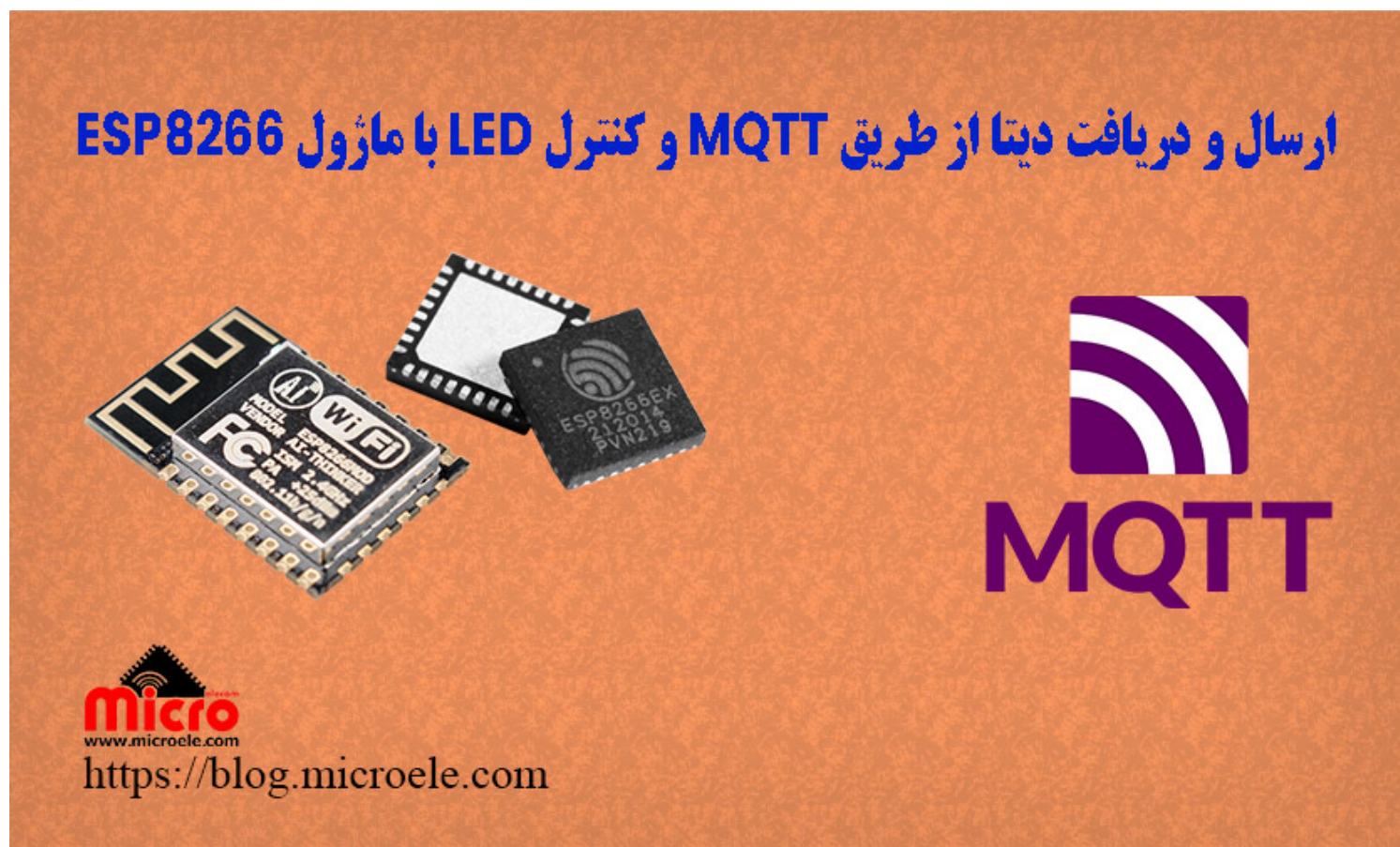




ارسال و دریافت دیتا از طریق MQTT و کنترل LED با ماژول ESP8266



تاریخ انتشار ۱۴ تیر، ۱۴۰۲ توسط سید حسین سلطانی

سلام و درود خدمت همراهان همیشگی مایکروالکام. در مطالب قبلی از سری مطالب مربوط به سری **آموزش‌های ESP8266** به معرفی **سریال مانیتور تحت وب** پرداخته شد. در این مطلب به بررسی کامل ارسال و دریافت دیتا/پیام با استفاده از پروتکل MQTT و بروکر رایگان HiveMQ و همچنین کنترل GPIO ماژول ESP8266 پرداخته خواهد شد. پس با من تا انتهای مطلب همراه باشید. همچنین شما میتونید سایر مطالب من رو از **این قسمت** دنبال کنید.



