants (setup())
2. Entrée dans la boucle principale (loop())
    - Lecture de l'état du bouton
    - Mise à jour de l'état du jeu (positions et vitesses du dino et de l'obstacle, détection des collisions, etc.)
    - Dessin de l'état du jeu sur l'écran OLED
3. Retour à l'étape 2
4. Cette boucle continue jusqu'à ce que le système soit éteint.

Perspectives d'évolution :

Le système est fonctionnel et a atteint l'objectif initial : il offre une version minimaliste mais jouable du jeu du dinosaure de Google Chrome sur un microcontrôleur. Cependant, plusieurs améliorations pourraient être envisagées pour le rendre plus attractif et augmenter sa complexité.

1. **Introduction de niveaux de difficulté** : La vitesse de l'obstacle pourrait augmenter progressivement, rendant le jeu plus difficile au fur et à mesure que le temps passe.

2. **Introduction de différents types d'obstacles** : Actuellement, le jeu n'a qu'un seul type d'obstacle. Il serait intéressant d'introduire différents types d'obstacles nécessitant des sauts de différentes hauteurs.

3. **Améliorations graphiques** : Bien que les graphiques soient limités par la résolution de l'écran OLED, il serait possible d'introduire une certaine animation du dinosaure lorsqu'il saute ou lorsqu'il y a une collision avec un obstacle.

4. **Introduction d'un système de score** : Actuellement, le jeu n'a pas de système de score. Il serait intéressant d'introduire un tel système, où le joueur gagne des points pour chaque obstacle qu'il évite.

Problématiques rencontrées :

- Gestion de l'affichage OLED avec la bibliothèque U8g2lib

La gestion de l'affichage OLED par le biais de la bibliothèque U8g2lib s'est révélée être une tâche délicate. Bien que cette bibliothèque soit puissante et flexible, sa complexité a entravé la progression initiale du projet. Comprendre la manipulation des objets graphiques et leur mise à jour sur l'écran a nécessité un effort significatif d'apprentissage et de débogage.

- Logique de détection de collision

Le développement de la logique de collision dans le jeu a également été une difficulté majeure. En raison de l'importance de cette fonctionnalité pour le fonctionnement global du jeu, nous avons dû consacrer beaucoup de temps à l'affinage de notre approche pour obtenir une détection précise et fiable. Cette précision était nécessaire pour assurer l'équité du jeu et une expérience de jeu agréable pour l'utilisateur.

- Gestion des entrées du bouton

La gestion des entrées du bouton pour faire sauter le Dino a été une autre problématique. Assurer la détection précise de chaque pression du bouton et que le Dino réagisse en conséquence a requis un réglage fin. Nous avons rencontré des défis pour parvenir à une synchronisation précise entre la pression du bouton et l'action dans le jeu, en assurant à la fois la réactivité du jeu et l'élimination des faux positifs.

Chacune de ces problématiques a exigé des efforts considérables pour être résolue, et bien qu'elles aient été difficiles à surmonter, chaque solution nous a rapprochés de notre objectif : un jeu de Dino fonctionnel et amusant à jouer.

Conclusion :

Dans l'ensemble, ce projet a été une excellente occasion de mettre en pratique et d'approfondir les connaissances en programmation C++, en programmation orientée objet et en développement de systèmes embarqués. Il a permis de comprendre et de surmonter les défis associés à l'optimisation de la mémoire, à la gestion des interruptions et à l'interaction entre le matériel et le logiciel dans un système embarqué. Malgré les difficultés rencontrées, l'objectif a été atteint et le jeu fonctionne comme prévu. Avec les perspectives d'évolution proposées, ce projet a encore un grand potentiel d'amélioration et pourrait être une base intéressante pour de futurs travaux dans le domaine des systèmes embarqués.