INSTITUTO POLITÉCNICO DE BRAGANÇA
Escola Superior de Educação

INTERNET DAS COISAS

TP2 - Dispositivos de Desenvolvimento para Internet das Coisas

- Ano Letivo 2022/2023 –

A48530 – Diogo Brandão Ferreira

A48542 – José Nuno Marinho Carvalho

3. Os dois códigos de exemplo - Blink e BlinkWithoutDelay – servem para ilustrar como um LED é ligado e desligado num microcontrolador ESP32 ou ESP8266.A principal diferença entre os dois códigos é que o Blink usa a função delay() para controlar o tempo em que o LED fica ligado e desligado . Essa função faz com que o programa pare de executar por um determinado período de tempo, o que significa que outras tarefas que precisam ser realizadas pelo microcontrolador ficam em espera durante esse mesmo tempo. Enquanto o BlinkWithoutDelay usa a técnica de não-bloqueio para controlar o tempo de ativação do LED. Essa função faz com que o programa pare de executar por um determinado período de tempo, o que significa que outras tarefas que precisam de ser realizadas pelo microcontrolador ficam em espera durante esse tempo.

4. A função void setup() é usada no código do Arduino para executar uma série de instruções que precisam ser realizadas apenas uma vez. Ela é usada para inicializar variáveis, definir os pinos de entrada e saída, definir a velocidade de transmissão da porta, etc. Já a função void loop() é usada para executar o código principal do programa repetidamente, em um loop infinito. Ela é usada para implementar o corpo do programa, como ler entradas de sensores, processar dados, enviar informações para dispositivos externos, etc.. O código dentro dessa função é executado repetidamente, em uma taxa determinada pelo tempo de execução do programa e pela velocidade do processador do Arduino.

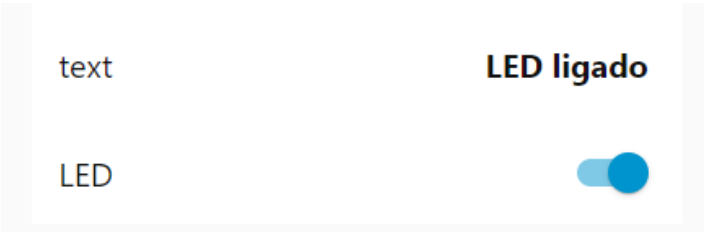
7. "void callback()": Esta função é um “callback” que é chamado sempre que uma mensagem MQTT é recebida. Ela é registrada no cliente MQTT usando a função client.setCallback(callback). Quando uma mensagem é recebida pelo cliente MQTT, esta função é chamada com a mensagem como um parâmetro. A função callback() pode ser personalizada para realizar uma ação específica, dependendo do conteúdo da mensagem recebida.

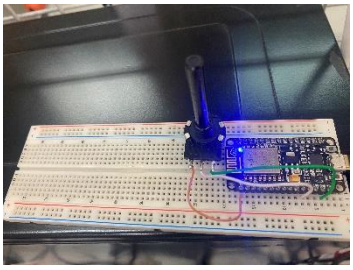
• "void reconect()": Esta função é usada para reconectar o cliente MQTT ao broker MQTT, caso a conexão seja perdida ou interrompida. Ela é chamada pelo loop principal do programa quando o cliente MQTT perde a conexão com o broker MQTT. A função reconect() tenta reconectar o cliente MQTT ao broker MQTT, em seguida, inscrevendo-se novamente nos tópicos MQTT que o cliente estava inscrito antes da desconexão.

• "if (now - lastTime > 2000)": Este trecho de código é usado para verificar se uma quantidade de tempo especificada passou desde a última vez que uma determinada ação foi realizada. A variável lastTime armazena o tempo em milissegundos desde o início do programa, enquanto a variável now armazena o tempo.