مقدمه

در مطالب قبلی به معرفی و موارد استفاده پروتکل MQTT پرداخته شد. در این مطلب بصورت کامل نحوه استفاده از آن جهت تبادل دیتا (ارسال و دریافت) بررسی شده است. می‌توان با ارتقا و ویرایش سورس کد این آموزش آن را شخصی سازی کرده و متناسب با پروژه خود از آن استفاده کنید. در این مطلب علاوه بر بحث ارسال و دریافت دیتا، یک بروکر (MQTT Broker) رایگان که استفاده از آن نیز راحت می‌باشد معرفی گردیده است. همچنین علاوه بر بررسی تبادل دیتا در ESP8266، از یک نرم افزار کلاینت با قابلیت نصب روی ویندوز استفاده شده است که مضاف بر رصد آنلاین دیتا در سایت اصلی، می‌توان از آن جهت ارسال و دریافت نیز استفاده کرد.

بروکر MQTT Broker

سایت و ارائه کنندگانی در سراسر اینترنت وجود دارند که خدماتی که شامل پروتکل MQTT است را ارائه می‌دهند. لذا طریقه کلی استفاده از آنها تقریباً مشابه و دارای روش مشابهی می‌باشند. برای این منظور از بروکر **HiveMQ** استفاده شده است. اولین قدم برای استفاده از آن ایجاد یک اکانت بوده تا بتوانیم از Cloud آن استفاده کنیم.

ثبت اکانت در بروکر HiveMQ

در ابتدا وارد سایت **HiveMQ** شده و از بالا سمت راست گزینه Get Started را بزنید. در صفحه جدید تعرفه و قیمت های استفاده از آن را نمایش می‌دهد که گزینه Serverless که رایگان است را انتخاب کنید. در این قسمت گزینه Sign up free را انتخاب و اکانت خود را ایجاد نمایید.



Pricing

We have the right solution for you

Fully-Managed

HiveMQ Cloud

Self-Managed

HiveMQ

HiveMQ Cloud

Fully-managed, **unlimited connections**, growing with you

Serverless

Free

[Sign Up Free](#)

No credit card required

A basic MQTT broker designed for learning and experimenting with MQTT.

A great place to play

- Multi-tenant MQTT broker
- 100 connections free
- No uptime SLA
- Community supported

RECOMMENDED

Starter

Starting from **\$0.34/hour***
+ \$0.80/million messages

[Try with Free Credits](#)

Sign up and get \$100 in credits

Complete MQTT platform for testing and small-scale production.

A great place to start

- Single-tenant MQTT platform
- No fees for connections
- 99.95% uptime with Multi-AZ
- Up to 24/7 Support

Professional

Starting from **\$0.57/hour***
+ \$0.80/million messages

[Join the Waitlist](#)

Production-ready, complete MQTT platform for scalable workloads.

Everything in Starter, plus

- Deep observability
- Advanced management
- Enhanced security
- 24/7 Support available

Enterprise

Custom pricing

[Contact Sales](#)

Fully customizable MQTT platform for sophisticated workloads.


Everything in Professional, plus

- Dedicated infrastructure
- Custom systems integration
- Any region
- Dedicated CSM with 24/7 Support

بروکر MQTT HiveMQ

پس از ورود به اکانت خود از بخش Cluster با کلیک روی New Cluster یک خوشه جدید ایجاد کنید. در بخش جدید باز شده مجدداً از بخش Serverless، گزینه Get Start را انتخاب کنید. مشابه تصویر زیر AWS را انتخاب و گزینه Creat را کلیک می‌کنیم.



☰  Create cluster

Data

- ☁ Clusters

Billing

- 📄 Billing & Payment

← Configure your HiveMQ Cloud plan

Select your cloud service provider
Cloud selection is only available in our Professional Plan

COMING SOON

Your Selection


Create

Serverless

For learning and experimenting with MQTT. Basic MQTT broker.

A great place to play

- ✓ Multi-tenant MQTT broker
- ✓ 100 connections free
- ✓ No uptime SLA
- ✓ Community supported

 وبلاگ میکروالکام

NEW What's new

- 🔍 Help
- 📄 Documentation
- 💬 Feedback
- 🚪 Logout

ایجاد Cluster در بروکر MQTT

حال به منوی Cluster هدایت شده یا می‌توانید با انتخاب آن خوشه‌هایی که ایجاد کرده‌اید را مشاهده نمایید. با انتخاب Manage Cluster وارد جزئیات آن شده، با انتخاب گزینه Access Management یک username و password را تعریف کرده تا



بتوانیم توسط ESP8266 یا میکروکنترلر خود به آن متصل شده و تبادل دیتا نماییم. با انتخاب Web Client می‌توانیم به اکانت دسترسی داشته و با یک topic جلسه یا Session برقرار کرده و دیتا ارسال یا دیتای دریافتی را مشاهده نماییم.

