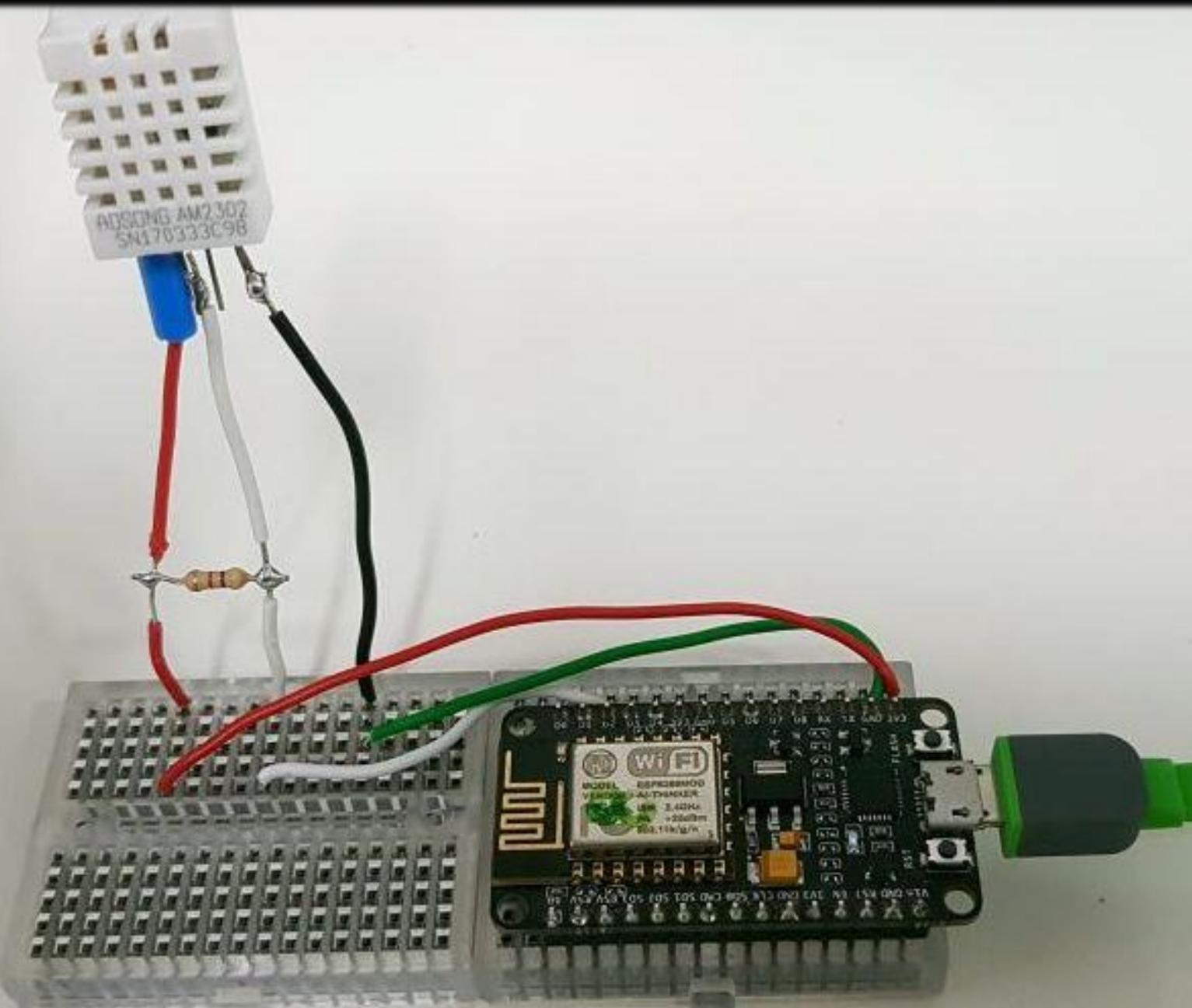


# ESP com MQTT

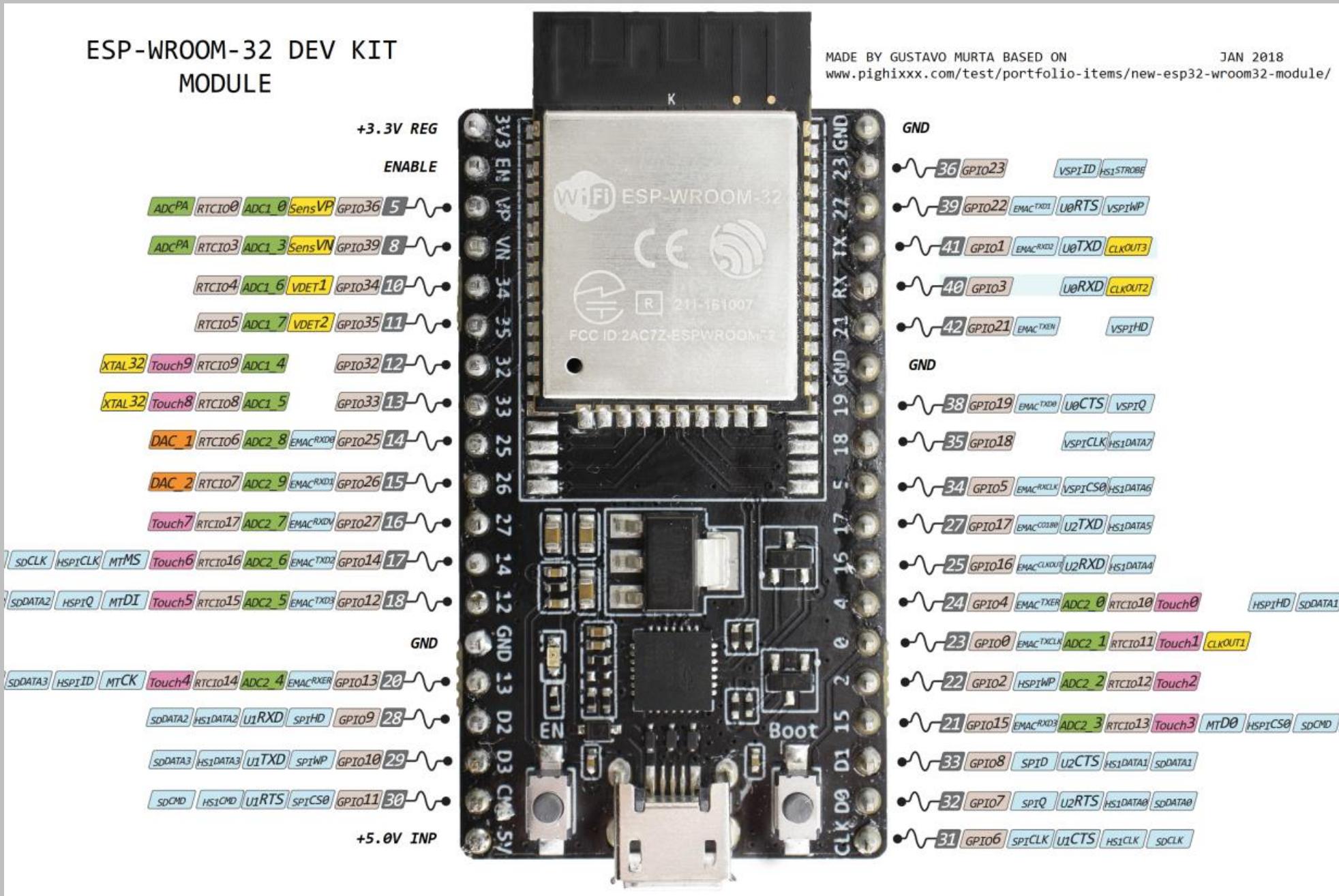


Por Fernando Koyanagi

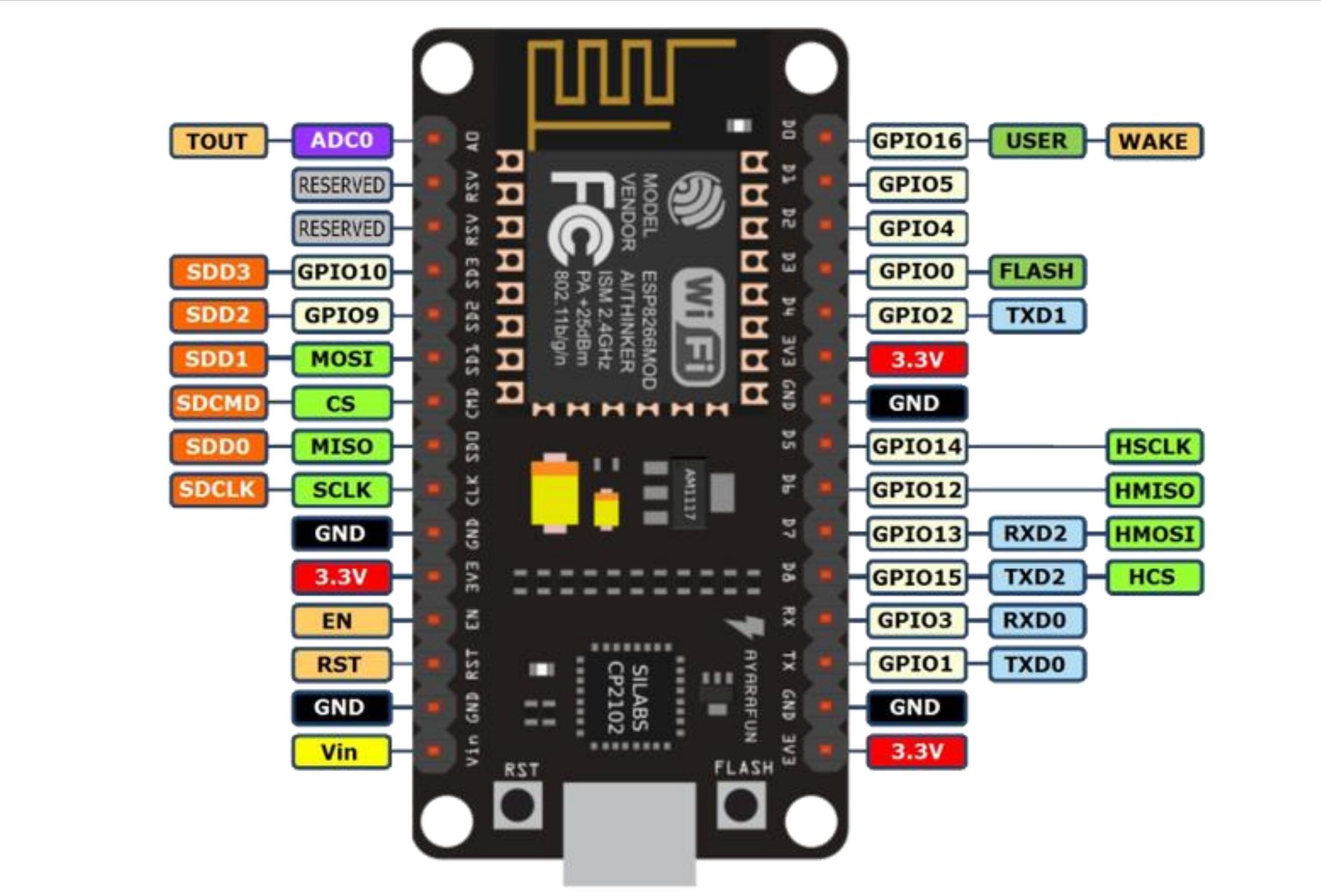
# **Intenção dessa aula**

- 1. Introdução ao MQTT**
- 2. Enviar dados do sensor de temperatura e umidade utilizando MQTT**
- 3. Verificar o gráfico dos dados em uma página web**

