



POLYTECH[®]
NICE-SOPHIA



PROJETS ARDUINO – PEIP2

Année scolaire 2018-2019

"OttoBox : la boîte connectée"

Étudiants : Vogt Hugo et Garcia Axel

Encadrant : Mr. Pascal Masson

REMERCIEMENTS

Nous remercions Mr. Pascal Masson professeur d'électronique pour ses conseils et recommandations tout au long du projet ainsi que pour la mise à disposition de tout le matériel nécessaire. Nous remercions également Pierre Gatto pour nous avoir permis de réaliser une très bonne impression 3D pour notre projet en un temps record. Nous tenons également à remercié l'ensemble de l'équipe enseignante présentent lors de la soutenance de notre projet.

SOMMAIRE

Introduction	7
I. Déroulement du projet	8
I.1. Nos objectifs et motivations	8
I.2. Les différents modules	8
I.3. Algorithme	10
I.3.1. Le code Arduino	10
I.3.2. Le code MIT (App Inventor)	14
I.4. Réalisation du projet	17
I.5. Planning	18
Conclusion	19
Bibliographie	20

Introduction

Depuis quelques années la domotique prend une place de plus en plus importante dans le quotidien, nous avons voulu à travers ce projet toucher à ce domaine en réalisant une interface servant de communication entre le smartphone de l'utilisateur et la domotique de sa maison. Nous voulions que seulement grâce à son smartphone l'utilisateur puisse demander la météo, allumer les pièces de sa maison ou encore lancer de la musique et tout cela commander par sa voix.

Tout au long de ce rapport nous vous présenterons comment nous avons réalisé ce projet durant les 8 séances qui nous été accordées (environ 6 mois).

Nous commencerons par vous présenter nos objectifs et motivations, puis nous vous présenterons les différents modules que nous avons intégré dans le projet ainsi que l'algorithme qui se cache dans cette boîte.

Nous finirons par vous présenter les étapes de réalisation et enfin un planning afin d'avoir une idée de la répartition des tâches dans le temps.

I. Déroulement du projet

I.1. Nos objectifs et motivations

Comme abordé dans l'introduction, notre objectif principal était de réaliser une boîte connectée, c'est-à-dire une interface de communication pour la domotique. Nous voulions que cette boîte soit la seule à gérer les tâches classiques de la domotique. Nous voulions donc construire une boîte capable d'allumer les lumières de différentes pièces, de donner la météo actuelle ou encore de jouer de la musique si on le lui demande. Nous voulions que toutes les fonctionnalités soient accessibles depuis une application sur notre smartphone commandée par notre voix.

I.2. Les différents modules

Pour rendre notre projet réel nous avons eu besoin d'un certain nombre de modules que nous allons vous présenter ci-dessous.

Dès la première séance en vue d'obtenir une utilisation à distance de notre boîte nous avons décidé d'implanter un module Bluetooth. Le module choisi fut le Hc-06, celui-ci nous a permis de communiquer avec la carte Arduino Mega via notre smartphone afin d'envoyer et de recevoir des données.

Le Hc-06 est un module Bluetooth esclave, par conséquent il ne se connecte pas automatiquement à notre smartphone. Nous verrons dans la partie concernant le code MIT comment cette communication a telle eu lieu.

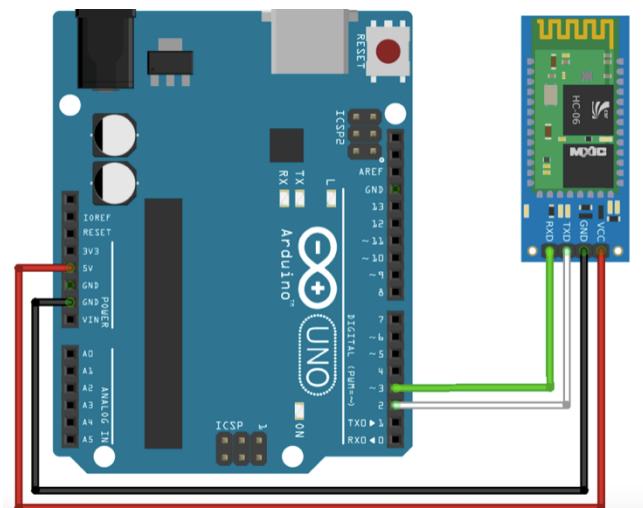


Figure 1 : Branchement du module Hc-06

Le module mp3 est directement branché à un potentiomètre lui-même connecté à deux haut-parleurs permettant de moduler la puissance du son émit.

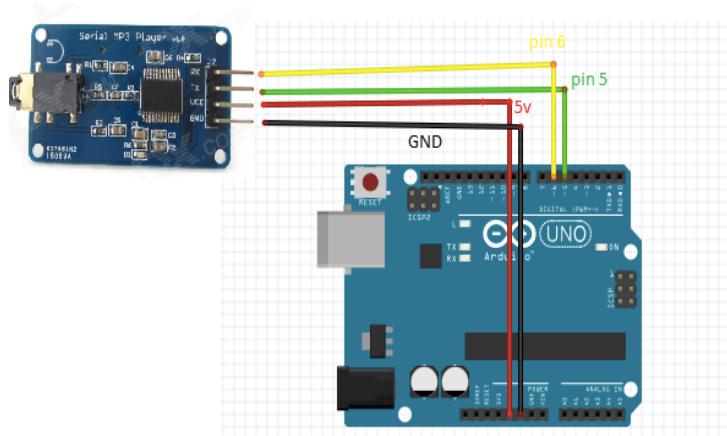


Figure 2 : Branchement du module mp3

Afin de rajouter une fonctionnalité utile, nous avons implanter dans la boîte un module RTC (Real Time Clock) qui n'est autre qu'un module d'horloge. Ce module est connecté à la carte Arduino Mega via les ports SCL et SDA. Afin d'avoir l'affichage de l'heure en permanence sur la boîte, nous avons câbler le module RTC sur un écran LCD (16x2), nous avons donc en permanence et en temps réel un affichage de la date et de l'heure en français sur la devanture de la boîte.

