

MoodBox

Boîte modulaire : mini-station météo et juke box Arduino.

 Difficulté **Moyen**

 Durée **100 heure(s)**

 Catégories **Art, Décoration, Électronique, Musique & Sons**

 Coût **50 EUR (€)**

Sommaire

Introduction

Préambule

Spécifications générales

Démarrage de la boîte

Étape 1 - Conception 2D (nouvel étage de la boîte)

Étape 2 - Conception des éléments pour l'impression 3D

Étape 3 - Découpe laser et gravure du nouvel étage de la boîte

Étape 4 - Impression 3D

Étape 5 - Schéma du circuit

Entrées

Sorties

Moteurs

Alimentation

Étape 6 - Programmation Arduino

Bibliothèques

Étape 7 - Assemblage, tests et finitions

Étape 8 - Now let's dance!

Notes et références

Commentaires

Introduction

Cette boîte modulaire est composée de 3 étages. Une fois allumée, elle fournit par les informations d'une mini-station météo (température/humidité/pression atmosphérique) mais se transforme en une "Mood box", à savoir en juke box, à la demande.

Préambule

Ce projet est un projet de groupe (FaB team) réalisé dans le cadre de la formation hybride "Fabrication Numérique", promotion juillet 2018. Ce projet de fin de parcours met en œuvre une partie des savoir-faire acquis au cours de celle-ci et scelle la formation. Il vient donc en complément du projet "bentolux" qui sera bientôt documenté et référencé sur wikifab.org .

Les instructions de base étaient les suivantes :

- concevoir un nouvel étage sur une boîte existante conçue au fil de la formation (3 étages possibles à disposition : 1 socle, 1 en plexiglas, 1 pour l'écran LCD) ;
- utiliser pour ce nouvel étage les connaissances acquises : impression 3D, découpe laser, etc ;

- programmer via Arduino au moins une interaction avec l'utilisateur (à nous de choisir cette dernière).

Spécifications générales

Après réflexion, notre groupe a décidé de créer l'étage musical "Mood Box" qui viendrait compléter la station météo déjà prévue avec l'étage socle et l'étage écran LCD.

Le comportement prévu est décrit dans les paragraphes ci-dessous.

Démarrage de la boîte

Une fois le bloc d'alimentation correctement branché, le démarrage de la boîte est réalisé par le biais de l'interrupteur en façade.

Ce démarrage déclenche les événements suivants :

- la figurine au sommet de la boîte tourne sur son support ;
- cette "danse" est agrémentée d'une animation lumineuse via l'anneau de LEDs situé sous le support de la figurine ;
- l'écran LCD affiche un message de bienvenue.

S'il n'y a aucune autre action dans la foulée, la boîte passe en mode "veille".

Dans ce mode, l'écran LCD affiche la température et le taux d'humidité accompagnés d'un message.

Ces messages comme la couleur appliquée à l'anneau de Leds et au fond de l'écran LCD dépendent de la température détectée, à savoir :

- la couleur (LCD/Led-ring) varie du bleu au rouge proportionnellement à la température ;
- le led-ring scintille à une fréquence aléatoire pour donner de l'animation ;
- un message variable par tranche est affiché sur l'écran LCD.



Matériaux

Les matériaux nécessaires ont été regroupés par catégorie.

Pour la boîte

- panneau de contreplaqué en 3mm d'épaisseur pour la découpe laser ;

 le peuplier n'est pas très cher, se découpe très bien et surtout très rapidement.

- languette de laiton 80x7x2mm (Lxlxh), pour modifier et étendre la surface sensible de la touche capacitive ;

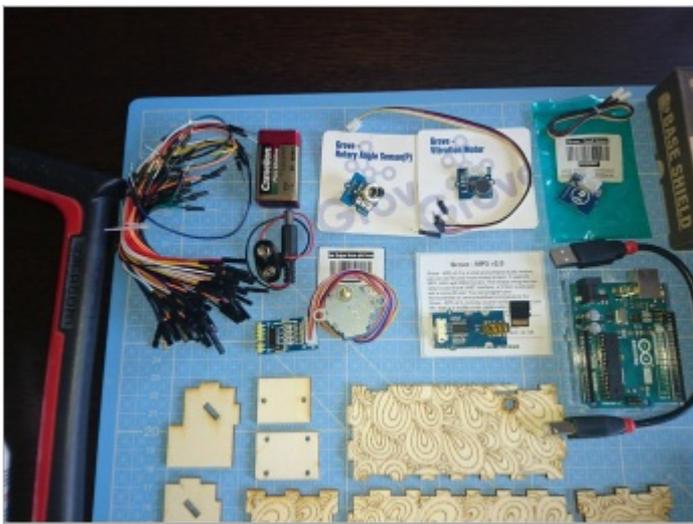
 La tige laiton se trouve dans le commerce en barre d'un mètre (~7€). On peut la remplacer par n'importe quel élément conducteur, de section et forme différente.

