import grafica.\*;

import processing.serial.\*;

Serial myPort; // The serial port

int xPos = 1; // horizontal position of the graph

float height\_old = 0;

float height\_new = 0;

float inByte = 0;

int BPM = 0;

int beat\_old = 0;

float[] beats = new float[500]; // Used to calculate average BPM

int beatIndex;

float threshold = 620.0; //Threshold at which BPM calculation occurs

boolean belowThreshold = true;

PFont font;

GPlot plot1;

int x,y,z = 0;

GPointsArray p1 = new GPointsArray(2000);

char[] arr;

String val; // Data received from the serial port

boolean switchCase = true;

void setup () {

// set the window size:

size(1000, 700);

// List all the available serial ports

//println(Serial.list());

// Open whatever port is the one you're using.

myPort = new Serial(this, "COM3", 115200);

// don't generate a serialEvent() unless you get a newline character:

//myPort.bufferUntil('\n');

// set inital background:

background(0xff);

font = createFont("Times New Roman", 12, true);

}

void draw () {

//Map and draw the line for new data point

if ( myPort.available() > 0) { // If data is available,

val = myPort.readStringUntil('\n'); // read it and store it in val

}

boolean skip = true;

if (switchCase) {

try {

val = val.replace("\n", "").replace("\r", "");

val.trim();

arr = val.toCharArray();

if(!Character.isDigit(arr[0])){

skip = false;

}

}

catch (NullPointerException e) {

skip = false;

}

if(skip) {

x = Integer.parseInt(new String(arr));

}

switchCase = false;

} else {

try {

val = val.replace("\n", "").replace("\r", "");

val.trim();

arr = val.toCharArray();

if(!Character.isDigit(arr[0])){

skip = false;

}

switchCase = true;

}

catch (NullPointerException e) {

skip = false;

}

if(skip) {

z = Integer.parseInt(new String(arr));

print("Yay");

}

}

//inByte = map(millis()-10, z, millis(), 0);

//height\_new = height - z;

println(z);

if(z == 0) {

z = 300;

}

line(xPos - 10, 300, xPos, z-300);

height\_old = height\_new;

// at the edge of the screen, go back to the beginning:

if (xPos >= width) {

xPos = 0;

background(0xff);

}

else {

// increment the horizontal position:

xPos+=10;

}

long y = millis();

// draw text for BPM periodically

if (millis() % 128 == 0){

fill(0xFF);

rect(0, 0, 200, 20);

fill(0x00);

text("BPM: " + inByte, 15, 10);

}

//val = inByte;

//skip = true;

// try {

// val = val.replace("\n", "").replace("\r", "");

// val.trim();

// arr = val.toCharArray();

// if(!Character.isDigit(arr[0])){

// skip = false;

// }

// }

// catch (NullPointerException e) {

// skip = false;

// }

// if(skip) {

// x = Integer.parseInt(new String(arr));

// }

// x = (int)inByte;

//println(val);

plot1 = new GPlot(this);

plot1.setPos(0,300);

plot1.setDim(700, 300);

//plot1.setPointColor(color(255, 0, 255, 0));

plot1.setPointSize(2);

plot1.getXAxis().setAxisLabelText("Time");

plot1.getYAxis().setAxisLabelText("Respiration");

plot1.setTitleText("Respiration over Time");

plot1.setPointSize(5);

p1.add(y,x);

plot1.addLayer("red", p1);

plot1.getLayer("red").setPointColor(color(140,12,3,255));

plot1.beginDraw();

plot1.drawBackground();

plot1.drawBox();

plot1.drawXAxis();

plot1.drawYAxis();

plot1.drawTitle();

plot1.drawGridLines(GPlot.BOTH);

plot1.drawPoints();

plot1.endDraw();

fill(140, 12, 3);

rect(800, 0, 200, 40);

fill(0);

text("Maximum: 90-100% ", 815, 25);

fill(240, 16, 0);

rect(800, 40, 200, 40);

fill(0);

text("Hard: 80-90%", 815, 65);

fill(240, 152, 0);

rect(800, 80, 200, 40);

fill(0);

text("Moderate: 70-80%", 815, 105);

fill(255, 255, 0);

rect(800, 120, 200, 40);

fill(0);

text("Light: 60-70%", 815, 145);

fill(0, 255, 8);

rect(800, 160, 200, 40);

fill(0);

text("Very Light: 50-60%", 815, 185);

fill(153);

rect(800, 200, 200, 40);

fill(0);

text("No Data", 815, 225);

}

void serialEvent (Serial myPort)

{

// get the ASCII string:

String inString = myPort.readStringUntil('\n');

if (inString != null)

{

// trim off any whitespace:

inString = trim(inString);

// If leads off detection is true notify with blue line

if (inString.equals("!"))

{

stroke(0, 0, 0xff); //Set stroke to blue ( R, G, B)

inByte = 512; // middle of the ADC range (Flat Line)

}

// If the data is good let it through

else

{

stroke(0xff, 0, 0); //Set stroke to red ( R, G, B)

inByte = float(inString);

// BPM calculation check

if (inByte > threshold && belowThreshold == true)

{

calculateBPM();

belowThreshold = false;

}

else if(inByte < threshold)

{

belowThreshold = true;

}

}

}

}

void calculateBPM ()

{

int beat\_new = millis(); // get the current millisecond

int diff = beat\_new - beat\_old; // find the time between the last two beats

float currentBPM = 60000 / diff; // convert to beats per minute

beats[beatIndex] = currentBPM; // store to array to convert the average

float total = 0.0;

for (int i = 0; i < 500; i++){

total += beats[i];

}

BPM = int(total / 500);

beat\_old = beat\_new;

beatIndex = (beatIndex + 1) % 500; // cycle through the array instead of using FIFO queue

}