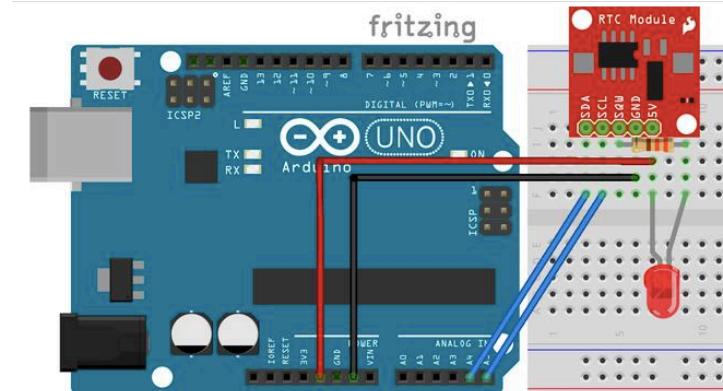


Figure 3 : Branchement du module RTC

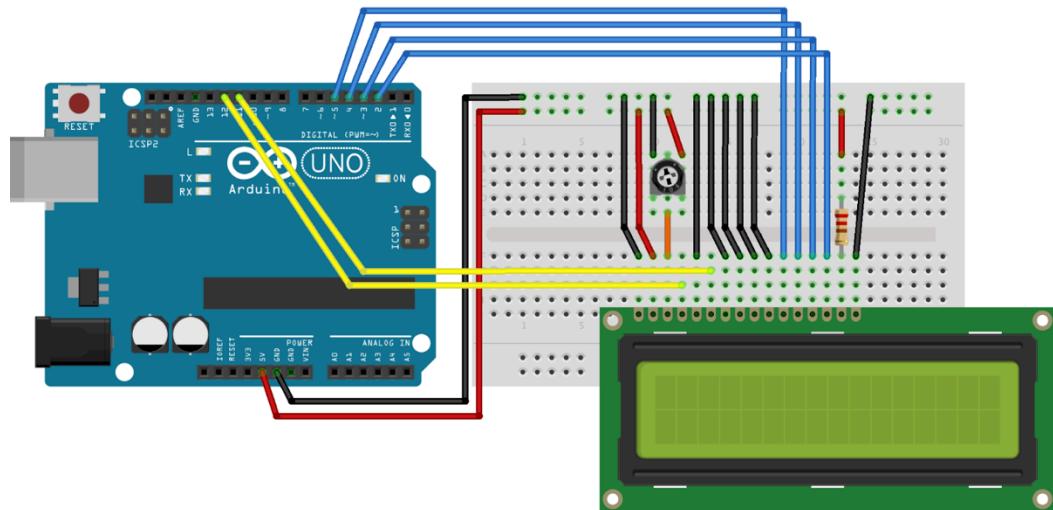


Figure 4 : Branchement de l'écran LCD

Finalement nous avons rajouté une diode infrarouge pour permettre à l'OTTOBOX de fonctionner comme une télécommande universelle. Par l'intermédiaire du module de récepteur IR nous avons analysé le signal envoyé par une télécommande classique pour ensuite en fonction de l'ordre envoyé à la boîte transcrire ce signal grâce à cette diode.

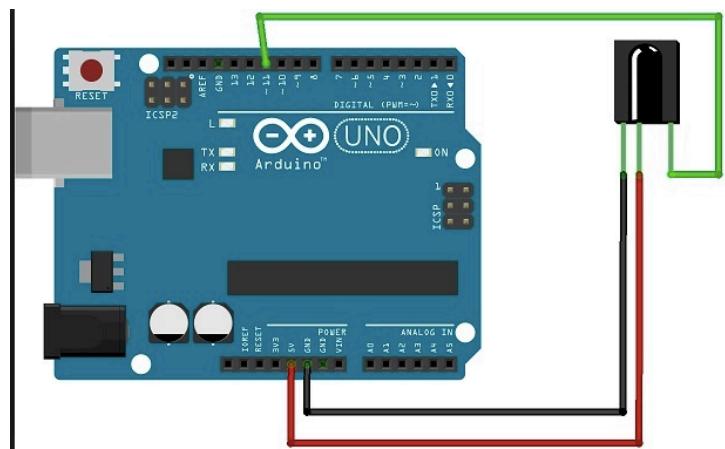


Figure 5 : Branchement du module récepteur IR

I.3 Algorithme

I.3.1. Le code Arduino

Dans cette section nous allons nous intéresser à la partie code de notre projet. Voici un aperçu général du fonctionnement de celui-ci :

