25/02/2018

NEU Robin - REBILLARD Maxime - LABARRE Julien ROUSSE Thomas

IBO 2

PoC Design Report

Outil d’apprentissage de piano

Table des matières

[Introduction 2](#_Toc507342619)

[Matériel 3](#_Toc507342620)

[Schéma fonctionnel 4](#_Toc507342621)

[Feature 1 : Leds, capteurs et Connexion I²C 5](#_Toc507342622)

[Feature 2 : Web Server / Gestion de fichiers MIDI 6](#_Toc507342623)

[Feature 3 : Front end / Site Web / Interface utilisateur 7](#_Toc507342624)

[Feature 4: Précision de la session / Scoring / Précision de la session 8](#_Toc507342625)

# GIT HUB

https://github.com/trousse/pianoMidi2.git

# Introduction

Le but de ce projet est de créer un outil qui permet à un utilisateur de pouvoir apprendre des morceaux de piano grâce à un système de lumière. Le concept est simple : des LEDS s’allument et s’éteignent pour signaler à l’utilisateur de jouer une touche dans le bon timing. Ce projet s’articule autour de plusieurs fonctionnalités. La première est la possibilité de pouvoir jouer les notes sur des capteurs (de lumière, de pression, de boutons…), et ce dans le bon rythme : cela signifie que l’utilisateur doit pouvoir anticiper ces notes à jouer. Pour l’aider, il y aura 2 rangées de LEDS au-dessus de la touche à jouer, le but étant de lui donner une estimation du moment ou il doit appuyer sur la touche.

L’utilisateur pourra jouer des morceaux directement récupérés sur une interface web qui servira de stockage de ces morceaux (en format MIDI, ou autre). Ce morceau sélectionné sera ensuite converti dans un format qui pourra permettre la traduction en signaux luminaux, au fur et à mesure que l’on avance dans le morceau. Un son sera émis dès lors que l’utilisateur activera une touche, via un système de fichiers sons mp3 lié aux touches.

L’historique des touches jouées par l’utilisateur sera enregistré, afin de pouvoir comparer la précision avec celle du morceau choisi. Ce score sera calculé par comparaison du fichier récupéré sur le web server et du fichier enregistré par l’utilisateur, avec ses touches, et il sera ensuite affiché sur l’interface web avec le résumé du nombre de faute, du pourcentage de précision, et un historique des morceaux joués…

# Matériel

\_Carte Raspberry Pi

\_Ecran/ordinateur (pour accéder au web server et sélectionner le morceau à jouer)

\_Carte Arduino UNO (pour connecter tous les composants : les 12 LEDS, les 6 Capteurs, les résistances…)

\_Câbles

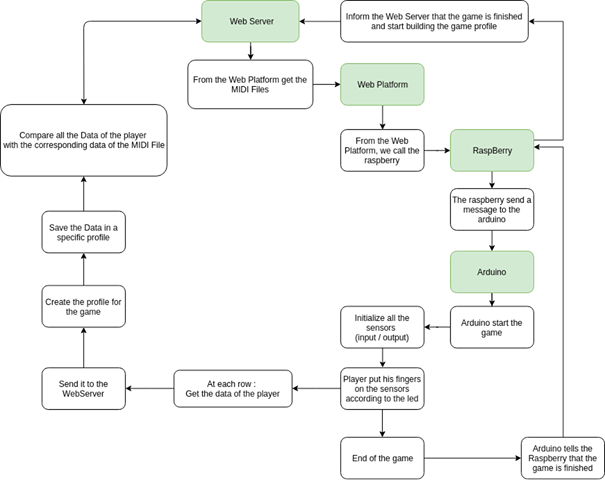
\_12 LEDS (Signaux visuels pour l’utilisateur)

\_18 résistances (6 de 220 Ohms, 6 de 10kOhms, 6 de 470 Ohms -- Les leds/résistances changeront sans doute à cause de la disponibilité)

