Fabrication 3ème étage Bentolux

Le troisième étage de notre Bentolux, une station météo que nous avons créer durant notre formation à la Fabrication Hybride et au prototypage rapide

Difficulté **Facile** | Durée **2 jour(s)** | Catégories **Électronique** | Coût **35 EUR (€)**

Sommaire

Introduction

Pour notre formation au prototypage, nous avons du créer le troisième étage d'une boite que nous avons construite durant 6 mois. Cette boite permet de vérifier la qualité de l'air dans la pièce, à l'aide d'un capteur BME280, qui mesure la température, l'humidité, et la pression atmosphérique. Un écran LCD d'environ 3cm de diagonale nous affiche ces valeurs, tandis qu'un LED Ring s'allumera en rouge lorsque le taux d'humidité dans la pièce est trop important.

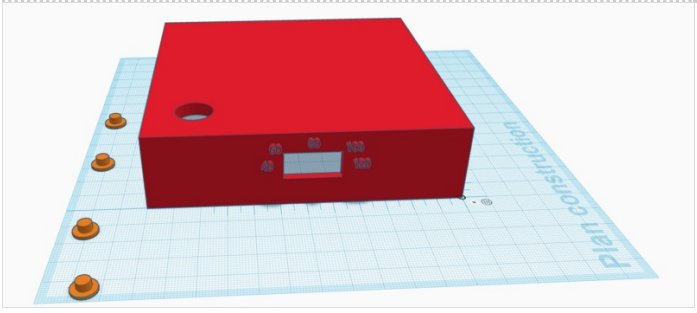
Pour le troisième étage, nous avions la possibilité de faire ce qui nous plaisait, même si ce n'était pas raccord avec la station météo. J'ai alors choisi d'y insérer un capteur de pulsation cardiaque, ainsi qu'un servomoteur. Ce dernier aura comme fonction de tourner en accord avec les pulsations captées par le capteur.

Matériaux

Outils

Étape 1 - Dessiner le 3ème étage

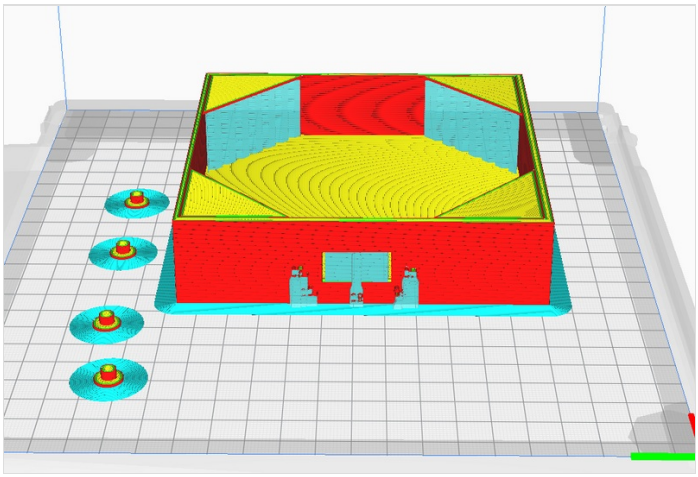
Premièrement, j'ai dessiner ,à l'aide de Tinkercad, ce à quoi aller ressembler mon 3ème étage. J'ai fait attention à laisser un trou sur le somme de l'étage pour laisse passer le capteur de pulsation, ainsi qu'un interstice sur un des côté pour laisser passer le servomoteur. J'ai également penser à dessiner les picot, qui me permettront d'assembler le 3ème et le 2ème étage ensemble sans avoir à les coller ensemble.



Étape 2 - Ultimaker Cura

J'ai utilisé le logiciel Ultimaker Cura comme trancheur, mais vous êtes libre d'utiliser celui qui vous convient le mieux.

J'ai arrangé les paramètres d'impression de façon à réduire le temps d'impression au maximum, ce qui à fait approximativement 9h49min, ce qui est déjà conséquent pour une pièce creuse de 3cm de haut, et 20cm de côté.