- PLA transparent pour imprimer en 3D :
 - la trappe d'accès à la boîte contenant l'Arduino,
 - les supports de la figurine dont un qui va également diffuser la lumière du led-ring,
 - le bouton de volume,
 - les clips d'assemblage ;
- PLA (couleur au choix) pour créer le décor animé placé au sommet de la boîte.

Pour l'électronique

- 1 carte Arduino Uno Rev.3 ;
- 1 alimentation électrique MW, modèle 1500mA Rotary Switch Adaptor ;
- 1 inverseur KNX-1, 3A, 250V AC, pour servir d'interrupteur général ;
- 1 potentiomètre Seeed Studio, modèle Grove Rotary Angle Sensor (P) ;
- 1 touche capacitive Seeed Studio, modèle Grove Touch Sensor ;
- 1 ledring Adafruit, modèle Neopixel Ring 12 x 5050 RGB ;
- 1 lecteur MP3 Seeed Studio, modèle Grove MP3 v2.0 ;
- 1 carte micro SD, pour stocker la bibliothèque musicale de son choix ;
- 1 moteur Seeed Studio, type Gear Stepper Motor with Driver,





- 📄 Supports - Bras et pivot - V4.stl
- 📄 Fermeture box V2.stl
- 📄 Capot arrière V3.stl

modèle Step Motor 28BYJ-48 5V DC ;

- 1 capteur BME280 Adafruit : T°, humidité, pression (ADA2652) ;
- 1 "Base Shield V2" de marque Seeed Studio.

Autres

- colle à bois, pour l'assemblage des différents étages de la boîte ;
- colle cyanoacrylate ;
- visserie pour fixer certains composants à la boîte ;
- câbles et connecteurs.

Outils

Les outils utilisés sont répartis ci-dessous suivant leur nature.

CAO & PAO

- Inkscape : logiciel de CAO 2D pour créer ou modifier le schéma de découpe et/ou de gravure des différents éléments de la boîte ;
- TinkerCAD, Blender ou autre logiciel de CAO 3D de son choix pour créer ou modifier les éléments de décor et autres parties imprimées de la boîte ;
- Arduino IDE : environnement de développement adapté à la carte Arduino pour créer le programme de commande de la boîte et gérer les différentes interactions entre les utilisateurs et les capteurs et autres composants.

💡 Tous les fichiers 2D et 3D (.svg, .stl) sont mis à disposition dans l'onglet "fichier" de cette page.

Machines & outils

- découpeuse laser, accessible dans un Fablab ;

⚠️ Plus les panneaux à découper seront plans (et non cintrés), plus la découpe sera précise. Il peut être judicieux de mentionner au vendeur que les panneaux vont aller en découpe laser.

- imprimante 3D, également accessible via un FabLab au besoin ;
- fer à souder et matériel de soudure ;

⚠️ Attention, en plus du risque de brûlure en cas de mauvaise manipulation du fer lui-même, certains composants électroniques sont sensibles à la chaleur et nécessitent de minimiser le temps de chauffe des différentes parties à souder. Il faut donc rester vigilant et procéder par étapes courtes (de 2 à 3 secondes maximum).