The screenshot shows the 'Cluster Details' page in the Micro Cloud interface. It features a sidebar with navigation options like Clusters, Billing, and Billing & Payment. The main content area is divided into several sections: 'MQTT Client Sessions' showing 0 / 100, 'Data Traffic' showing 0 B / 10 GB, 'Connection Settings' with fields for Cluster URL, Port, and Websocket Port, and 'Cluster Information' showing the Cloud Provider as Amazon Web Services and the Current Plan as Free. There are two red callouts: one pointing to the 'Host Name' field in the Connection Settings section, and another pointing to the 'MQTT Client Sessions' section.

برنامه نویسی جهت اتصال به بروکر MQTT با ESP8266

سورس کد این بخش جهت استفاده و برنامه نویسی ماژول ESP8266 می‌باشد. برای ماژول ESP32 نیز مشابه بوده و نیاز به ویرایش خیلی جزئی چند قسمت کد ها دارد. در ابتدا نیاز است که کتابخانه PubSubClient برای اتصال و تبادل دیتا از طریق پروتکل MQTT نصب شود. برای دانلود و نصب آن از [این لینک](#) استفاده کنید.

نکته: برای ارتباط با سرور به چند مورد اصلی نیاز است که در ادامه به آن اشاره شده است (این موارد در بخش Overview اکانت خود قابل مشاهده است). همچنین دقت شود که نوع برقراری ارتباط با سرور بر بستر TLS بوده و باید ارتباط برقرار شده ماژول نیز ایمن باشد. برای این منظور از کتابخانه WiFiClientSecure استفاده شده و Certificate مورد نیاز سرور نیز در سورس کد آمده است.



- Server: همان آدرس یکتایی که سرور به شما اختصاص داده است.
- username: نام کاربری معرفی شده در Access Management
- password: رمز عبور معرفی شده در Access Management
- شماره پورت: شماره پورت TLS

معرفی کردن کتابخانه و متغیر های لازم

در ابتدای کد ها کتابخانه های مورد نیاز را معرفی کرده و از آنجا که اتصال به بروکر MQTT مبتنی بر پروتکل TLS می باشد، بایستی Certificate آن نیز در کد ها درج شده باشد. برای همین از متغیر PROGMEM *root_ca استفاده شده است. همچنین از کتابخانه WiFiClientSecure نیز به همین منظور استفاده شده است.

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

const int led = 2;
const char* ssid = "SSID";
const char* password = "Password";

const char* mqtt_server = "*****.eu.es.cloud";
const char* mqtt_username = "username";
const char* mqtt_password = "password";
const int mqtt_port = 8883;

WiFiClientSecure espClient;
PubSubClient client(espClient);

