CODE FILES MILD- TEAM 5

- **VIBRATION CODE ARDUINO FOR EARPHONE CASE MICROCONTROLLER**

```
#include "BluetoothSerial.h"
#define LEDC_CHANNEL_0     0
BluetoothSerial BtSerial; // Object for Bluetooth
bool IsBluetoothOn=false;

// use 12 bit precission for LEDC timer
#define LEDC_TIMER_12_BIT  12

// use 5000 Hz as a LEDC base frequency
#define LEDC_BASE_FREQ     5000

// fade LED PIN (replace with LED_BUILTIN constant for built-in LED)
#define LED_PIN           LED_BUILTIN
#define Motor_PIN         0

int brightness = 0;    // how bright the LED is
int fadeAmount = 5;    // how many points to fade the LED by

// Arduino like analogWrite
// value has to be between 0 and valueMax
void ledcAnalogWrite(uint8_t channel, uint32_t value, uint32_t valueMax = 255) {
  // calculate duty, 4095 from 2 ^ 12 - 1
  uint32_t duty = (4095 / valueMax) * min(value, valueMax);

  // write duty to LEDC
  ledcWrite(channel, duty);
}

void setup() {
  // Setup timer and attach timer to a led pin


  ledcSetup(LEDC_CHANNEL_0, LEDC_BASE_FREQ, LEDC_TIMER_12_BIT);

  /*
  ledcAttachPin(LED_PIN, LEDC_CHANNEL_0);
  pinMode(Motor_PIN,OUTPUT);
  digitalWrite(Motor_PIN, LOW);
 */


  ledcAttachPin(Motor_PIN, LEDC_CHANNEL_0);
  pinMode(LED_PIN,OUTPUT);
```

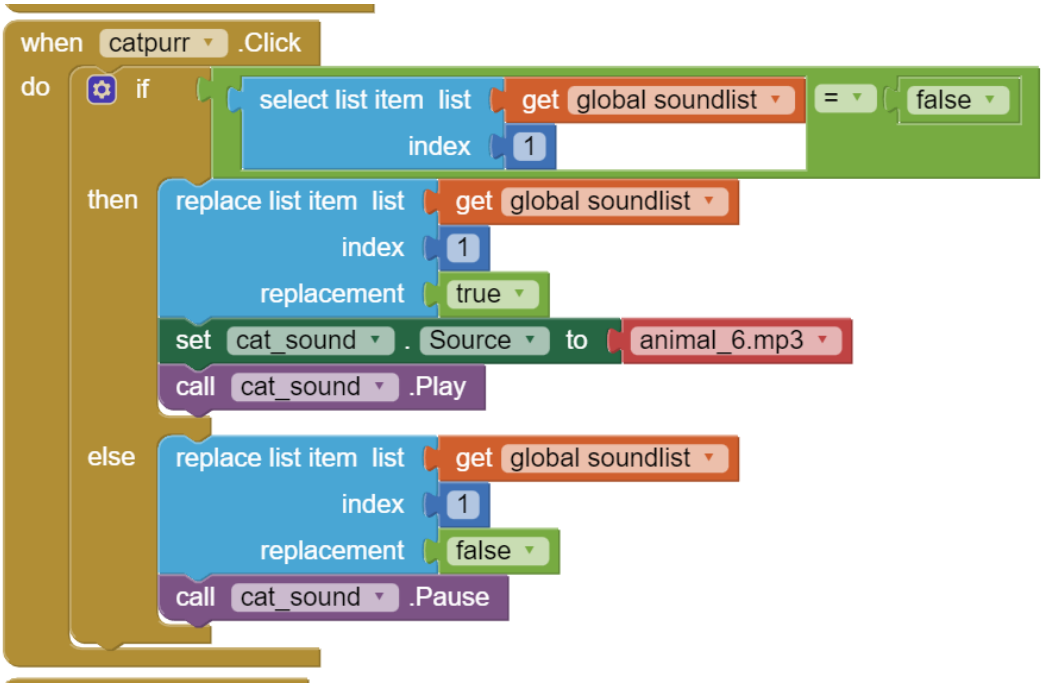
```
  digitalWrite(LED_PIN, LOW);


  Serial.begin(115200); // Start Serial monitor
  BtSerial.begin("ESP32_LED_Control"); // Name of Bluetooth Module
  Serial.println("Bluetooth Device is Ready to Pair");


}

void loop() {
  // set the brightness on LEDC channel 0
    if (BtSerial.available()) { // Check if we receive anything from Bluetooth
      char incoming = BtSerial.read(); //Read what we recevive
      Serial.print("Received:");
      Serial.println(incoming);

      if (incoming == '1') {
         IsBluetoothOn=true;
         BtSerial.println("LED or Motor turned ON");
         brightness = 0;
      }
      else if (incoming == '0') {
         IsBluetoothOn=false;
         BtSerial.println("LED or Motor turned OFF");
         brightness=0;
         ledcAnalogWrite(LEDC_CHANNEL_0, brightness);
      }
  }
    if(IsBluetoothOn){
      // change the brightness for next time through the loop:
      brightness = brightness + fadeAmount;
      Serial.println(brightness);
      // reverse the direction of the fading at the ends of the fade:
      if (brightness < 0 || brightness >= 255) {
        fadeAmount = -fadeAmount;
      }
      ledcAnalogWrite(LEDC_CHANNEL_0, brightness);
    }
  // wait for 30 milliseconds to see the dimming effect
  delay(118);
}
```

