



# BENTO BOX - SPACE FLAN

Une machine neurogastronomique permettant le gobbage d'un flan lacté salé dans une ambiance lumineuse et sonore de décollage de fusée.

Les prémisses d'un voyage sensoriel amenant le "Mangeur" dans les enjeux d'une conquête spatiale à la fois parodique, transgressive et délicieuse.

Projet réalisé pour le troisième étage de ma Bentolux pendant la formation de l'IMT Mines Alès entre septembre 2021 et février 2022.

 Difficulté **Moyen**

 Durée **10 heure(s)**

 Catégories **Art, Alimentation & Agriculture**

 Coût **80 EUR (€)**

## Sommaire

Introduction

Video d'introduction

Étape 1 - Découpe des éléments

Étape 2 - Construction du circuit

Étape 3 - Création du code

Étape 4 - Peinture de la structure

Étape 5 - Impression 3D du support pour l'afficheur Oled

Étape 6 - Montage final des étages de la boite

Étape 7 - Réalisation de flans

Étape 8 - Lancement !

Commentaires

## Introduction

Fermez les yeux ! Vous entrez dans un restaurant.

Mais, en est-ce vraiment un ? Vous commencez à en douter quand un serveur vous invite à le suivre dans une salle plongée dans la pénombre. Il vous accompagne jusqu'à votre place.

Maintenant assis confortablement, il commence à vous raconter le début d'une histoire dont vous serez le héros.

Il s'éloigne ensuite en vous prévenant que le départ pour ce voyage immersif est imminent. Vous entendez dans la salle le son lointain de techniciens échangeant sur la configuration d'une mise à feu.

Le serveur revient et pose devant vous une assiette contenant en son centre une drôle de structure surmontée d'une sorte de flan. Il vous indique que tout est prêt pour le décollage. Il vous remet un ordre de mission, vous le lisez et savez désormais ce qu'il vous reste à faire.

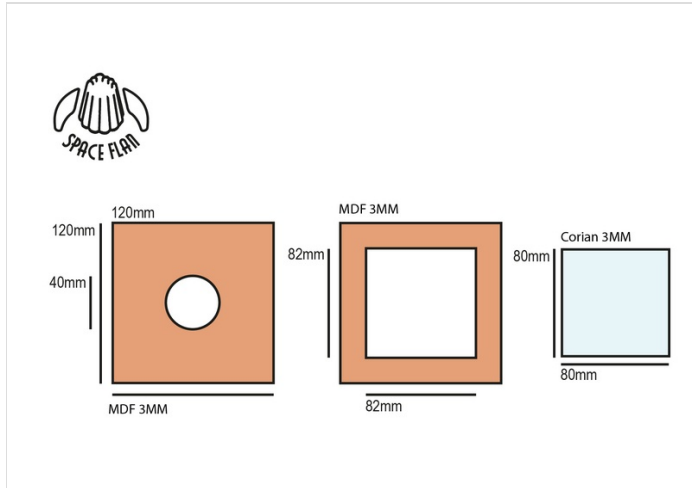
Aurez-vous le courage d'appuyer sur le bouton ?

## Matériaux

## Outils

# Étape 1 - Découpe des éléments

- Découpe au laser des deux plaques de MDF 3mm
- Découpe à la scie sauteuse de la plaque de Corian 3mm



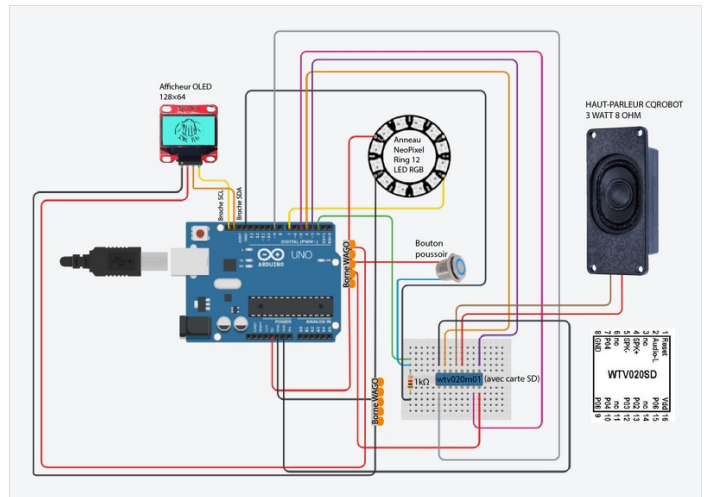
# Étape 2 - Construction du circuit

## En entrée :

- Un capteur analogique "bouton poussoir"

