13_Cloud_MQTT_ESP8266_Relazione

Citterio Giorgio e Colombo Umberto

Lo scopo di questa è utilizzare il protocollo MQTT per lo scambio di messaggi fra dispositivi IoT su ESP8266.

parte 1

Nella prima parte dell'attività abbiamo installato le librerie necessarie al controllo dell'ESP8266 dall' IDE di Arduino.

parte 2

SENSORE:

Con il seguente programma andiamo a prendere i valori del sensore dal pin A0 dell'ESP8266 e li pubblichiamo sul topic *tps/inviaSensore* in formato json ogni 10 secondi.

sketch_sensore_esp8266:

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

// WiFi
const char *WIFI_SSID = "Greppi-2G";  // inserire i dati della rete WiFi
const char *WIFI_PASSWORD = "withProxy";

// MQTT Broker
const char *MQTT_BROKER = "172.17.4.29";  // indirizzo IP del broker
const char *MQTT_USERNAME = ""; // se necessario
const char *MQTT_PASSWORD = "";
const int MQTT_PORT = 1883;

// topic
const char *INVIA = "tps/inviaSensore";

// oggetti per wifi e mqtt
WiFiClient espClient;
PubSubClient client(espClient);

// setup
void setup()
{
  Serial.begin(111520);
```

```
  Serial.println("Inizio");
  Serial.print("Connessione al WiFi..");
  WiFi.begin(WIFI_SSID, WIFI_PASSWORD);
  while (WiFi.status() != WL_CONNECTED)
  {
    delay(500);
    Serial.print(".");
  }

  Serial.println("");
  Serial.println("Connesso");

  // assegna un nome questo client
  String client_id = "esp8266-client-gioUmbe" + String(WiFi.macAddress());
  Serial.printf("%s in connessione al MQTT ..", client_id.c_str());
  client.setServer(MQTT_BROKER, MQTT_PORT);
  while (!client.connected())
  {
    if (client.connect(client_id.c_str(), MQTT_USERNAME, MQTT_PASSWORD))
    {
      Serial.println("Connesso al broker mqtt");
    }
    else
    {
      Serial.print("Fallito con codice di errore: ");
      Serial.println(client.state());
      Serial.println("Ritento");
      delay(2000);
    }
  }

  // pubblicazione di un primo messaggio
  client.publish(INVIA, "Attivo");
}

// ciclo di invio messaggi
void loop()
{
  StaticJsonDocument<200> doc;
  int num = analogRead(A0);
  char s[5];
  sprintf(s, "%04d", num);
  JsonObject jsonObj = doc.to<JsonObject>();
  jsonObj["valoreSensore"] = s;
  char buffer[200];
  serializeJson(jsonObj, buffer);
  Serial.write(buffer);
  Serial.println();
  client.publish(INVIA, buffer);
  delay(10000);
  client.loop();
}
```

MOTORE:

Con il seguente programma andiamo a prendere i valori inviati in formato json sul topic *tps/riceviAttuatore* li deserializziamo e andiamo a controllare il motore.

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

#define AVANTI_PIN 12 //D6
#define INDIETRO_PIN 14 //D5
#define VELOCITA_PIN 0 //D3

// WiFi
const char *WIFI_SSID = "Greppi-2G";  // inserire i dati della rete WiFi
const char *WIFI_PASSWORD = "withProxy";

// MQTT Broker
const char *MQTT_BROKER = "172.17.4.29"; // indirizzo IP del broker
const char *MQTT_USERNAME = ""; // se necessario
const char *MQTT_PASSWORD = "";
const int MQTT_PORT = 1883;

// topic
const char *RICEVI = "tps/riceviAttuatore";

// oggetti per wifi e mqtt
WiFiClient espClient;
PubSubClient client(espClient);

// setup
void setup()
{
  pinMode(AVANTI_PIN, OUTPUT);
  pinMode(INDIETRO_PIN, OUTPUT);
  Serial.begin(111520);
  Serial.println("Inizio");
  Serial.print("Connessione al WiFi..");
  WiFi.begin(WIFI_SSID, WIFI_PASSWORD);
  while (WiFi.status() != WL_CONNECTED)
  {
    delay(500);
    Serial.print(".");
  }

  Serial.println("");
  Serial.println("Connesso");

  // assegna un nome questo client
  String client_id = "esp8266-client-gioUmbe" + String(WiFi.macAddress());
  Serial.printf("%s in connessione al MQTT ..", client_id.c_str());
  client.setServer(MQTT_BROKER, MQTT_PORT);
  client.setCallback(callback);
  while (!client.connected())
  {
    if (client.connect(client_id.c_str(), MQTT_USERNAME, MQTT_PASSWORD))
    {
      Serial.println("Connesso al broker mqtt");
    }
    else
    {
      Serial.print("Fallito con codice di errore: ");
      Serial.println(client.state());
      Serial.println("Ritento");
```

```
      delay(2000);
    }
  }
  //sottoscrizione
  client.subscribe(RICEVI);
}

void loop()
{
  client.loop();
}

// funzione di callback
void callback(char *topic, byte *payload, unsigned int length)
{
  StaticJsonDocument<200> jsonDoc;
  Serial.print("Arrivato un messaggio nel topic: ");
  Serial.println(topic);
  Serial.println("Messaggio:");
  for (int i = 0; i < length; i++)
  {
    Serial.print((char) payload[i]);
  }
  Serial.println();
  Serial.println("-----------------------");

  deserializeJson(jsonDoc, payload, length);
  const char* direzione = jsonDoc["direzione"];
  int velocita = jsonDoc["velocita"];
  Serial.println(velocita);
  Serial.println(direzione);
  if (strcmp("A", direzione) == 0)
  {
    digitalWrite(INDIETRO_PIN, LOW);
    digitalWrite(AVANTI_PIN, HIGH);
    analogWrite(VELOCITA_PIN, velocita);
  }
  if (strcmp("I", direzione) == 0)
  {
    digitalWrite(INDIETRO_PIN, HIGH);
    digitalWrite(AVANTI_PIN, LOW);
    analogWrite(VELOCITA_PIN, velocita);
  }
}
```