MQTT esp32

1. Aggiunto esp32 secondo le specifiche di Vincenzo
2. Installare le librerie EspMQTTClient e PubSubClient (forse anche WiFi)
3. Usare questo programma di esempio, andando sul server esterno ***broker.hivemq.com***

```
#include "EspMQTTClient.h"

int i = 0;

EspMQTTClient client(
  "Infostrada-BA5E3C",
  "QcJwF7EmUf",
  //"broker.hivemq.com",  // MQTT Broker server
  "192.168.1.110",  // Mosquitto
  "",   // Can be omitted if not needed
  "",   // Can be omitted if not needed
  "TestClient",     // Client name that uniquely identify your device
  1883              // The MQTT port, default to 1883. this line can be omitted
);

void setup()
{
  Serial.begin(115200);
  client.enableDebuggingMessages();
}

void ricevi(String payload)
{
  Serial.println(payload);
}

void invia()
{
  if (client.isConnected()) {
    i = i + 1;
    client.publish("mytopic/testinvia", String(i));
    client.executeDelayed(10000, invia);
  }
}

void onConnectionEstablished()
{
  client.subscribe("mytopic/testricevi", ricevi);
  client.publish("mytopic/testinvia", "Attivo");
  client.executeDelayed(5000, invia);
}

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

4. Usare su Android IoTMQTTPanel o simili
5. Installare Mosquitto, impostando le seguenti opzioni nel file ***mosquitto.conf***

```
listener 1883
allow_anonymous true
```

6. Lanciare mosquitto con le seguenti opzioni:

```
mosquitto –v –c mosquitto.conf
```

7. Lanciare ngrok

```
ngrok tcp 1883
```

8. Versione senza l'uso della libreria EspMQTTClient

```
#include <WiFi.h>
#include <PubSubClient.h>

// WiFi
const char *ssid = "emilio";
const char *password = "mafeking";

// MQTT Broker
const char *mqtt_broker = "192.168.43.48";
const char *invia = "mytopic/invia";
const char *ricevi = "mytopic/ricevi";
const char *mqtt_username = "";
const char *mqtt_password = "";
const int mqtt_port = 1883;

WiFiClient espClient;
PubSubClient client(espClient);

void setup() {

  Serial.begin(115200);
  Serial.println("Inizio");

  Serial.print("Connecting to WiFi..");
  WiFi.begin(ssid, password);
  while (WiFi.status() != WL_CONNECTED) {
    delay(500);
    Serial.print(".");
  }
  Serial.println("");
  Serial.println("Connected to the WiFi network");

  String client_id = "esp32-client-" + String(WiFi.macAddress());
  Serial.printf("%s connecting to MQTT broker .." , 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("mqtt broker connected");
    } else {
      Serial.print("failed with state ");
      Serial.println(client.state());
      Serial.println("retrying");
      delay(2000);
    }
  }
  // publish and subscribe
  client.publish(invia, "Attivo");

  client.subscribe(ricevi);
}

void callback(char *topic, byte *payload, unsigned int length) {
  Serial.print("Message arrived in topic: ");
  Serial.println(topic);
  Serial.print("Message:");
  for (int i = 0; i < length; i++) {
    Serial.print((char) payload[i]);
  }
  Serial.println();
  Serial.println("-----------------------");
}
int n = 0;
void loop() {
  char s[20];
  n = n + 1;
  sprintf(s,"%d",n);
  client.publish(invia, s);
  delay(10000);
  client.loop();
}
```

9. Versione con Arduino e esp8266

```
#include <SoftwareSerial.h>
#include "WiFiEsp.h"
#include <PubSubClient.h>
SoftwareSerial ESP8266(10, 11);
// WiFi
const char *ssid = "emilio";
const char *password = "mafeking";

// MQTT Broker
const char *mqtt_broker = "192.168.43.48";
//const char *mqtt_broker = "172.17.5.14";
const char *invia = "mytopic/invia";
const char *ricevi = "mytopic/ricevi";
const char *mqtt_username = "";
const char *mqtt_password = "";
const int mqtt_port = 1883;

WiFiEspClient espClient;
PubSubClient client(espClient);
```

```
void setup() {

  Serial.begin(115200);
  ESP8266.begin(9600);
  WiFi.init(&ESP8266);
  Serial.println("Inizio");

  Serial.print("Connecting to WiFi..");
  int status = WiFi.begin(ssid, password);
  if (status == WL_CONNECTED) {
    Serial.println();
    Serial.println("Connected to WiFi network.");
  } else {
    WiFi.disconnect(); // remove the WiFi connection
    Serial.println();
    Serial.println("Connection to WiFi network failed.");
  }
  Serial.println("");
  Serial.println("Connected to the WiFi network");

  String client_id = "esp32-client-" ;//+ String(WiFi.macAddress());
  //Serial.printf("%s connecting to MQTT broker .." , 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("mqtt broker connected");
    } else {
      Serial.print("failed with state ");
      Serial.println(client.state());
      Serial.println("retrying");
      delay(2000);
    }
  }
  // publish and subscribe
  client.publish(invia, "Attivo");

  client.subscribe(ricevi);
}

void callback(char *topic, byte *payload, unsigned int length) {
  Serial.print("Message arrived in topic: ");
  Serial.println(topic);
  Serial.print("Message:");
  for (int i = 0; i < length; i++) {
    Serial.print((char) payload[i]);
  }
  Serial.println();
  Serial.println("-----------------------");
}
int n = 0;
void loop() {
  char s[20];
  n = n + 1;
  sprintf(s,"%d",n);
  client.publish(invia, s);
  delay(5000);
  client.loop();
}
```