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



```
//root certificate
static const char *root_ca PROGMEM = R"EOF(
                                     -----BEGIN CERTIFICATE-----
MIIFazCCA10gAwIBAgIRAIIQz7DSQ0NZRGPgu20CiwAwDQYJKoZIhvcNAQELBQAw
TzELMAkGA1UEBhMCVVMxKTAnBgNVBAoTIEludGVybmV0IFNlY3VyaXR5IFJlc2Vh
cmNoIEdyb3VwMRUwEwYDVQQDEwxFb3VyaXR5IFNlY3VyaXR5IFJlc2VhcmNoIE
WhcNMzUwNjA0MTEwNDM4WjBPMQswCQYDVQQGEwJVUzEpMCCGA1UEChMwSW50ZXJ
ZXQgU2VjdXJpdHkgUmVzZWZyY2ggR3JvdXAFTATBgNVBAMTDElUkcgUm9vdCBY
MTCCAiIwDQYJKoZIhvcNAQEBBQADggIPADCCAggIBAK3oJHP0FDfzm54rVygch
h77ct984kIxuP0ZXoHj3dcKi/vVqbvYATyjb3miGbESTtrFj/RQSa78f0uoxmyF+
0TM8ukj13Xnfs7j/EvEhmkvBioZxaUpmZmyPfxwv60pIgbz5MDmgK7iS4+3mX6U
A5/TR5d8mUgjU+g4rk8Kb4Mu0ULXjIB0ttov0DiNewNwIRt18jA8+o+u3dpjq+sW
T8K0EUt+zwvo/7V3LvSye0rgTBIldHCNAymg4VMk7BPZ7hm/ELNKjD+Jo2FR3qyH
B5T0Y3HsLuJvW5iB4YlcnHlsdu87kGJ55tukmi8mxdAQ4Q7e2RC0Fvu396j3x+UC
B5iPNgiV5+I3lg02dZ77DnKxHZu8A/LJBdiB3QW0KtZB6awBdpUKD9jf1b0SHzUv
KBds0pjbQalkd25HN7r0rFleaJ1/ctaJxQZBKT5ZPt0m9STJEadao0xAH0ahmbWn
0lFuhjuefXKnEgV4We0+UXgVCw0PjdAvBbI+e0ocS3MFEVzG6uBQE3xDk3SzynTn
jh8BCNAw1FtxNrQHusEwMFxIt4I7mKZ9YIqioymCzLq9gwQbooMDQaHwBfEbwrwb
qHyG00aoSCqI3Haadr8faqu9GY/r0PNk3sgrDQoo//fb4hVC1CLQJ13hef4Y53CI
rU7m2Ys6xt0nUW7/vGT1M0NPAGMBAAGjQjBAMA4GA1UdDwEB/wQEAwIBBjAPBgNV
HRMBAf8EBTADAQH/MB0GA1UdDgQWBBR5tFnme7bl5AFzGaiIyBpY9umbbjANBgkq
hkiG9w0BAQsFAA0CAgEAVR9YqbyyqFDQDLHYGmkGjYkIrGF1XIpU+ILlaS/V9lZL
ubhzEFnTIZd+50xx+7LSYK05qAvqFyFWhfFQDlnrzubZ6brJFe+GnY+EgPbk6ZGQ
3BebYhtF8GaV0nxvwo77x/Py9auJ/GpsMiu/X1+mvoiB0v/2X/qkSsisRc0j/KK
NFtY2PwByVS5uCbMiogziUwthDyC3+6WVwW6LLv3xLfHTjuCvjHIInNzktHCgKQ5
0RAzI4JMPJ+GslWYHb4phowim57iaztx0oJwTdwJx4nLCgdNb0hdjsnvzqvHu7Ur
TkXWStAmz0VyyghqpZXjFaH3p03JLF+l+/+sKAIuvtd7u+Nxe5AW0wdeRlN8NwdC
jNPElpzVmbUq4JUagEiuTDkHszxHpFKVK7q4+63SM1N95R1NbdWhscdCb+ZAJzVc
oyi3B43njT0Q5y0f+1CceWxG1bQV5SzufpsMlj4Ui0/1lvh+wjChP4kqK0J2qxq
4RgqsahDYVvTH9w7jXbyLeiNdd8XM2w9U/t7y0Ff/9yi0GE44Za4rF2LN9d11TPA
mRGunUHBcnWEvgJBQl9nJEiU0Zsnvgc/ubhPgXRR4Xq37Z0j4r7g1SgEEzwxA57d
emyPxgcYxn/eR44/KJ4EBs+LVDR3veyJm+kXQ99b21/+jh5Xos1AnX5iItreGCc=
```



```
-----END CERTIFICATE-----  
)EOF";
```

اتصال ESP8266 به مودم

برای برقراری ارتباط با بروکر MQTT نیاز به اتصال به مودم و اینترنت داریم. در بالا در متغیرهای ssid و password نام و پسورد مودم یا روتر خود را جایگزین کنید. از تابع زیر جهت اتصال به مودم استفاده شده است.

```
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());  
}
```

تابع اتصال ESP8266 به بروکر MQTT

پس از اتصال به اینترنت، باید به سرور یا بروکر MQTT خود متصل شویم. برای این کار نیاز به چند پارامتر اصلی داریم که در ابتدای کد ها معرفی شده است. این موارد شامل آدرس سرور، پورت، یوزرنیم و پسورد می باشد. در کد زیر بجای Client_ID می توانیم یک نام دلخواه وارد نماییم.

```
void reconnect() {  
    while (!client.connected()) {
```




```
MQTT connection...");  
  
Serial.print("Attempting  
  
// Attempt to connect  
if  
(client.connect("Client_ID", mqtt_username, mqtt_password)) {  
Serial.println("connected");  
client.subscribe("test"); //subscribe to topic  
}  
else {  
Serial.print("failed, rc=");  
Serial.print(client.state()); //MQTT connection status  
Serial.println(" try again in 3 seconds"); //Wait 3 sec. before retrying  
delay(5000);  
}  
}  
}
```

دریافت دیتا از بروکر MQTT

علاوه بر ارسال دیتا به سرور از طریق پروتکل MQTT باید پیام های دریافتی که ارسال می شود را نیز بتوانیم دریافت کنیم. به همین منظور پس از اصطلاحا شرکت کردن در یک topic یا برقراری یک Session، پیام های آن را بوسیله تابع زیر دریافت خواهیم کرد. می توانیم با نوشتن یک شرط ساده، یک LED را روشن و خاموش کنیم. در شرط if کد زیر، از تابع strcmp استفاده شده است. چنانچه topic معرفی شده در آرگومان تابع هنگام فراخوانی با topic مورد نظر خود که در اینجا test نام دارد برابر بود، شرط اجرا خواهد شد.