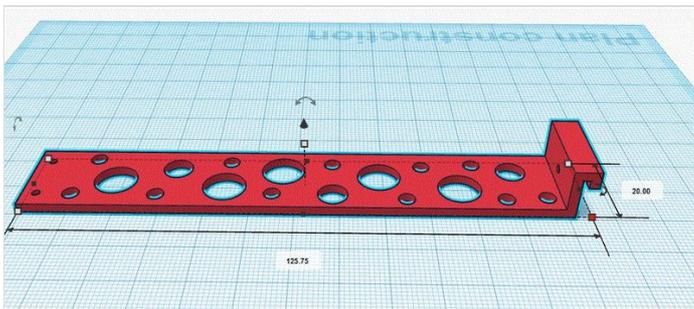
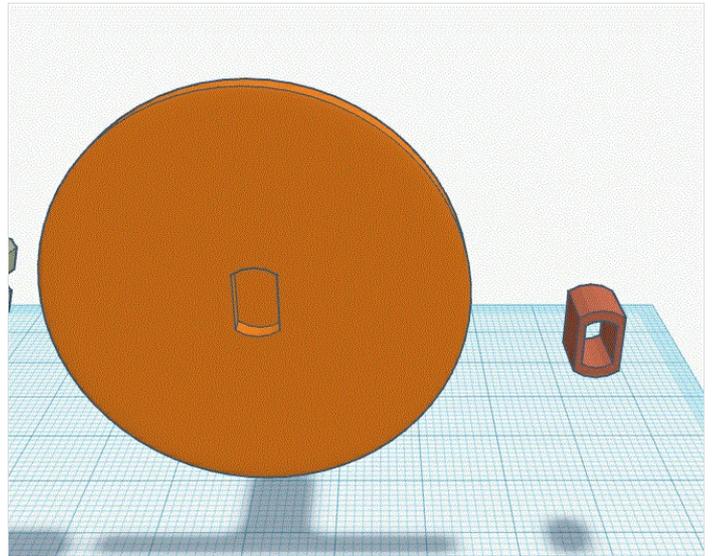
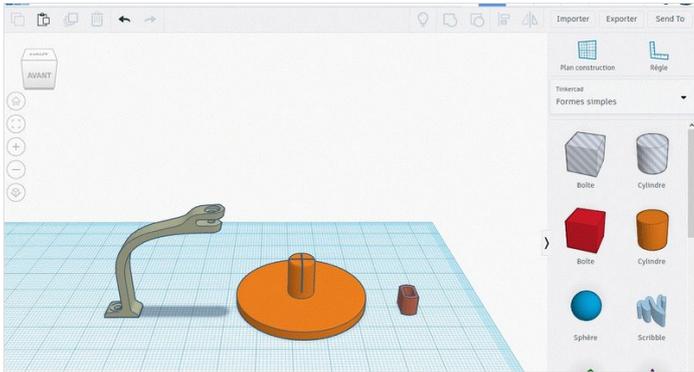
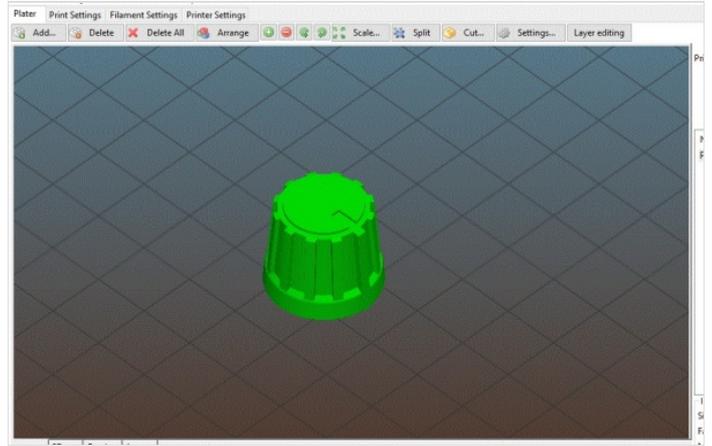
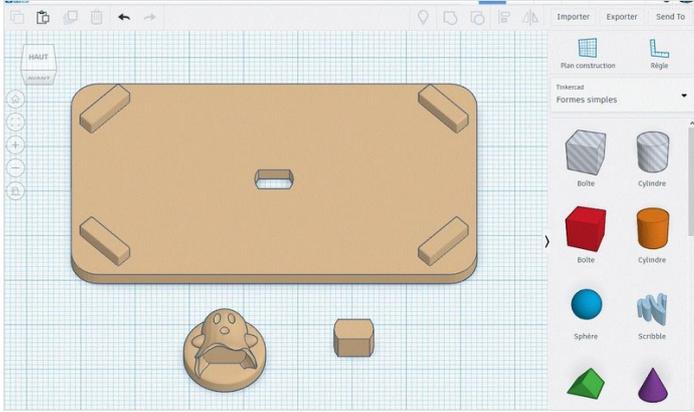
- tournevis, pinces coupantes, scie, etc. ;
- outil type Dremel.

Étape 2 - Conception des éléments pour l'impression 3D

Les éléments destinés à être imprimés en 3D sont les suivants :

- trappe d'accès pour l'étage de base sur lequel l'Arduino est placé ainsi que l'interrupteur et le potentiomètre ;
- bouton pour la molette de réglage du potentiomètre ;
- l'embase sous le fantôme qui lui sert de support et diffuse la lumière du ledring ;
- la figurine, ici le fantôme animé "Funkgeist" issu du site Thingiverse (<https://www.thingiverse.com/thing:570654>) et ses éléments de support ;
- les clips de fermeture latéraux de la boîte.

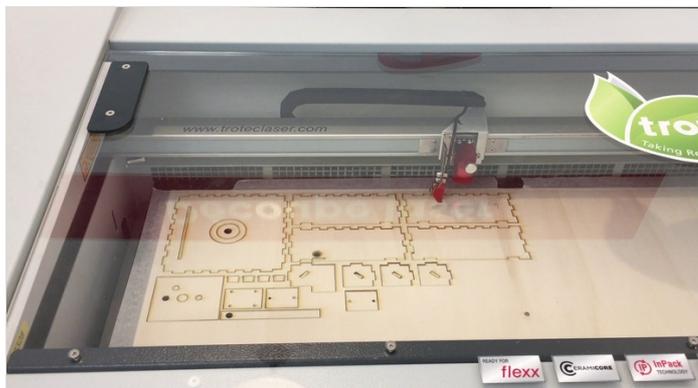
Pour cette étape, le travail a été réalisé à l'aide des logiciels Inkscape, TinkerCAD et Blender.



