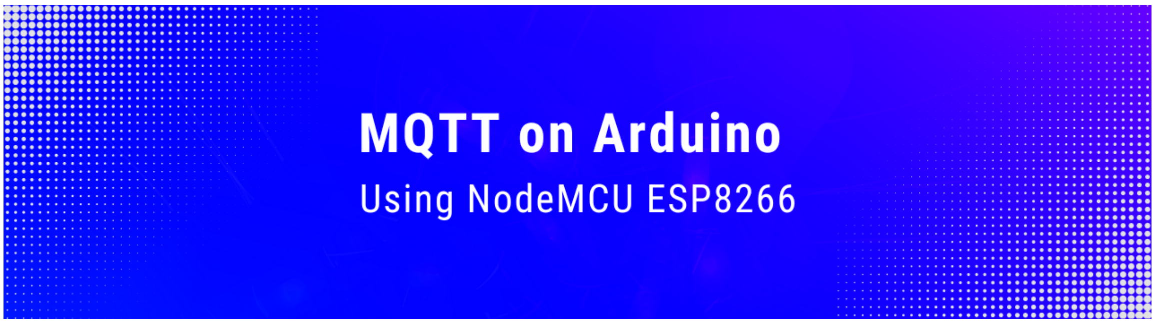


Getting Started with MQTT on Arduino Using NodeMCU ESP8266

Written by Kudzai Manditereza
Category: HiveMQ Cloud • IoT
Published: February 3, 2023

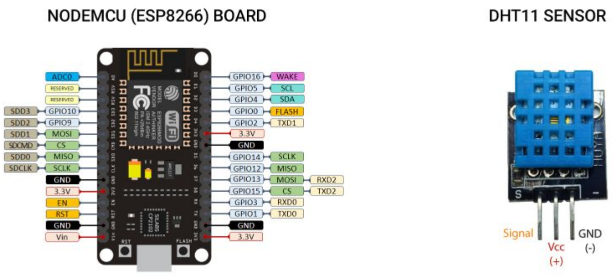
In this tutorial, I will show you how to use the Arduino IDE to program an ESP8266 microcontroller to read sensor data and publish it using MQTT to a HiveMQ Cloud MQTT broker.

Further, I'll show you how to receive MQTT messages as commands from the cloud and activate the microcontroller output to switch ON and OFF a Light Emitting Diode (LED).

What are ESP8266 NodeMCU Unit and DHT11 Sensor?

Let us look briefly at the hardware components we will use for this demo, the NodeMCU Micontroller Unit, and the DHT11 Temperature and Humidity Sensor.

Based on the ESP8266 WLAN chip, the NodeMCU is a small, fast, and Wi-Fi-capable microcontroller board. You can easily program it in the popular Arduino Integrated Development Environment (IDE). It has many Input/Output ports and various serial interfaces such as SPI, I2C, or RS232.

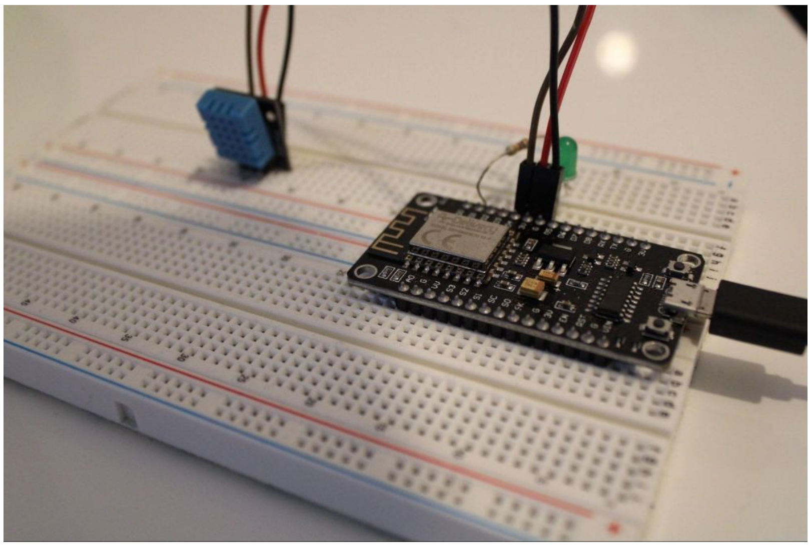


NodeMCU Microcontroller Board and DHT11 Sensor

The DHT11 is a sensor capable of measuring the relative humidity and temperature of the surrounding environment. It consists of three pins, two for power, and one is a signal line for transmitting the sensor data to the NodeMCU microcontroller.

Hardware Wiring and Circuit Diagram

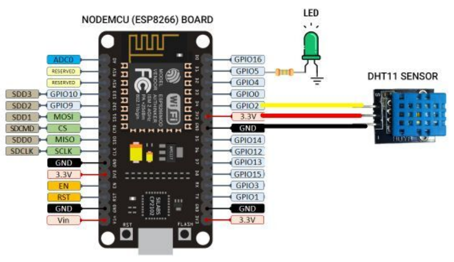
Next, let’s look at how to wire the hardware components for this demonstration.



The image above shows the setup for this demonstration, and the one below shows the visualization of the specific connection using a circuit diagram.

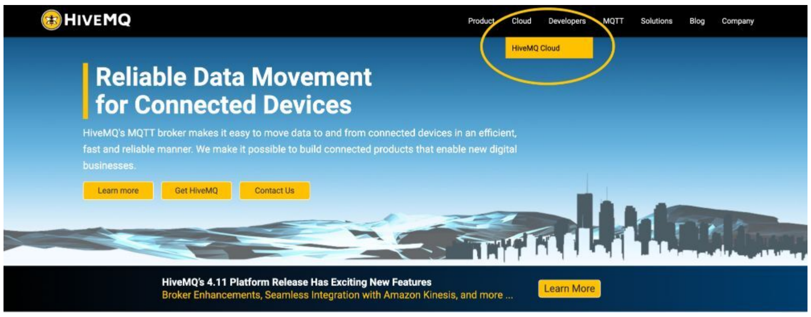
As shown, the connections are as follows:

- Vcc of the DHT11 goes to +3v of the NodeMCU,
- Signal pin of the DHT11 goes into the Digital Pin GPIO2 of the NodeMCC, and the
- Ground pin of the DHT11 goes into the Ground Pin of the NodeMCU.
- A resistor connects the LED to the NodeMCU digital pin, GPIO5.



Setting up a HiveMQ Cloud MQTT Broker

Since I'll be publishing MQTT messages from the NodeMCU board to an MQTT Broker, I must create the MQTT Broker on HiveMQ Cloud and retrieve the broker details to use in the Arduino code.