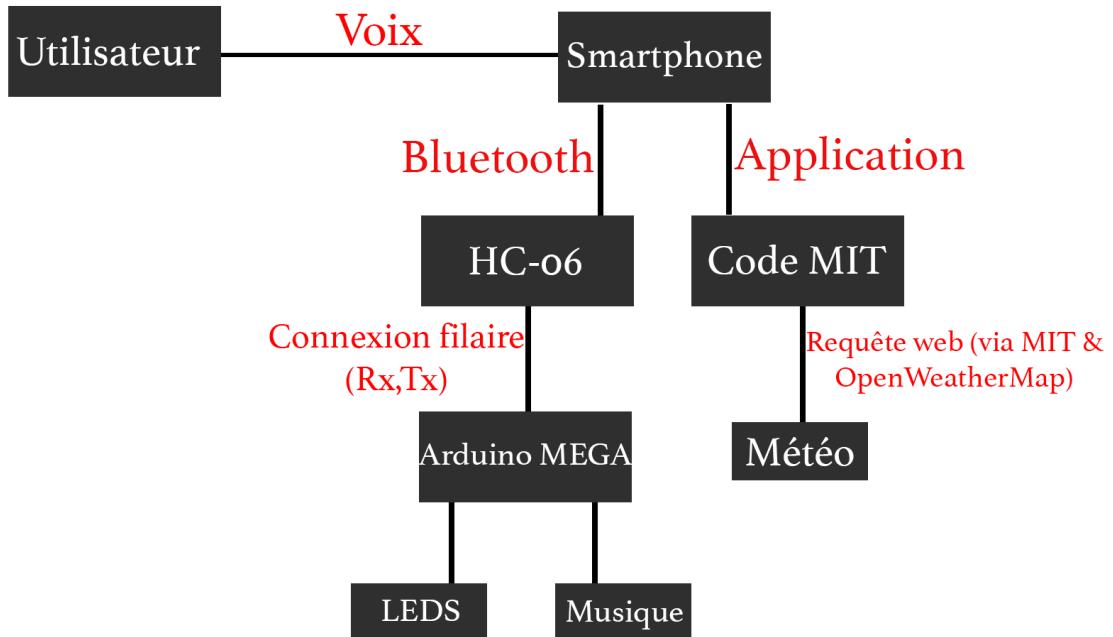


Figure 6-a : Fonctionnement du projet

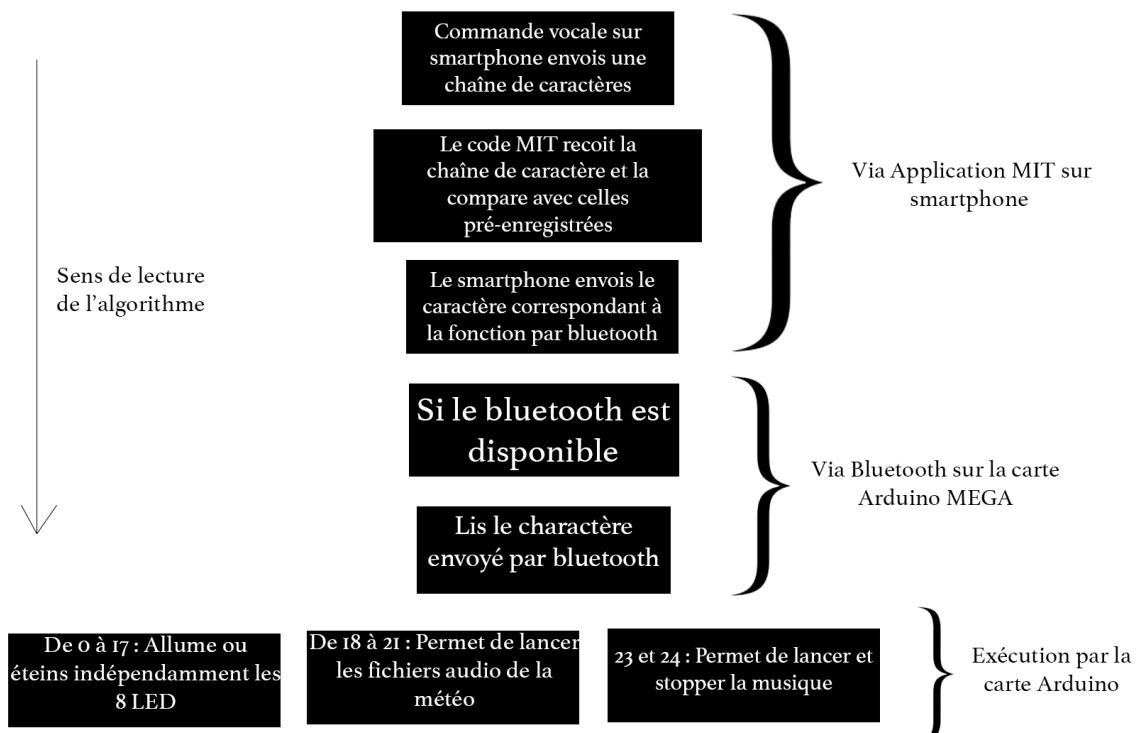


Figure 6-b : Algorithme du programme

Concernant l'éclairage des différentes parties de la maison.

Un code simple et concis a été développé pour permettre l'allumage des LEDs sur la boîte et par extension de l'éclairage de votre maison (domotique classique).

On commence dans un premier temps par établir l'allumage du Hc-06 afin de lui permettre de recevoir les données que nous souhaitons lui envoyer et aussi d'émettre en fonction de la commande.

```
#include <SoftwareSerial.h>
#define RX 10
#define TX 11
SoftwareSerial BlueT(RX,TX);
void setup() {
  BlueT.begin(9600);
  // put your setup code here, to run once:
}
}
```

Figure 7 : Programmation du module Hc-06

Lors de la réception des informations sur le module Hc-06 grâce à la fonction « if » nous allons analyser toutes les possibilités. Autrement dit lorsque le module Hc-06 reçoit une information émise par l'utilisateur, cette information (chaine de caractères) va tous d'abord être transformer en un chiffre pour ensuite être analyser à l'intérieur du code arduino grâce à la boucle « if » et l'opérateur d'égalité nous évaluons l'ensemble des possibilité et retournons une fonctionnalité (partie de code).