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



```
}  
  
message = incomingMessage;  
Serial.println("Incoming Message: " + message);  
if(strcmp(topic, "test") == 0){  
  if(message == "on"){  
    Serial.println("LED is ON");  
    digitalWrite(led, LOW);  
  }  
  else if(message == "off"){  
    Serial.println("LED is OFF");  
    digitalWrite(led, HIGH);  
  }  
}  
}
```

ارسال دیتا به بروکر MQTT با ESP8266

برای ارسال دیتا ابتدا باید در topic مورد نظر عضو شده و پس از آن دیتای خود را ارسال نماییم.

```
void publishMessage(const char* topic, String payload , boolean retained){  
  if (client.publish(topic, payload.c_str(), true))  
    Serial.println("Message publised [" + String(topic) + "]: " + payload);  
  else  
    Serial.println("Unable to publishing!");  
}
```

با استفاده از کد زیر می‌توانیم در صورت نیاز خود در برنامه، پیام مد نظر خود را ارسال کنیم. به یاد داشته باشید بجای Your_topic، همان topic که قصد ارسال پیام به آن را دارید عینا وارد کنید. و بجای mqtt_message نیز پیام/دیتا دلخواه خود را وارد نمایید.



```
publishMessage("Your_topic", "mqtt_message", true);
```

تابع setup

در این تابع تنظیمات اولیه جهت راه اندازی، تنظیمات سریال، اتصال به مودم، فراهم کردن اتصال به بروکر MQTT و مواردی از این قبیل را انجام خواهیم داد.

```
void setup() {  
    pinMode(led, OUTPUT);  
    digitalWrite(2, HIGH);  
    Serial.begin(115200);  
    setup_wifi();  
    #ifdef ESP8266  
        espClient.setInsecure();  
    #else  
        espClient.setCACert(root_ca); // enable this line and  
the "certificate" code for secure connection  
    #endif  
    client.setServer(mqtt_server, mqtt_port); //setting server  
and port  
    client.setCallback(callback);  
}
```

تابع loop

در این تابع یک شرط قرار داده شده است که در صورت عدم اتصال یا قطع ارتباط با بروکر، مجدداً تلاش به برقراری ارتباط می‌کند. می‌توانیم با افزودن delay مدت زمان آن را محدود نماییم.

```
void loop() {  
    if (!client.connected()) reconnect(); // check if client is not
```



```
connected, try again
    client.loop();
    //delay(1000);
}
```

سورس کد کامل

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

const int led = 2;
const char* ssid = "SSID";
const char* password = "Password";

const char* mqtt_server = "*****.eu.es.cloud";
const char* mqtt_username = "username";
const char* mqtt_password = "password";
const int mqtt_port = 8883;

WiFiClientSecure espClient;
PubSubClient client(espClient);

unsigned long lastMsg = 0;
#define MSG_BUFFER_SIZE (50) // تعیین مقدار بافر برای پیام
char msg[MSG_BUFFER_SIZE];

