# Programmation Orientée Objet

# -

# Bureau d’étude

## Mawdulo, l’instrument de musique numérique modulaire




OMARI Larbi - SANCHEZ Vittorio 4AE-SE3 2019-2020

# Introduction

L'objectif de ce bureau d'étude est de développer une bibliothèque modulaire extensible qui s'interface avec des capteurs et des actionneurs afin de créer un nouveau service. Afin de réaliser ce projet nous avons utilisé le langage C++. Néanmoins à cause du confinement et de l'impossibilité d'accéder au matériel requis, ce projet a été réalisé en simulation.
A partir d'une base de simulateur fournie par nos enseignants que nous avons enrichie, nous avons développé un programme arduino qui se rapproche le plus possible de la réalité. Pour ce projet, nous avons choisi de réaliser un instrument de musique modulaire que chacun pourrait enrichir selon ses besoins et ses envies.

# Concept et fonctionnement de l'instrument

L'idée de concevoir cet instrument nous est venu du thérémine, instrumet de musique numérique inventé en 1920 par Mr Leon Theremin. L'instrument que nous avons conçu dépend de la personne qui va l'utiliser, de ses gouts musicaux et de sa créavité. En effet, l'originalité de cet instrument réside dans le fait qu'il est modulaire c'est à dire que l'on peut rajouter autant de "modules musicaux" que l'on veut. Un module musical est composé d'un ou plusieurs sons et d'un actionneur et/ou capteur. Afin de jouer des sons, nous avons utilisé la librairie SFML. Pour notre application, nous avons choisi 4 modules : 2 modules "ButtonSound", 1 module "AnalogSensorUltrasoundSoundDevice" et 1 module "AnalogSensorLuminositySoundDevice".

# Fonctionnement des modules choisis

## ButtonSound

Le principe de ce module est d'appuyer sur un bouton afin de lancer un son choisi par l'utilisateur au préalable. Tant que le bouton est allumé le son est joué. Nous avons choisi pour ce modules 2 sons de batterie.

## AnalogSensorUltrasoundSoundDevice

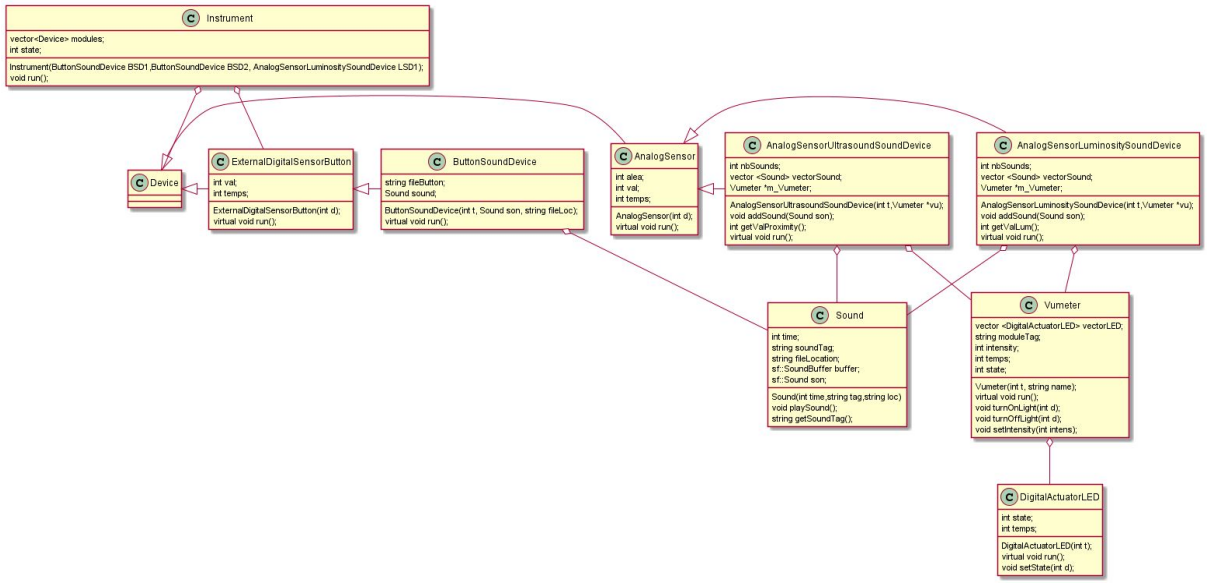
L'utilisateur va choisir différents sons qu'il va intégrer à ce module. Selon la distance détectée par le capteur, un son différent va être joué. Ce module intègre aussi un vumètre afin que l'utilisateur puisse savoir quel son il est en train de jouer. Nous avons choisi pour ce modules 5 sons de synthétiseur.

