Rapport de Projet C++

Lucas LE BRAZIDEC - Laura MENDES

Mai 2022

1 Introduction

Notre projet de programmation C++ s'est tourné sur la réalisation des concepts de base d'un système d'entrainement au laser game. La problématique a ici été de faire fonctionner une chaine de capteurs et actionneurs à travers des classes. Cela permet alors de simplifier la mise en marche et l'usage de ces derniers.

2 Présentation de notre système

Le système de laser game réalisé est composé de plusieurs éléments : - une diode infrarouge, - un capteur infrarouge, - un bouton poussoir, - une diode verte, - une diode rouge.

Le bouton poussoir est alors ce qui actionne ou non la diode infrarouge. Dans le cas où cette diode est actionnée et mis en direction du detecteur infrarouge, la diode verte s'allume pour signaler que la cible est atteinte. Sinon, dans le cas où la diode n'est pas bien en direction du capteur, la diode rouge s'allume. Ce système permet alors de vérifier le fonctionnement de détection et de donner une idée au joueur de si il a échoué ou non.

3 La conception orientée objet : Diagramme de classe

Le projet est composé d'une classe `Composant` qui permet de gérer l'affectation de pin à tous les composants et les configurer. De cette classe hérite la classe `Digital` qui a pour fonction de gérer les composants discrets uniquement tels que les trois diodes et le bouton poussoir. Cette classe complète sa classe mère car elle permet de configurer en `INPUT` ou `OUTPUT` les pin sur lesquelles sont les composants et de lire la valeur du niveau dans le cas où il s'agit d'une entrée `INPUT`. Enfin, la classe `Diode` ne concerne que les trois diodes. Elle permet alors de régler le niveau de la diode en plus des fonctions dont elle hérite des autres classes.

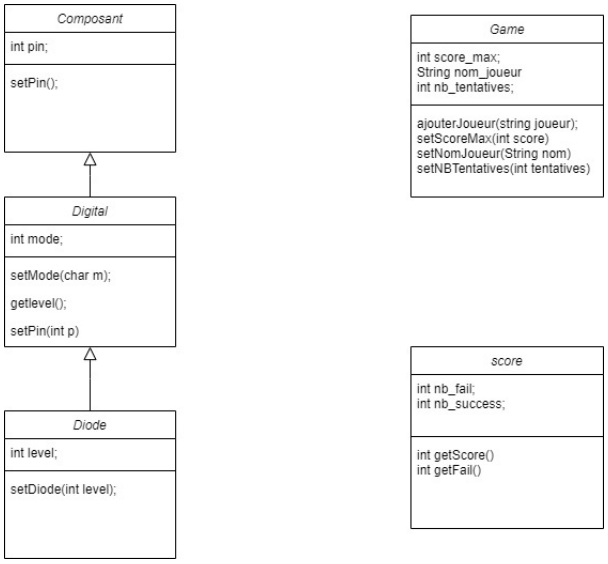

Figure 1 – Diagramme de classe de l'ensemble du projet

Nous avons également créé la classe `Game` qui permet de gérer la création d'une partie en configurant : le nom du joueur, le nombre de tentatives et le nombre de succès à atteindre. Enfin, la classe `Score` permet de gérer l'évolution du nombre de succès et d'échecs d'un joueur pendant une de ses tentatives.

4 Fonctionnement du système

4.1 Lancement de la partie

Avant que la partie ne commence, le joueur doit entrer son nom, le nombre max de succès à atteindre pour mettre fin à la partie et son nombre de tentatives. Ces informations sont stockées dans une des constantes globales. Un tableau est alors créé pour contenir pour chaque tentative, le nombre de fois qu'il a tiré à côté et le nombre de fois qu'il a atteint la cible.

4.2 Déroulement de la partie

Une fois que la partie est lancée :
— le joueur a un score de succès qui augmente quand il touche la cible,
— le joueur a son nombre d'échecs qui augmente quand il tire à côté.

4.3 Fin de partie

Une fois la partie terminée, le système communique le score du joueur à chacune de ses tentatives.

5 Difficultés rencontrées et pistes d'améliorations

5.1 Fonctionnement de la librairie STL sur Arduino

[1] Pour répondre au cahier des charges, nous voulions faire un objet `Map` pour contenir :
— le nom du joueur,
— son score dans la partie.
Il aurait alors été demandé aux joueurs d'entrer leurs noms au début de la partie et faire évoluer leurs scores associés au cours de la partie. Cependant, nous n'avons pas réussi à importer une librairie compatible avec ESP8266. Ainsi, nous avons simplifié le problème en ne prennant en compte qu'un joueur. Cela nous a également permis de faire un projet cohérent avec le nombre de capteurs à notre disposition.

5.2 Choix des paramètres de création de game

Pour créer une game il est nécessaire de donner :
— le score max à atteindre,
— le nom du joueur,
— le nombre de tentatives autorisées.
Nous aurions voulu communiquer avec le joueur pour lui demander d'entrer ces informations. Néanmoins, nous n'avions pas beaucoup de compétences en Arduino et préféré simplifier le problème en mettant en constantes ces informations. Il serait alors possible d'améliorer le projet en proposant une communication entre l'écran et le clavier.

5.3 Création d'une exception

[1] Afin de détecter un disfonctionnement du capteur IR, nous souhaitions identifier les valeurs aberrantes(supérieures à 1000 ou inférieures à 600).
Une exception a ainsi été codée dans la fonction `Touche(int pin)`. Lors de la compilation l'IDE nous informe de l'interdiction de faire des exceptions. Nous n'avons pas su résoudre ce problème à temps. Il serait intéressant de travailler dessus afin d'améliorer le projet.

6 Conclusion

Ce projet nous a permis de mettre en oeuvre nos nouvelles compétences en C++ et de découvrir le monde ARDUINO. L'utilisation de classes nous a apporté une méthode simplifiée et modulable pour réaliser un projet.
Malgré tout, nous avons rencontré des problèmes de compatibilité avec les librairies C++(cf 5.1) et de configuration (cf 5.3) nous empêchant d'atteindre l'objectif souhaité.