Étape 3 - Découpe laser et gravure du nouvel étage de la boîte

La découpe laser (avec la gravure) a été faite au sein du EdFab à Saint-Denis sous la houlette bienveillante de Lola et Arthur : un grand merci à eux deux !

Une fois la boîte découpée et assemblée, on peut laisser libre court à son imagination pour la peindre, la décorer, la personnaliser selon ses envies.



Étape 4 - Impression 3D

Toutes les impressions nécessaires au projet ont été réalisées sur une imprimante Prusa i3 MK3.

Ces éléments sont tous visibles sur les photos ci-contre :

- étage de base
 - trappe d'accès,
 - capot,
- étage Mood Box
 - supports de la figurine (fixe et tournant),
 - figurine,
 - clips de fermeture (sur les côtés).





Étape 5 - Schéma du circuit

Les différents capteurs et éléments sont branchés sur le shield implanté sur la carte Arduino sur les ports comme décrit ci-dessous.

Entrées

- Potentiomètre : entrée analogique A0 ;
- Touche capacitive : entrée analogique A1 ;
- Capteur BME280 :
 - Vin > 5V,
 - GND > GND,
 - CS > 10,
 - SDI > 11,
 - SDO > 12,
 - SCK > 13.

Sorties

- LCD : port I2C ;
- Lecteur MP3 : D2 ;
- Ledring : D3.

Moteurs

- Moteur pas-à-pas :
 - IN 1 > 10 ;
 - IN 2 > 9 ;
 - IN 3 > 11 ;
 - IN 4 > 8.

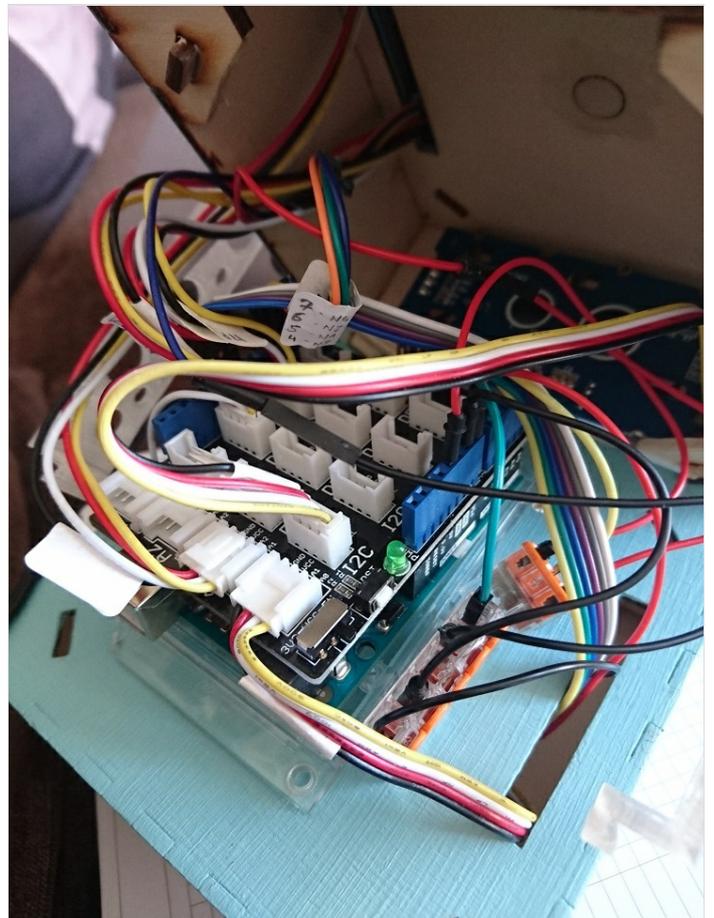
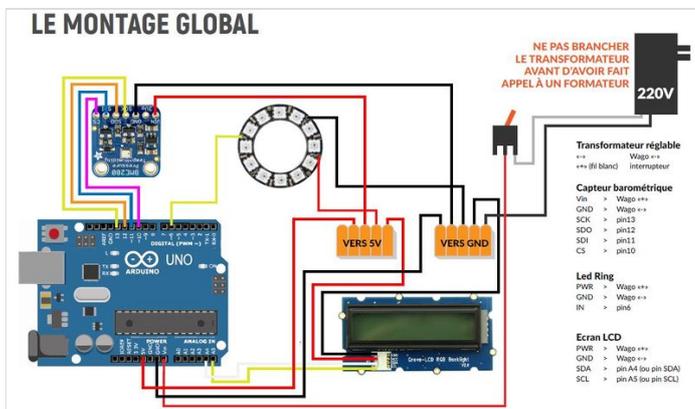
Alimentation

Il est à noter que l'ensemble des éléments sont assez gourmands en énergie, l'alimentation délivrée par un USB est ainsi trop faible (~5V), idem pour une pile 9V embarquée, elle se décharge relativement vite.