\_6 capteurs (pression, lumière, ou boutons, dans cet ordre pour la disponibilité afin de pouvoir jouer des notes)

(\_Alimentation extérieure, 5V de sortie, et suffisamment pour alimenter tout le système)

# Schéma fonctionnel



# Feature 1 : Leds, capteurs et Connexion I²C

Sur le schéma ci-dessus, nous avons le montage tel qu’il devra être pour faire fonctionner les 6 capteurs (ici les photorésistances) que l’utilisateur devra actionner en rythme avec les LEDS juste au-dessus. Les deux rangées de LEDS fonctionnent en rythme, la première rangée, en rouge est la première à s’activer, puis clignote une fois. La deuxième clignote après un intervalle de temps et l’utilisateur doit appuyer sur le capteur avec le même intervalle. On laisse à l’utilisateur 2 temps afin qu’il puisse anticiper la frappe. Les différentes photorésistances correspondent à une note différente.

Ce qu’à joué l’utilisateur sera enregistré puis transmis via une connexion I2C à la Raspberry afin de l’écrire dans un fichier pour la future comparaison.

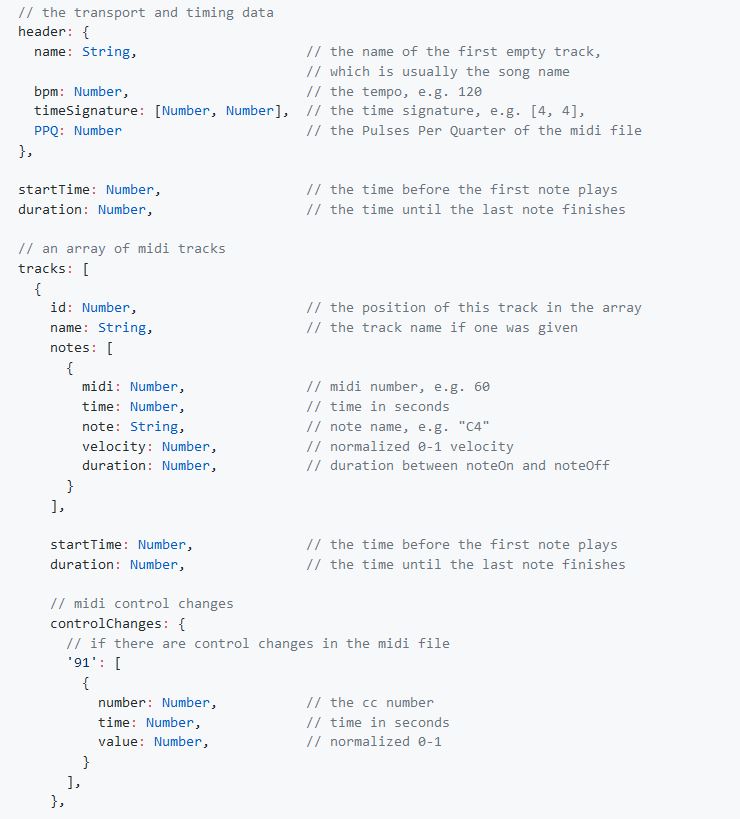
La Raspberry envoie les notes à l’Arduino, et l’Arduino renvoie le résultat de l’utilisateur à la Raspberry.

Quand l’utilisateur actionnera un capteur, le son correspondant sera joué, afin de pouvoir garder une trace auditive.

L’algorithme devra être capable de dire quoi faire à l’Arduino, et la Raspberry devra interpréter le fichier MIDI, et transmettre ces informations, tout en écrivant le fichier de l’utilisateur au fur et à mesure de sa session.