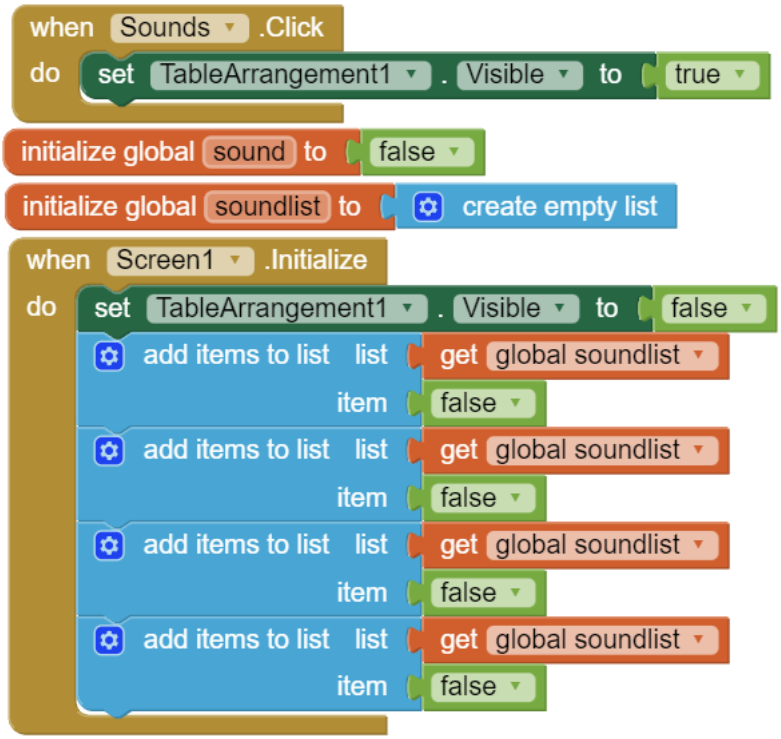
- **HEART RATE CODE ARDUINO FOR THE BRACELET MICROCONTROLLER**

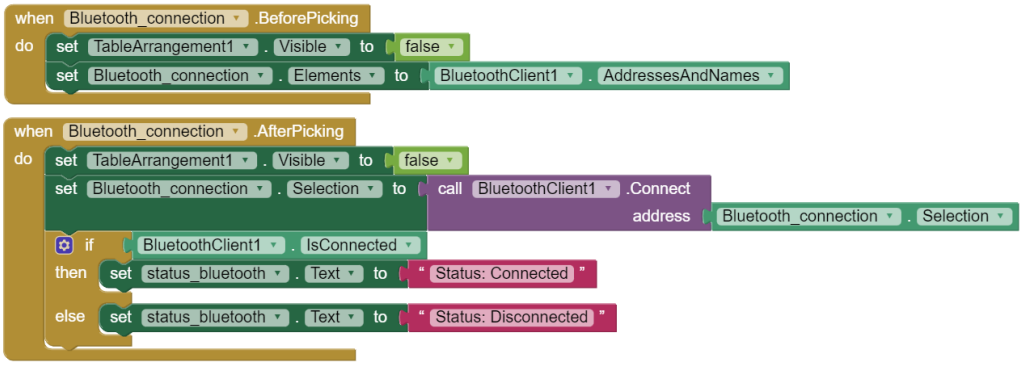
```
#include <math.h>
#include <Wire.h>
#include "MAX30105.h"

#include "heartRate.h"

MAX30105 particleSensor;

const byte RATE_SIZE = 4; //Increase this for more averaging. 4 is good.
byte rates[RATE_SIZE]; //Array of heart rates
byte rateSpot = 0;
long lastBeat = 0; //Time at which the last beat occurred

float beatsPerMinute;
int beatAvg;


void setup()
{

 //Wire.begin(SDA,SCL);
 Serial.begin(115200);
 Serial.println("Initializing...");


 // Initialize sensor
 if (!particleSensor.begin(Wire, I2C_SPEED_FAST)) //Use default I2C port, 400kHz
speed
 {
  Serial.println("MAX30105 was not found. Please check wiring/power. ");
  while (1);
 }
 Serial.println("Place your index finger on the sensor with steady pressure.");

 particleSensor.setup(); //Configure sensor with default settings
 particleSensor.setPulseAmplitudeRed(0x0A); //Turn Red LED to low to indicate sensor
is running
 particleSensor.setPulseAmplitudeGreen(0); //Turn off Green LED
}

void loop()
{ float r_interval = 0;
 static float r_current=0;
 static float r_previous=0;
 float r_difference=0;
 long counter=0;
 static float mean_of_r_sum=0;
```