Profile: Fast - 0.2mm

Search settings

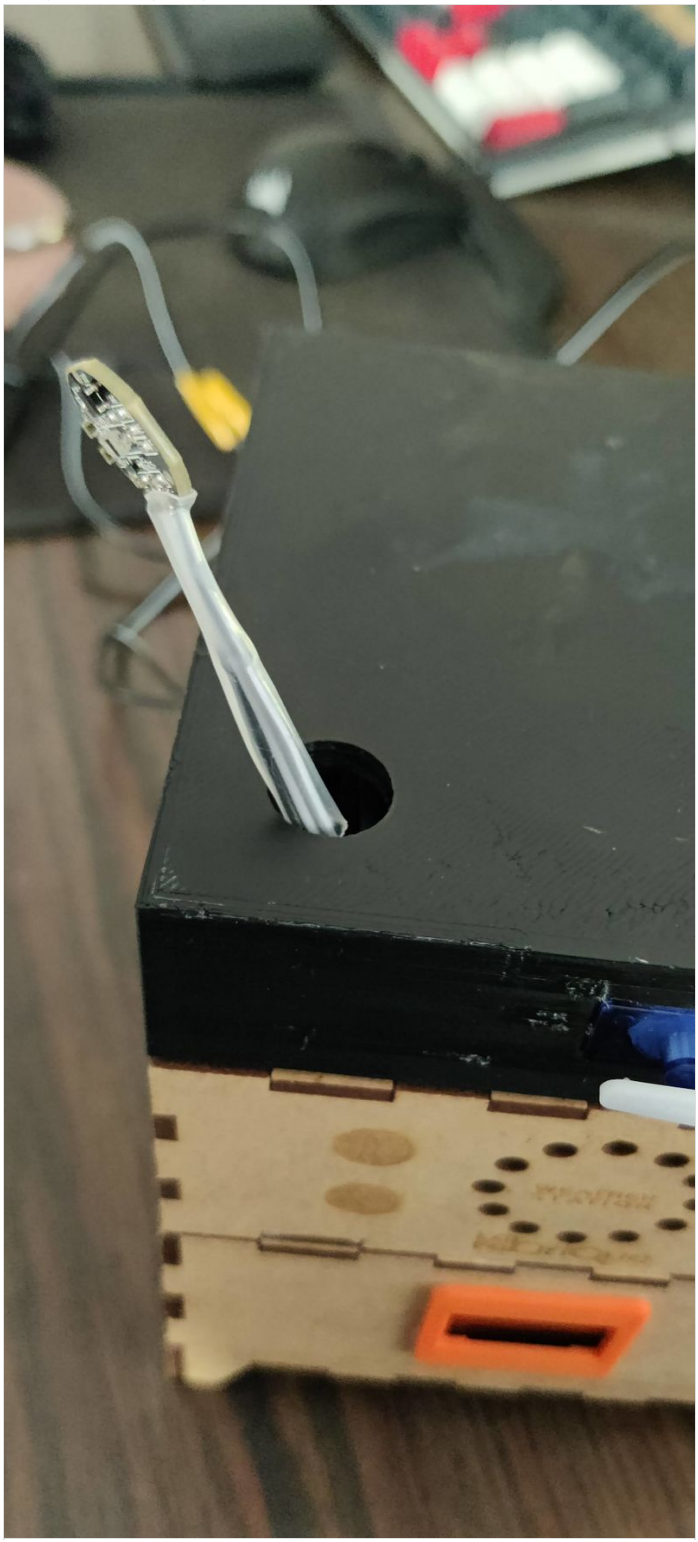
Quality		
Layer Height	0.2	mm
Walls		
Wall Thickness	1.0	mm
Wall Line Count	3	
Horizontal Expansion	0.0	mm
Top/Bottom		
Top/Bottom Thickness	1.2	mm
Top Thickness	1.2	mm
Top Layers	6	
Bottom Thickness	1.2	mm
Bottom Layers	6	
Infill		
Infill Density	20.0	%
Infill Pattern	Triangles	
Material		
Printing Temperature	240.0	°C
Build Plate Temperature	80	°C
Speed		
Print Speed	60.0	mm/s
Travel		
Enable Retraction	✓	
Z Hop When Retracted	✓	
Cooling		
Enable Print Cooling	✓	
Fan Speed	5.0	%
Support		
Generate Support	✓	
Support Extruder	Extruder 1	
Support Placement	Everywhere	
Support Overhang Angle	60.0	°
Build Plate Adhesion		
Enable Prime Blob	✓	
Build Plate Adhesion Type	Brim	
Build Plate Adhesion Extruder	Extruder 1	
Dual Extrusion		
Enable Prime Tower		

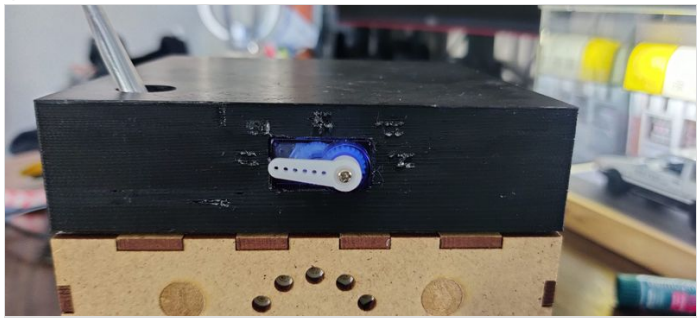
Étape 3 - Montage

Une fois que l'impression est terminée, il faut retirer les supports. Ces derniers étant en PLA, un bain d'eau froide pendant une dizaine d'heures suffit.