Nous avons donc opté pour une alimentation avec un transformateur réglable.



Penser à régler l'alimentation sur 9V en sortie afin d'avoir la puissance nécessaire à des affichages/actions claires.





Étape 6 - Programmation Arduino

La programmation a posé quelques difficultés du fait du fonctionnement séquentiel natif de l'Arduino.

Des contournements ont dus être trouvés afin d'animer plusieurs des composants de façon simultanée et non plus seulement séquentielle.

Les 3 premières images ci-contre montrent les déclarations pour chacun des éléments constituant la boîte et son étage Mood Box.

Le programme principal consiste en une boucle exécutant la séquence demandée en fonction des interactions avec l'utilisateur. Chaque séquence est identifiée par un chiffre :

- 0 : séquence au cours de laquelle il ne se passe rien ;
- 1 : séquence de démarrage ;
- 2 : séquence de "réveil", soit au moins un appui sur la barre ;
- 3 : séquence musicale.

La gestion des temps d'attente, notamment pour la surveillance du nombre d'appuis sur la barre, a été faite en utilisant la fonction millis() afin d'éviter la fonction delay() qui met en pause tout le programme.

Pour contourner le fonctionnement séquentiel, l'utilisation de la bibliothèque AccelStepper.h et l'implantation des commandes, associées au cœur même des routines appelées séquentiellement par la boucle principale, permet d'avoir la simultanéité des actions principales, à savoir : les animations de l'anneau de LEDs et la rotation du moteur pas-à-pas.

 C'est cette astuce dans le code qui permet de ne pas arrêter le moteur pendant le temps dédié à la surveillance d'une interaction avec la barre.

Bibliothèques

Le programme s'articule autour de bibliothèques spécifiques aux composants utilisés, en voici la liste :