Voici un exemple :

```
void loop(){
  int recvChar;
  while(true){
    if(BlueT.available()){
      recvChar = BlueT.read();
      Serial.print(recvChar);
      if(recvChar==1){
        digitalWrite(led1,LOW);
      }
      else if (recvChar == 0){
        digitalWrite(led1,HIGH);
      }
    }
  }
}
```

Figure 8 : Exemple d'utilisation d'arduino

Comme nous pouvons le voir dans la figure 6 le code va dans un premier temps évaluer si le Bluetooth est disponible. Ensuite une variable va prendre comme élément ce que reçoit le Bluetooth. Finalement si cette variable reçoit le caractère « 1 » une led de notre boîte va s'allumer et si la variable reçoit « 0 » cette même led va s'éteindre.

Il reste plus qu'à étendre ce code pour l'ensemble des pièces de votre maison et vous avez une boîte vous permettant d'allumer une pièce grâce à la voix.

L'OTTOBOX propose également une fonctionnalité permettant de donner la météo en temps réel.

Intéressons-nous donc à la programmation du module mp3 qui nous permet de faire « parler » la boîte. Tout d'abord il a fallu enregistrer dans une carte micro SD des fichiers sons contenant les éléments que l'on souhaite faire dire à notre boîte.

Ensuite nous revenons au même code que pour la programmation des leds. Une simple boucle « if » va vérifier si le Bluetooth a reçu le chiffre correspondant à la commande concernant la demande de la météo. Nous développerons dans la partie code MIT comment nous avons réussi à récupérer la météo du jour.

```

else if (recvChar == 18){
    sendCommand(CMD_PLAY_WITHFOLDER, 0X0F00203);
    delay(3500);
    sendCommand(CMD_PLAY_WITHFOLDER, 0X0F00204);
}
else if (recvChar == 19){
    sendCommand(CMD_PLAY_WITHFOLDER, 0X0F00203);
    delay(3500);
    sendCommand(CMD_PLAY_WITHFOLDER, 0X0F00202);
}

```

Figure 9 : Exemple d'utilisation du module mp3

Nous pouvons voir comme dans l'exemple précédent la même utilisation de la boucle « if » et de l'opérateur d'égalité, cependant à l'intérieur de la boucle nous avons rajouter le code permettant de lire le fichier son correspondant à la météo du jour.

Par exemple si la météo est nuageuse le module mp3 va chercher dans le dossier 02 le fichier son correspondant à la météo nuageuse.

Pour la partie code de l'horloge, nous avons fait en sorte que l'affichage se fasse sur les deux lignes de l'écran LCD.

Sur la première l'écran affichera le jour (LUN pour lundi, MAR pour mardi, etc...) suivi de la date et sur la deuxième ligne il affichera l'heure en temps réel (l'heure s'actualise toutes les secondes).

Ce programme est divisé en plusieurs fonctions afin de faciliter et de rendre plus clair l'affichage de l'heure.

La fonction *donne_jour_semaine()* permet en fonction de la date du PC qui compile le programme de retourner le jour de la semaine sous ce format.

La fonction *affiche_date_heure()* permet quant à elle comme son nom l'indique d'afficher la date et l'heure sur l'écran.

```

void loop(){
    DateTime now=RTC.now(); //Récupère l'heure et la date courante
    affiche_date_heure(now); //Converti la date en langue humaine
    delay(1000); // delais de 1 seconde
}

//Converti le numéro de jour en jour /\ la semaine commence un dimanche
String donne_jour_semaine(uint8_t j){
    switch(j){
        case 0: return "DIM";
        case 1: return "LUN";
        case 2: return "MAR";
        case 3: return "MER";
        case 4: return "JEU";
        case 5: return "VEN";
        case 6: return "SAM";
        default: return " ";
    }
}

// affiche la date et l'heure sur l'écran
void affiche_date_heure(DateTime datetime){

    // Date
    String jour = donne_jour_semaine(datetime.dayOfWeek()) + " " +
        Vers2Chiffres(datetime.day())+ "/" +
        Vers2Chiffres(datetime.month())+ "/" +
        String(datetime.year(),DEC);

    // heure
    String heure = "";
    heure = Vers2Chiffres(datetime.hour())+ ":" +
        Vers2Chiffres(datetime.minute())+ ":" +
        Vers2Chiffres(datetime.second());
}

```

Figure 10-a : 1^{ère} partie du Code pour le module RTC et l'écran LCD

Ici on voit l'affichage sur l'écran ainsi que la dernière fonction qui est *Vers2Chiffres()* et qui permet d'afficher l'heure sous 2 chiffres afin que l'heure soit correctement afficher sur l'écran LCD .

```
//affichage sur l'écran
lcd.clear();
lcd.setCursor(0,0);
lcd.print(jour);
lcd.setCursor(0,1);
lcd.print(heure);
Serial.println(heure);
}

//permet d'afficher les nombres sur deux chiffres
String Vers2Chiffres(byte nombre) {
    String resultat = "";
    if(nombre < 10)
        resultat = "0";
    return resultat += String(nombre,DEC);
}
```

Figure 10-b : 2^{ème} partie du Code pour le module RTC et l'écran LCD

I.3.2. Le code MIT (App Inventor)

Notre projet est basé sur l'accessibilité des fonctionnalités de la boîte par utilisation de la voix. De base nous voulions intégrer à la carte arduino un module de reconnaissance vocale, cependant celui présentait beaucoup de dysfonctionnement et fonctionnait uniquement sous environnement Windows. Nous avons eu l'idée de développer une application disponible sur Android permettant de servir d'intermédiaire entre l'utilisateur et l'OTTOBOX.