## AnalogSensorLuminositySoundDevice

Ce module possède le même fonctionnement que le module précédent mais au lieu d'être basé sur la distance du membre de l'utilisateur, il est basé sur la luminosité que le capteur va recevoir. Il possède aussi un vumètre intégré composé de LED. Nous avons choisi pour ce modules 5 sons de guitare. Une autre différence avec la classe précédente est le fait que si l'utilisateur éloigne sa main du capteur (comme s'il ne voulait pas jouer de note) ce module jouera tout de même un son alors que sur le précédent non. Cela permet de l'utiliser comme module "d'ambiance".

L'utilisateur pourrait utiliser d'autre modules basés sur d'autres capteurs s'il en a envie et rajouter autant de modules qu'il le souhaite.

# Diagramme de classe




# Implémentation en simulation

Ne pouvant pas implémenter notre instrument en réel, nous devions le faire en simulation. Nous avons créé des classes de modules musicaux qui héritent des classes des différents capteurs/actionneurs. Afin de simuler les capteurs, nous avons dû passer par la manipulation de fichiers texte. La valeur contenue dans le fichier représente la distance entret la main et le capteur, la valeur de luminosité detectée par le capteur. Pour le bouton, l'état du bouton change par la création d'un fichier approprié. Nous avons créé un script de test qui automatise la gestion de ces fichiers (création et destruction, modification du contenu) afin de faire une démo.

Pour s'assurer du bon fonctionnement de l'instrument, nous avons sur la console l'affichage de l'état des différents éléments à intervalles réguliers: le nombre de LEDs allummées pour les vumètres, les élements qui jouent un son et le son en question. Aussi, les sons sont joués en direct grâce à l'utilisation de la librairie SFML.

# Conclusion

Lors de ce bureau d'étude nous avons pu réaliser une base fonctionnelle de notre instrument en simulation. Nous avons néanmoins eu quelques difficultés, notamment avec la librairie SFML qui nous permet de lancer les fichiers audios. Pour une raison inconnue, les sons son parfois joués et parfois non.

La suite logique serait de pouvoir l'implémenter sur une carte arduino avec les capteurs et actionnerus correspondants. Au niveau de l'implémentation réelle, il y aurait des modifications au niveau du code à faire, en fonction des protocoles utilisés par les capteurs. Par exemple, le capteur ultrason de ref. HC-SR04 souvent utilisé avec les cartes arduino, a un pin qui est HIGH pendant un temps donné qui correspond à la distance mesurée. Dans ce cas il y a donc un traitement supplémentaire à réaliser pour extraire les données que nous souhaitons récupérer.

Au niveau de l'instrument en lui-même, il y a plein d'améliorations possibles. Tout d'abord il est tout à fait imaginable d'étendre notre bibliothèque de classes et fonctions pour accueillir d'autres capteurs ou d'autres comportements des capteurs existants. En effet, les capteurs que nous utilisons reposent sur le fait de lancer un son en fonction d'une valeur donnée sur une plage de valeurs (luminosité, proximité) mais on pourrait très bien imaginer utiliser les capteurs pour moduler le volume, le pitch, ajouter des effets ou pour tout autre comportoment qui nous intéresserait. Enfin, au niveau de l'interface homme-machine, il serait intéressant d'ajouter une interface graphique afin de pouvoir augmenter la modularité de l'instrument, en pouvant choisir les sons et à quels capteurs les lier, configurer les entrées/sorties, et avoir un retour visuel supplémentaire qui complémenterai les vumètres.