ENTREGA #2 DISEÑO ELECTRÓNIO

**CODIGO FUENTE**

#include <UbiConstants.h>

#include <UbiTypes.h>

#include <UbidotsEsp32Mqtt.h>

#include "DHT.h"

#include <TFT\_eSPI.h>

#include <SPI.h>

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

 \* Define Constants

 \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

#define DHTPIN 2            // Pin conectado al sensor DHT11

#define DHTTYPE DHT22       // Tipo de sensor DHT11

const char \*UBIDOTS\_TOKEN = "BBUS-UmLWaIxEkCI6nYJel8pIL3zd96IPJa";  // Token de Ubidots

const char \*WIFI\_SSID = "Lasso";      // Nombre de tu red Wi-Fi

const char \*WIFI\_PASS = "24825207";   // Contraseña de tu red Wi-Fi

const char \*DEVICE\_LABEL = "esp32";   // Nombre del dispositivo en Ubidots

const char \*TEMP\_LABEL = "temp";      // Etiqueta para la temperatura

const char \*HUMIDITY\_LABEL = "humidity"; // Etiqueta para la humedad

const char \*SWITCH\_LED\_RED = "sw1";   // Etiqueta para el switch del LED rojo

const char \*SWITCH\_LED\_GREEN = "sw2"; // Etiqueta para el switch del LED verde

const int PUBLISH\_FREQUENCY = 5000;   // Frecuencia de envío en milisegundos

const uint8\_t RED\_LED = 12;           // Pin del LED rojo

const uint8\_t GREEN\_LED = 13;         // Pin del LED verde

unsigned long timer;

// Inicializa el sensor DHT

DHT dht(DHTPIN, DHTTYPE);

// Inicializa el cliente Ubidots

Ubidots ubidots(UBIDOTS\_TOKEN);

// Variables para rastrear el estado de los LEDs

int RED\_led = 0; // 0 = apagado, 1 = encendido

int GREEN\_led = 0; // 0 = apagado, 1 = encendido

// Inicializa la pantalla OLED

TFT\_eSPI tft = TFT\_eSPI(135, 240); // Pantalla de la TTGO OLED

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

 \* Funciones Auxiliares

 \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

void callback(char \*topic, byte \*payload, unsigned int length)

{

  Serial.print("Mensaje recibido [");

  Serial.print(topic);

  Serial.print("]: ");

  // Verifica si el mensaje es para el switch del LED rojo (sw1)

  if (strstr(topic, SWITCH\_LED\_RED) != NULL)

  {

    if ((char)payload[0] == '1')

    {

      digitalWrite(RED\_LED, HIGH); // Encender el LED rojo

      Serial.println("LED ROJO ON");

      tft.fillCircle(50, 110, 20, TFT\_RED);

      RED\_led = 1; // Actualizar el estado del LED rojo

    }

    else

    {

      digitalWrite(RED\_LED, LOW); // Apagar el LED rojo

      Serial.println("LED ROJO OFF");

      tft.fillCircle(50, 110, 20, TFT\_DARKGREY);

      RED\_led = 0; // Actualizar el estado del LED rojo

    }

  }

  // Verifica si el mensaje es para el switch del LED verde (sw2)

  if (strstr(topic, SWITCH\_LED\_GREEN) != NULL)

  {

    if ((char)payload[0] == '1')

    {

      digitalWrite(GREEN\_LED, HIGH); // Encender el LED verde

      Serial.println("LED VERDE ON");

      tft.fillCircle(150, 110, 20, TFT\_GREEN);

      GREEN\_led = 1; // Actualizar el estado del LED verde

    }

    else

    {

      digitalWrite(GREEN\_LED, LOW); // Apagar el LED verde

      Serial.println("LED VERDE OFF");

      tft.fillCircle(150, 110, 20, TFT\_DARKGREY);

      GREEN\_led = 0; // Actualizar el estado del LED verde

    }

  }

  Serial.println();

}

void initDisplay()