## En sortie :

- Un module sonore wtv020m01 dans lequel est insérer une carte SD 1 Go avec un son au format AD4 connecté à un Haut parleur
- Un anneau NeoPixel Ring 12 LED RGB
- Un afficheur OLED 128x64



# Étape 3 - Création du code

```
// Variables qui ne peuvent être modifiées,
const int buttonPin = 2; // Bouton poussoir
const int ledPin = 7; // Anneau NeoPixel Ring 12 LED RGB

// Bibliothèque utilisée pour écran OLED
#include <Adafruit_SSD1306.h>

// =====
// Paramètres écran OLED
// =====
#define nombreDePixelsEnLargeur 128 // Taille de l'écran OLED, en pixel, au niveau de sa largeur
#define nombreDePixelsEnHauteur 64 // Taille de l'écran OLED, en pixel, au niveau de sa hauteur
#define brocheResetOLED -1 // Reset de l'OLED partagé avec l'Arduino (d'où la valeur à -1, et non un numéro de pin)
#define adresseI2CecranOLED 0x3C // Adresse de "mon" écran OLED sur le bus i2c (généralement égal à 0x3C ou 0x3D)
Adafruit_SSD1306 écranOLED(nombreDePixelsEnLargeur, nombreDePixelsEnHauteur, &Wire, brocheResetOLED);

// =====
// Image à afficher
// =====
#define largeurDeLimage 128 // Largeur de l'image à afficher, en pixels
#define hauteurDeLimage 64 // Hauteur de l'image à afficher, en pixels

const unsigned char imageAafficher [] PROGMEM = {
// Lien SPACE FLAN (image BITMAP) / CD Assistant / Éditeur de texte / https://www.electronicart.fr/espaces-3d/3d-objets/
```

```
// Logo SPACE FLAIN (image BITMAP / LCD Assistant / Eclaireur de texte / https://passionelectronique.fr/ecran-bleu-izc-arduino/)
```

```
0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff,
0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0x07, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff,
0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xfc, 0x03, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff,
0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xf0, 0xf0, 0x3f, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff,
0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xe0, 0xf0, 0x0f, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff,
0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xc4, 0x73, 0x83, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff,
0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0x0e, 0x77, 0x03, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff,
0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xef, 0x06, 0xf7, 0x39, 0xe7, 0xff, 0xff, 0xff, 0xff, 0xff,
0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0x06, 0x63, 0xff, 0x30, 0xe0, 0xff, 0xff, 0xff, 0xff, 0xff,
0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xc, 0x26, 0x73, 0xff, 0xe0, 0xe4, 0x3f, 0xff, 0xff, 0xff, 0xff,