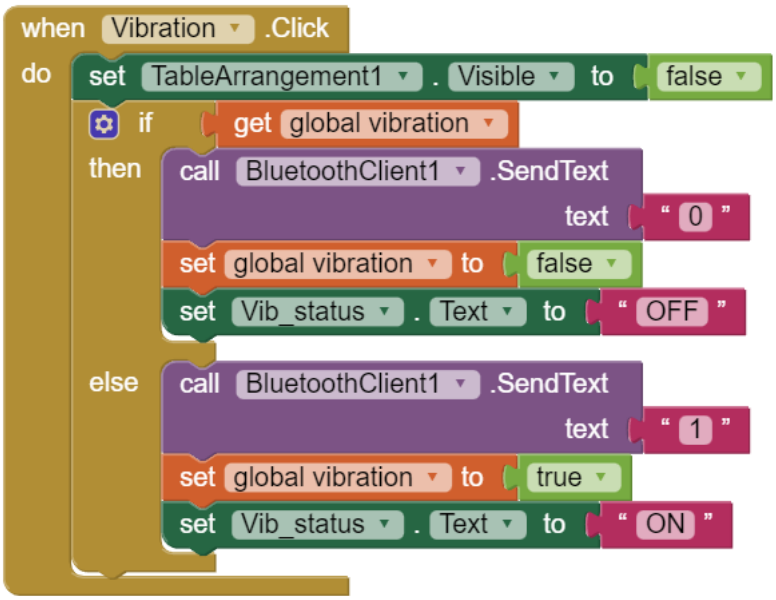
```
 long long averagerinterval=0;
 long irValue = particleSensor.getIR();

 if (checkForBeat(irValue) == true)
 {
  //We sensed a beat!
  long delta = millis() - lastBeat;
  lastBeat = millis();

  beatsPerMinute = 60 / (delta / 1000.0);

  if (beatsPerMinute < 255 && beatsPerMinute > 20)
  {
   rates[rateSpot++] = (byte)beatsPerMinute; //Store this reading in the array
   rateSpot %= RATE_SIZE; //Wrap variable

   //Take average of readings
   beatAvg = 0;
   for (byte x = 0 ; x < RATE_SIZE ; x++)
    beatAvg += rates[x];
   beatAvg /= RATE_SIZE;
   r_interval = 60.0/beatAvg;
   r_current=delta;
   r_difference=((r_previous-r_current)*(r_previous-r_current))/1000;
   r_previous=r_current;
   counter++;

   averagerinterval=(r_difference+averagerinterval);

   mean_of_r_sum=sqrt(averagerinterval/counter);
  }
 }

 Serial.print("IR=");
 Serial.print(irValue);
 Serial.print(", BPM=");
 Serial.print(beatsPerMinute);
 Serial.print(", Avg BPM=");
 Serial.print(beatAvg);
 Serial.print(",averagerinterval=");
 Serial.print(averagerinterval/beatsPerMinute);


 if (irValue < 50000)
  Serial.print(" No finger?");

 Serial.println();
}
```

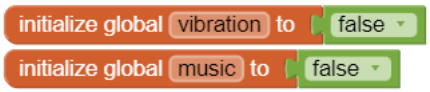
- **APP CODE MIT APP INVENTOR**

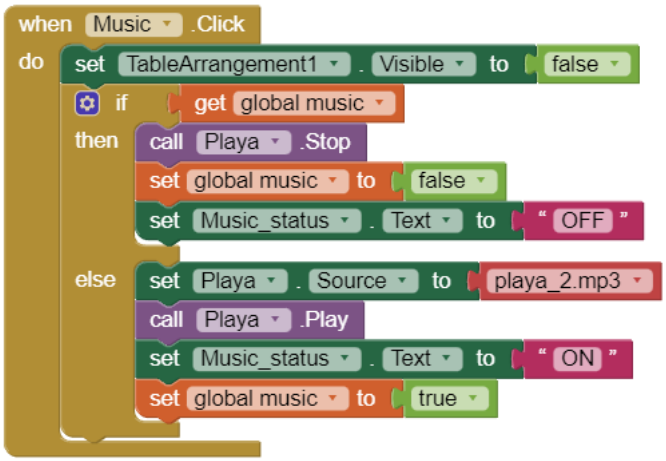












JACOBINE CAMPS, JULIAN BODE, JULIA DO NASCIMENTO, GEMA ROMERA