RGB Number neopixel with Task scheduler

```
#include <TaskScheduler.h>

//function
void neopixelCB();

void distanceCB();

void soundCB();

void oledCB();


//task
Task neopixel(50, TASK_FOREVER, &neopixelCB);

Task distance(40, TASK_FOREVER, &distanceCB);

Task sound(10, TASK_FOREVER, &soundCB);

Task oled(50, TASK_FOREVER, &oledCB);


//OLED DISPLAY

#include <Arduino.h>
#include <U8g2lib.h>

#ifdef U8X8_HAVE_HW_SPI
#include <SPI.h>
#endif
#ifdef U8X8_HAVE_HW_I2C
#include <Wire.h>
#endif



///DISTANCE SENSORE
#include <Wire.h>
```

```
//Click here to get the library: http://librarymanager/All#SparkFun_VCNL4040
#include "SparkFun_VCNL4040_Arduino_Library.h"

VCNL4040 proximitySensor;

long startingProxValue = 0;
long deltaNeeded = 0;
boolean nothingThere = false;

//SOUND SENSOR

#define PIN_ANALOG_IN A0

//NEOPIXEL
#include <Adafruit_NeoPixel.h>
#ifdef __AVR__
#include <avr/power.h> // Required for 16 MHz Adafruit Trinket
#endif

// Which pin on the Arduino is connected to the NeoPixels?
#define PIN        6 // On Trinket or Gemma, suggest changing this to 1

// How many NeoPixels are attached to the Arduino?
#define NUMPIXELS 1 // Popular NeoPixel ring size

// When setting up the NeoPixel library, we tell it how many pixels,
// and which pin to use to send signals. Note that for older NeoPixel
// strips you might need to change the third parameter -- see the
// strandtest example for more information on possible values.
Adafruit_NeoPixel pixels(NUMPIXELS, PIN, NEO_RGB + NEO_KHZ800);

#define DELAYVAL 500 // Time (in milliseconds) to pause between pixels

//need scheduler
Scheduler runner;

int red = 0;

int green = 0;

int blue = 150;
```

```
//oled display

U8G2_SSD1327_EA_W128128_1_HW_I2C u8g2(U8G2_R0, /* reset=*/ U8X8_PIN_NONE);


//const char *text = "xxxxxx "; // scroll this text from right to left



void setup() {


  //OLED DISPLAY

  u8g2.begin();

//  u8g2.setFont(u8g2_font_inb30_mr);  // set the target font to calculate the pixel width
//  width = u8g2.getUTF8Width(text);    // calculate the pixel width of the text
//
  u8g2.setFontMode(0);    // enable transparent mode, which is faster



  //SOUND SENSOR


  //DISTANCE SENSOR
  Serial.begin(9600);
  Wire.begin(); //Join i2c bus

  if (proximitySensor.begin() == false)
  {
    Serial.println("Device not found. Please check wiring.");
    while (1); //Freeze!
  }

  //Set the current used to drive the IR LED - 50mA to 200mA is allowed.
  proximitySensor.setLEDCurrent(200); //For this example, let's do max.

  //The sensor will average readings together by default 8 times.
  //Reduce this to one so we can take readings as fast as possible
  proximitySensor.setProxIntegrationTime(8); //1 to 8 is valid
```

```
  //Take 8 readings and average them
  for (byte x = 0 ; x < 8 ; x++)
  {
    startingProxValue += proximitySensor.getProximity();
  }
  startingProxValue /= 8;

  deltaNeeded = (float)startingProxValue * 0.05; //Look for 5% change
  if (deltaNeeded < 5) deltaNeeded = 5; //Set a minimum

  //NEOPIXEL CODE
  // put your setup code here, to run once:
  // These lines are specifically to support the Adafruit Trinket 5V 16 MHz.
  // Any other board, you can remove this part (but no harm leaving it):
#if defined(__AVR_ATtiny85__) && (F_CPU == 16000000)
  clock_prescale_set(clock_div_1);
#endif
  // END of Trinket-specific code.

  pixels.begin(); // INITIALIZE NeoPixel strip object (REQUIRED)

  //what tasks im adding
  runner.addTask(neopixel);
  runner.addTask(distance);
  runner.addTask(sound);
  runner.addTask(oled);

  //what task u enabling(start)
  neopixel.enable();
  distance.enable();
  sound.enable();
  oled.enable();



}

void loop() {
  // put your main code here, to run repeatedly:

  //just one line runnning all tasks and functions in a loop
  runner.execute();
}
```

```
void neopixelCB() {

  pixels.clear(); // Set all pixel colors to 'off'

  // The first NeoPixel in a strand is #0, second is 1, all the way up
  // to the count of pixels minus one.
  for (int i = 0; i < NUMPIXELS; i++) { // For each pixel...

    // pixels.Color() takes RGB values, from 0,0,0 up to 255,255,255
    // Here we're using a moderately bright green color:
    pixels.setPixelColor(i, pixels.Color(red, green, blue));

    pixels.show();   // Send the updated pixel colors to the hardware.

    //delay(DELAYVAL); // Pause before next pass through loop
  }

}

void distanceCB() {

  unsigned int proxValue = proximitySensor.getProximity();

  int redVal = map(proxValue, 0, 12000, 0, 255);

  red = redVal;


  //  Serial.print("Prox: ");
  //  Serial.print(proxValue);
  //  Serial.print(" ");

  //Let's only trigger if we detect a 5% change from the starting value
  //Otherwise, values at the edge of the read range can cause false triggers
  if (proxValue > (startingProxValue + deltaNeeded))
  {
    Serial.print("Something is there!");
    nothingThere = false;
  }
  else
  {
    if (nothingThere == false) Serial.print("I don't see anything");
```

```
    nothingThere = true;
  }


}

void soundCB() {

//MAP MEANS COLLECTION OF VALUES IS USED IN INPUT

  // Check the envelope input
  int value = analogRead(PIN_ANALOG_IN);
  Serial.print(value);

  int greenVal = map(value, 0, 150, 0, 255);

  green = greenVal;
  // Convert envelope value into a message
  Serial.print("Status: ");

  if (value <= 50)
  {
    Serial.println("Quiet.");
  }
  else if ( (value > 50) && ( value <= 60) )
  {
    Serial.println("Moderate.");
  }
  else if (value > 70)
  {
    Serial.println("Loud.");
  }
  //delay(100);

}


void oledCB() {


  uint8_t r= red;

  char r_str[4];
```

```
  strcpy(r_str, u8x8_u8toa(r, 3));    /* convert m to a string with three digits */

///////////
  uint8_t g= green;

  char g_str[4];
  strcpy(g_str, u8x8_u8toa(g, 3));

 ////////////////
  uint8_t b= blue;

  char b_str[4];
  strcpy(b_str, u8x8_u8toa(b, 3));


//    int blueVal = map( value, 150, 0, 255);
//    blue = blueVal;

//neopixel what color in rgb

  u8g2.firstPage();
  do {
    u8g2.setFont(u8g2_font_ncenB14_tr);
    u8g2.drawStr(0,21,r_str); // displays red value on oled
    u8g2.drawStr(0,42,g_str);  // displays green value oled
    u8g2.drawStr(0,63,b_str);  // displays blue value oled
  } while ( u8g2.nextPage() );

}
```