//root certificate
static const char *root_ca PROGMEM = R"EOF(
                                     -----BEGIN CERTIFICATE-----
MIIFazCCA10gAwIBAgIRAIQz7DSQONZRGPGu20CiwAwDQYJKoZIhvcNAQELBQAw
```



TzELMAkGA1UEBhMCVVMxKTAnBgNVBAoTIEludGVybmV0IFNlY3VyaXR5IFJlc2Vh
cmNoIEdyb3VwMRUwEwYDVQQDEwXJU1JHIFJvb3QgWDEwHhcNMTUwNjA0MTEwNDM4
WhcNMzUwNjA0MTEwNDM4WjBPMQswCQYDVQQGEwJVUzEpMCCGA1UEChMgSW50ZXJw
ZXQgU2VjdXJpdHkgUmVzZWZyY2ggR3JvdXAxFtATBgNVBAMTDElUkcgUm9vdCBY
MTCCAiIwDQYJKoZIhvcNAQEBBQADggIPADCCAgoCggIBAK3oJHP0FDfzm54rVygc
h77ct984kIxuPOZxHj3dcKi/vVqbvYATyjb3miGbESTtrFj/RQSa78f0uoxmyF+
0TM8ukj13Xnfs7j/EvEhmkvBioZxaUpmZmyPfxwv60pIgbz5MDmgK7iS4+3mX6U
A5/TR5d8mUgUj+g4rk8Kb4Mu0ULXjIB0ttov0DiNewNwIRt18jA8+o+u3dpjq+sW
T8K0EUt+zwvo/7V3LvSye0rgTBIldHCNAymg4VMk7BPZ7hm/ELNKjD+Jo2FR3qyH
B5T0Y3HsLuJvW5iB4YlcnHlsdu87kGJ55tukmi8mxdAQ4Q7e2RC0Fvu396j3x+UC
B5iPNgiV5+I3lg02dZ77DnKxHZu8A/LJBdiB3QW0KtZB6awBdpUKD9j f1b0SHzUv
KBds0pjBqAlkd25HN7r0rFleaJ1/ctaJxQZBKT5ZPt0m9STJEadao0xAH0ahmbWn
0lFuhjuefXKnEgV4We0+UXgVCw0PjdAvBbI+e0ocS3MFEvzG6uBQE3xDk3Szyntn
jh8BCNAw1FtxNrQHusEwMFxIt4I7mKZ9YIqiOymCzLq9gwQbooMDQaHwBfEbwrwb
qHyG00aoSCqI3Haadr8faqu9GY/r0PNk3sgrDQoo//fb4hVC1CLQJ13hef4Y53CI
rU7m2Ys6xt0nUw7/vGT1M0NPAGMBAAGjQjBAMA4GA1UdDwEB/wQEAwIBBjAPBgNV
HRMBAf8EBTADAQH/MB0GA1UdDgQWBRR5tFnme7bl5AFzgAiIyBpY9umbbjANBgkq
hkiG9w0BAQsFAA0CAgEAVR9YqbyyqFDQDLHYGmkGjYkIRGF1XIPu+ILlaS/V9lZL
ubhzEFnTIZd+50xx+7LSYK05qAvqFyFWhfFQDlnrzuBZ6brJFe+GnY+EgPbk6ZGQ
3BebYhtF8GaV0nxvwo77x/Py9auJ/GpsMiu/X1+mvoiB0v/2X/qkSsisRc0j/KK
NFtY2PwByVS5uCbMiogziUwthDyC3+6WVwW6LLv3xLfHTjuCvjHIIInNzktHCgKQ5
ORazI4JMPJ+GslWYHb4phowim57iaztx0oJwTdwJx4nLCgdNb0hdj snvzqvHu7Ur
TkXWStAmz0VyyghqpZXjFaH3p03JLF+l+/+sKAIuvtd7u+Nxe5AW0wdeRlN8NwdC
jNPElpzVmbUq4JUagEiuTDkHszxHpFKVK7q4+63SM1N95R1NbdWhscdCb+ZAJzVc
oyi3B43njT0Q5y0f+1CceWxG1bQVs5ZufpsMlj4Ui0/1lvh+wjChP4kqK0J2qxq
4RgqsahDYVvTH9w7jXbyLeiNdd8XM2w9U/t7y0Ff/9yi0GE44Za4rF2LN9d11TPA
mRGunUHBcnWEvgJBQl9nJEiU0Zsnvgc/ubhPgXRR4Xq37Z0j4r7g1SgEEzwxA57d
emyPxcYxn/eR44/KJ4EBs+lVDR3veyJm+kXQ99b21/+jh5Xos1AnX5iI treGCc=

-----END CERTIFICATE-----

)EOF";

//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());
}

//Connect to MQTT Broker
void reconnect() {
    while (!client.connected()) {
        Serial.print("Attempting
MQTT connection...");

        // Attempt to connect
        if
(client.connect("Client_ID", mqtt_username, mqtt_password)) {
            Serial.println("connected");
            client.subscribe("test"); //subscribe to topic
        }
        else {
            Serial.print("failed, rc=");
            Serial.print(client.state()); //MQTT connection status
            Serial.println(" try again in 3 seconds"); //Wait 3 sec. before retrying
            delay(5000);
        }
    }
}
```



```
    }  
}  
  
//Receiving MQTT messages  
void callback(char* topic, byte* payload, unsigned int length) {  
    String  
    incommingMessage = "";  
    for (int i  
    = 0; i < length; i++){  
        incommingMessage += (char)payload[i];  
    }  
    String  
    message = incommingMessage;  
    Serial.println("Incomming Message: " + message);  
    if(strcmp(topic, "test") == 0){  
        if(message == "on"){  
            Serial.println("LED is ON");  
            digitalWrite(led, LOW);  
        }  
        else if(message == "off"){  
            Serial.println("LED is OFF");  
            digitalWrite(led, HIGH);  
        }  
    }  
}  
  
//Publishing MQTT Message  
void publishMessage(const char* topic, String payload , boolean retained){  
    if (client.publish(topic, payload.c_str(), true))  
        Serial.println("Message publised [" + String(topic) + "]: " + payload);  
    else  
        Serial.println("Unable to publishing!");  
}
```



