INSA INSTITUT NATIONAL DES SCIENCES APPLIQUÉES TOULOUSE

Compte-rendu de BE C++ : Le Sheet-Writer

Réalisé par
Amélie MAIER
Arthur ZAPPA

Dans le cadre du cours
BE C++

Travail présenté à
Raphaël Deau

Table des matières

1 Introduction et Présentation du Sheet Writer

Ce Bureau d'étude de C++ a plusieurs objectifs comme réfléchir à une conception orientée objet, utiliser le langage C++, imaginer un nouveau service dans le monde des objets connectés et de l'Internet des Objets. Notre imagination nous a amené vers l'idée d'un Song Writer. Il s'agit d'un petit système qui, en écoutant une mélodie simple jouée sur un piano ou une guitare, est capable d'en écrire la partition et de l'afficher sur une application web. Pour ce projet, nous avons utilisé un microcontrôleur ESP8266 programmable en arduino. Nous avons choisi d'ajouter un Sound Sensor v1.6 Grove (entrée analogique) qui nous permet de capter les ondes sonores envoyées par l'instrument. Ensuite, l'ESP8266 effectue une Transformée de Fourier pour reconnaître la note jouée et ainsi l'afficher sur un écran LCD Grove 3,3V (sortie numérique). La carte ESP8266 embarque une carte Wifi permettant de se connecter à un réseau Wifi et donc d'utiliser une application web.

2 Conception du système et du logigiel

2.1 Diagramme UML

Voyons d'abord la conception de notre diagramme UML composé de 5 classes :