0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xf8, 0xe6, 0xff, 0xff, 0xe6, 0x67, 0x1f, 0xff, 0xff, 0xff, 0xff,
0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xe3, 0xec, 0xff, 0xff, 0xe6, 0x67, 0x87, 0xff, 0xff, 0xff, 0xff,
0xff, 0xff, 0xff, 0xff, 0xff, 0xc7, 0xcc, 0xff, 0xff, 0xfe, 0x77, 0xe3, 0xff, 0xff, 0xff, 0xff,
0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0x9f, 0xcc, 0xff, 0xff, 0xfe, 0x33, 0xf1, 0xff, 0xff, 0xff, 0xff,
0xff, 0xff, 0xff, 0xff, 0xff, 0x3f, 0xd9, 0xff, 0xff, 0xfe, 0x33, 0xfc, 0xff, 0xff, 0xff, 0xff,
0xff, 0xff, 0xff, 0xff, 0xfe, 0x7f, 0x99, 0x7f, 0xff, 0xfe, 0x33, 0xfe, 0x7f, 0xff, 0xff, 0xff,
0xff, 0xff, 0xff, 0xff, 0xfc, 0xff, 0x99, 0x7f, 0xff, 0xff, 0x3b, 0xff, 0x3f, 0xff, 0xff, 0xff,
0xff, 0xff, 0xff, 0xff, 0xf9, 0xff, 0x9a, 0x7f, 0xff, 0xf9, 0x19, 0xff, 0x1f, 0xff, 0xff, 0xff,
0xff, 0xff, 0xff, 0xff, 0xf3, 0xff, 0xb2, 0x77, 0xff, 0x79, 0x19, 0xff, 0x9f, 0xff, 0xff, 0xff,
0xff, 0xff, 0xff, 0xff, 0xf3, 0xff, 0x32, 0x67, 0xff, 0x39, 0x1d, 0xff, 0xcf, 0xff, 0xff, 0xff,
0xff, 0xff, 0xff, 0xff, 0xe7, 0xff, 0x32, 0xe7, 0xdf, 0x39, 0x9d, 0xff, 0xef, 0xff, 0xff, 0xff,
0xff, 0xff, 0xff, 0xff, 0xc, 0xff, 0x74, 0xe7, 0xdf, 0x3d, 0x8c, 0xff, 0xe7, 0xff, 0xff, 0xff,
0xff, 0xff, 0xff, 0xff, 0xc, 0xff, 0x64, 0xe7, 0xdf, 0x3d, 0x8c, 0xff, 0xf7, 0xff, 0xff, 0xff,
0xff, 0xff, 0xff, 0xff, 0xdf, 0xfe, 0x64, 0xe7, 0xdf, 0x3c, 0x8e, 0xff, 0xf3, 0xff, 0xff, 0xff,
0xff, 0xff, 0xff, 0xff, 0x9f, 0xfe, 0x64, 0xe7, 0xdf, 0x3c, 0x86, 0x7f, 0xf3, 0xff, 0xff, 0xff,
0xff, 0xff, 0xff, 0xff, 0x9f, 0xfc, 0xed, 0xef, 0xdf, 0xbc, 0xc6, 0x7f, 0xf9, 0xff, 0xff, 0xff,
0xff, 0xff, 0xff, 0xff, 0xbf, 0xfc, 0xc9, 0xc, 0xdf, 0xbc, 0xc7, 0x3f, 0xf9, 0xff, 0xff, 0xff,
0xff, 0xff, 0xff, 0xff, 0x3f, 0xf1, 0xc9, 0xc, 0x9f, 0x9e, 0xc7, 0x1f, 0xf9, 0xff, 0xff, 0xff,
0xff, 0xff, 0xff, 0xff, 0x3f, 0xe3, 0xc9, 0xc, 0x9f, 0x9e, 0xc3, 0x8f, 0xfd, 0xff, 0xff, 0xff,
0xff, 0xff, 0xff, 0xff, 0x7f, 0xc7, 0xdb, 0xc, 0x9f, 0x9e, 0x63, 0xe7, 0xfd, 0xff, 0xff, 0xff,
0xff, 0xff, 0xff, 0xff, 0x7f, 0x8f, 0x93, 0xc, 0x9f, 0x9e, 0x63, 0xf3, 0xfd, 0xff, 0xff, 0xff,
0xff, 0xff, 0xff, 0xff, 0x7f, 0x1f, 0x93, 0xc, 0x9f, 0x9e, 0x63, 0xf9, 0xfd, 0xff, 0xff, 0xff,
0xff, 0xff, 0xff, 0xff, 0x7f, 0x3f, 0x93, 0xdf, 0x9f, 0x9e, 0x63, 0xfc, 0xfd, 0xff, 0xff, 0xff,
0xff, 0xff, 0xff, 0xff, 0x7e, 0x7f, 0xb7, 0xdf, 0x9f, 0x9f, 0x73, 0xfc, 0xfd, 0xff, 0xff, 0xff,
0xff, 0xff, 0xff, 0xff, 0x7c, 0xff, 0xa7, 0x9f, 0x9f, 0x9f, 0x33, 0xfe, 0x7d, 0xff, 0xff, 0xff,
0xff, 0xff, 0xff, 0xff, 0x7c, 0xff, 0x87, 0x9f, 0x9f, 0x9f, 0x33, 0xfe, 0x7d, 0xff, 0xff, 0xff,