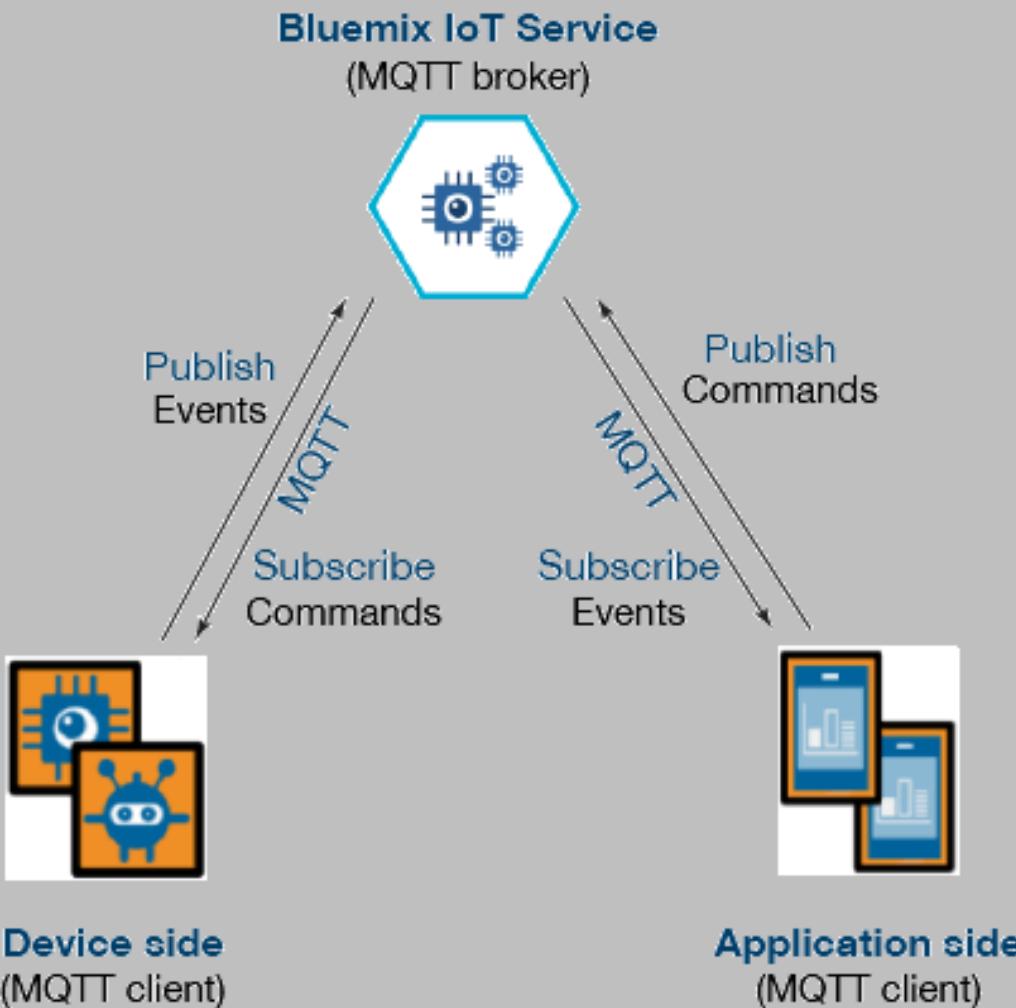
# ESP32 Pinout



# NodeMCU Pinout



# MQTT

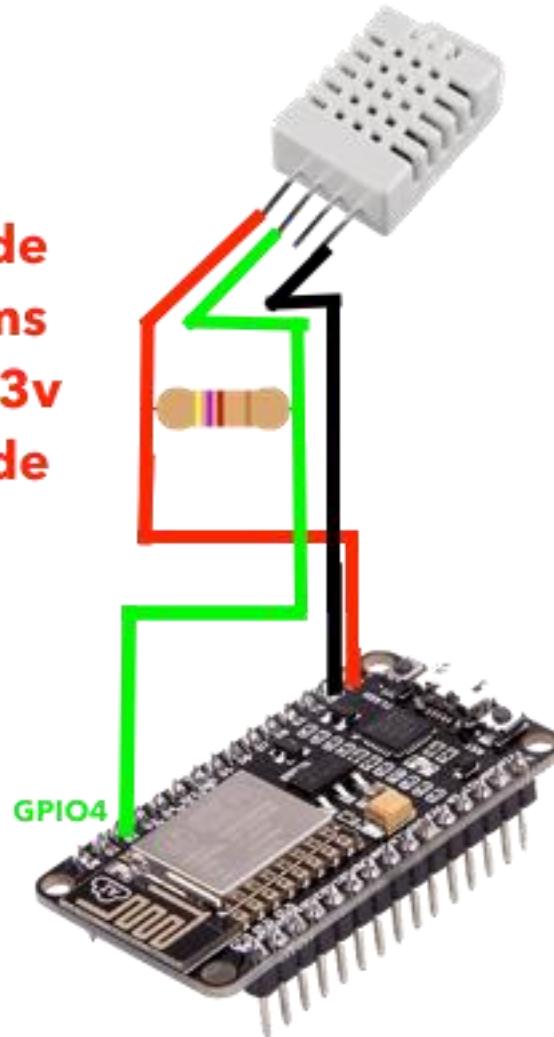


MQTT é um protocolo máquina a máquina usado em IoT, portanto foi criado para ser **leve e rápido**. Utiliza um sistema de **subscribe/publish** onde um dispositivo se “**inscreve**” em um tópico cujas informações lhe interessa e recebe as informações sempre que algum dispositivo **publicar** dados neste tópico.