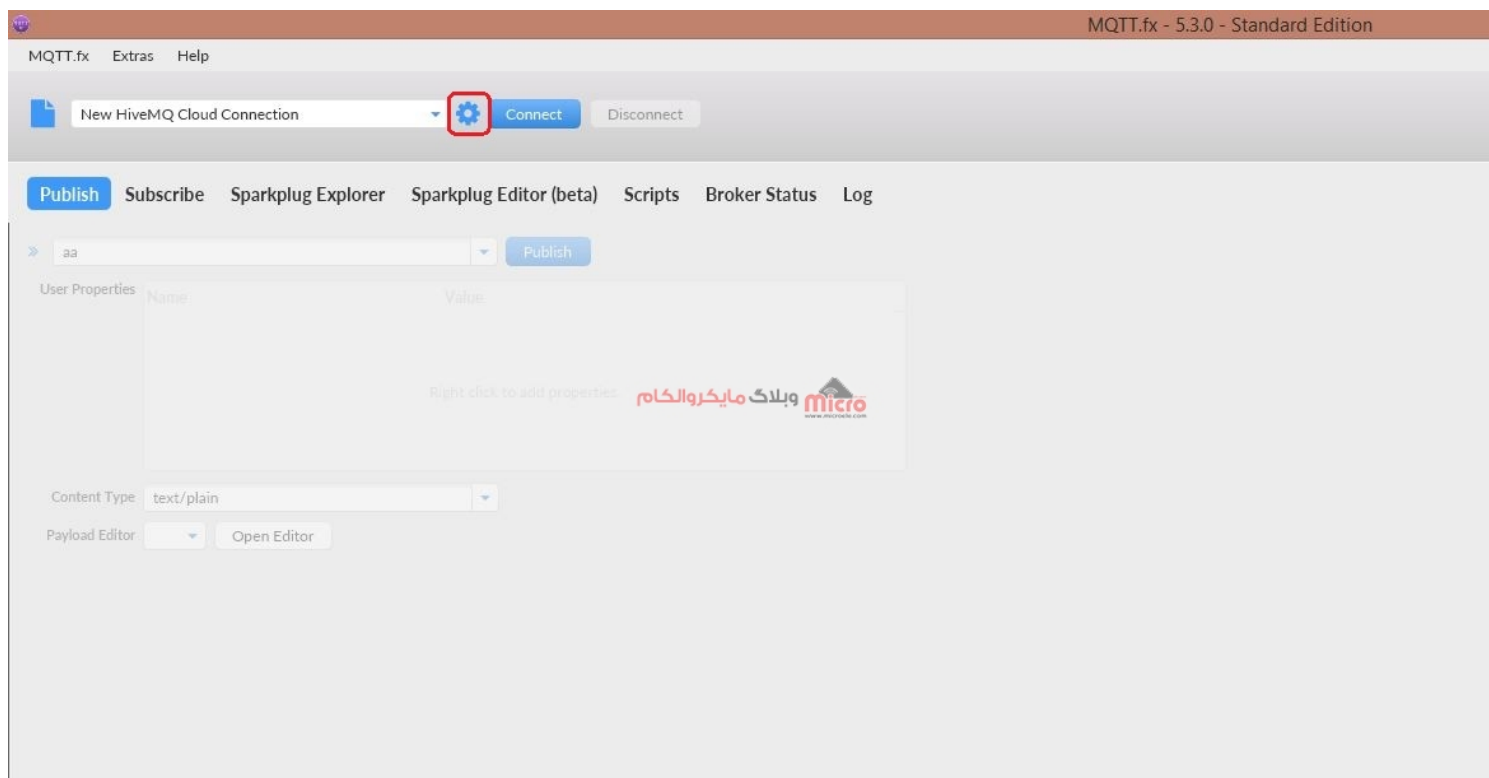
```
    }  
  
void setup() {  
    pinMode(led, OUTPUT);  
    digitalWrite(2, HIGH);  
    Serial.begin(115200);  
    setup_wifi();  
    #ifdef ESP8266  
        espClient.setInsecure();  
    #else  
        espClient.setCACert(root_ca); // enable this line and  
the "certificate" code for secure connection  
    #endif  
    client.setServer(mqtt_server, mqtt_port); //setting server  
and port  
    client.setCallback(callback);  
}  
  
void loop() {  
    if (!client.connected()) reconnect(); // check if client is not  
connected, try again  
    client.loop();  
    //delay(1000);  
}
```

کلاینت MQTT برای ویندوز

می‌توانیم از طریق نرم افزار MQTT.Fx نیز از طریق ویندوز به بروکر متصل شده و تبادل دیتا نماییم. برای دانلود این نرم افزار از [این لینک](#) اقدام نمایید. دقت کنید پس از نصب نیاز به لایسنس دارید برای درخواست آن نیز از [این لینک](#)