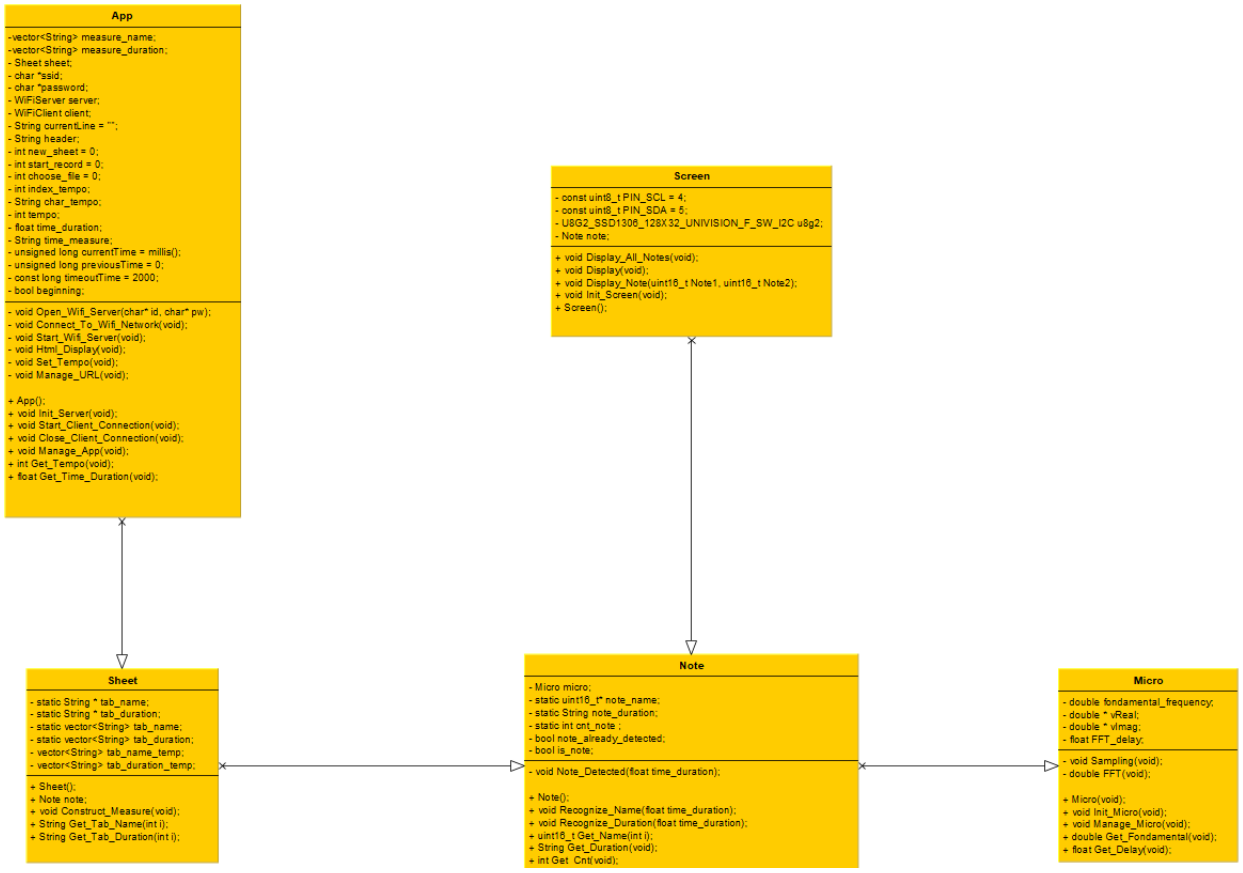

FIGURE 1 – Diagramme UML réalisé sous yEd

La classe Micro permet de récupérer les données analogiques envoyées par l'instrument, de les échantilloner avec la fonction *Sampling* puis d'appliquer une transformée de Fourier afin d'en déduire la fréquence fondamentale de chaque note jouée (*Get_Fondamental*). Nous avons utilisé la bibiothèque ArduinoFFT pour effectuer facilement la transformée de Fourier. Toutes les données de cette classe sont utilisées dans la classe Note. qui crée une note lorsque la fréquence est bonne.

La classe Note utilise donc les données de la classe Micro pour créer une note lorsque la fréquence fondamentale renvoyée par la FFT correspond à une note connue avec *Recognize_Name*. La durée de la note est aussi reconnue grâce à *Recognize_Duration*. Elle envoie également les notes à la classe Sheet afin de remplir chaque mesure par des notes et à la classe Screen pour l'affichage des notes sur l'écran LCD.

La classe Screen récupère les données la classe Note et affiche (sur l'écran LCD) le nom de chaque note récupérée par le micro grâce à la fonction *Display*. L'importation d'une librairie a été nécessaire pour gérer facilement le transfert de donnée et l'affichage sur l'écran LCD.

Ensuite, la classe Sheet permet de faire le lien entre la classe Note et la classe App. Elle regroupe les notes créées et les envoie à App pour lui permettre de remplir les mesures à afficher. La fonction principale est *Construct_Measure* qui permet de remplir un tableau de note, correspondant à une ou plusieurs mesures, qui seront envoyées à App lors du remplissage des mesures à afficher.

Enfin, la classe App s'occupe de l'aplication web, de la connexion wifi, et de l'affichage de la partition. C'est dans cette classe qu'est créé le serveur wifi. On initie d'abord le serveur grâce à *Init_Server* qui crée le serveur et se connecte au réseau Wifi. La connexion au serveur est démarrée lorsqu'un nouveau client fait une requête grâce à *Start_Client_Connection* puis fermée par *Close_Client_Connection*. Lorsque le client est connecté, on appelle *Manage_App* qui récupère les données envoyées par le client et s'occupe de l'affichage HTML et donc de la partition. La bibliothèque Java Script que nous utilisons pour l'affichage de la partition permet d'écrire la partition mesure par mesure. Nous avons eu besoin d'inclure Sheet.h pour pouvoir récupérer les mesures à afficher.

2.2 Fonctionnement du fichier main

Regardons maintenant comment fonctionne notre fichier main. Dans le *void setup()*, nous ouvrons un serial monitor afin de pouvoir afficher différentes informations nécessaires au fonctionnement du système. Ensuite, *app.Init_Server*() initialise le serveur web et gère la connection au réseau Wifi. Il y a aussi l'initialisation de l'écran LCD qui est faite.