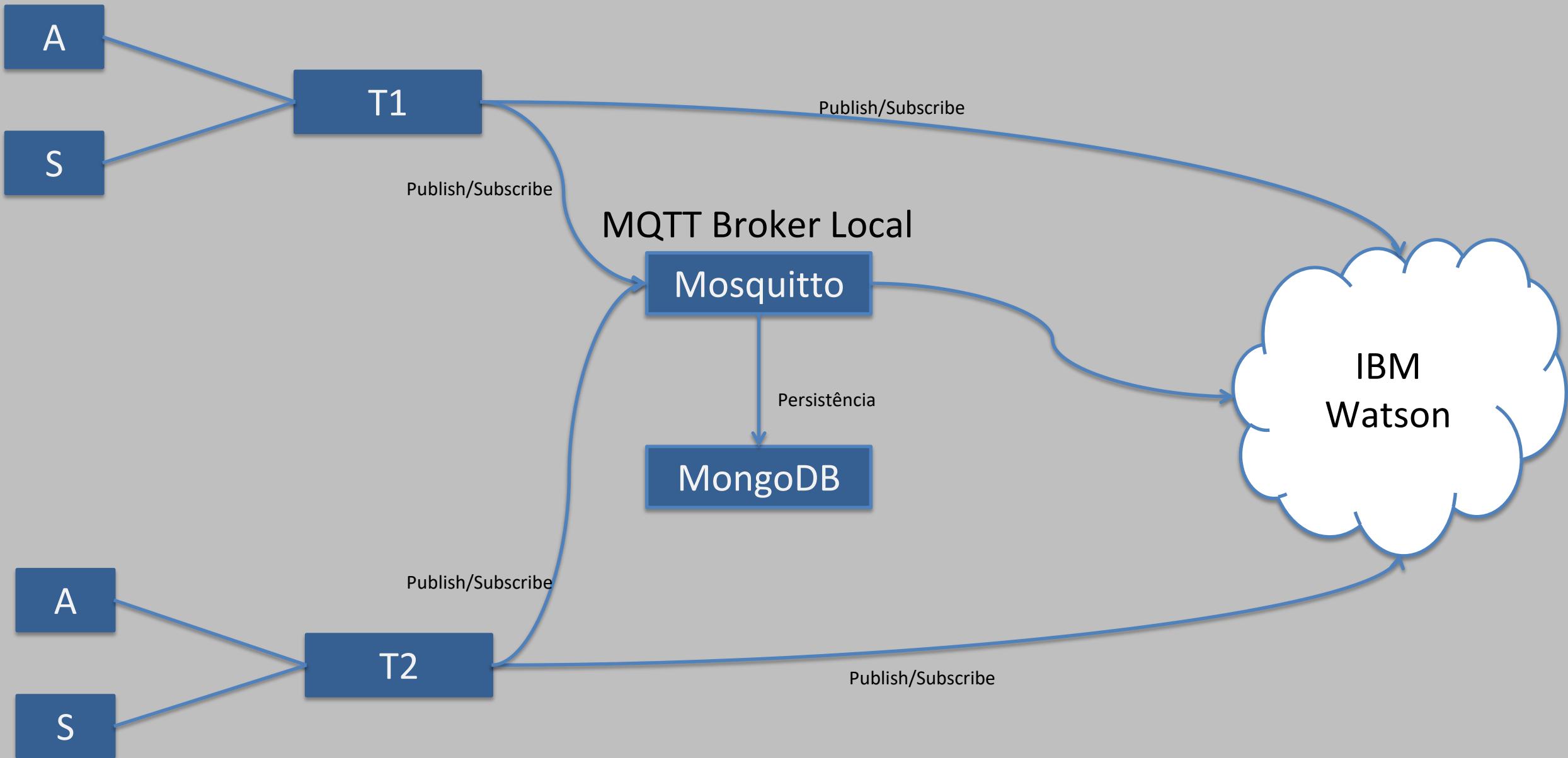
# Montagem

ESP32  
ou  
NodeMCU  
utilize o GPIO 4