0xff, 0xff, 0xff, 0xff, 0x79, 0xff, 0x87, 0x9f, 0x9f, 0x9f, 0x33, 0xff, 0x3d, 0xff, 0xff, 0xff,
0xff, 0xff, 0xff, 0xff, 0x79, 0xff, 0xe7, 0x9f, 0x9f, 0xdf, 0x23, 0xff, 0x3d, 0xff, 0xff, 0xff,
0xff, 0xff, 0xff, 0xff, 0x7b, 0xff, 0xe7, 0x9f, 0x9f, 0xdf, 0x07, 0xff, 0x9d, 0xff, 0xff, 0xff,
0xff, 0xff, 0xff, 0xff, 0x73, 0xff, 0xe7, 0xbf, 0x9f, 0xdf, 0x8f, 0xff, 0x9d, 0xff, 0xff, 0xff,
0xff, 0xff, 0xff, 0xff, 0x33, 0xff, 0xe3, 0x3f, 0x9f, 0xc, 0x9f, 0xff, 0xc9, 0xff, 0xff, 0xff,
0xff, 0xff, 0xff, 0xff, 0x87, 0xff, 0xf0, 0x3f, 0x9f, 0xc, 0x3f, 0xff, 0xc1, 0xff, 0xff, 0xff,
0xff, 0xff, 0xff, 0xff, 0xff, 0xf8, 0xfc, 0x3f, 0x9f, 0xcc, 0x3e, 0x3f, 0xff, 0xff, 0xff, 0xff,
0xff, 0xff, 0xff, 0xff, 0xff, 0xf2, 0x7f, 0x9f, 0x9f, 0xc0, 0x7c, 0x9f, 0xff, 0xff, 0xff, 0xff,
0xff, 0xff, 0xff, 0xff, 0xff, 0xf7, 0xff, 0xc0, 0x0f, 0x01, 0xfc, 0xc, 0xff, 0xff, 0xff, 0xff,
0xff, 0xff, 0xff, 0xff, 0xff, 0xf7, 0x8f, 0xe0, 0x00, 0x3f, 0xee, 0x67, 0xff, 0xff, 0xff, 0xff,
0xff, 0xff, 0xff, 0xff, 0xff, 0xf7, 0x8f, 0xfd, 0xe0, 0x7f, 0xc3, 0x37, 0xff, 0xff, 0xff, 0xff,
0xff, 0xff, 0xff, 0xff, 0xff, 0xf7, 0x27, 0xff, 0xff, 0xff, 0x93, 0xb3, 0xff, 0xff, 0xff, 0xff,
0xff, 0xff, 0xff, 0xff, 0xff, 0xf7, 0x64, 0x7f, 0xff, 0xfb, 0x99, 0x99, 0xff, 0xff, 0xff, 0xff,
0xff, 0xff, 0xff, 0xff, 0xfe, 0xe4, 0x69, 0x3f, 0xff, 0xcb, 0xdd, 0xcc, 0xff, 0xff, 0xff, 0xff,
0xff, 0xff, 0xff, 0xff, 0xfe, 0x0c, 0x0b, 0x3c, 0x47, 0x89, 0xcc, 0xe6, 0xff, 0xff, 0xff, 0xff,
0xff, 0xff, 0xff, 0xff, 0xff, 0x1c, 0x13, 0x30, 0xc7, 0x9d, 0xec, 0x77, 0xff, 0xff, 0xff, 0xff,
0xff, 0xff, 0xff, 0xff, 0xff, 0xfd, 0xf3, 0x73, 0xdf, 0x9c, 0xe0, 0x73, 0xff, 0xff, 0xff, 0xff,
0xff, 0xff, 0xff, 0xff, 0xff, 0xf9, 0xe7, 0x67, 0xdf, 0xdc, 0xe3, 0x7f, 0xff, 0xff, 0xff, 0xff,
0xff, 0xff, 0xff, 0xff, 0xff, 0xf3, 0xe2, 0x6f, 0xdf, 0xc6, 0xe3, 0x3f, 0xff, 0xff, 0xff, 0xff,
0xff, 0xff, 0xff, 0xff, 0xff, 0xf3, 0xe0, 0xc, 0xdf, 0xc6, 0x7b, 0xbf, 0xff, 0xff, 0xff, 0xff,
0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xec, 0x4f, 0xc, 0xce, 0x79, 0xff, 0xff, 0xff, 0xff, 0xff,
0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xcc, 0xc, 0xc7, 0xc, 0x7d, 0xff, 0xff, 0xff, 0xff, 0xff,
0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xdd, 0xc, 0xc7, 0xef, 0x0f, 0xff, 0xff, 0xff, 0xff, 0xff,
0xff, 0xff, 0xff, 0xff, 0xff, 0xf9, 0xe7, 0xdf, 0xef, 0x0f, 0xff, 0xff, 0xff, 0xff, 0xff,
0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xe7, 0xdf, 0xe7, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff,
0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xf1, 0xc7, 0xe7, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff,