Le montage aurait du être plug and play, malheureusement j'ai oublié de prendre en compte la rétractation du plastic ABS, et j'ai du limer l'interstice pour le Servomoteur pour le faire rentrer. Les chiffre en façade n'était pas assez gros, et se sont aussi mal imprimé. J'ai préféré les couper avec un cutter.

Je n'avais pas prévu de faire passer le capteur de pulsation de cette manière, malheureusement c'est la seule façon si je veux pouvoir capter une pulsation, puisqu'il ne faut pas cacher le dos de la pièce si l'on ne veut pas avoir de résultats erronés.





Étape 4 - Le codage

Dernière étape, mais non des moindre, le code.

Puisqu'une image vaut mieux que 1000 mots, et qu'il ne sert à rien de réinventer la roue, je vous montre directement le code utilisé, que j'ai moi-même récupérer dans une vidéo trouvé sur Youtube

```
#include <Servo.h>
#define USE_ARDUINO_INTERRUPTS true //--> Set-up low-level interrupts for most acurate BPM math.
#include <PulseSensorPlayground.h> //--> Includes the PulseSensorPlayground Library.
Servo monServo; //On donne le nom monServo à un servomoteur
//-----------------------------------------Variable Declaration
const int PulseWire = 1; //--> Le capteur de pulsation est branché sur A1
int Threshold = 550; //--> Determine quel Signal je dois compter comme un battement et quel signal je dois ingnorer

//-----------------------------------------

PulseSensorPlayground pulseSensor; //--> Creates an instance of the PulseSensorPlayground object called "pulseSensor"

//--------------------------------------------------------------------------------void setup
void setup() {
  monServo.attach (3); //Commande du servomoteur en D3
  Serial.begin(9600); //--> Set's up Serial Communication at certain speed.

  //-----------------------------------------Configure the PulseSensor object, by assigning our variables to it.
  pulseSensor.analogInput(PulseWire);
  pulseSensor.setThreshold(Threshold);
  //-----------------------------------------

  //-----------------------------------------Double-check the "pulseSensor" object was created and "began" seeing a signal.
  if (pulseSensor.begin()) {
    Serial.println("We created a pulseSensor Object !"); //--> This prints one time at Arduino power-up,  or on Arduino reset.
  }
  //-----------------------------------------
}
//--------------------------------------------------------------------------------

//--------------------------------------------------------------------------------void loop
void loop() {
  int myBPM = pulseSensor.getBeatsPerMinute(); //--> Calls function on our pulseSensor object that returns BPM as an "int". "myBPM" hold this BPM value now.

  //-----------------------------------------Constantly test to see if "a beat happened".
  if (pulseSensor.sawStartOfBeat()) { //--> If test is "true", then the following conditions will be executed.
    Serial.println("♥  Oh ! Un battement de coeur ! "); //--> Print a message "a heartbeat happened".
    Serial.print("BPM: "); //--> Print phrase "BPM: "
    Serial.println(myBPM); //--> Print the value inside of myBPM.
  }
```

```
  //-----------------------------------------
  delay(20); //--> considered best practice in a simple sketch.
    if(myBPM < 40){
      monServo.write (180);
    }
    if((myBPM >= 41) && (myBPM<=49)){
      monServo.write (170);
    }
    if((myBPM >= 50) && (myBPM<=59)){
      monServo.write (160);
    }
    if((myBPM >= 60) && (myBPM<=69)){
      monServo.write (160);
    }
    if((myBPM >= 70) && (myBPM<=79)){
      monServo.write (150);
    }
    if((myBPM >= 80) && (myBPM<=89)){
      monServo.write (140);
    }
    if((myBPM >= 90) && (myBPM<=99)){
      monServo.write (130);
    }
    if((myBPM >= 100) && (myBPM<=109)){
      monServo.write (120);
    }
    if((myBPM >= 110) && (myBPM<=119)){
      monServo.write (110);
    }
    if((myBPM >= 120) && (myBPM<=129)){
      monServo.write (100);
    }
    if((myBPM >= 130) && (myBPM<=139)){
      monServo.write (90);
    }
    if((myBPM >= 140) && (myBPM<=149)){
      monServo.write (80);
    }
    if((myBPM >= 150) && (myBPM<=159)){
      monServo.write (70);
    }
    if((myBPM >= 160) && (myBPM<=169)){
      monServo.write (60);
    }
    if((myBPM >= 170) && (myBPM<=179)){
      monServo.write (50);
    }
    if((myBPM >= 180) && (myBPM<=189)){
      monServo.write (40);
    }
    if((myBPM >= 190) && (myBPM<=199)){
      monServo.write (30);
    }
}
```