اقدام کنید. پس از چند دقیقه فایل مربوطه ایمیل شده و آن را در notepad باز کرده و کامل کپی کرده و در پنجره باز شده هنگام باز کردن نرم افزار دقیقا paste نمایید. پس از وارد شدن به نرم افزار بر روی چرخ دنده مشخص شده در تصویر زیر کلیک کنید. در پنجره باز شده مطابق با تصویر دوم اطلاعات خواسته شده را وارد کنید. نهایتا بر روی OK کلیک کرده و به صفحه اصلی باز خواهید گشت. اینبار بر روی دکمه Connect که در کنار آیکن چرخ دنده است کلیک کرده و منتظر بمانید تا به بروکر متصل شوید.



نرم افزار MQTT.Fx



The screenshot shows the 'Edit Connection Profiles' dialog box in a software application. The title bar reads 'Edit Connection Profiles' and 'Edition'. The main content area is titled 'HiveMQ Cloud Profile Settings' and features the HiveMQ Cloud logo. The following fields are visible:

- Profile Name: New HiveMQ Cloud Connection
- Profile Type: HiveMQ Cloud
- BrokerAddress: آدرس سرور خود
- Client ID: MQTT_FX_Client (with a 'Generate' button)
- Username: در اکانت خود username
- Password: در اکانت خود password
- MQTT Version: Use Default (MQTT v3.1) with a dropdown menu set to 3.1 (highlighted with a red box)

At the bottom of the dialog, there are buttons for 'Revert', 'Cancel', 'OK', and 'Apply'.

تنظیمات نرم افزار MQTT.FX

پس از اتصال موفقیت آمیز، از قسمت subscribe می‌توانیم در مورد نظر عضو شده و دیتا های دریافتی را مشاهده نماییم. پس از وارد کردن نام تاپیک بصورت صحیح، بروی subscribe کلیک کنید. پس از موفقیت آمیز بودن همه موارد، تصویر نرم افزار همانند تصویر زیر خواهد شد.



The screenshot shows the MQTT.fx web client interface. At the top, there are navigation buttons: Publish, Subscribe, Sparkplug Explorer, Sparkplug Editor (beta), Scripts, Broker Status, and Log. Below this, a search bar contains the topic 'test' with a 'Subscribe' button. The main area is divided into two sections. The top section shows a message received on the 'test' topic at 21:26:05.77165843. The bottom section shows the message details for 'mqtt_message', which is a retained message with a QoS of 0, received on 04-07-2023 at 21:26:05.77165843. The message content is in Persian: 'آخرین پیامی که در سرور دریافت شده است یا پیام های دریافتی در سرور در این قسمت نمایش داده می شود.' The content type is 'text/plain' and the payload is decoded with 'Plain Text Decoder'.

اتصال به یک topic در MQTT.Fx

در منوی Publish نیز می توانیم به topic مورد نظر، دیتا ارسال نماییم. در تصویر زیر در محل مشخص شده، topic مد نظر را وارد و سپس روی گزینه Publish کلیک نمایید. پس از آن می توانید در محلی که در تصویر زیر با کادر قرمز مشخص شده، پیام خود را وارد کرده و نهایتاً دکمه publish را زده تا دیتا ارسال گردد. می توانیم در بروکر یا سرور خود، پیام های دریافتی را در قسمت Cluster و زیر قسمت Web Client مشاهده نماییم.



ارسال (Publish) دیتا به تاپیک مورد نظر توسط MQTT.Fx

نتیجه گیری

در این مطلب به معرفی و نحوه ایجاد یک Session در بروکر MQTT پرداخته شد. همچنین روش تبادل دیتا و ارسال و دریافت از طریق پروتکل MQTT جهت کنترل پایه های GPIO ماژول ESP8266 بررسی شد. با استفاده از کد هایی که در این مطلب قرار دارد براحتی می توانیم متناسب با پروژه IoT خود آن ها را ویرایش و متناسب با پروژه و کاربرد خود آن را در بستر اینترنت اشیا پیاده سازی کنیم.

امیدوارم از این آموزش کمال بهره را برده باشید. در صورتی که هرگونه نظر یا سوال داشتید درباره این آموزش لطفاً اون رو در انتهای همین صفحه در قسمت دیدگاه ها قرار بدید. در کوتاه ترین زمان ممکن به اون ها پاسخ خواهم داد. اگر این مطلب براتون مفید بود، اون رو حتماً به اشتراک بگذارید. همینطور میتونید این آموزش را توی اینستاگرام با هشتگ #microelecom به اشتراک بگذارید و **پیج مایکروالکام** (@microelecom) رو هم منشن کنید.