0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xf9, 0xc7, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff,
0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff, 0xff
```

```
};
```

```
// Array of all bitmaps for convenience. (Total bytes used to store images in PROGMEM = 1040)
```

```
const int epd_bitmap_allArray_LEN = 1;
```

```
***** Routine pour afficheur LCD *****
```

```
#if (SSD1306_LCDHEIGHT != 64)
```

```
#endif
```

```

// ANNEAU RING LED 12 PIXELS
#include <Adafruit_NeoPixel.h>
#define PIN 7

// Parameter 1 = number of pixels in strip
// Parameter 2 = pin number (most are valid)
// Parameter 3 = pixel type flags, add together as needed:
// NEO_KHZ800 800 KHz bitstream (most NeoPixel products w/WS2812 LEDs)
// NEO_KHZ400 400 KHz (classic 'v1' (not v2) FLORA pixels, WS2811 drivers)
// NEO_GRB Pixels are wired for GRB bitstream (most NeoPixel products)
// NEO_RGB Pixels are wired for RGB bitstream (v1 FLORA pixels, not v2)
Adafruit_NeoPixel strip = Adafruit_NeoPixel(12, PIN, NEO_GRB + NEO_KHZ800);

// Module sonore wtv020m01
// boolean déclare une variable de type binaire
boolean boutonWasUp = true;
boolean ledEnabled = false;

const int clockPin = 6; // clockpin sur la broche 6
const int dataPin = 9; // datapin sur la broche 9
const int resetPin = 3; // resetpin sur la broche 3

const unsigned int VOLUME_7 = 0xFFFF7; //unsigned = variable entière non signée

const unsigned int PLAY_PAUSE = 0xFFFE;
const unsigned int STOP = 0xFFFF;

void setup()
{
  Serial.begin(9600); //Initialise la communication entre le PC et Arduino

  // Initialisation de l'écran OLED
  if(!ecranOLED.begin(SSD1306_SWITCHCAPVCC, adresse12CecranOLED))
    while(1); // Arrêt du programme (boucle infinie) en cas d'échec de l'initialisation

  // Affichage d'une image au centre de l'écran
  ekranOLED.clearDisplay(); // Effaçage de la mémoire tampon de l'écran OLED

  ekranOLED.drawBitmap(
    (ekranOLED.width() - largeurDeLimage) / 2, // Position de l'extrême "gauche" de l'image (pour centrage écran, ici)
    (ekranOLED.height() - hauteurDeLimage) / 2, // Position de l'extrême "haute" de l'image (pour centrage écran, ici)
    imageAafficher,
    largeurDeLimage,
    hauteurDeLimage,
    WHITE); // "couleur" de l'image

  ekranOLED.display(); // Transfert de la mémoire tampon à l'écran OLED, pour affichage

{
  // LedPin en sortie
  pinMode(ledPin, OUTPUT);
  // Bouton poussoir en entrée
  pinMode(buttonPin, INPUT);
}

{
  strip.begin();
  strip.setBrightness(255); //adjust brightness here, maximum à 255
  strip.show(); // Initialize all pixels to 'off'
}

pinMode(clockPin, OUTPUT);
pinMode(dataPin, OUTPUT);
pinMode(resetPin, OUTPUT);