{

  tft.init();

  tft.setRotation(1);

  tft.fillScreen(TFT\_BLACK);

  tft.setTextSize(2);

  tft.setTextColor(TFT\_GREEN, TFT\_BLACK);

  tft.setTextDatum(MC\_DATUM);

  tft.drawString("Inicializando...", tft.width() / 2, tft.height() / 2);

  delay(2000);

  tft.fillScreen(TFT\_BLACK);

}

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

 \* Funciones principales

 \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

void setup()

{

  // Configuración inicial

  Serial.begin(115200);

  dht.begin(); // Inicia el sensor DHT

  // Inicializa la pantalla OLED

  initDisplay();

  // Configura los pines de los LEDs como salida

  pinMode(RED\_LED, OUTPUT);

  pinMode(GREEN\_LED, OUTPUT);

  // Conexión Wi-Fi y Ubidots

  ubidots.connectToWifi(WIFI\_SSID, WIFI\_PASS);

  ubidots.setCallback(callback);

  ubidots.setup();

  ubidots.reconnect();

  // Suscribirse a los temas de los switches

  ubidots.subscribeLastValue(DEVICE\_LABEL, SWITCH\_LED\_RED);

  ubidots.subscribeLastValue(DEVICE\_LABEL, SWITCH\_LED\_GREEN);

  timer = millis();

}

void displayData(float temperature, float humidity)

{

  tft.fillScreen(TFT\_BLACK);

  tft.setTextSize(3);

  tft.setCursor(0, 20);

  tft.setTextColor(TFT\_GREEN, TFT\_BLACK);

  // Mostrar temperatura y humedad

  tft.printf("Temp: %.2f C\n", temperature);

  tft.printf("Hum: %.2f %%\n", humidity);

  // Dibujar el círculo rojo según el estado del LED rojo

  if (RED\_led == 1)

  {

    tft.fillCircle(50, 110, 20, TFT\_RED); // Círculo rojo encendido

  }

  else

  {

    tft.fillCircle(50, 110, 20, TFT\_DARKGREY); // Círculo rojo apagado

  }

  // Dibujar el círculo verde según el estado del LED verde

  if (GREEN\_led == 1)

  {

    tft.fillCircle(150, 110, 20, TFT\_GREEN); // Círculo verde encendido

  }

  else

  {

    tft.fillCircle(150, 110, 20, TFT\_DARKGREY); // Círculo verde apagado

  }

  Serial.printf("Temp: %.2f C, Hum: %.2f %%\n", temperature, humidity);

}

void loop()

{

  // Reconexión a Ubidots si es necesario

  if (!ubidots.connected())

  {

    ubidots.reconnect();

    ubidots.subscribeLastValue(DEVICE\_LABEL, SWITCH\_LED\_RED);

    ubidots.subscribeLastValue(DEVICE\_LABEL, SWITCH\_LED\_GREEN);

  }

  // Enviar datos periódicamente

  if (abs((long)(millis() - timer)) > PUBLISH\_FREQUENCY)

  {

    float temperature = dht.readTemperature(); // Lectura de temperatura

    float humidity = dht.readHumidity();      // Lectura de humedad

    // Verifica si las lecturas son válidas

    if (isnan(temperature) || isnan(humidity))

    {

      Serial.println("Error al leer el sensor DHT11");

      tft.fillScreen(TFT\_RED);

      tft.setTextDatum(MC\_DATUM);

      tft.drawString("Error sensor!", tft.width() / 2, tft.height() / 2);

    }

    else

    {

      // Muestra los datos en la pantalla OLED

      displayData(temperature, humidity);

      // Enviar datos a Ubidots

      ubidots.add(TEMP\_LABEL, temperature);   // Añadir temperatura

      ubidots.add(HUMIDITY\_LABEL, humidity); // Añadir humedad

      ubidots.publish(DEVICE\_LABEL);         // Publicar al dispositivo

    }

    timer = millis(); // Reinicia el temporizador

  ubidots.subscribeLastValue(DEVICE\_LABEL, SWITCH\_LED\_RED);

  ubidots.subscribeLastValue(DEVICE\_LABEL, SWITCH\_LED\_GREEN);

  }

  ubidots.loop();

}