Reliable Data Movement for Connected Devices

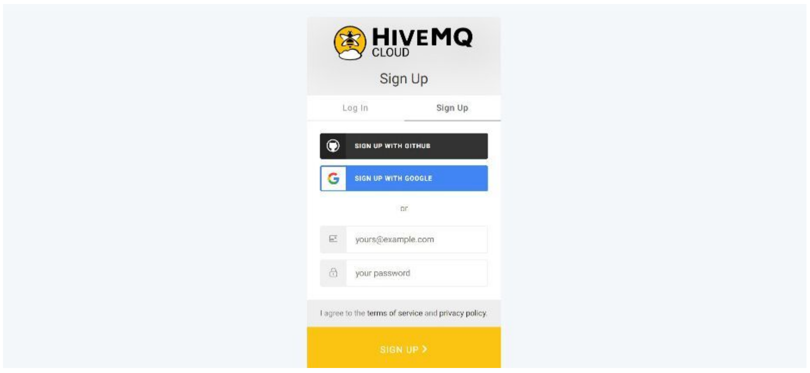
HiveMQ's MQTT broker makes it easy to move data to and from connected devices in an efficient, fast and reliable manner. We make it possible to build connected products that enable new digital businesses.

HiveMQ's 4.11 Platform Release Has Exciting New Features
Broker Enhancements, Seamless Integration with Amazon Kinesis, and more ...

I'll go to the **HiveMQ website** and select '*Cloud*' on the menu. I'll get the **HiveMQ Cloud** page and then click the '*Try out for free*' button. If you'd like, you can **sign up for HiveMQ Cloud directly here**.

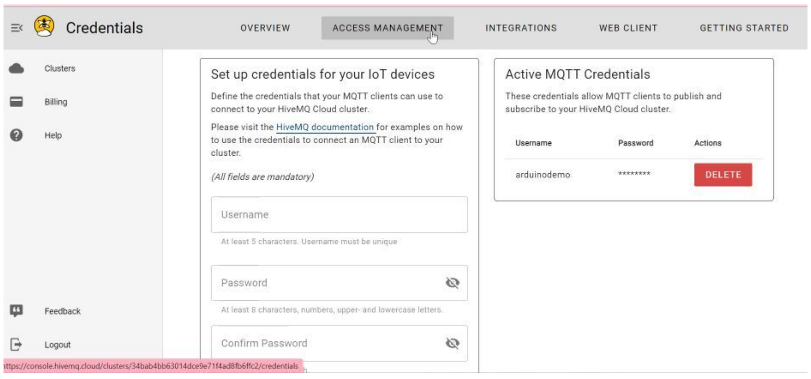
We are taken to a sign-up page and prompted to create a free account that allows us to connect up to 100 devices.

If you are accessing the HiveMQ Cloud portal for the first time, you'll need to provide an email address and password and follow the simple steps to confirm your email and create your account.



I agree to the terms of service and privacy policy.

An MQTT broker cluster will automatically be created when you've successfully created your account. Start by setting up credentials to successfully created your account. Start by setting up credentials to allow devices to connect to your MQTT broker.


Credentials
OVERVIEW ACCESS MANAGEMENT INTEGRATIONS WEB CLIENT GETTING STARTED
Clusters
Billing
Help
Feedback
Logout

Set up credentials for your IoT devices
Define the credentials that your MQTT clients can use to connect to your HiveMQ Cloud cluster.
Please visit the HiveMQ documentation for examples on how to use the credentials to connect an MQTT client to your cluster.
(All fields are mandatory)
Username
At least 5 characters. Username must be unique
Password
At least 8 characters, numbers, upper- and lowercase letters.
Confirm Password

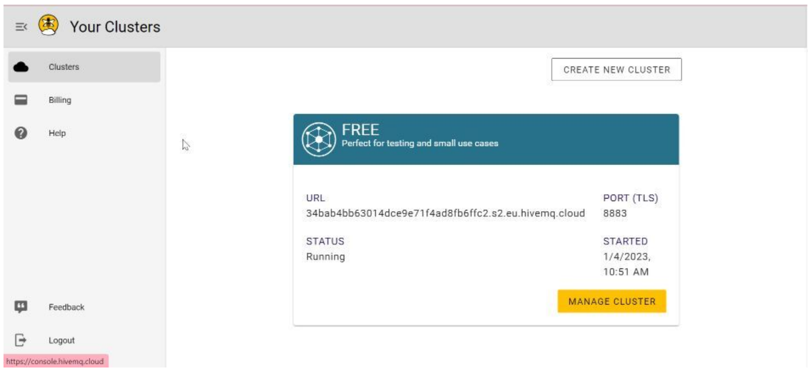
Active MQTT Credentials
These credentials allow MQTT clients to publish and subscribe to your HiveMQ Cloud cluster.

Username	Password	Actions
arduinodemo	********	DELETE

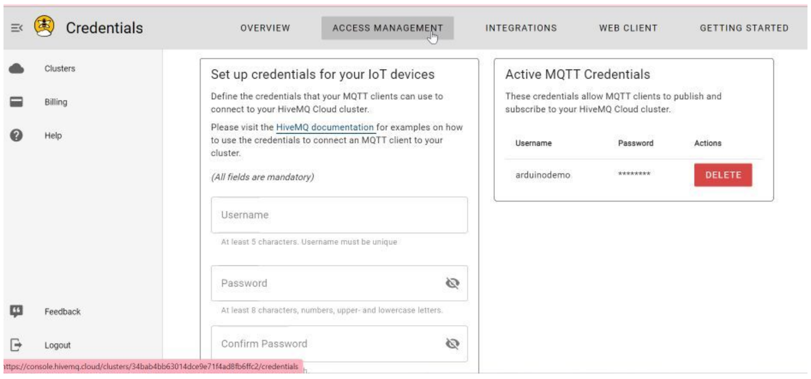
ttps://console.hivemq.cloud/clusters/34bab4bb63014dce9e71f4ad8fb6ffc2/credentials

So, I'll define the username and password and click on *ADD*.

Clicking on '*Clusters*' will provide the list of my MQTT broker clusters. Since I just created this account, I only have one. So, my MQTT broker setup is ready. Now, I need to get the cluster URL, port number, and access credentials for use in my Arduino code.


Your Clusters
Clusters
Billing
Help
Feedback
Logout
CREATE NEW CLUSTER
FREE
Perfect for testing and small use cases
URL
34bab4bb63014dce9e71f4ad8fb6ffc2.s2.eu.hivemq.cloud
PORT (TLS)
8883
STATUS
Running
STARTED
1/4/2023, 10:51 AM
MANAGE CLUSTER
https://console.hivemq.cloud

If I wish to update my connection credentials, I can always log in to my portal, click on 'MANAGE CLUSTER,' and select 'ACCESS MANAGEMENT'.


