# IoT Communication - MQTT

Welcome to Day 3 of our IoT Workshop Series! Building upon the foundation laid on Day 2, where you delved into the intricacies of sensors and their protocols, today is all about application. Get ready to harness that knowledge and dive into the world of data transmission as we explore how to gather and send sensor data using MQTT.

## 1. MQTT in detail

## 2. ESP8266 and WiFi

Next, we'll take a closer look at connecting the ESP8266 to WiFi. This step is crucial for enabling wireless communication, allowing your devices to seamlessly interact with the digital world.

In this example code, you will first of all connect to a WiFi network and then try to access a predefined API. This API is the root of our public weather station API. You can find out more about the project right here: HSRW Weather Station at Campus Kamp-Lintfort.

wifi-http-api-example.ino

```
#include <ESP8266WiFi.h>
#include <WiFiClientSecure.h>

const uint16_t HTTP_PORT = 443; // Use 443 for HTTPS and 80 for HTTP

// Replace with your WiFi credentials
const char* SSID = "iotlab";
const char* PASSWORD = "iotlab18";

// Replace with your API details
const char* API_HOST = "weather.eolab.de";
const char* API_ENDPOINT = "/api";

const uint8_t PERIOD_MINUTES = 1;

// Create an instance of WiFiClientSecure
WiFiClientSecure wifi_client;

void setup() {
  Serial.begin(115200);

  // Connect to Wi-Fi
  WiFi.begin(SSID, PASSWORD);
```

```
  Serial.print("\n\nConnecting to ");
  Serial.print(SSID);

  while (WiFi.status() != WL_CONNECTED) {
    delay(1000);
    Serial.print(".");
  }

  Serial.println("\nConnected to WiFi");

  // Set the client to verify the server's certificate
  wifi_client.setInsecure();

  // Give the client a chance to perform the handshake
  delay(1000);
}

void loop() {
  if (WiFi.status() == WL_CONNECTED) {

    wifi_client.connect(API_HOST, HTTP_PORT);
    // Make an HTTPS GET request
    wifi_client.println("GET " + String(API_ENDPOINT) + " HTTP/1.1");
    wifi_client.println("Host: " + String(API_HOST));
    wifi_client.println("Connection: close");
    wifi_client.println();

    // Wait for the response
    while (wifi_client.connected()) {
      if (wifi_client.available()) {
        // read an incoming byte from the server and print it to serial
monitor:
        char c = wifi_client.read();
        Serial.print(c);
      }
    }

    wifi_client.stop();
  }

  // Wait for a while before making the next request
  delay(PERIOD_MINUTES * 60 * 1000); // convert from min to sec (60)
and from sec to ms (1000)
}
```

## 3. Our first MQTT Publish

Stepping ahead, you'll have the chance to implement your first MQTT publish from the ESP8266.

While the following example provides a static demonstration, we encourage you to take it a step further by incorporating one of the sensors available to you.

To start, search and install the library “PubSubClient ” by Nick O'Leary in Arduino IDE or download and install it manually from source Github PubSubClient by knolleary.

mqtt-publish.ino

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

const uint16_t HTTP_PORT = 443; // Use 443 for HTTPS and 80 for HTTP

// Replace with your WiFi credentials
const char* SSID = "Eutopiq"; // [USER INPUT]
const char* PASSWORD = "Eutopiq321"; // [USER INPUT]

// Create an instance of WiFiClientSecure
WiFiClientSecure wifi_client;

// ==================
// MQTT
// ==================

#define MQTT_MSG_BUFFER_SIZE 50 // [USER INPUT]
StaticJsonDocument<MQTT_MSG_BUFFER_SIZE> doc;

const uint8_t PERIOD_MINUTES = 1;

const char* MQTT_SERVER = "broker.hivemq.com"; // [USER INPUT]
const uint16_t MQTT_PORT = 8883; // TLS TCP PORT for HTTPS connections,
otherwise use 1883
const char* MQTT_OUTPUT_TOPIC = "colmayor/workshop/2024/user1"; //
[USER INPUT]
const char* MQTT_INPUT_TOPIC = "colmayor/workshop/2024/to-user1"; //
[USER INPUT]

PubSubClient mqtt_client(wifi_client);
unsigned long lastMsg = 0;
char msg[MQTT_MSG_BUFFER_SIZE];

// count the number of msg
uint value = 0;

void setup_wifi() {
  // Connect to Wi-Fi
  WiFi.begin(SSID, PASSWORD);

  Serial.print("\n\nConnecting to ");
```

```
  Serial.print(SSID);

  while (WiFi.status() != WL_CONNECTED) {
    delay(1000);
    Serial.print(".");
  }

  Serial.println("\nConnected to WiFi");
  Serial.print("IP address: ");
  Serial.println(WiFi.localIP());

  // Set the client to verify the server's certificate
  wifi_client.setInsecure();

  // Give the client a chance to perform the handshake
  delay(1000);
}

void callback(char* topic, byte* payload, unsigned int length) {
  // RAW payload
  Serial.print("Message arrived [");
  Serial.print(topic);
  Serial.print("] ");
  for (int i = 0; i < length; i++) {
    Serial.print((char)payload[i]);
  }
  Serial.println();

  // JSON formated payload
  //deserializeJson(doc, (const byte*)payload, length);
  //serializeJson(doc, Serial);
}

void mqtt_reconnect() {
  while (!mqtt_client.connected()){
    Serial.println("Attempting MQTT re-connection..");

    // Create a random client ID
    String clientId = "ESP8266-";
    clientId += String(random(0xffff), HEX);

    // Attempt to connect
    if (mqtt_client.connect(clientId.c_str())) {
      Serial.println("Successful reconnection.");
      // Once connected, publish an announcement...
      //JsonDocument doc;
      doc["msg"] = "hello world";
      doc["count"] = value;
      serializeJson(doc, msg);
      mqtt_client.publish(MQTT_OUTPUT_TOPIC, msg);
      mqtt_client.subscribe(MQTT_INPUT_TOPIC);
```

```
    } else {
      Serial.print("failed, rc=");
      Serial.print(mqtt_client.state());
      Serial.println(" try again in 10 seconds");
      // Wait before retrying
      delay(10000);
    }
  }
}

void setup() {
  Serial.begin(115200);

  setup_wifi();

  // to generate a random client ID
  // during reconnection with the
  // mqtt server connection
  randomSeed(micros());
  mqtt_client.setServer(MQTT_SERVER, MQTT_PORT);
  mqtt_client.setCallback(callback);
}

void loop() {
  if (!mqtt_client.connected()) {
    mqtt_reconnect();
  }

  mqtt_client.loop();

  unsigned long now = millis();

  if (now - lastMsg > (PERIOD_MINUTES*(60*1000))) {
    lastMsg = now;
    ++value;
    // JsonDocument doc;
    doc["msg"] = "hello world";
    doc["count"] = value;
    serializeJson(doc, msg);
    mqtt_client.publish(MQTT_OUTPUT_TOPIC, msg);
  }


}
```

## 4. Subscribe

Now, let's move on to an essential aspect of our workshop. We'll guide you through the process of

subscribing to an MQTT topic. This step is a practical gateway to interactive IoT applications. You'll learn how to exchange MQTT messages between devices, potentially triggering actions like LED reactions or serial outputs. While we'll provide initial guidance, feel free to explore further and experiment with this dynamic feature.

TODO.

## Recording