Après quelques recherches nous avons découvert le site MIT App Inventor, qui propose de coder sa propre application et d'intégrer de nombreuses fonctionnalités à l'intérieur qui peuvent être reliés à un code arduino.

Dans un premier temps il a fallu s'occuper du design de l'application et de choisir les éléments qui seront disponibles dessus.

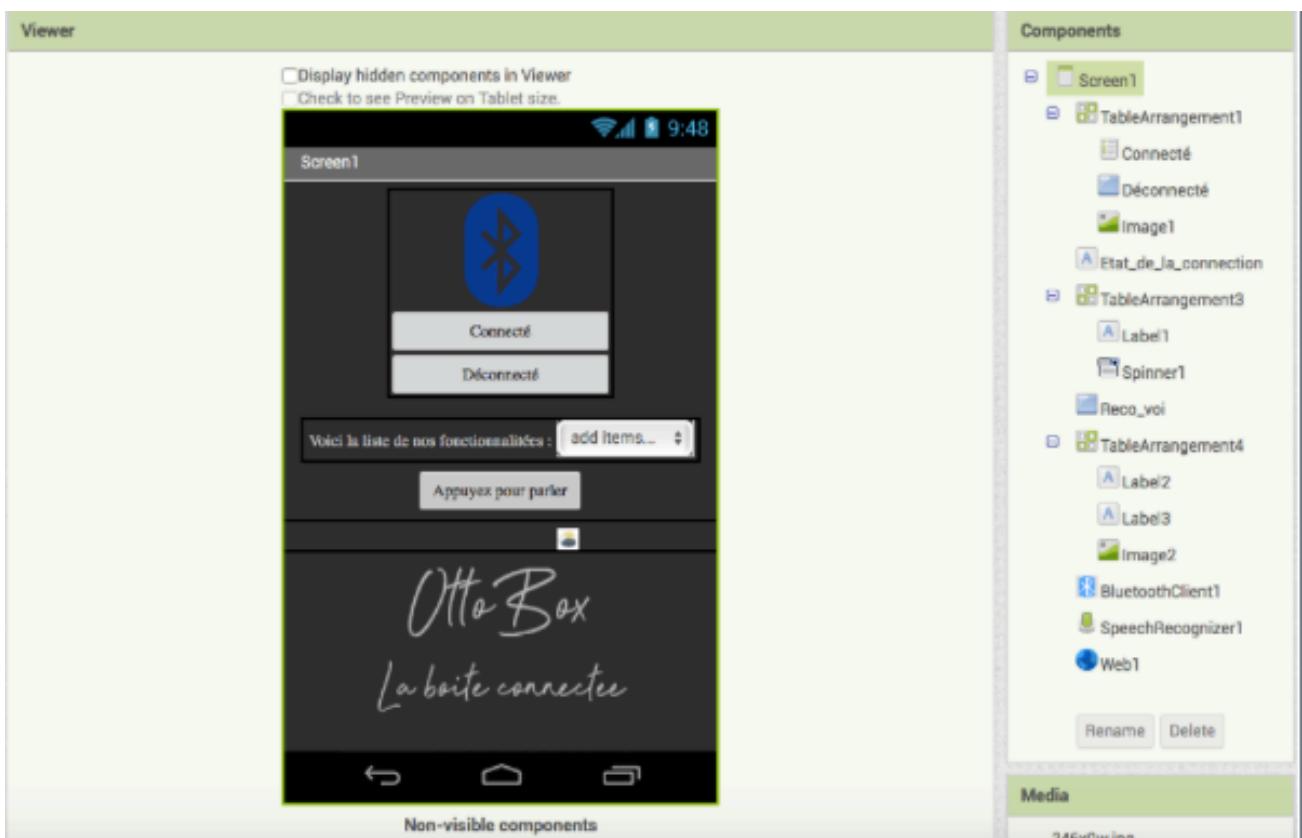


Figure 11 : Design de l'application

Voici l'aspect graphique de notre application. Comme nous pouvons le voir différents « modules » sont importés. Chaque module offre différentes fonctionnalités disponibles dans la partie code que nous verrons par la suite. Par exemple le module web permet de récupérer la météo, le Speech Recognizer permet d'activer la reconnaissance vocale de Google et enfin le Bluetooth Client permet d'envoyer les informations à la carte arduino.

Ensuite différents boutons sont présents sur l'application. Chaque bouton engendre une fonctionnalité bien précise. Par exemple le bouton « Connecté » permet d'afficher l'ensemble des Bluetooth disponibles dans un rayon de 10 mètres (porté du HC-06). Le bouton « Appuyez pour parler » quand a lui déclenche le Speech Recognizer.

Nous allons maintenant nous intéresser à la partie code de cette application, cette dernière fonctionne par « bloc » et chacun d'eux permettent de : vérifier des conditions (If, When, ...), effectuer des calculs, lire des chaînes de caractères, Ces blocs peuvent également être reliés aux fonctionnalités importées depuis la partie design pour y programmer le fonctionnement.

Commençons par regarder la fonctionnalité Bluetooth disponible par l'intermédiaire du Bluetooth Client.