```

```

digitalWrite(clockPin, HIGH); // aucune différence si je le met en HIGH ou LOW
digitalWrite(dataPin, LOW);

// reset the module (si les 2 lignes dessous retirer le son ne s'allume qu'une fraction de secondes)
digitalWrite(resetPin, HIGH);
delay(100);

sendCommand(VOLUME_7); // Empêche le son de se répéter, si enlevé le son est en boucle

}

void loop()
{

// Programme de lumière LED de 12 secondes
colorWipe(strip.Color(255, 255, 255), 84); // Blanc
colorWipe(strip.Color(255, 0, 0), 84); // Rouge
colorWipe(strip.Color(255, 255, 255), 84); // Blanc
colorWipe(strip.Color(255, 0, 0), 84); // Rouge
colorWipe(strip.Color(255, 255, 255), 84); // Blanc
colorWipe(strip.Color(255, 0, 0), 84); // Rouge
colorWipe(strip.Color(255, 255, 255), 84); // Blanc
colorWipe(strip.Color(255, 0, 0), 84); // Rouge
colorWipe(strip.Color(255, 255, 255), 5); // Blanc
colorWipe(strip.Color(255, 0, 0), 5); // Rouge
colorWipe(strip.Color(255, 255, 255), 5); // Blanc
colorWipe(strip.Color(255, 0, 0), 5); // Rouge
colorWipe(strip.Color(255, 255, 255), 5); // Blanc
colorWipe(strip.Color(255, 0, 0), 5); // Rouge
colorWipe(strip.Color(255, 255, 255), 5); // Blanc
colorWipe(strip.Color(255, 0, 0), 5); // Rouge
colorWipe(strip.Color(255, 255, 255), 5); // Blanc
colorWipe(strip.Color(255, 0, 0), 5); // Rouge
colorWipe(strip.Color(255, 255, 255), 5); // Blanc
colorWipe(strip.Color(255, 0, 0), 5); // Rouge
colorWipe(strip.Color(255, 255, 255), 5); // Blanc
colorWipe(strip.Color(255, 0, 0), 5); // Rouge
colorWipe(strip.Color(255, 255, 255), 5); // Blanc
colorWipe(strip.Color(255, 0, 0), 5); // Rouge
colorWipe(strip.Color(255, 255, 255), 5); // Blanc
colorWipe(strip.Color(255, 0, 0), 5); // Rouge
colorWipe(strip.Color(255, 255, 255), 5); // Blanc
colorWipe(strip.Color(255, 0, 0), 5); // Rouge
colorWipe(strip.Color(255, 255, 255), 5); // Blanc
colorWipe(strip.Color(255, 0, 0), 5); // Rouge
colorWipe(strip.Color(255, 255, 255), 5); // Blanc
colorWipe(strip.Color(255, 0, 0), 5); // Rouge

colorWipe(strip.Color(0, 0, 0), 5); // Eteindre

while (1);
}

// Remplir les points l'un après l'autre avec une couleur (si supprimé, système de points ne fonctionne plus)
void colorWipe(uint32_t c, uint8_t wait)
{
for(uint16_t i=0; i<strip.numPixels(); i++) {
strip.setPixelColor(i, c);
strip.show();
delay(wait);
}
}
}