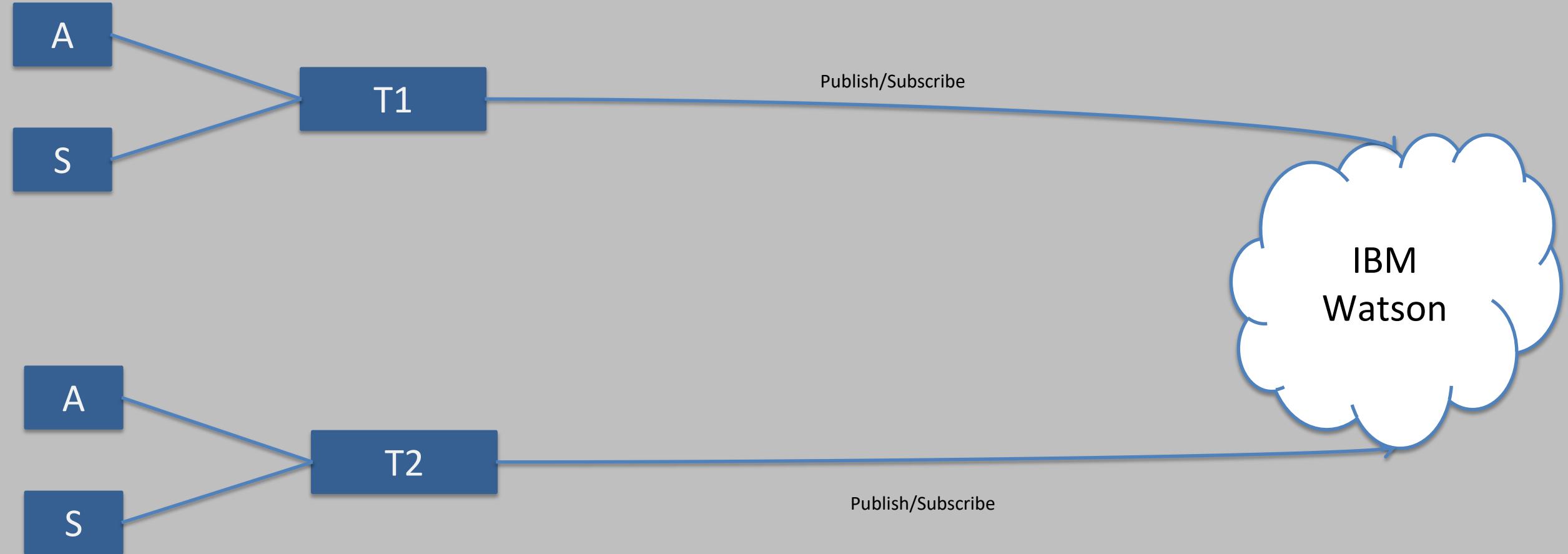
**Resistor de  
4.7k Ohms  
entre o 3.3v  
e o pino de  
dados**