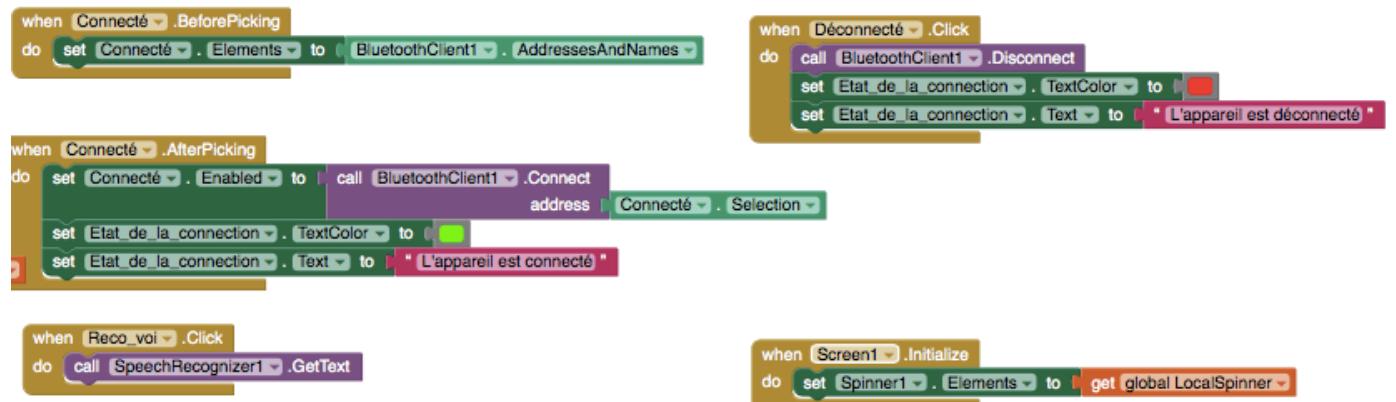


Figure 12 : Code MIT du Bluetooth

On voit bien ici comment se déroule la programmation par « bloc ».

Lorsque l'utilisateur va appuyer sur le bouton « Connecté », le Client Bluetooth va afficher la liste des appareils Bluetooth disponible (dont le module Hc-06). Après la sélection, le Client se connecte et l'application affiche le message « L'appareil est connecté ».

De même pour la déconnection du Bluetooth (cf. en haut à gauche Figure 12).

Du côté météo, la fonctionnalité est en grande partie programmée sur MIT.

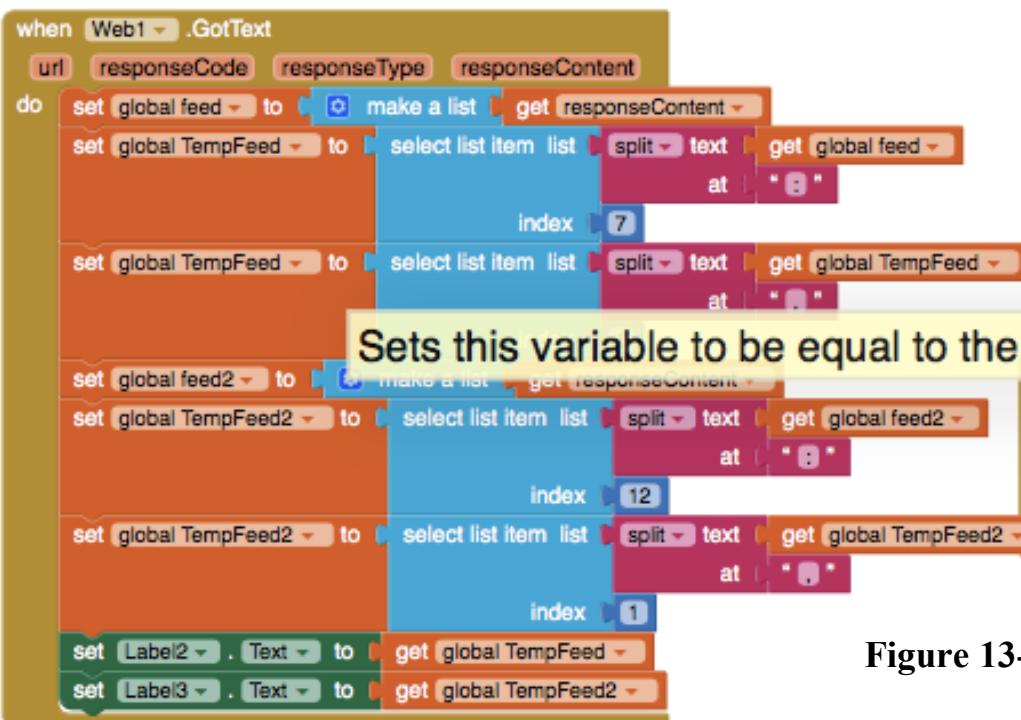


Figure 13-a : Code MIT de la météo

Le bloc « web1 » nous permet quant à lui d'effectuer des requêtes http via des URL afin de récupérer du contenu html ou autre. Dans notre cas nous voulions récupérer des données météo. Nous sommes donc passés par un site web nommé OpenWeatherMap proposant une API gratuite qui fournit en temps réel un fichier JSON regroupant l'intégralité des données météo de la ville choisie.

L'URL que nous avons donc utilisé est le suivant :

```
http://api.openweathermap.org/data/2.5/weather?q=Antibes&APPID=1d7a7c2444395f0ea4f2064f4110dff9&units=metric
```

Une fois la requête envoyée et la réponse obtenu, nous nous retrouvons avec un fichier JSON exploitable, par la suite le programme va découper en plusieurs listes les informations obtenus en utilisant les « : » pour séparer les éléments, puis on sélectionne les éléments désirés en inscrivant l'index (numéro) correspondant. Ici les éléments en index 7 et 12 correspondent à la description de la météo et à la température.

Ces éléments sont stockés dans des variables (TempFeed et TempFeed2) qui seront utilisées pour vérifier quelle est la météo (« Clouds », « Rain », « Clear », « Snow ») et envoyer le caractère correspondant par bluetooth à l'arduino afin que cette dernière joue les fichiers mp3 en conséquence.