Credentials
OVERVIEW ACCESS MANAGEMENT INTEGRATIONS WEB CLIENT GETTING STARTED
Clusters
Billing
Help
Feedback
Logout

Set up credentials for your IoT devices
Define the credentials that your MQTT clients can use to connect to your HiveMQ Cloud cluster.
Please visit the HiveMQ documentation for examples on how to use the credentials to connect an MQTT client to your cluster.
(All fields are mandatory)
Username
At least 5 characters. Username must be unique
Password
At least 8 characters, numbers, upper- and lowercase letters.
Confirm Password

Active MQTT Credentials
These credentials allow MQTT clients to publish and subscribe to your HiveMQ Cloud cluster.

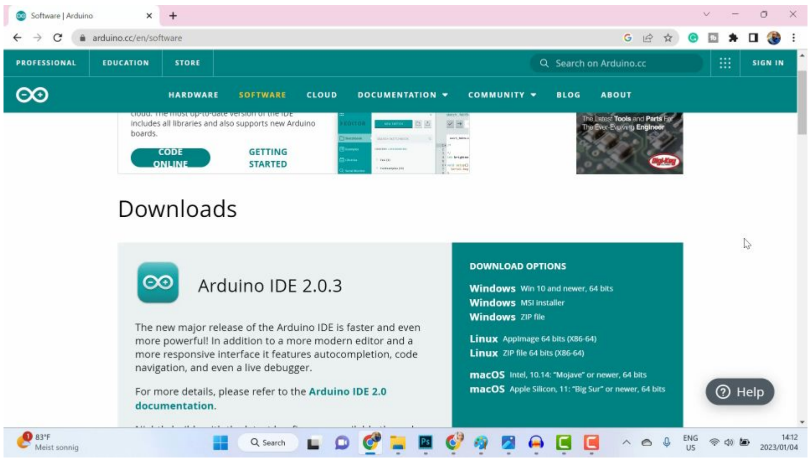
Username	Password	Actions
arduinodemo	********	DELETE

ttps://console.hivemq.cloud/clusters/34bab4bb63014dce9e71f4ad8fb6ffc2/credentials

Installing and Configuring Arduino IDE

Downloading and Installing

Next, we program the ESP8266 NodeMCU board using Arduino IDE. You can download Arduino IDE by going to the Software **downloads** page of their website.



includes all libraries and also supports new Arduino boards.

Downloads

Arduino IDE 2.0.3

The new major release of the Arduino IDE is faster and even more powerful! In addition to a more modern editor and a more responsive interface it features autocompletion, code navigation, and even a live debugger.

For more details, please refer to the **Arduino IDE 2.0 documentation**.

DOWNLOAD OPTIONS
Windows Win 10 and newer, 64 bits
Windows MSI installer
Windows ZIP file
Linux AppImage 64 bits (X86-64)
Linux ZIP file 64 bits (X86-64)
macOS Intel, 10.14: "Mojave" or newer, 64 bits
macOS Apple Silicon, 11: "Big Sur" or newer, 64 bits

For this demonstration, I'll download Arduino for Windows and install it.

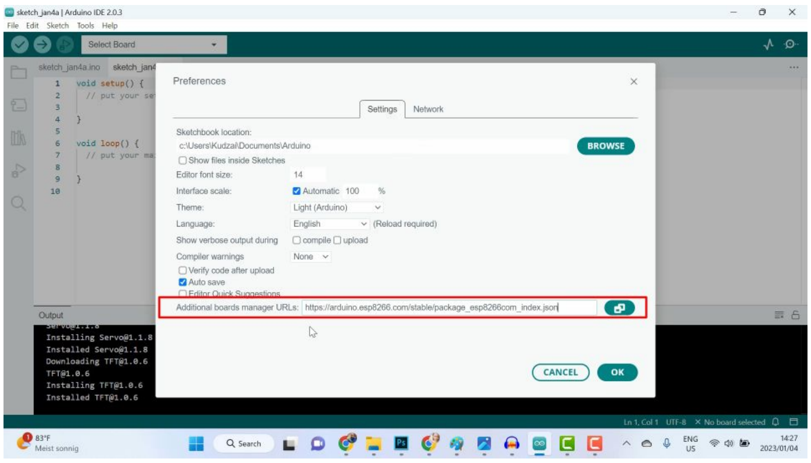
Enabling Support for the ESP8266 chip

After installing the Arduino IDE, you can program various Arduino boards. However, to program a NodeMCU board instead of an Arduino board, you will need to add a package to the IDE's board management system.

To do this, go to the '*File*' menu and select '*Preferences*.'

Then under the '*Additional Boards Manager*' field, I'll enter this URL:

https://arduino.esp8266.com/stable/package_esp8266com_index.json.



```
void setup() {
  // put your se

}

void loop() {
  // put your ma

}
```

Preferences

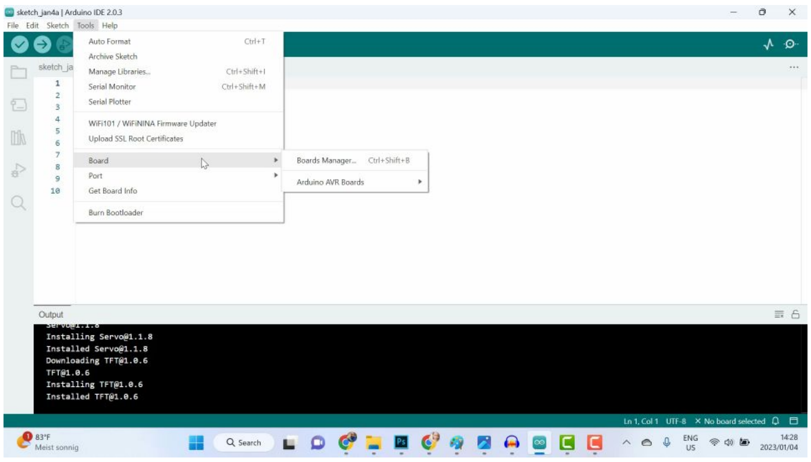
Sketchbook location: c:\Users\Kudzai\Documents\Arduino
☐ Show files inside Sketches
Editor font size: 14
Interface scale: ☑ Automatic 100 %
Theme: Light (Arduino)
Language: English (Reload required)
Show verbose output during ☐ compile ☐ upload
Compiler warnings None
☐ Verify code after upload
☑ Auto save
☐ Editor Quick Suggestions
Additional boards manager URLs: https://arduino.esp8266.com/stable/package_esp8266com_index.json

```
Installing Servo@1.1.8
Installed Servo@1.1.8
Downloading TFT@1.0.6
TFT@1.0.6
Installing TFT@1.0.6
Installed TFT@1.0.6
```

Ln 1, Col 1 UTF-8 ✕ No board selected

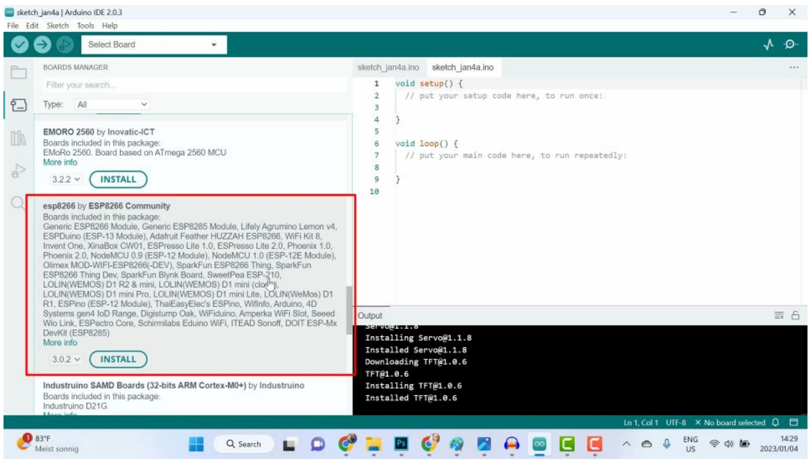
Then click OK.