```

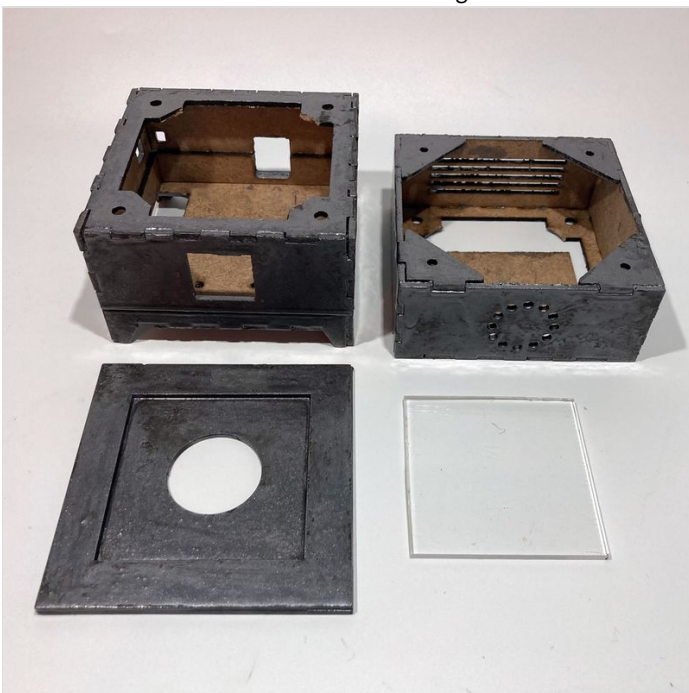
```
uint32_t Wheel(byte WheelPos) // je ne sais pas à quoi ça sert
```

```
{  
  // lecture son "0000.wav"  
  sendCommand(0x0001);  
}  
  
void sendCommand(int addr) {  
  digitalWrite(clockPin, LOW);  
  delay(2);  
  for (int i=15; i>=0; i--)  
  {  
    delayMicroseconds(50);  
    if((addr>>i)&0x0001 >0)  
    {  
      digitalWrite(dataPin, HIGH);  
      //Serial.print(1);  
    }  
    else  
    {  
      digitalWrite(dataPin, LOW);  
      // Serial.print(0);  
    }  
    delayMicroseconds(50);  
    digitalWrite(clockPin, HIGH);  
    delayMicroseconds(50);  
  
    if(i>0)  
    digitalWrite(dataPin, LOW);  
    else  
    digitalWrite(dataPin, HIGH);  
    delayMicroseconds(50);  
  
    if(i>0)  
    digitalWrite(clockPin, LOW);  
    else  
    digitalWrite(clockPin, HIGH);  
    delay(20);  
  }  
}
```