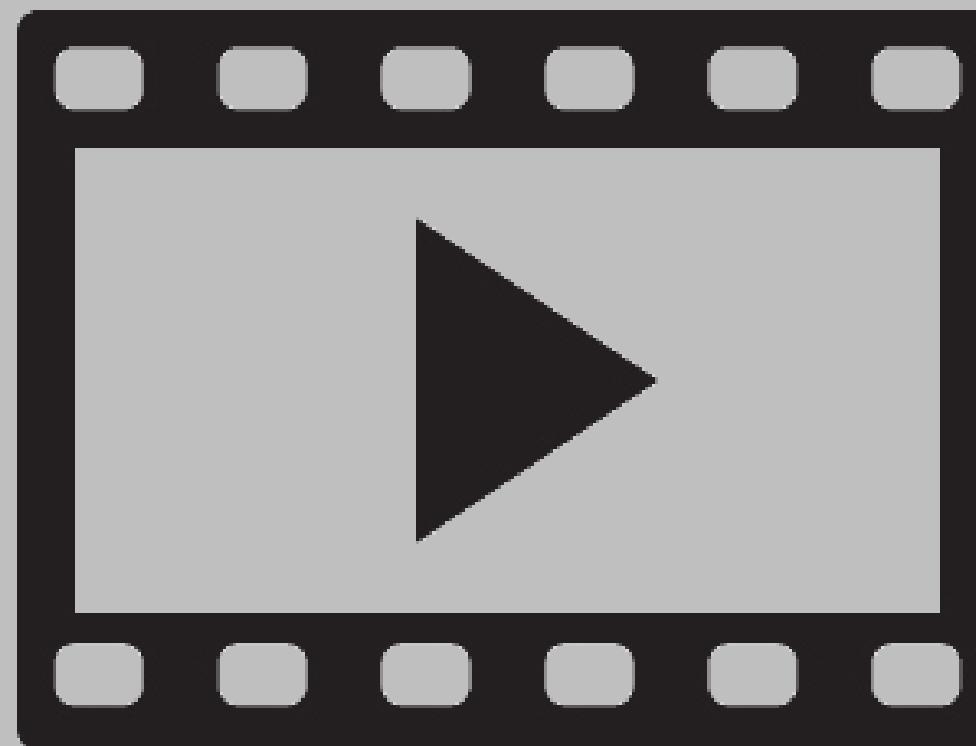
# Diagrama



# Diagramma



# Demonstração



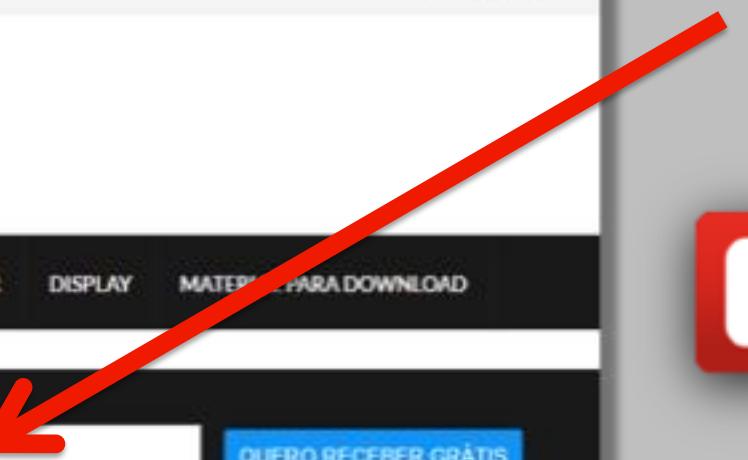


PRINCIPAL SOBRE FERNANDO K ARDUINO ESP8266 ESP32 LORAWAN MOTOR DISPLAY MATERIAIS PARA DOWNLOAD

Receba o meu conteúdo  
GRATUITAMENTE

Insira aqui seu melhor email...

QUERO RECEBER GRÁTIS



Seu e-mail



Inscreva-se



Motor de Passo Nema 23 com Driver TB6600 e Arduino Due

by Fernando K Tecnologia · 2:44 PM

Hoje vamos voltar a falar de Motor de Passo. Vamos utilizar um Nema 23 que será controlado por um Driver TB6600 e um Arduino Due. É p...

Leia mais

QUAL ASSUNTO VOCÊ TEM?

- Arduino
- ESP8266
- ESP32
- Motor
- Display
- Sensors

You may select multiple answers.

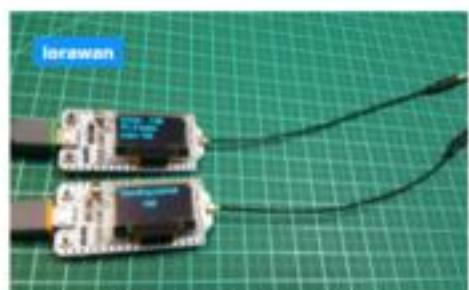
Votar Exibir resultados

Votes ate a momento: 12  
Das votantes para votar: 49

FACEBOOK



Fernando K Tec  
16.235 curtidas



ESP32 Longa Distância - LoRaWan

by Fernando K Tecnologia · 9:45 AM

Neste artigo vamos tratar da LoRaWAN, uma rede que vai longe gastando pouca energia. Mas, o quanto "longe"? Com o chip que uso no vídeo...

Leia mais



Motor de HD com Arduino

by Fernando K Tecnologia · 2:00 PM

# Bibliotecas

Na **ide do Arduino** vá no menu **Sketch -> Incluir Biblioteca -> Gerenciar Bibliotecas...**

Na tela que abrir digite na busca **DHT** e instale a lib **DHT sensor library**.

Depois digite **PubSubClient** e instale a lib PubSubClient



# Biblioteca para leitura de Temperatura e Umidade

Gerenciador de Biblioteca

Tipo Todos Tópico Todos dht

**DHT sensor library by Adafruit Versão 1.3.0 INSTALLED**  
**Arduino library for DHT11, DHT22, etc Temp & Humidity Sensors** Arduino library for DHT11, DHT22, etc Temp & Humidity Sensors  
[More info](#)

**DHT sensor library for ESPx by beegee\_tokyo**  
**Arduino ESP library for DHT11, DHT22, etc Temp & Humidity Sensors** Optimized libray to match ESP32 requirements. Last changes:  
Updated example.  
[More info](#)

**Grove Temperature And Humidity Sensor by Seeed Studio**  
**Arduino library to control Grove Temperature And Humidity Sensor, it contains chip DHT11 AM2302.** This temperature & humidity sensor provides a pre-calibrated digital output. A unique capacitive sensor element measures relative humidity and the temperature is measured by a negative temperature coefficient (NTC) thermistor. It has excellent reliability and long term stability.  
[More info](#)