- LCD : "rgb_lcd.h"
 - https://github.com/Seeed-Studio/Grove_LCD_RGB_Backlight
- Ledring : "FastLED.h"
 - <https://github.com/FastLED/FastLED>
- MP3 : "SoftwareSerial.h" et "MP3Player_KT403A.h"
 - https://github.com/Seeed-Studio/Grove_Serial_MP3_Player_V2.0/archive/master.zip
- Moteur : "AccelStepper.h"
 - <https://github.com/adafruit/AccelStepper>
- Capteur BME280 : "SPI.h", "Adafruit_Sensor.h" et "Adafruit_BME280.h"
 - https://github.com/adafruit/Adafruit_BME280_Library
 - https://github.com/adafruit/Adafruit_Sensor

```
// 1- LCD
// -----
// Bibliothèques nécessaires à l'afficheur LCD :
// - I2C : pour "Inter-Integrated Circuit", gère le lien entre un microprocesseur et des circuits, module "Wire.h" du package Arduino.
// - LCD : on utilise la bibliothèque fournie par le fabricant Seeed "Grove_LCD_RGB_Backlight", module "rgb_lcd.h".
// cf. https://github.com/Seeed-Studio/Grove\_LCD\_RGB\_Backlight
#include <Wire.h>
#include "rgb_lcd.h"

// Déclaration du nom de l'afficheur LCD : la construction de la bibliothèque Grove impose "lcd" comme nom par défaut.
rgb_lcd lcd;

// Rem : par définition une sortie I2C ne nécessite pas de définition de brochage, ceci est autogéré par le protocole.

// Définition de caractères spéciaux :
byte heart[8] = {0b000000,0b01010,0b11111,0b11111,0b11110,0b01100,0b00000,0b00000};
byte smiley[8] = {0b000000,0b00000,0b01010,0b00000,0b00000,0b10001,0b01110,0b00000};
byte frownie[8] = {0b000000,0b00000,0b01010,0b00000,0b00000,0b00000,0b01110,0b10001};
byte armzDown[8] = {0b00100,0b01010,0b00100,0b01110,0b01101,0b00100,0b01010};
byte armzUp[8] = {0b00100,0b01010,0b00100,0b01010,0b01110,0b00100,0b00100,0b01010};

// Définition de la variable qui va contenir le message à afficher :
String Message = "Message de base";

// Définition de la luminosité de l'écran (0-255)
const int LCD_Seuil = 255;

// 2- LED-RING
// -----
// Bibliothèques nécessaires à la gestion de la led-ring :
// - FastLED : bibliothèque dédiée aux led-rings "Mempixels" du fabricant Adafruit.
#include "FastLED.h"

// Définition du nombre de leds du led-ring et de la broche de sortie correspondante
#define NBR_LEDS 12 // 12 leds sur l'anneau
#define LEDR_PIN 3 // Broche du led-ring (D3 sur le Base Shield V2 Seeed)

// Déclaration du tableau de valeurs de couleurs des leds
CRGB leds[NBR_LEDS];

// Variable de stockage de la dernière couleur utilisée
byte Latest_Color[3] = {0,20,50};

// Variable et valeur de brillance de base
float BaseBrightness = 20;

// 3- TOUCHE CAPACITIVE
// -----
// Broche d'entrée de la touche/ie de la barre laiton
#define TCAP_PIN A1

// Variable d'entrée associée à la touche
int TCAP_Value = 0;
```

```
// 4- LECTEUR MP3
// -----
// Le module MP3 Grove de la marque Seeed (https://wiki.seeedstudio.com/Grove-MP3\_V2.0/)
// lit une carte micro-SD et renvoie vers une sortie jack.
// Ce module utilise la bibliothèque "MP3Player_KT403A.h" contenue dans le paquet "Grove_Serial_MP3_V2.0"
// (https://github.com/Seeed-Studio/Grove\_Serial\_MP3\_Player\_V2.0/archive/master.zip).
#include <SoftwareSerial.h>
#include <MP3Player_KT403A.h>

// Comme pour le moteur (ci-dessous) il faut définir un nom à l'objet lecteur et son nom est
// obligatoirement "mp3" et une broche de connexion :
#define MP3_PIN 2
SoftwareSerial mp3(MP3_PIN,3); // Rem : le 2nd argument définit la broche de sortie sur la carte MP3
// et "3" est la prise jack.

// Commandes de pilotage du lecteur :
// 1- Play (P)
// 2- Pause/Resume (p)
// 3- Next (N)
// 4- Previous (L)
// On définit le statut du lecteur au départ :
int PlayStatus = 0;

// 5- POTENTIOMETRE
// -----
// Broche d'entrée du potentiomètre
#define POT_PIN A0

// Variables d'entrée associée au potentiomètre
int PotValue;
int PotValueRef = 14;

// 6- MOTEUR
// -----
// Le moteur pas-à-pas bipolaire (https://www.gotronic.fr/art-moteur-24bv148-19028.htm)
// est commandé via la bibliothèque "Stepper.h" qui est pré-installée sous Arduino.
// Déclaration de la bibliothèque :
#include <Stepper.h>
#include <AccelStepper.h>

// Déclaration du nom du moteur ("Ghost_Stepper") et passage des arguments (type et numéros des broches) à la bibliothèque.
// ATTENTION : le brochage n'est pas "linéaire" sur le driver !!!
// = nom_stepper(nbr_tours, INA, INB, IN1, IN2)
AccelStepper Ghost_Stepper(AccelStepper::FULL4WIRE, 4, 5, 6, 7);

// Définition des vitesses typiques (en RPM) :
const int LOW_SPEED_Stepper = 4096/8/4;
const int NORMAL_SPEED_Stepper = 4096/8/2;
const int HIGH_SPEED_Stepper = 4096/8;
const int ACC_Stepper = 4096/8/4;
int SPEED_Stepper = HIGH_SPEED_Stepper;
```

```
// 7- CAPTEUR "METEO" BME280
// -----
// Ce multiocteur BME280 (https://learn.adafruit.com/adafruit-bme280-humidity-barometric-pressure-temperature-sensor-breakout/rview=all)
// est monté ici en mode SPI ("Serial Peripheral Interface").
// Il utilise 3 bibliothèques. Une commune à tous les capteurs Adafruit, une spécifique au capteur en question et une dédiée
// à la communication SPI (native Arduino). On les déclare donc :
#include <SPI.h>
#include <Adafruit_Sensor.h>
#include <Adafruit_BME280.h>

// Brochage : - CS (Slave Select)      : broche 10
//            - MOSI (Master Out Slave In) : broche 11
//            - MISO (Master In Slave Out) : broche 12
//            - SCK (Serial Clock)       : broche 13

// Définition de la pression de référence locale (le du "sea level pressure") nécessaire à l'étalonnage du capteur :
#define BME_SEALEVELPRESSURE_HPA (1024.6)

// Déclaration du nom du capteur et passage des arguments (nécessaire au module SPI) :
Adafruit_BME280 bme(10, 11, 12, 13);

// N- Autres variables
// -----
// Variable des différentes séquences, en fonction de sa valeur :
// - 0 : il ne se passe rien;
// - 1 : séquence de démarrage;
// - 2 : séquence de "réveil";
// - 3 : séquence musicale.
// Au démarrage on a de l'affichage et une animation led-ring :
int Seq_Box = 1;

// Création d'un compteur de temps :
unsigned long TimeCounter;
unsigned long ElapsedTime;
```

```
// Fonction d'attente avec, ou sans, la rotation du moteur
void Task_Wait(int waiting_time,boolean Moteur){
// ===== //
// La clef ici est de pouvoir faire tourner le moteur en simultané //
// avec la fonction d'attente. C'est l'ASTUCE du code, mettre la //
// commande dans la boucle while, avec un check booléen. //
// ===== //

unsigned long previousMillis;
unsigned long currentMillis;

previousMillis = millis();
currentMillis = millis();
while (currentMillis - previousMillis<waiting_time) {
    currentMillis = millis();

    // Moteur en simultané en fonction de la variable booléenne
    // *****
    if (Moteur) Ghost_Stepper.runSpeed();
}
}
```