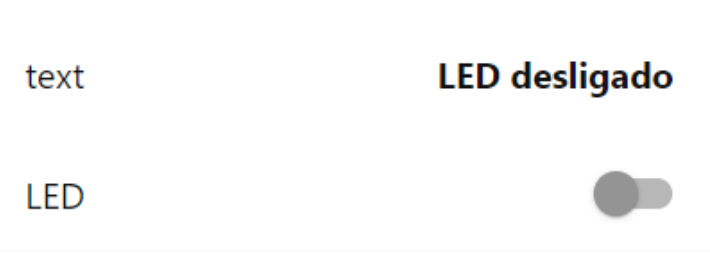
8

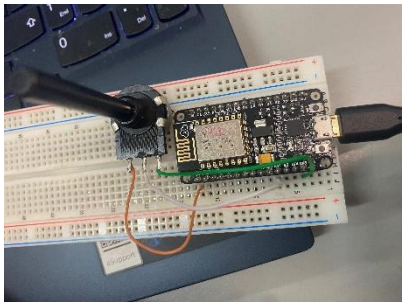
LED LIGADO:

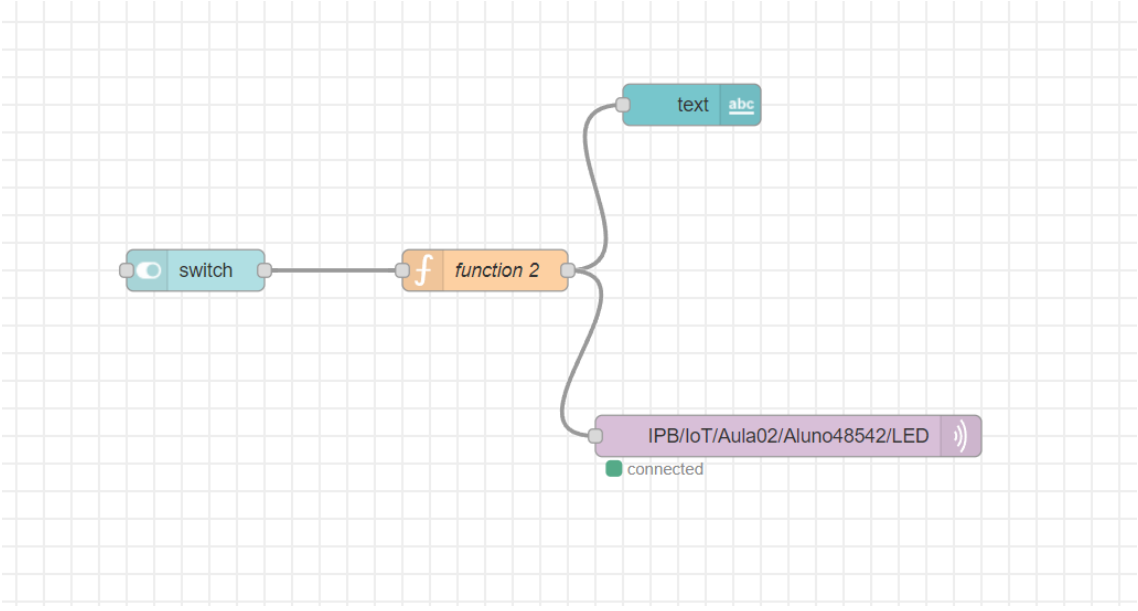




LED DESLIGADO:







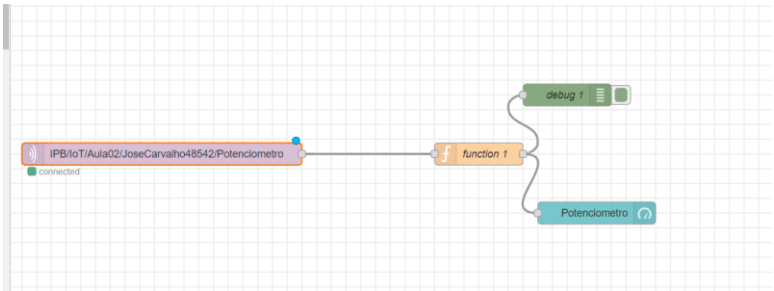
```
#include <ESP8266WiFi.h>
#include <PubSubClient.h>
#include <ArduinoJson.h>

const char* mqtt_server = "broker.mqtt-dashboard.com";
const char* ssid = "agents";
const char* password =
"QgC9O8VucAByqvVu5Rruv1zdpqM66cd23KG4ElV7vZiJND580bzYvaHqz5k07G2";

WiFiClient espClient;
PubSubClient client(espClient);

long lastTime = 0;
const int Msg_Size = 50;
char msg[Msg_Size];
int value = 0;


void setup_wifi() {
  Serial.println("Starting Setup Wifi");
  delay(10);
  // We start by connecting to a WiFi network
  Serial.println();
  Serial.print("Connecting to Wifi ");
   WiFi.begin(ssid, password);

  while (WiFi.status() != WL_CONNECTED) {
    delay(500);
    Serial.print(".");
  }
  randomSeed(micros());
  Serial.println("");
  Serial.println("WiFi connected");
}

void callback(String topic, byte* payload, unsigned int length) {

  Serial.print("Message arrived [");
  Serial.print(topic);
  Serial.print("] ");
```

```
  String strPayload="";
  for (int i=0;i<length;i++) {
    strPayload = strPayload + (char)payload[i];
  }

  Serial.println(strPayload);
  StaticJsonBuffer<200> jsonBuffer;
  JsonObject& data = jsonBuffer.parseObject(strPayload);

  if(topic == "IPB/IoT/Aula02/Aluno48542/LED" ){
    Serial.println("12345");
    if( strPayload == "LED desligado"){
      digitalWrite(LED_BUILTIN, HIGH); //Turn off the LED
    }else{
      digitalWrite(LED_BUILTIN, LOW); //Turn on the LED
    }
  }
  Serial.println();
}

void reconnect() {
  // Loop until we're reconnected
  while (!client.connected()) {
    Serial.print("Attempting MQTT connection...");
    // Create a random client ID
    String clientId = "AulaIoT-";
    clientId += String(random(0xffff), HEX);
    // Attempt to connect
    if(client.connect(clientId.c_str())) {
      Serial.println("connected to the Broker");
      client.subscribe("IPB/IoT/Aula02/Aluno48542/LED"); //Tópico que o
ESP subscreve.
    } else {
      Serial.print("failed, rc=");
      Serial.print(client.state());
      Serial.println("try again in 5 seconds");
      delay(1000);
    }
  }
}

void setup() {
  Serial.begin(9600);
  pinMode(BUILTIN_LED, OUTPUT);
  Serial.println("\n Start Setup");
  setup_wifi();
  client.setServer(mqtt_server, 1883);
  client.setCallback(callback);
}
```

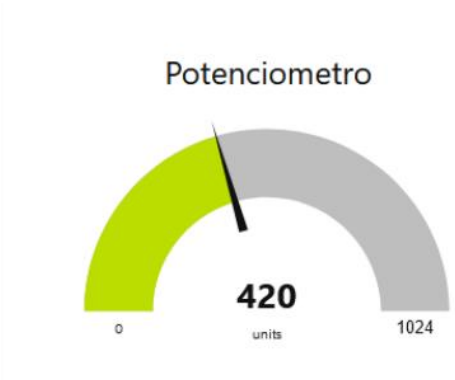
```
void loop() {
  while(WiFi.status() != WL_CONNECTED){ //Check Wifi connection
    WiFi.reconnect(); //If disconnected, try to reconnect
  }
  if(!client.connected()) {//Check Check Broker connection
    Serial.println("Node disconnected of Broker. Trying to connect.. ");
    reconnect(); //try reconect to broker.
  }
  client.loop();
  }
```

9.

Diagrama node-red



Representação do gauge



Codigo Arduino

```
void loop() {
  while(WiFi.status() != WL_CONNECTED){
    WiFi.reconnect();
  }
  if(!client.connected()) {
    Serial.println("Node disconnected of Broker. Trying to connect.. ");
    reconnect();
  }
  client.loop();
  long now = millis();
  if (now - lastTime > 2000) {
    lastTime = now;
    ++value;
    sensor = analogRead(A0);
    StaticJsonBuffer<200> jsonBuffer;
    JsonObject& sensorValue =jsonBuffer.createObject();
    sensorValue["potenciometro"]=sensor;
    sensorValue.printTo(potenciometro,sizeof(potenciometro));
    Serial.print("Publish message: ");
    Serial.println(potenciometro);
    client.publish("IPB/IoT/Aula02/JoseCarvalho48542/Potenciometro",
potenciometro);
  }
}
```