Dans le *void loop()* nous appelons toutes les fonctions faisant fonctionner le système. On commence par attendre que l'utilisateur renvoie un tempo via l'application. Ensuite, de façon à ce que l'application est l'utilisateur soit bien synchrinisés, on attend qu'une note de plus forte amplitude que bruit ambient soit jouée. Les fonctions sont ensuite appelées dans l'ordre : on reconnait d'abord les notes, ensuite on affiche ces notes sur l'écran LCD et on construit la mesure à afficher sur la page web. Enfin, on appelle les fonctions de gestion de la connexion au serveur web.

3 Comment faire fonctionner notre Sheet Writer ?

Voyons comment faire fonctionner notre Sheet Writer. Tout d'abord, il faut connecter l'ESP8266 au réseau Wifi. Pour cela, il faut absolument rentrer le nom du réseau ainsi que le code d'accès dans le fichier App.cpp (voir les deux define $WIFI\_LOG$ et $PASSWORD$). Il faut ensuite flasher la carte avec le code et attendre la connexion au réseau Wifi. Il est important de ne pas faire trop de bruit pendant tout la phase de démarrage afin de ne pas déclencher le micro.

```
Connecting to Emmintal
.......
WiFi connected.
IP address:
192.168.1.54
```

FIGURE 2 – Affichage sur le moniteur Arduino lors de la connexion au réseau Wifi

Une fois connecté, l'adresse IP est affichée sur le monitor de Arduino. On peut ensuite accéder au site web en entrant directement l'adresse IP dans le navigateur de l'ordinateur qui doit être connecté au même réseau Wifi. On arrive alors sur la page d'accueil qui nous propose deux options :

- "Open a record" avec un Bouton OPEN qui permet d'ouvrir une partition déjà enregistrée sur l'ordinateur sous le format html
- "Create a new record" et donc une nouvelle partition avec un Bouton START

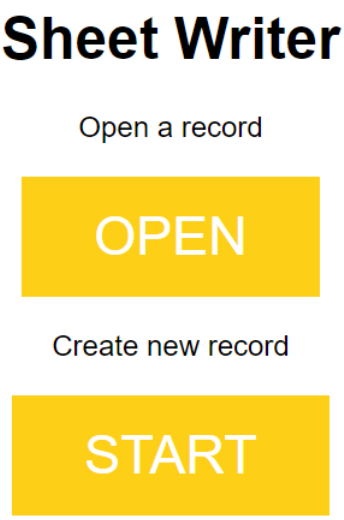

FIGURE 3 – Page web d'accueil

Lorsque que l'on choisit de créer une nouvelle partition, il est indispensable de rentrer le tempo de la partition à créer puis d'appuyer sur la touche ENTER du clavier. On peut désormais commencer l'écriture de la partition en jouant les premières notes. Ces notes se doivent d'appartenir à la gamme numéro 4 d'un piano.

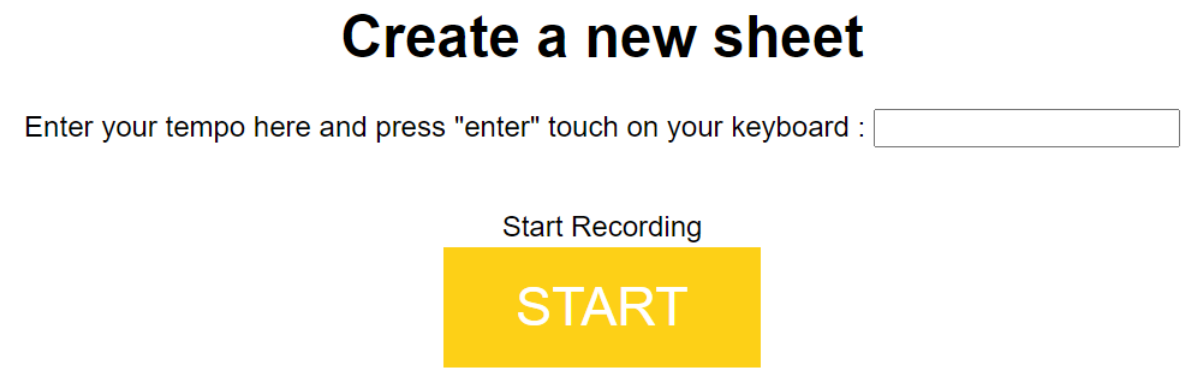

FIGURE 4 – Page web de création d'une nouvelle partition

Pour basculer sur l'écran d'affichage des mesures, il suffit d'appuyer sur START. On arrive sur une nouvelle page qui se rafraichit automatiquement et qui affiche la ou les mesures jouées.