Next, go to the '*Tools*' menu, and select '*Board*', followed by '*Boards Manager*'.


sketch_jan4a | Arduino IDE 2.0.3
File Edit Sketch Tools Help
Auto Format Ctrl+T
Archive Sketch
Manage Libraries... Ctrl+Shift+I
Serial Monitor Ctrl+Shift+M
Serial Plotter
WiFi101 / WiFiNINA Firmware Updater
Upload SSL Root Certificates
Board
Port
Get Board Info
Burn Bootloader
Boards Manager... Ctrl+Shift+B
Arduino AVR Boards
Output
Ln 1, Col 1 UTF-8 × No board selected

```
Installing Servo@1.1.8
Installed Servo@1.1.8
Downloading TFT@1.0.6
TFT@1.0.6
Installing TFT@1.0.6
Installed TFT@1.0.6
```

Look for ESP8266, select it, and click on the Install button.


sketch_jan4a | Arduino IDE 2.0.3
File Edit Sketch Tools Help
Select Board
BOARDS MANAGER
Filter your search...
Type: All
sketch_jan4a.ino
Output
Ln 1, Col 1 UTF-8 × No board selected

EMORO 2560 by **Inovatic-ICT**
Boards included in this package:
EMoRo 2560. Board based on ATmega 2560 MCU
More info
3.2.2 INSTALL

esp8266 by **ESP8266 Community**
Boards included in this package:
Generic ESP8266 Module, Generic ESP8285 Module, Lifely Agrumino Lemon v4, ESPDuino (ESP-13 Module), Adafruit Feather HUZZAH ESP8266, WiFi Kit 8, Invent One, XinaBox CW01, ESPresso Lite 1.0, ESPresso Lite 2.0, Phoenix 1.0, Phoenix 2.0, NodeMCU 0.9 (ESP-12 Module), NodeMCU 1.0 (ESP-12E Module), Olimex MOD-WIFI-ESP8266(-DEV), SparkFun ESP8266 Thing, SparkFun ESP8266 Thing Dev, SparkFun Blynk Board, SweetPea ESP-210, LOLIN(WEMOS) D1 R2 & mini, LOLIN(WEMOS) D1 mini (clone), LOLIN(WEMOS) D1 mini Pro, LOLIN(WEMOS) D1 mini Lite, LOLIN(WeMos) D1 R1, ESPino (ESP-12 Module), ThaiEasyElec's ESPino, WifInfo, Arduino, 4D Systems gen4 IoD Range, Digistump Oak, WiFiduino, Amperka WiFi Slot, Seeed Wio Link, ESPectro Core, Schirmilabs Eduino WiFi, ITEAD Sonoff, DOIT ESP-Mx DevKit (ESP8285)
More info
3.0.2 INSTALL

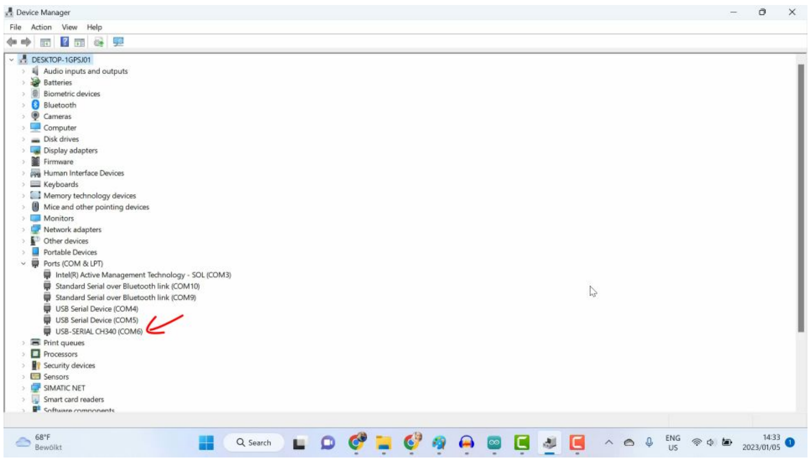
Industruino SAMD Boards (32-bits ARM Cortex-M0+) by Industruino
Boards included in this package:
Industruino D21G

```
void setup() {
  // put your setup code here, to run once:

}

void loop() {
  // put your main code here, to run repeatedly:

}
```

```
Installing Servo@1.1.8
Installed Servo@1.1.8
Downloading TFT@1.0.6
TFT@1.0.6
Installing TFT@1.0.6
Installed TFT@1.0.6
```

Connecting the NodeMCU board to the Arduino IDE via USB

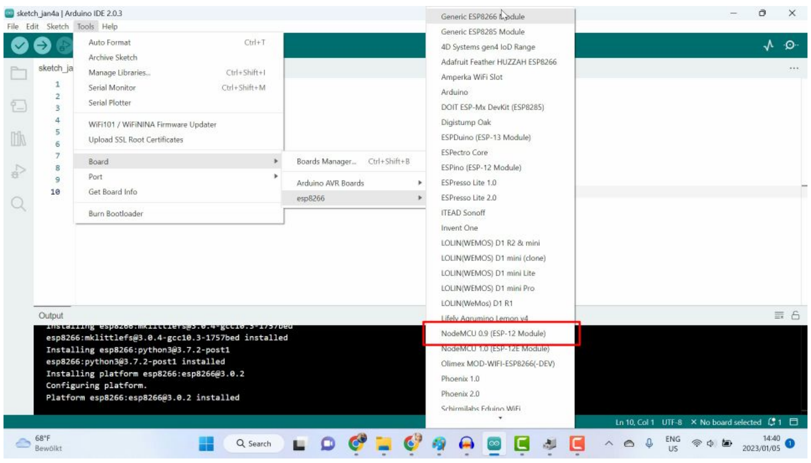
Next, I need to specify the ESP8266 board I want to program in the Arduino IDE. So, I'll connect my NodeMCU board to my PC using a micro USB cable with my Windows Device Manager open.