**SimpleDHT by Wintlin Versão 1.0.6 INSTALLED**  
**Arduino Temp & Humidity Sensors for DHT11 and DHT22.** Simple C++ code with lots of comments. strictly follow the standard DHT

**Fechar**



# Biblioteca MQTT

Gerenciador de Biblioteca

Tipo Todos Tópico Todos PubSubClient

**MFUthings** by Wathanyu Phromma  
This is the library that be used in [www.mfuthings.com](http://www.mfuthings.com) which belongs to Mae Fah Luang University this library has dependencies that are PubSubClient and ESP8266WiFi so make sure you installed these libraries also make sure the ArduinoIDE version is 1.6.8 or greater  
[More info](#)

**PubSubClient** by Nick O'Leary Versão 2.6.0 **INSTALLED**  
**A client library for MQTT messaging.** MQTT is a lightweight messaging protocol ideal for small devices. This library allows you to send and receive MQTT messages. It supports the latest MQTT 3.1.1 protocol and can be configured to use the older MQTT 3.1 if needed. It supports all Arduino Ethernet Client compatible hardware, including the Intel Galileo/Edison, ESP8266 and TI CC3000.  
[More info](#)

**PubSubClientTools** by Simon Christmann  
**Tools for easier usage of PubSubClient** Provides useful tools for PubSubClient, however they may consume more power and storage. Therefore it's recommended for powerful microcontrollers like ESP8266.  
[More info](#)

**Fechar**



# MQTT.ino

```
//Verifica qual ESP está sendo utilizado
//e importa a lib e wifi correspondente
#if defined(ESP8266)
#include <ESP8266WiFi.h>
#else
#include <WiFi.h> // ESP32
#endif

//Lib de MQTT
#include <PubSubClient.h>
//Lib do sensor de temperatura e umidade
#include <DHT.h>
```



# MQTT.ino

```
//Intervalo entre os envios
#define INTERVAL 1000

//Substitua pelo SSID da sua rede
#define SSID "TesteESP"

//Substitua pela senha da sua rede
#define PASSWORD "87654321"

//Server MQTT que iremos utilizar
#define MQTT_SERVER "quickstart.messaging.internetofthings.ibmcloud.com"

//Nome do tópico que devemos enviar os dados
//para que eles apareçam nos gráficos
#define TOPIC_NAME "iot-2/evt/status/fmt/json"

//ID que usaremos para conectar
//QUICK_START deve permanecer como está
const String QUICK_START = "d:quickstart:arduino:";
```



# MQTT.ino

```
//No DEVICE_ID você deve mudar para um id único  
//Aqui nesse exemplo utilizamos o MAC Address  
//do dispositivo que estamos utilizando  
//Servirá como identificação no site  
//https://quickstart.internetofthings.ibmcloud.com  
const String DEVICE_ID = "240ac40e3fd0";  
  
//Concatemos o id do quickstart com o id do nosso  
//dispositivo  
const String CLIENT_ID = QUICK_START + DEVICE_ID;
```



# MQTT.ino

```
//Cliente WiFi que o MQTT irá utilizar para se conectar
WiFiClient wifiClient;

//Cliente MQTT, passamos a url do server, a porta
//e o cliente WiFi
PubSubClient client(MQTT_SERVER, 1883, wifiClient);

//Tempo em que o último envio foi feito
long lastPublishTime = 0;

//Objeto que realiza a leitura da temperatura e da umidade
DHT dht(4, DHT22);

//Variável para guardarmos o valor da temperatura
float temperature = 0;

//Variável para guardarmos o valor da umidade
float humidity = 0;
```



# MQTT.ino - setup

```
void setup() {  
    Serial.begin(115200);  
    //Incializamos o dht  
    dht.begin();  
    //Conectamos à rede WiFi  
    setupWiFi();  
    //Conectamos ao server MQTT  
    connectMQTTSERVER();  
}
```



# MQTT.ino - loop

```
void loop() {
    //Tempos agora em milisegundos
    long now = millis();

    //Se o tempo desde o último envio for maior que o intervalo de envio
    if (now - lastPublishTime > INTERVAL) {
        //Atualizamos o tempo em que ocorreu o último envio
        lastPublishTime = now;
        //Fazemos a leitura da temperatura e umidade
        readSensor();
        Serial.print("Publish message: ");
        //Criamos o json que enviaremos para o server mqtt
        String msg = createJsonString();
        Serial.println(msg);
        //Publicamos no tópico onde o servidor espera para receber
        //e gerar o gráfico
        client.publish(TOPIC_NAME, msg.c_str());
    }
}
```



# MQTT.ino - setupWiFi

```
//Função responsável por conectar à rede WiFi
void setupWiFi() {
    Serial.println();
    Serial.print("Connecting to ");
    Serial.print(SSID);

    //Manda o esp se conectar à rede através
    //do ssid e senha
    WiFi.begin(SSID, PASSWORD);

    //Espera até que a conexão com a rede seja estabelecida
    while (WiFi.status() != WL_CONNECTED) {
        delay(500);
        Serial.print(".");
    }

    //Se chegou aqui é porque conectou
    Serial.println("");
    Serial.println("WiFi connected");
}
```



