MQTT et ESP32

tuto-mqtt-sur-esp32--mosquitto

ESP32 MQTT – Publier et s'abonner avec l'IDE Arduino

Utiliser le capteur DHT11 avec ESP32 et mqtt broker

Qu’est-ce que le MQTT server

Le MQTT (transport de télémétrie Message Queuing) est un protocole de messagerie Client / Serveur de publication / abonnement qui est léger, idéal et facile à mettre en œuvre pour les appareils connectés «Internet of things». Ce protocole vous permet d'envoyer et de recevoir des messages MQTT.

Le protocole MQTT fonctionne généralement sur le TCP / IP. Il est compatible avec plusieurs architectures, donc vous pouvez l’utiliser avec toutes les cartes Arduino ; ESP32 ; ESP8266 ; Raspberry pi et autre microcontrôleurs. MQTT client and MQTT broker

MQTT Client: tout appareil (comme esp32, ESP8266…) connecté à un MQTT broker via un réseau sans fil (wifi …). Le rôle du client est la collecte des informations et les publie sous forme de messages à un broker sur le réseau. MQTT Broker: Le rôle principal du MQTT broker est de recevoir les informations (température, humidité), filtrer les informations, déterminer les abonnés pour chaque message et envoyer ce dernier à ces clients abonnés (des applications mobile, des pages web…). MQTTLens MQTTLens est une application Google chrome qui vous permet de publier et de s’abonner sur le MQTT broker. DHT11sensor and ESP32 et cloudMQTT

Le but de ce projet est de lire la température et l'humidité par une application cloudMQTT via un réseau wifi.

Matériels utilisés

```
Une carte ESP32
Un capteur de température et d’humidité DHT11
Résistance 1k
Une plaque d’essai
Fils de connexion
Câble USB pour l’alimentation
```

Schéma de réalisation sous le logiciel fritzing

Affectation des composants aux broches d’ESP32

Le capteur d’humidité est connecté à la broche numéro 4 Configuration du compte MQTT Cloud

Pour configurer un compte sur Cloud MQTT, accédez à son site Web officiel ici et inscrivez-vous à

l'aide de votre Gmail ou avec votre compte GitHub.

Après la connexion, cliquez sur «+ Créer New Instance» pour créer une nouvelle instance.

Entrez maintenant le nom qui décrit votre instance et sélectionnez le plan que vous voulez dans l'option de plan.

Dans l'onglet Région, sélectionnez la région et cliquez sur «Examiner».

Maintenant, votre instance est créée et vous pouvez afficher vos données comme le Server, le Port, le User et le Password.

Pour vous abonner et publier des messages sur le broker MQTT, vous pouvez utiliser une application MQTTlens de Google Chrome ou une autre application. Vous pouvez télécharger l'application MQTTlens ici.

Lancez cette application et configurez une connexion avec MQTT broker comme suite:cliquez sur « connexions » et, dans la fenêtre suivante, saisissez les détails de la connexion à partir du compte CloudMQTT.

Programme_de_simulation.ino

```
#include <WiFi.h>

#include <PubSubClient.h>

#include "DHT.h"

const char* ssid = "xxxxxxxxxxxx";

const char* password =  "xxxxxxxx";

const char* mqttServer = "xxxxxxxxxxxxxxxxx";

const int mqttPort = xxxxxxxx;

const char* mqttUser = "xxxxxxx";

const char* mqttPassword = "xxxxxxxxx";

#define DHTPIN D2

#define DHTTYPE DHT11

DHT dht(DHTPIN, DHTTYPE);

//#define humidity_topic "sensor/humidity"

//#define humidity_topic "sensor/humidity"

WiFiClient espClient;
```

```
PubSubClient client(espClient);

long lastMsg = 0;

char msg[50];

int value = 0;

void setup() {

    Serial.begin(115200);

    dht.begin();

    pinMode(hsensor1,INPUT);

    WiFi.begin(ssid, password);

    while (WiFi.status() != WL_CONNECTED) {

    delay(500);

    Serial.println("Connecting to WiFi..");

    }

    Serial.println("Connected to the WiFi network");

    client.setServer(mqttServer, mqttPort);

    //client.setCallback(callback);

    while (!client.connected()) {

    Serial.println("Connecting to MQTT...");

    if (client.connect("ESP32Client", mqttUser, mqttPassword )) {

    Serial.println("connected");

    } else {

    Serial.print("failed with state ");

    Serial.print(client.state());

    delay(2000);

          }

    }
```

```
}

void reconnect() {

  // Loop until we're reconnected

  while (!client.connected()) {

    Serial.print("Attempting MQTT connection...");

    // Attempt to connect

    if (client.connect("ESP32Client")) {

      Serial.println("connected");

      // Subscribe

      client.subscribe("esp32/output");

    } else {

      Serial.print("failed, rc=");

      Serial.print(client.state());

      Serial.println(" try again in 5 seconds");

      // Wait 5 seconds before retrying

      delay(5000);

    }

  }

}

void loop() {

  if (!client.connected()) {

    reconnect();

  }

  client.loop();
  long now = millis();

  // Send a message every minute
```

```
  if (now - lastMsg > 100) {

    lastMsg = now;

   // Lire l'entrée numérique dans une valeur boolienne

  float humidite = dht.readHumidity();

  float temperature = dht.readTemperature();

    return;

  }

  // Les conditions d'humidité

  if(temperature<=30){

   // Pompe  marche

   Serial.print("la temperature est bas");

   client.publish("information", String("la temperature est
bas").c_str(), true);}

    else {

    Serial.print("la température est élevée");

    client.publish("information", String("la température est
élevée").c_str(), true);
   delay(500);

       }

   Serial.print("Humidité:");

   Serial.print(humidite);

   Serial.println("%") ;

   Serial.print(temperature);

   Serial.println("°C") ;

   client.publish("esp32/Humidité", String(Humidité).c_str(), true);

   client.publish("esp32/temperature", String(temperature).c_str(),
true);
```

```
  }

  //delay(2000);
```