Étape 7 - Assemblage, tests et finitions

La carte Arduino, le potentiomètre ainsi que l'interrupteur relié au bloc d'alimentation sont installés dans et sur l'étage de base de la boîte (étage inférieur).

L'écran LCD est installé dans l'étage éponyme (étage intermédiaire).

L'étage Mood Box, quant à lui, comprend les composants et éléments suivants.

Sur les parois de la boîte sont fixés :

- le moteur servant à animer la figurine avec sa carte contrôleur (la carte contrôleur est visible sur la gauche de la boîte sur la photo) ;
- le lecteur mp3 dans lequel est inséré la carte microSD contenant les titres de la liste de lecture (en bas sur la photo) ;
- la touche capacitive sur laquelle vient se brancher la barre de laiton (composant et fil de connexion orange visibles à droite sur la photo).

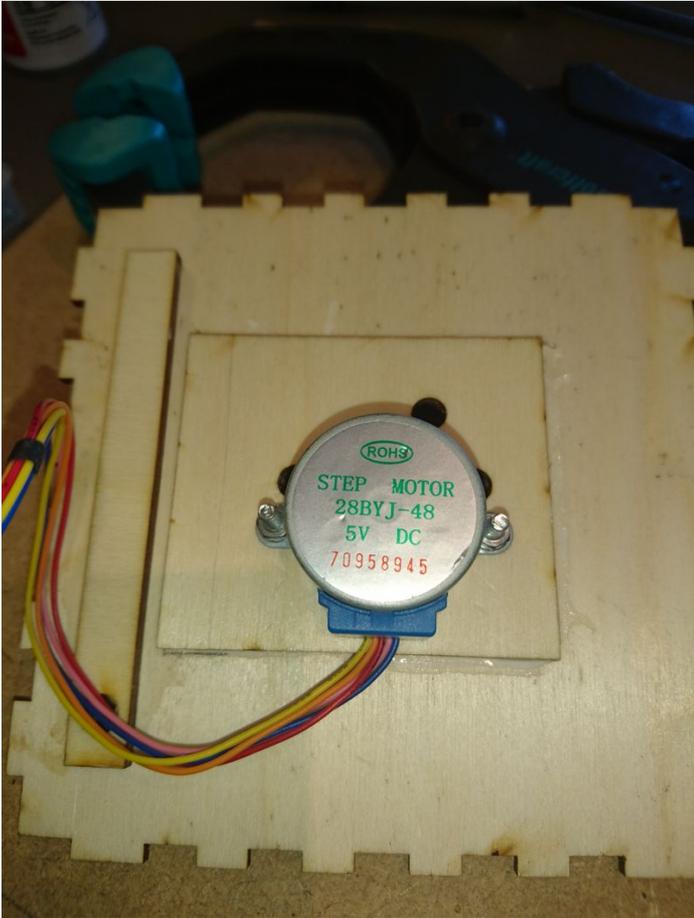
Dans le couvercle de cet étage sont fixés :