---

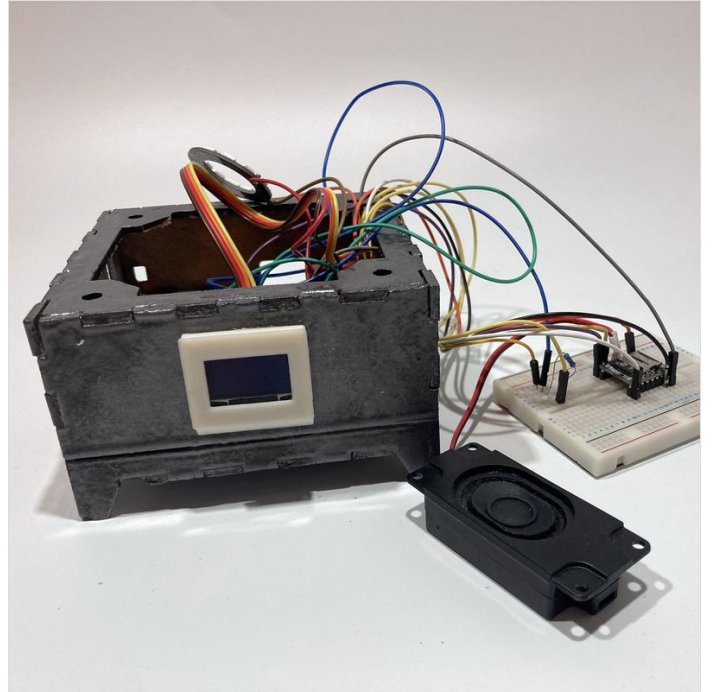
## Étape 4 - Peinture de la structure

Passer deux couches de résine effet métal argenté sur la structure



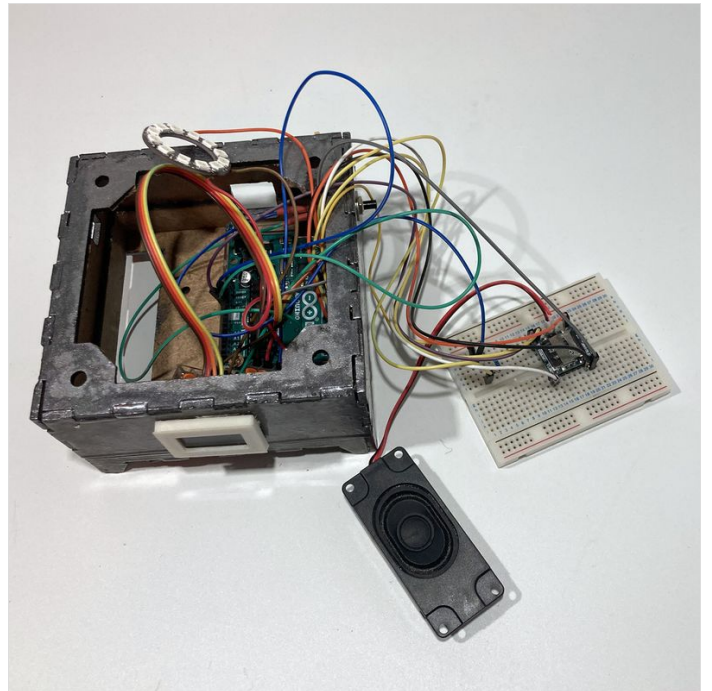
---

## Étape 5 - Impression 3D du support pour l'afficheur Oled



---

## Étape 6 - Montage final des étages de la boîte

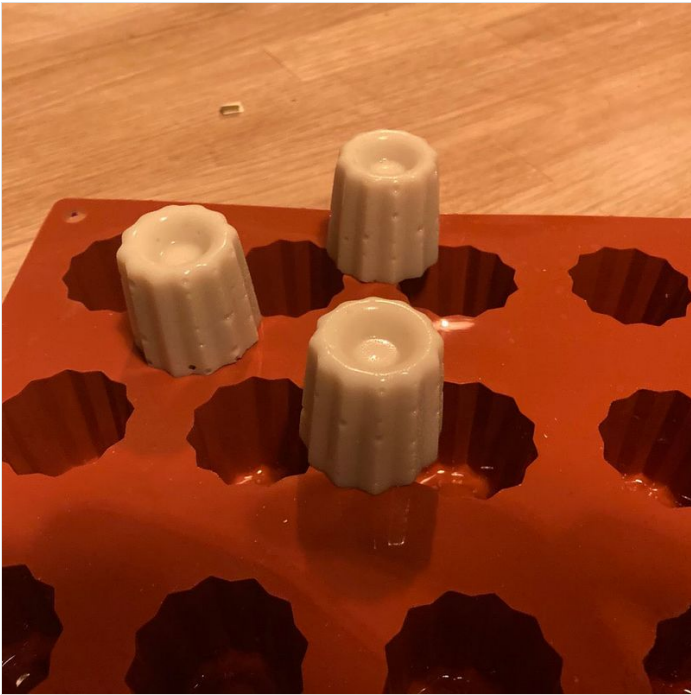


---

## Étape 7 - Réalisation de flans

Réalisez des flans sucrés ou salés dans des moules d'environ 4cm de diamètre.







## Étape 8 - Lancement !

Lancement de la séquence après avoir posé le flan au centre de la plaque de Corian.