DESKTOP-1GPSJ01
- Audio inputs and outputs
- Batteries
- Biometric devices
- Bluetooth
- Cameras
- Computer
- Disk drives
- Display adapters
- Firmware
- Human Interface Devices
- Keyboards
- Memory technology devices
- Mice and other pointing devices
- Monitors
- Network adapters
- Other devices
- Portable Devices
- Ports (COM & LPT)
  - Intel(R) Active Management Technology - SOL (COM3)
  - Standard Serial over Bluetooth link (COM10)
  - Standard Serial over Bluetooth link (COM9)
  - USB Serial Device (COM4)
  - USB Serial Device (COM5)
  - USB-SERIAL CH340 (COM6)
- Print queues
- Processors
- Security devices
- Sensors
- SIMATIC NET
- Smart card readers
- Software components

As soon as I plug in my board, a new COM port appears on the Device Manager.

If you experience problems with your USB drivers, Windows Device Manager can help you troubleshoot. However, generally, it doesn't need to be open. You can usually access the COM port by going to the *Port* menu item under tools in the Arduino IDE.


sketch_jan4a | Arduino IDE 2.0.3
File Edit Sketch Tools Help
Auto Format Ctrl+T
Archive Sketch
Manage Libraries... Ctrl+Shift+I
Serial Monitor Ctrl+Shift+M
Serial Plotter
WiFi101 / WiFiNINA Firmware Updater
Upload SSL Root Certificates
Board
Port
Get Board Info
Burn Bootloader
Boards Manager... Ctrl+Shift+B
Arduino AVR Boards
esp8266
Generic ESP8266 Module
Generic ESP8285 Module
4D Systems gen4 IoD Range
Adafruit Feather HUZZAH ESP8266
Amperka WiFi Slot
Arduino
DOIT ESP-Mx DevKit (ESP8285)
Digistump Oak
ESPDuino (ESP-13 Module)
ESPectro Core
ESPino (ESP-12 Module)
ESPresso Lite 1.0
ESPresso Lite 2.0
ITEAD Sonoff
Invent One
LOLIN(WEMOS) D1 R2 & mini
LOLIN(WEMOS) D1 mini (clone)
LOLIN(WEMOS) D1 mini Lite
LOLIN(WEMOS) D1 mini Pro
LOLIN(WeMos) D1 R1
Lifely Agrumino Lemon v4
NodeMCU 0.9 (ESP-12 Module)
NodeMCU 1.0 (ESP-12E Module)
Olimex MOD-WIFI-ESP8266(-DEV)
Phoenix 1.0
Phoenix 2.0
Schirmilabs Eduino WiFi
Output
Ln 10, Col 1 UTF-8 No board selected
68°F Bewölkt
Search
ENG US
14:40 2023/01/05

```
esp8266:mklittlefs@3.0.4-gcc10.3-1757bed installed
Installing esp8266:python3@3.7.2-post1
esp8266:python3@3.7.2-post1 installed
Installing platform esp8266:esp8266@3.0.2
Configuring platform.
Platform esp8266:esp8266@3.0.2 installed
```

To select the board for programming, go to the '*Tools'* menu item, followed by '*Board,"* and then NodeMCU 12 module.

Now we are ready to start programming my NodeMCU using the Arduino IDE.

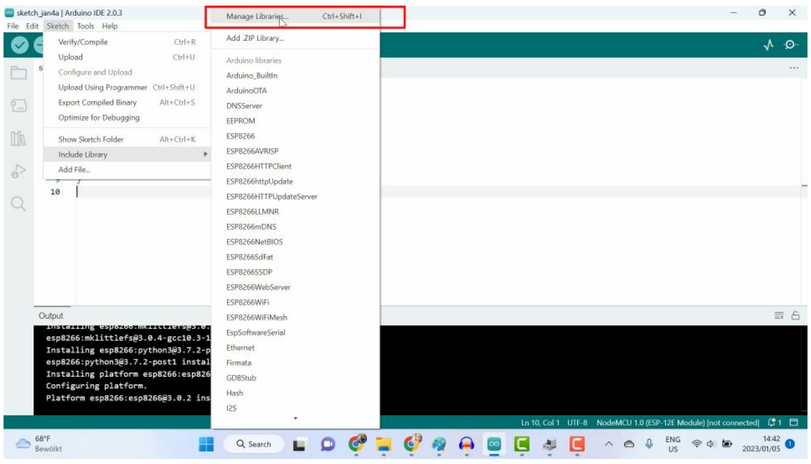
Installing Required Software Libraries

Before moving directly to coding, let's install the required libraries.

Installing DHT Sensor Library

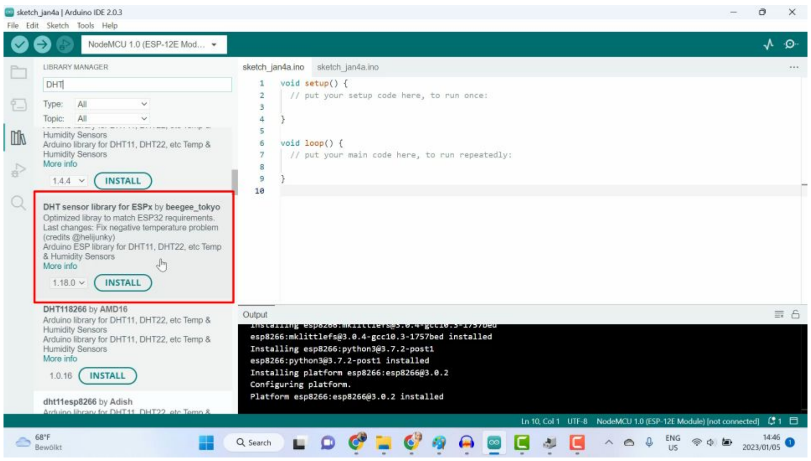
You must install the DHT sensor library available from the Arduino library manager to read sensor data.

So navigate to Sketch > Include Library > Manage Libraries.



```
esp8266:mklittlefs@3.0.4-gcc10.3-1
Installing esp8266:python3@3.7.2-p
esp8266:python3@3.7.2-post1 instal
Installing platform esp8266:esp826
Configuring platform.
Platform esp8266:esp8266@3.0.2 ins
```

Search for DHT and choose one from a number of DHT libraries.