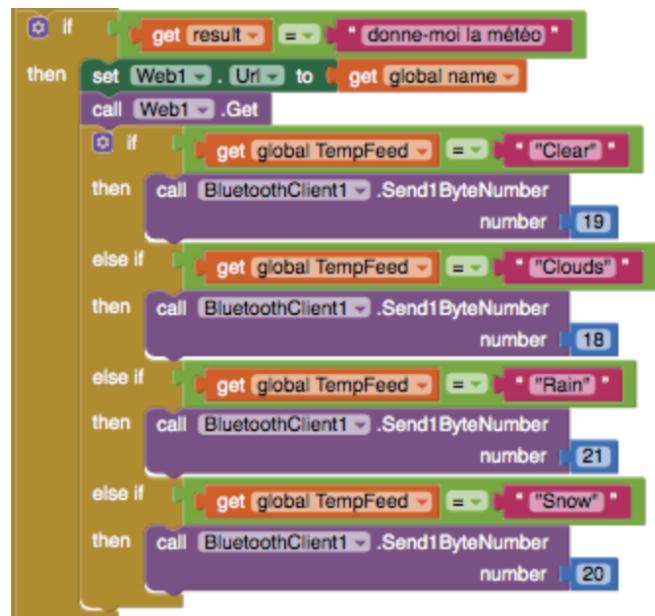


Figure 13-b : Code MIT de la météo

Regardons de plus près l'allumage des leds grâce à notre code MIT. Lorsque l'utilisateur déclenche la commande vocale et annonce une instruction au téléphone tel que « allume la cuisine » le code en bloc de MIT test l'égalité avec l'ensemble des données que nous avons préalablement enregistrés. Une fois l'égalité vérifiée le client Bluetooth envoi le byte correspondant à l'Arduino.

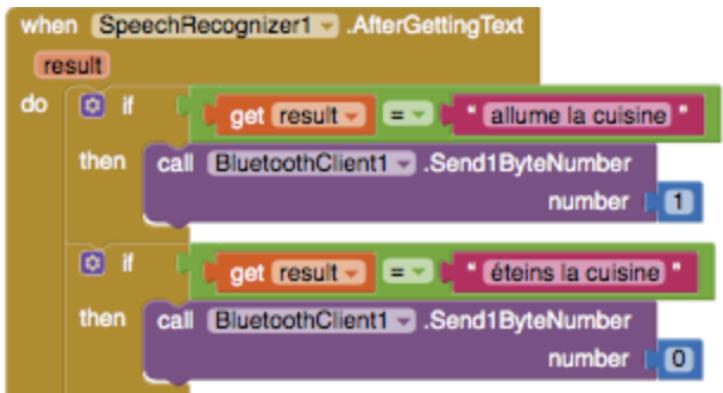


Figure 14 : Code MIT des leds

I.4. Réalisation du projet

Voici comment le projet a été réalisé (chronologie) :

Pour Axel :

- Branchements et programmation module RTC
- Branchements et affichage de l'heure sur l'écran LCD
- Programmation module ESP32
- Recherche API météo
- Abandon module ESP32

Pour Hugo :

- Branchements et programmation des leds (allumage par commande vocale)
- Design/Conception 3D de la boîte
- Programmation sur MIT
- Branchements et programmation module MP3
- Soudures des leds et du transformateur (220-5V)

En commun :

- Branchements et programmation module Infrarouge (pour la télé)
- Passage sur carte Arduino Mega
- Impression 3D de la boîte et soudures
- Finitions de la boîte

