09

Últimos ajustes

Transcrição

Realizando os últimos ajustes, observamos primeiro nosso bloco `loop` .

```
void loop() {
    if(!client.connected()){
        reconnect();
    }

    client.loop();
    int trigger = digitalReader(PIR);
    client.publish("outTopic", String(trigger).c_str());

    if(trigger == HIGH){
        liga_laser();
        // Serial.println("Detectado")
    } else {
        desliga_laser();
        // Serial.println("NÃO Detectado");
    }
    delay(1000);
}
```

Nossas primeiras mudanças serão: remover a linha que executa a função `client.loop()` , por que essa função está relacionada a leitura de mensagens do servidor. Como nossa aplicação não precisa disso, vamos removê-la. Vamos também configurar o tópico de publicação através de um ponteiro, assim organizamos melhor o código. Considerando isso teremos dentro do loop:

```
void loop() {
    if(!client.connected()){
        reconnect();
    }

    int trigger = digitalReader(PIR);
    client.publish(mqtt_topic, String(trigger).c_str());

    if(trigger == HIGH){
        liga_laser();
        // Serial.println("Detectado")
    } else {
        desliga_laser();
        // Serial.println("NÃO Detectado");
    }
    delay(1000);
}
```

E próximo das inicializações dos outros ponteiros do MQTT, teremos:

```
const char** mqtt_topic = "monitoracao/lasercat/status";
```

Também desfaremos o comentário da inicialização da serial por que temos várias mensagens sendo impressas no código. Porém, no final, todo esse código de impressão de mensagens pode ser removido. No bloco `setup` removeremos o comentário da seguinte linha:

```
Serial.begin(9600);
```

Por fim, removeremos as seguintes linhas da função `reconnect` :

```
client.public("outTopic", "hello world");
client.subscribe("inTopic");
```

Elas escrevem no tópico `outTopic` e fazem leitura do tópico `inTopic` , o que de fato não precisamos já que temos configurado isso em outro ponto. Por fim o código da nossa aplicação estará da seguinte forma:

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

#define POS_INICIAL 90
#define X_MINIMO 30
#define Y_MINIMO 50
#define X_INTERVALO 12
#define Y_INTERVALO 4
#define LASER D1
#define PIR D2

Servo servoX;
Servo servoY;

const char* ssid = "SEU SSID DE REDE";
const char* password = "SUA SENHA";
const char* mqtt_server = "Disponível no CloudMQTT";
int mqtt_port = PORTA_DA_INSTANCIA;
const char** mqtt_topic = "monitoracao/lasercat/status";
const char* mqtt_user = USUARIO_DA_INSTANCIA;
const char* mqtt_password = SENHA_DA_INSTANCIA;

WiFiClient espClient;
PubSubClient client(espClient);

void setup() {
    servoX.attach(D3);
    servoY.attach(D4);
    servoX.write(POS_INICIAL);
    servoY.write(POS_INICIAL);

    pinMode(LASER, OUTPUT);
    digitalWrite(LASER, LOW);
    Serial.begin(9600);
```

```
    setup_wifi();
    client.setServer(mqtt_server, mqtt_port);
}

void loop() {
    if(!client.connected()){
        reconnect();
    }

    int trigger = digitalReader(PIR);
    client.publish(mqtt_topic, String(trigger).c_str());

    if(trigger == HIGH){
        liga_laser();
        // Serial.println("Detectado")
    } else {
        desliga_laser();
        // Serial.println("NÃO Detectado");
    }
    delay(1000);
}

// FUNÇÕES AUXILIARES

void posiciona_motores() {
    int posicaoX = (random(0, (X_INTERVALO)) * 10 + X_MINIMO);
    int posicaoY = (random(0, (Y_INTERVALO)) * 10 + Y_MINIMO);
    servoX.write(posicaoX);
    servoY.write(posicaoY);
    // Serial.print(posicaoX);
    // Serial.print("  ,  ");
    // Serial.println(posicaoY);
    // delay(2000);
}

void liga_laser() {
    for (int i = 0; i < TEMPO_LASER/2; ++i){
        digitalWrite(LASER, HIGH);
        posiciona_motores();
        delay(2000);
    }
}

void desliga_laser() {
    digitalWrite(LASER, LOW);
}

void setup_wifi() {
    delay(10);
    Serial.println();
    Serial.print("Connecting to");
    Serial.println(ssid);

    Wifi.begin(ssid, password);

    while(Wifi.status() != WL_CONNECTED){
        delay(500);
        Serial.print(".");
```

```
    }

    Serial.println("");
    Serial.println("WiFI connected");
    Serial.println("IP adress: ");
    Serial.println(WiFi.localIP());
}

void reconnect(){
    while(!client.connected()){
        Serial.print("Attempting MQTT connection...");

        if(client.connect("ESP8266Client", mqtt_user, mqtt_password)){
            Serial.println("connected")
        } else {
            Serial.print("falied, rc=");
            Serial.print(client.state());
            Serial.println(" try again in 5 seconds ");

            delay(2000);
        }
    }
}
```

Note também que reduzimos o `delay` da função `reconnect` para que a espera pela nova tentativa de conexão ao servidor não precise esperar tanto. Vamos testar? Lembre-se de configurar a aplicação cliente no seu *smartphone*.