LIBRARY MANAGER

DHT

Type: All

Topic: All

Humidity Sensors
Arduino library for DHT11, DHT22, etc Temp & Humidity Sensors
More info
1.4.4 INSTALL

DHT sensor library for ESPx by **beegee_tokyo**
Optimized libray to match ESP32 requirements.
Last changes: Fix negative temperature problem (credits @helijunky)
Arduino ESP library for DHT11, DHT22, etc Temp & Humidity Sensors
More info
1.18.0 INSTALL

DHT118266 by **AMD16**
Arduino library for DHT11, DHT22, etc Temp & Humidity Sensors
Arduino library for DHT11, DHT22, etc Temp & Humidity Sensors
More info
1.0.16 INSTALL

dht11esp8266 by **Adish**

```
void setup() {
  // put your setup code here, to run once:

}

void loop() {
  // put your main code here, to run repeatedly:

}
```

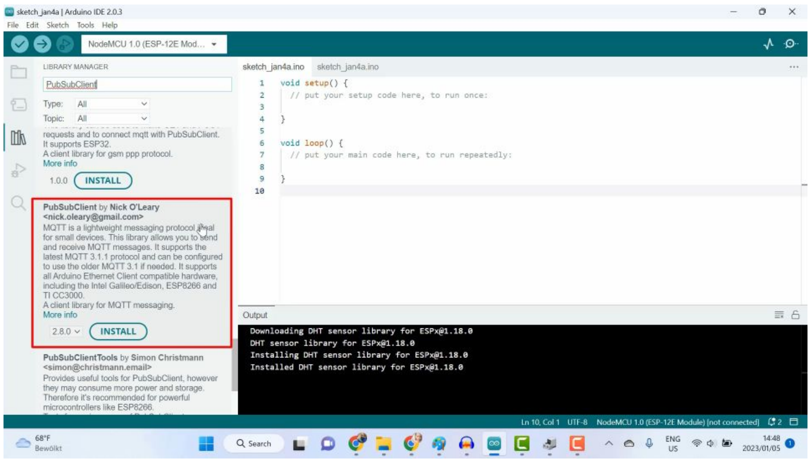
```
esp8266:mklittlefs@3.0.4-gcc10.3-1757bed installed
Installing esp8266:python3@3.7.2-post1
esp8266:python3@3.7.2-post1 installed
Installing platform esp8266:esp8266@3.0.2
Configuring platform.
Platform esp8266:esp8266@3.0.2 installed
```

Ln 10, Col 1 UTF-8 NodeMCU 1.0 (ESP-12E Module) [not connected]

sketch_jan4a | Arduino IDE 2.0.3
File Edit Sketch Tools Help
NodeMCU 1.0 (ESP-12E Mod...
sketch_jan4a.ino
Output
68°F Bewölkt
Search
ENG US
14:46 2023/01/05

Installing MQTT Client Library

We need an MQTT Client library to publish MQTT messages to my HiveMQ Cloud MQTT Broker. For this demonstration, we will use a library called PubSubClient. To add it, I'll go to Sketch > Include Library > Manage Libraries and search for PubSubClient.



LIBRARY MANAGER

PubSubClient

Type: All

Topic: All

requests and to connect mqtt with PubSubClient.
It supports ESP32.
A client library for gsm ppp protocol.
More info
1.0.0 INSTALL

PubSubClient by **Nick O'Leary <nick.oleary@gmail.com>**
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.
A client library for MQTT messaging.
More info
2.8.0 INSTALL

PubSubClientTools by **Simon Christmann <simon@christmann.email>**
Provides useful tools for PubSubClient, however they may consume more power and storage. Therefore it's recommended for powerful microcontrollers like ESP8266.

```
void setup() {
  // put your setup code here, to run once:

}

void loop() {
  // put your main code here, to run repeatedly:

}
```

```
Downloading DHT sensor library for ESPx@1.18.0
DHT sensor library for ESPx@1.18.0
Installing DHT sensor library for ESPx@1.18.0
Installed DHT sensor library for ESPx@1.18.0
```

Ln 10, Col 1 UTF-8 NodeMCU 1.0 (ESP-12E Module) [not connected]

sketch_jan4a | Arduino IDE 2.0.3
File Edit Sketch Tools Help
NodeMCU 1.0 (ESP-12E Mod...
sketch_jan4a.ino
Output
68°F Bewölkt
Search
ENG US
14:48 2023/01/05

There are many to choose from. I'll add this one from Nick O'Leary.

Programming NodeMCU to Send and Receive MQTT Messages

Now that we've installed the required libraries, let's go through the code for sending and receiving MQTT messages using NodeMCU on the Arduino platform.

First, I import the required libraries into my Arduino code.

```
#ifdef ESP8266
 #include <ESP8266WiFi.h>
 #else
 #include <WiFi.h>
#endif

#include "DHTesp.h"
#include <ArduinoJson.h>
#include <PubSubClient.h>
#include <WiFiClientSecure.h>
```

Next, declare GPIO2 as the pin to which my DHT sensor is connected. Also declare GPIO5 as the pin to which the LED is connected.

```
/**** DHT11 sensor Settings *******/
#define DHTpin 2   //Set DHT pin as GPIO2
DHTesp dht;

/**** LED Settings *******/
const int led = 5; //Set LED pin as GPIO5
```

After that, we declare variables for holding the Wi-Fi connection and HiveMQ Cloud MQTT Broker details

```
/****** WiFi Connection Details *******/
const char* ssid = "your_wifi_ssid";
const char* password = "your_wifi_password";

/******* MQTT Broker Connection Details *******/
const char* mqtt_server = "34bab4bb63014dce9e71f4ad8fb6ffc2.s2.eu.hivemq.cloud";
const char* mqtt_username = "your_mqtt_client_username";
const char* mqtt_password = "your_mqtt_client_password";
const int mqtt_port =8883;
```

Then I make sure to initialize a secure Wi-Fi client connection, create an MQTT client using PubSubClient library and attach it to the secure Wi-Fi connectivity. Also, here I declare MQTT message buffer variables.

```
/**** Secure WiFi Connectivity Initialisation *****/
WiFiClientSecure espClient;

/**** MQTT Client Initialisation Using WiFi Connection *****/
PubSubClient client(espClient);

unsigned long lastMsg = 0;
#define MSG_BUFFER_SIZE (50)
char msg[MSG_BUFFER_SIZE];
```