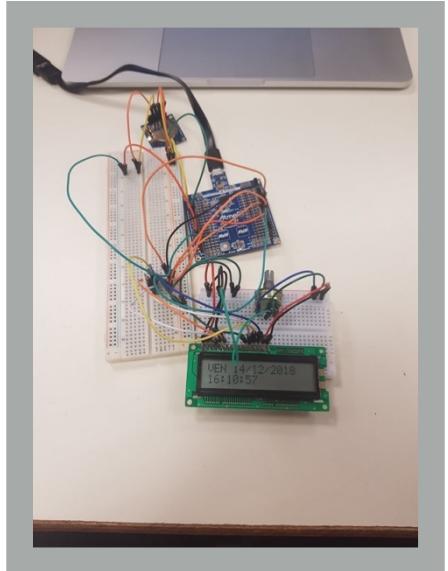


Figure 15 : Module RTC et écran LCD

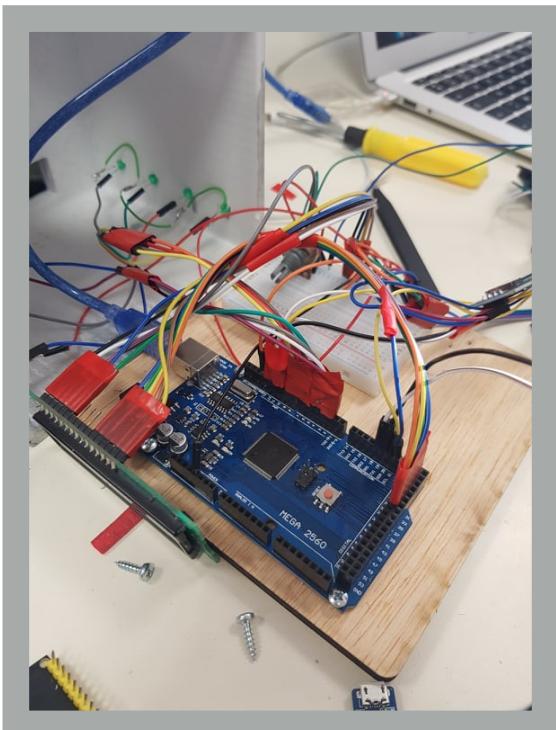


Figure 16 : Montage sur carte Arduino Mega

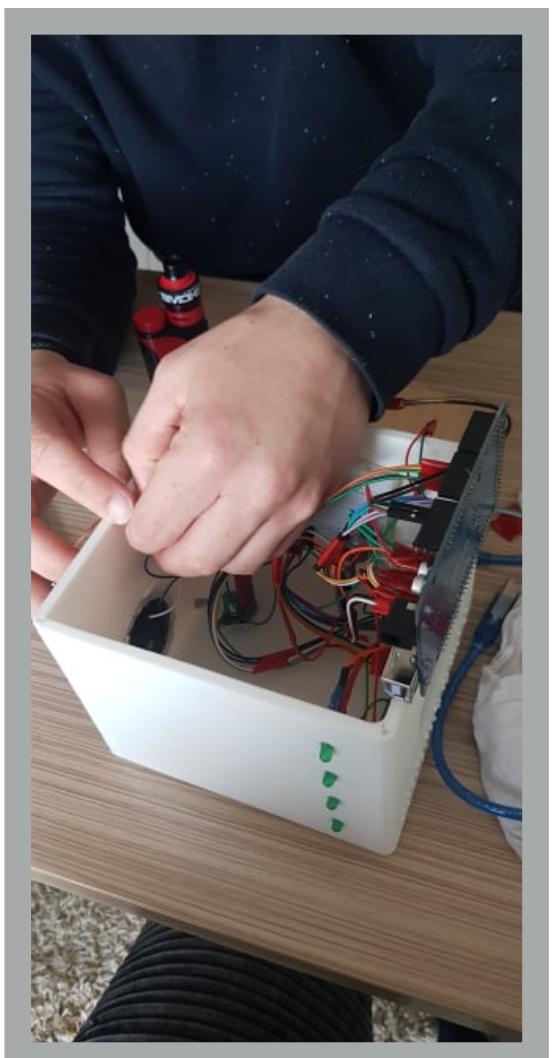


Figure 17 : Branchements dans la boîte

I.5. Planning

Voici le planning prévu à la suite des deux premières séances :

Cahier des charges - Planning théorique									
Réalisations	Séance 1	Séance 2	Séance 3	Séance 4	Séance 5	Séance 6	Séance 7	Séance 8	
Modélisation boîte	100%				50%				
Affichage heure (horloge) écran LCD	100%								
Branchements/Programmation LED			100%	100%					
Branchements du module ethernet et essais de connexions			100%	100%					
Programmation led commande vocale (MIT)					100%				
Branchements module WIFI (ESP) - Programmation ESP					100%				
Programmation Musique commande vocale (MIT)						100%			
Branchements des différents modules dans la boîte							100%	100%	
Branchements des différents modules dans la boîte							100%	100%	
Finitions boîte et branchements + derniers essais									100%
Finitions boîte et branchements + derniers essais									100%

CODE COULEUR:

HUGO
AXEL

Cependant, voici le planning réalisé au cours des 8 séances :

Planning Réalisé									
Réalisations	Séance 1	Séance 2	Séance 3	Séance 4	Séance 5	Séance 6	Séance 7	Séance 8	
Modélisation boîte				100%	100%	50%	50%		
Affichage heure (horloge) écran LCD	100%								
Branchements/Programmation LED	100%	100%	100%						
Branchements du module ethernet et essais de connexions		100%							
Programmation led commande vocale (MIT)			100%	100%					
Branchements module WIFI (ESP) - Programmation ESP				100%	100%	100%	100%	100%	
Programmation Musique commande vocale (MIT)					100%	100%			
Branchements des différents modules dans la boîte						100%	100%	100%	
Branchements des différents modules dans la boîte						100%	100%	100%	
Finitions boîte et branchements + derniers essais									100%
Finitions boîte et branchements + derniers essais									100%
Programmation météo MIT									100%
Programmation météo MIT									100%

CODE COULEUR:

HUGO
AXEL

On voit bien que même si le projet a été réalisé dans son intégralité, le planning prévu n'a pas été respecté, il a souvent fallu effectuer plusieurs tâches en simultané et certaines que l'on avait jugées réalisables rapidement on en réalité pris beaucoup plus de temps.

Conclusion

Pour conclure, nous pouvons affirmer que le projet est fonctionnel et qu'il répond à toutes nos attentes. Même si la réalisation a parfois été difficile, le rendu correspond à ce que nous voulions faire au début de ces 8 séances ; les LEDs s'allument, la météo est donnée par la boîte, la musique est jouée par la boîte et tout ceci commandé par la voix des utilisateurs.

Avec plus de temps nous aurions voulu intégrer un module de détecteur de température afin de contrôler les chauffages électriques d'une pièce, développer une application de contrôle de la consommation électrique dans la maison ainsi qu'ouvrir et fermer des portes et fenêtres depuis l'OTTOBOX.

Bibliographie

Liens utiles : https://www.elecrow.com/wiki/index.php?title=Tiny_RTC

<https://forum.arduino.cc/index.php?board=33.0>

<http://ai2.appinventor.mit.edu/>

<https://store.arduino.cc/arduino-ethernet-shield-2>

<https://www.carnetdumaker.net/articles/utiliser-un-lecteur-serie-de-fichiers-mp3-avec-une-carte-arduino-genuino/>

http://tiptopboards.free.fr/arduino_forum/viewtopic.php?t=8&p=8

<https://openweathermap.org/current>