# Feature 2 : Web Server / Gestion de fichiers MIDI

Les fichiers MIDI (les fichiers sons) seront stockés sur un Web Server à partir duquel il sera facile de sélectionner celui voulu. Le but sera de traduire ce fichier midi en fichier JSON qui contiendra toutes les informations du morceau. Les notes, le tempo, la vélocité (la force de pression sur le capteur). Pour cela nous pourrons utiliser la bibliothèque [Midi Json Parser](https://www.npmjs.com/package/midi-json-parser) ou bien [Tone.js](https://tonejs.github.io/).

Ces bibliothèques nous serviront au moment ou un morceau sera sélectionné pour interpréter les données MIDI en Json selon ce format :

Il devient alors plus facile de créer des informations à afficher sur les LEDS. En lisant ce fichier, l’algorithme sur la Raspberry fournira les informations à l’Arduino.

La Raspberry devra être capable d’écrire un JSON dans lequel toutes les informations seront mises quant à la partie jouée par l’utilisateur.

Resultat :

le web server utilise @pioug/MidiConvert pour parser le fichier midi une fois parser le json est transformé en un tableau a 2 dimensions [x][y] x étant l’unité de temp I et y les led allumé. Les fonctions permettant de parser ce situe dans le fichier parser.js, qui export la fonction getled, prenant en paramètre la position du fichier dans le file système.

Getled(filePath) => [ [0,1,1,1] , [0,1,1,0], … ]

Une fois le fichier parser on veut envoyer les rangers de led à la arduino suivant un tempo donné dans le fichier midi.

Le fichier i2c.js implémente le protocole de communication entre le server et la Arduino suivant :

« Start » => It[] + it+20[] => it+1[] + it+21[] => … => itn +itn+20 => « Stop »

Affin d’envoyer les données au bon moment on utilise une fonction récursive SendData qui envoyer les données et ce rappelle après l’intervalle de temp 60 000 000/bpm représentant le nombre de milliseconde entre chaque I

Cette fonction ce charge aussi d’enregistrer les notes que l’utilisateurs a joué dans un tableau

La fonction Start initialise le protocole et avertie la arduino,

la fonction end Stop la lecture de la arduino et lance la fonction charger de comparer les notes du morceau et celles jouées par l’utilisateur, le résultat est ensuite enregistré dans le dossier prévu à cette effet « Enregistrement »

les fonction lié au protcole entre node.js et la arduino ce situe dans le fichier protcole.js

les fonction lié a la gestion de l’envoi de donné ce situe dans stream.js

les fonctions lié a l’enregistrement ce situe dans parser.js

# Feature 3 : Front end / Site Web / Interface utilisateur

Sur ce Site web, l’utilisateur pourra voir les différentes pistes jouées, les scores atteints, et le pourcentage de précision par rapport à la version originale.

Il y aura un profil utilisateur avec un menu sur la gauche avec différents onglets entre son pourcentage de réussite de chacun de ses morceaux, de sélectionner les morceaux à jouer. Le but est de rendre l’interface la plus claire possible, tout en offrant à l’utilisateur un historique de toutes ses activités.

Le site sera codé en HTML, CSS, JavaScript, avec la technologie Node.JS (pour l’intégration des bibliothèques pour convertir les fichiers en MIDI/Json).

# Feature 4 : Précision de la session / Scoring / Précision de la session

Une fois la partie terminée, le morceau arrivé à son terme, il faut encore calculer le score de précision pour que l’utilisateur puisse voir ses erreurs. Il suffira pour ça que le JSON soit interpréter en termes de précision avec le pourcentage de ressemblance par exemple. Pour cela, différents outils existent sur internet, comme [diff-json](https://www.npmjs.com/package/diff-json).

Ces informations seront disponibles sur le web server et sur la vue de l’utilisateur. Ou il pourra voir les 2 Json et voir ce qu’il à bien joué ou non. Cela se fera via un script en Javascript. Il sera alors possible de mettre en place un historique général avec le nombre de fois ou le morceau à été essayé, quelle est la meilleure précision, la moins bonne, la moyenne… Nombre de fonctionnalités du site web pourront être rajoutées.