Now that we've set our connectivity, we can declare a variable for holding a root certificate that allows the client to connect securely to the HiveMQ Cloud MQTT broker.

```
/****** root certificate *********/

static const char *root_ca PROGMEM = R"EOF(
-----BEGIN CERTIFICATE-----
MIIFazCCA1OgAwIBAgIRAIIQz7DSQONZRGPgu2OCiwAwDQYJKoZIhvcNAQELBQAw
TzELMAkGA1UEBhMCVVMxKTAnBgNVBAoTIEludGVybmV0IFNlY3VyaXR5IFJlc2Vh
cmNoIEdyb3VwMRUwEwYDVQQDEwxJU1JHIFJvb3QgWDEwHhcNMTUwNjA0MTEwNDM4
WhcNMzUwNjA0MTEwNDM4WjBPMQswCQYDVQQGEwJVUzEpMCcGA1UEChMgSW50ZXJu
ZXQgU2VjdXJpdHkgUmVzZWFyY2ggR3JvdXAxFTATBgNVBAMTDElTUkcgUm9vdCBY
MTCCAiIwDQYJKoZIhvcNAQEBBQADggIPADCCAgoCggIBAK3oJHP0FDfzm54rVygc
h77ct984kIxuPOZXoHj3dcKi/vVqbvYATyjb3miGbESTtrFj/RQSa78f0uoxmyF+
0TM8ukj13Xnfs7j/EvEhmkvBioZxaUpmZmyPfjxwv60pIgbz5MDmgK7iS4+3mX6U
A5/TR5d8mUgjU+g4rk8Kb4Mu0UlXjIB0ttov0DiNewNwIRt18jA8+o+u3dpjq+sW
T8KOEUt+zwvo/7V3LvSye0rgTBIlDHCNAymg4VMk7BPZ7hm/ELNKjD+Jo2FR3qyH
B5T0Y3HsLuJvW5iB4YlcNHlsdu87kGJ55tukmi8mxdAQ4Q7e2RCOFvu396j3x+UC
B5iPNgiV5+I3lg02dZ77DnKxHZu8A/lJBdiB3QW0KtZB6awBdpUKD9jf1b0SHzUv
KBds0pjBqAlkd25HN7rOrFleaJ1/ctaJxQZBKT5ZPt0m9STJEadao0xAH0ahmbWn
OlFuhjuefXKnEgV4We0+UXgVCwOPjdAvBbI+e0ocS3MFEvzG6uBQE3xDk3SzynTn
jh8BCNAw1FtxNrQHusEwMFxIt4I7mKZ9YIqioymCzLq9gwQbooMDQaHWBfEbwrbw
qHyGO0aoSCqI3Haadr8faqU9GY/rOPNk3sgrDQoo//fb4hVC1CLQJ13hef4Y53CI
rU7m2Ys6xt0nUW7/vGT1M0NPAgMBAAGjQjBAMA4GA1UdDwEB/wQEAwIBBjAPBgNV
HRMBAf8EBTADAQH/MB0GA1UdDgQWBBR5tFnme7bl5AFzgAiIyBpY9umbbjANBgkq
hkiG9w0BAQsFAAOCAgEAVR9YqbyyqFDQDLHYGmkgJykIrGF1XIpu+ILlaS/V9lZL
ubhzEFnTIZd+50xx+7LSYK05qAvqFyFWhfFQDlnrzuBZ6brJFe+GnY+EgPbk6ZGQ
3BebYhtF8GaV0nxvwuo77x/Py9auJ/GpsMiu/X1+mvoiBOv/2X/qkSsisRcOj/KK
NFtY2PwByVS5uCbMiogziUwthDyC3+6WVwW6LLv3xLfHTjuCvjHIInNzktHCgKQ5
ORAzI4JMPJ+GslWYHb4phowim57iaztXOoJwTdwJx4nLCgdNbOhdjsnvzqvHu7Ur
TkXWStAmzOVyyghqpZXjFaH3pO3JLF+l+/+sKAIuvtd7u+Nxe5AW0wdeRlN8NwdC
jNPElpzVmbUq4JUagEiuTDkHzsxHpFKVK7q4+63SM1N95R1NbdWhscdCb+ZAJzVc
oyi3B43njTOQ5yOf+1CceWxG1bQVs5ZufpsMljq4Ui0/1lvh+wjChP4kqKOJ2qxq
4RgqsahDYVvTH9w7jXbyLeiNdd8XM2w9U/t7y0Ff/9yi0GE44Za4rF2LN9d11TPA
mRGunUHBcnWEvgJBQl9nJEiU0Zsnvgc/ubhPgXRR4Xq37Z0j4r7g1SgEEzwxA57d
emyPxgcYxn/eR44/KJ4EBs+lVDR3veyJm+kXQ99b21/+jh5Xos1AnX5iItreGCc=
-----END CERTIFICATE-----
)EOF";
```

Then declare the function that we will use to connect to the WiFi network.

```
/************* Connect to WiFi ***********/
void setup_wifi() {
  delay(10);
  Serial.print("\nConnecting to ");
  Serial.println(ssid);

  WiFi.mode(WIFI_STA);
  WiFi.begin(ssid, password);

  while (WiFi.status() != WL_CONNECTED) {
    delay(500);
    Serial.print(".");
  }
  randomSeed(micros());
  Serial.println("\nWiFi connected\nIP address: ");
  Serial.println(WiFi.localIP());
}
```

Next, we set a function for esp8266 connecting to the HiveMQ Cloud MQTT broker.

```
/************* Connect to MQTT Broker ***********/
void reconnect() {
  // Loop until we're reconnected
  while (!client.connected()) {
    Serial.print("Attempting MQTT connection...");
    String clientId = "ESP8266Client-";   // Create a random client ID
    clientId += String(random(0xffff), HEX);
    // Attempt to connect
    if (client.connect(clientId.c_str(), mqtt_username, mqtt_password)) {
      Serial.println("connected");

      client.subscribe("led_state");   // subscribe the topics here

    } else {
      Serial.print("failed, rc=");
      Serial.print(client.state());
      Serial.println(" try again in 5 seconds");   // Wait 5 seconds before retrying
      delay(5000);
    }
  }
}
```

Then let's add a callback method for receiving MQTT messages and switching the LED ON and OFF.

```
/***** Call back Method for Receiving MQTT messages and Switching LED ****/

void callback(char* topic, byte* payload, unsigned int length) {
  String incommingMessage = "";
  for (int i = 0; i < length; i++) incommingMessage+=(char)payload[i];

  Serial.println("Message arrived ["+String(topic)+"]"+incommingMessage);

  //--- check the incomming message
    if( strcmp(topic,"led_state") == 0){
      if (incommingMessage.equals("1")) digitalWrite(led, HIGH);   // Turn the LED on
    if (incommingMessage.equals("1")) digitalWrite(led, HIGH);   // Turn the LED on
    else digitalWrite(led, LOW);  // Turn the LED off
  }

}
```

After that, I implement a method for publishing MQTT messages to my HiveMQ Cloud MQTT broker.

```
/**** Method for Publishing MQTT Messages **********/
void publishMessage(const char* topic, String payload , boolean retained){
  if (client.publish(topic, payload.c_str(), true))
      Serial.println("Message publised ["+String(topic)+"]: "+payload);
}
```

We move to the default Arduino setup method, where we set up the DHT 11 sensor and LED, enable the certificate for a secure connection, setup up Wi-Fi, and create instances of the MQTT server and callback methods.

```
/**** Application Initialisation Function******/
void setup() {

  dht.setup(DHTpin, DHTesp::DHT11); //Set up DHT11 sensor
  pinMode(led, OUTPUT); //set up LED
  Serial.begin(9600);
  while (!Serial) delay(1);
  setup_wifi();

  #ifdef ESP8266
    espClient.setInsecure();
  #else
    espClient.setCACert(root_ca);      // enable this line and the the "certificate" code for secure c
  #endif

  client.setServer(mqtt_server, mqtt_port);
  client.setCallback(callback);
}
```

Finally, we are ready to add the main function for executing the application logic. If you are not connected, connect the MQTT client to the HiveMQ Cloud MQTT broker. Then read the temperature and humidity values from the DHT11 sensor, and serialize them, including ***deviceId*** and ***siteId*** properties, into a JSON object.