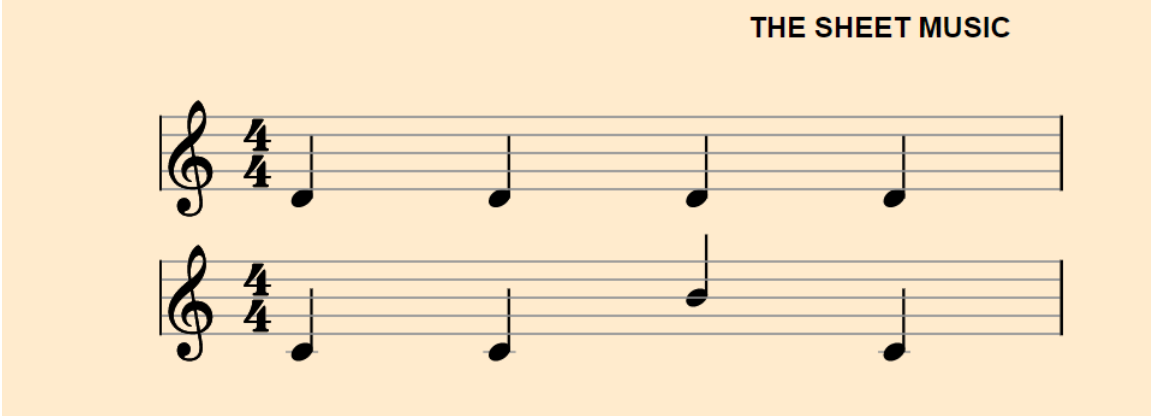

FIGURE 5 – Affichage de la partition sur la page web

4 Conclusion et Pistes d'amélioration

Pour conclure, ce BE de C++ nous a permis d'appliquer l'ensemble des notions acquises lors des cours théoriques. Travailler sur un système complet en ayant une liberté totale dans le choix du sujet est un projet très intéressant.

Malheureusement, plusieurs fonctionnalités que nous souhaitions implémenter n'ont pas fonctionnées. Effectivement, l'objectif était de reconnaitre le nom des notes (C, D, E..) mais aussi leur durée (croche, blanche, noire,...). Cependant, cette dernière fonctionnalité suppose que l'on puisse déterminer avec précision quelle note est jouée durant un temps donné. Or, ceci nécessite qu'une fonction écoute en permanence la sortie du micro, ce qui n'est pas possible dans notre programme comme nous l'avons implémenté. Effectivement, d'autres taches s'effectuent séquentiellement, dont $Manage\_App$ qui est particulièrement longue à s'exécuter, surtout quand la connexion Wifi est mauvaise. La reconnaissance de la durée des notes nécessite donc de mettre en place du multitâche. De cette façon, plusieurs tâches pourraient s'exécuter en même temps, mais nous n'avons malheuresement pas eu le temps de l'implémenter. Nous avons tout de même tenté de limiter la durée de chaque tâche, en évitant les *delay* qui sont bloquant et en les remplaçant par la fonction *millis()* qui elle n'est pas bloquante. D'autre part, afin d'améliorer notre système et après avoir régler le problème de multi-tache, nous pourrions essayer de reconnaitre davantages de notes (autres gammes, dièze, ...). Il serait aussi possible d'ajouter un bouton poussoir permettant de taper le tempo souhaité à la place de le rentrer manuellement lors de la phase de démarrage. On pourrait également par exemple ajouter une LED clignotant au rythme du tempo.

Ce BE nous a tout de même permis d'apprendre des notions de code html, ce qui nous a pris beaucoup de temps lors du démarrage. Nous avons aussi pu appliquer l'ensemble des notions apprises pendant les cours, les TDs et les TPs de C++. Porter le projet dans sa totalité, à partir de sa naissance et jusqu'à sa réalisation, en passant par la phase de conception et la programmation fut très formateur pour nous.