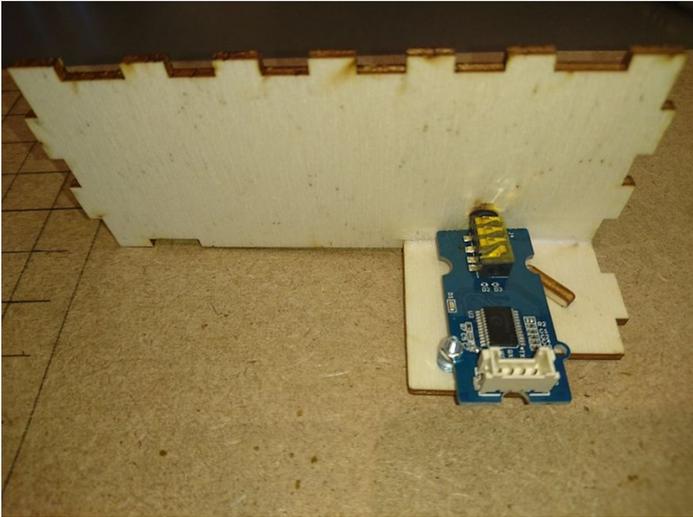
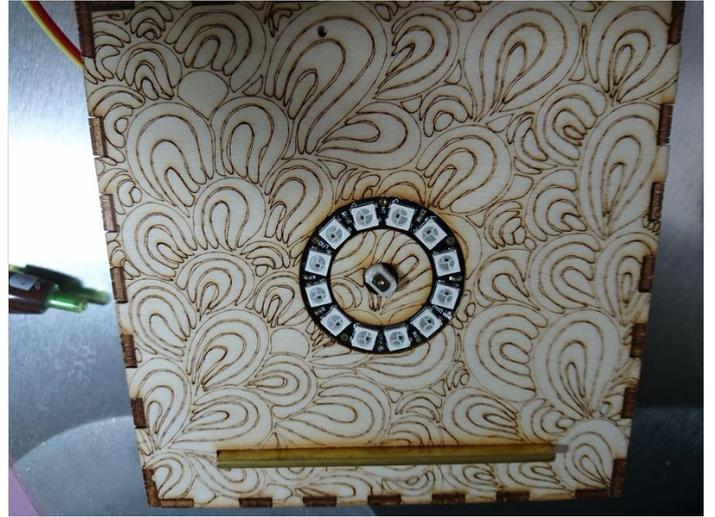
- l'anneau de LEDs ;
- le moteur mais seul son axe est visible de l'extérieur, au centre de l'anneau de LEDs ;
- la barre de laiton.

Sur le couvercle de l'étage sont fixés :

- le support de la figurine, juste au-dessus de l'anneau de LEDs pour en diffuser la lumière (et la rendre moins agressive à l'oeil étant donné la puissance des LEDs) ;
- la figurine elle-même, sur l'axe du moteur de façon à être entraînée lorsque celui-ci tourne.







Étape 8 - Now let's dance!

L'étage Mood Box est désormais terminé et fonctionnel.

==== **Activation de la Mood Box et comportement** ====

La Mood Box est activée lorsque l'utilisateur appuie sur la barre de laiton située au sommet de la boîte (cette barre est une extension de la touche capacitive installée à l'intérieur de la boîte).

Cette activation est accompagnée d'une animation conjointe de la figurine, de l'anneau de LEDs et de l'écran LCD.

⚠ Si aucun autre appui n'est effectué sur la barre dans les 5 secondes suivant l'activation, la Mood Box est désactivée et le mode veille reprend.

Un nouvel appui déclenche le mode lecture.

💡 Pour faciliter l'utilisation de la barre, un affichage itératif de leds permet un comptage visuel direct des "clics" effectués.

La première fois, la lecture de la musique démarre au tout début de la liste de lecture.

Le bouton du potentiomètre est alors actif et permet de régler le volume de la musique jouée.

La barre reste quant à elle disponible pour agir sur la musique jouée.

Pendant la lecture, si la barre est touchée :

- 1 fois : "pause/play", i.e. la musique est mise en pause ou en lecture suivant l'état précédent, avec un affichage spécifique sur l'écran LCD ;
- 2 fois : "next", i.e. la lecture passe au titre suivant de la liste de lecture ;
- 3 fois : "previous", i.e. la lecture repasse au titre précédent de la liste de lecture ;
- 4 fois ou plus : retour au mode "veille", avec une animation conjointe de l'anneau de LEDs, de la figurine et de l'écran LCD.

Remarque : chacune des actions pause/play/next/previous génère un changement de rotation de la figurine.

Pendant la lecture, l'anneau de LEDs est en mode "arc-en-ciel" et la figurine tourne sur elle-même.

==== **Désactivation de la Mood Box** ====

Après activation de la Mood Box ou après l'arrêt volontaire de la musique, si aucun appui n'est effectué dans les 5 secondes, le mode veille reprend et la Mood Box est désactivée.

Il est à noter que si l'on relance la lecture depuis le mode "veille", celle-ci va reprendre exactement là où on l'avait interrompue. Merci pour votre intérêt :)



Notes et références

La figurine utilisée pour la présentation finale est issue de Thingiverse (<https://www.thingiverse.com/thing:570654>).

Elle a été utilisée faute d'avoir eu le temps suffisant pour recréer en 3D le couple formé par l'hippopotame Hyacinth et son cavalier Ben Ali Gator dans Fantasia.

Projet "bentolux" accessible sur le site (à partir de juillet 2018).