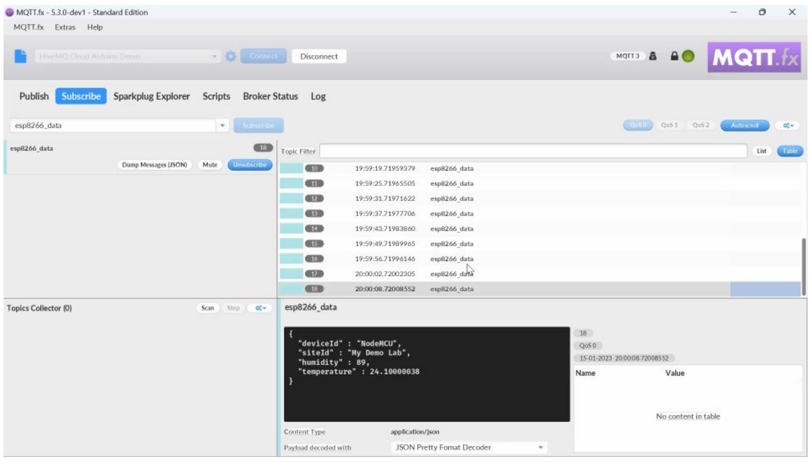
After serialization, I publish the JSON string to my HiveMQ Cloud MQTT Broker under the topic “esp8266_data”, and repeat the process every 5 seconds.

```
/******** Main Function *************/
void loop() {

  if (!client.connected()) reconnect(); // check if client is connected
  client.loop();

//read DHT11 temperature and humidity reading
  delay(dht.getMinimumSamplingPeriod());
  float humidity = dht.getHumidity();
  float temperature = dht.getTemperature();

  DynamicJsonDocument doc(1024);

  doc["deviceId"] = "NodeMCU";
  doc["siteId"] = "My Demo Lab";
  doc["humidity"] = humidity;
  doc["temperature"] = temperature;
  char mqtt_message[128];
  serializeJson(doc, mqtt_message);

  publishMessage("esp8266_data", mqtt_message, true);

  delay(5000);

}
```

Next, deploy this code onto the microcontroller board.

Now that my code is running and I'm publishing data to the HiveMQ Cloud MQTT Broker, I can use an MQTT Client application such as MQTTfx to subscribe to the broker and view the data.

When I open my MQTT.fx application, which is already connected to the HiveMQ Cloud MQTT broker and subscribed to the same topic, I can see the data coming.


MQTT.fx - 5.3.0-dev1 - Standard Edition
MQTT.fx Extras Help
Connect Disconnect
MQTT 3
MQTT.fx
Publish Subscribe Sparkplug Explorer Scripts Broker Status Log
Subscribe
QoS 0 QoS 1 QoS 2 Autoscroll
Dump Messages (JSON) Mute Unsubscribe
Topic Filter List Table
Topics Collector (0) Scan Stop

esp8266_data 18

```
10  19:59:19.71959379  esp8266_data
11  19:59:25.71965505  esp8266_data
12  19:59:31.71971622  esp8266_data
13  19:59:37.71977706  esp8266_data
14  19:59:43.71983860  esp8266_data
15  19:59:49.71989965  esp8266_data
16  19:59:56.71996146  esp8266_data
17  20:00:02.72002305  esp8266_data
18  20:00:08.72008552  esp8266_data
```

esp8266_data

```
{
  "deviceId" : "NodeMCU",
  "siteId" : "My Demo Lab",
  "humidity" : 89,
  "temperature" : 24.10000038
}
```

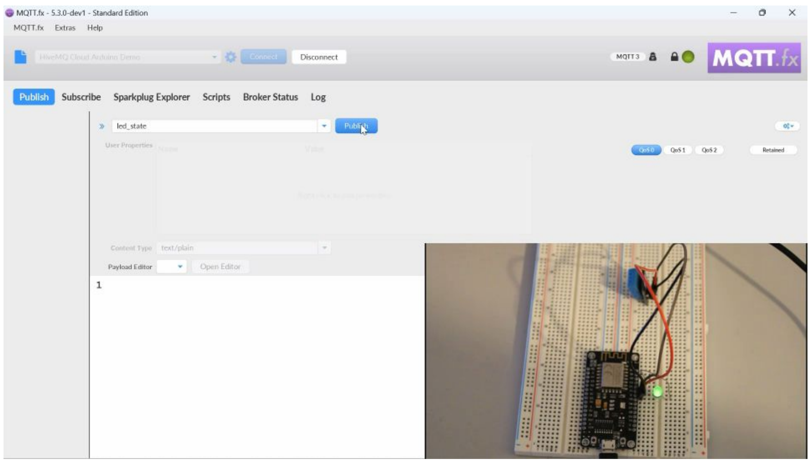
18
QoS 0
15-01-2023 20:00:08.72008552

Name	Value
No content in table	

Content Type: application/json
Payload decoded with: JSON Pretty Fomat Decoder

Seeing data flowing means you are successfully publishing data from the NodeMCU using MQTT.

Now, I will publish an MQTT command to switch on the LED light from MQTT.fx by publishing the value of "1" under the topic "led_state". When I do that, you can see the LED light switch ON.


MQTT.fx - 5.3.0-dev1 - Standard Edition
MQTT.fx Extras Help
Connect Disconnect
MQTT 3
MQTT.fx
Publish Subscribe Sparkplug Explorer Scripts Broker Status Log
Publish
QoS 0 QoS 1 QoS 2 Retained
User Properties
Open Editor

Topic: led_state
Content Type: text/plain
Payload Editor

```
1
```

To switch the light off, publish a value of "0".

Conclusion

Together, we have experienced using the Arduino IDE to program an ESP8266 microcontroller to send and receive MQTT messages from a HiveMQ Cloud MQTT broker. How was the process for you? Can you see other use cases?

Try out the above demonstration and get started with building MQTT applications by signing up for a free HiveMQ Cloud account.

Sign-up for HiveMQ Cloud Now

About Kudzai Manditereza

Kudzai is a Developer Advocate at HiveMQ and the Founder of **Industry40.tv**. He is the host of an IIoT Podcast and is involved in Industry4.0 research and educational efforts.

HiveMQ Community Edition 2023.1 is released

Insights from ABB Automotive Manufacturing Survey 2022

Product	Cloud	Developers	MQTT	Solutions	Blog	Company	Try HiveMQ
HiveMQ	HiveMQ Cloud	Resources	Overview	Connected-Cars	MQTT Client Tools	About us	Download HiveMQ
Features	Sign up	Getting Started	Glossary	Automotive	MQTT Introduction	Logos & Media Kit	Docker
Editions		Documentation HiveMQ	MQTT 5 Essentials	Manufacturing	Get started with MQTT	News	AWS
HiveMQ Swarm			MQTT Essentials	Transportation		Events	