# MQTT.ino - connectMQTTSERVER

```
//Função responsável por conectar ao server MQTT
void connectMQTTSERVER() {
    Serial.println("Connecting to MQTT Server...");
    //Se conecta ao id que definimos
    if (client.connect(CLIENT_ID.c_str())) {
        //Se a conexão foi bem sucedida
        Serial.println("connected");
    } else {
        //Se ocorreu algum erro
        Serial.print("error = ");
        Serial.println(client.state());
    }
}
```



# MQTT.ino - readSensor

```
//Função responsável por realizar a leitura  
//da temperatura e umidade  
void readSensor(){  
    float value;  
    //Faz a leitura da temperatura  
    value = dht.readTemperature();  
    //Se o valor lido é válido  
    if(!isnan(value)){  
        //Armazena o novo valor da temperatura  
        temperature = value;  
    }  
    //Faz a leitura da umidade  
    value = dht.readHumidity();  
    //Se o valor for válido  
    if(!isnan(value)){  
        //Armazena o novo valor da umidade  
        humidity = value;  
    }  
}
```



# MQTT.ino - createJsonString

```
//Função responsável por criar
//um Json com os dados lidos
String createJsonString() {
    String data = "{";
    data+= "\"d\": {";
    data+="\"temperature\":";
    data+=String(temperature);
    data+=",";
    data+="\"humidity\":";
    data+=String(humidity);
    data+="}";
    data+="}";
    return data;
}
```



# Gráfico

Para visualizar o gráfico do sensor vá até <https://quickstart.internetofthings.ibmcloud.com>

No campo Device id digite o DEVICE\_ID que você definiu no código.

É importante mudar no código este devido id para um id única, utilizado somente por você, para não dar conflito com dados enviados por outra pessoa.

Aceite os termos e clique em Go.

Mude aqui e envie  
o sketch para o ESP

```
//No DEVICE_ID você deve mudar para um id único
//Aqui nesse exemplo utilizamos o MAC Address
//do dispositivo que estamos utilizando
//Servirá como identificação no site
//https://quickstart.internetofthings.ibmcloud.com
const String DEVICE_ID = "240ac40e3fd0";
```



## Quickstart

No sign-up required to see how easy it is to connect your device to Watson IoT Platform and view live sensor data

I accept IBM's Terms of Use

240ac40e3fd0

Go

Copie para este campo  
e clique em Go





## Quickstart

No sign-up required to see how easy it is to connect your device to Watson IoT Platform and view live sensor data

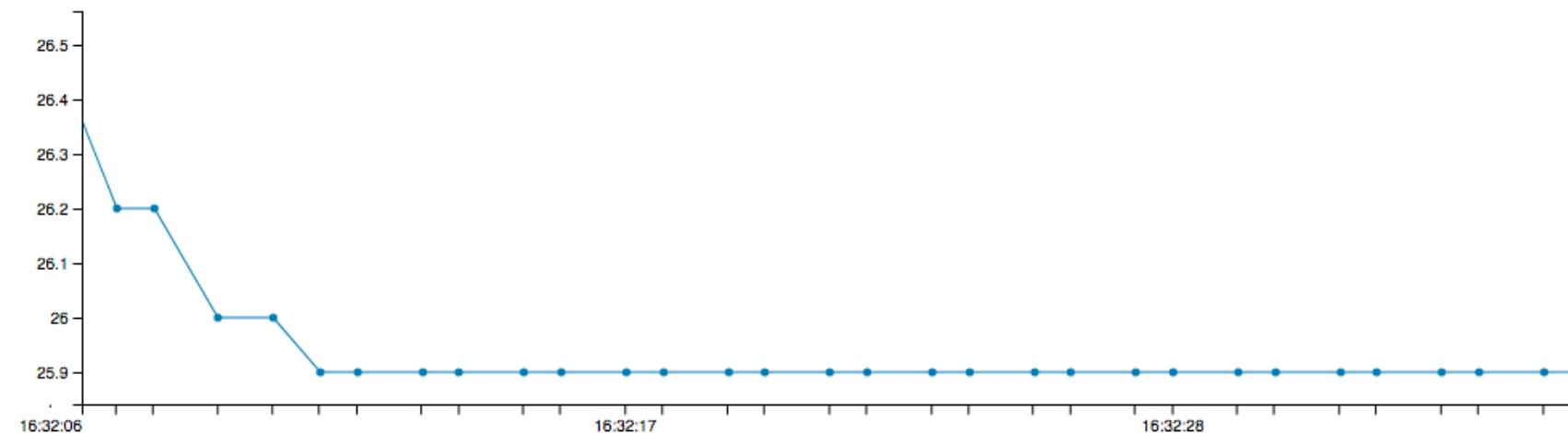
240ac40e3fd0

Go

Last message received at 4:32:36 PM

240ac40e3fd0

status.temperature



Event	Datapoint	Value	Time Received
status	temperature	25.9	Apr 20, 2018 4:32:36 PM
status	humidity	55	Apr 20, 2018 4:32:36 PM

Em [www.fernandok.com](http://www.fernandok.com)

Download arquivos PDF e INO do código fonte

