



IOT201 - LẬP TRÌNH IOT CƠ BẢN

BÀI 7.1 LẬP TRÌNH GIAO THỨC KẾT NỐI MQTT(P1)

- ⊙ Kết thúc bài học này, sinh viên có khả năng
 - ⊙ Lập trình giao thức kết nối MQTT



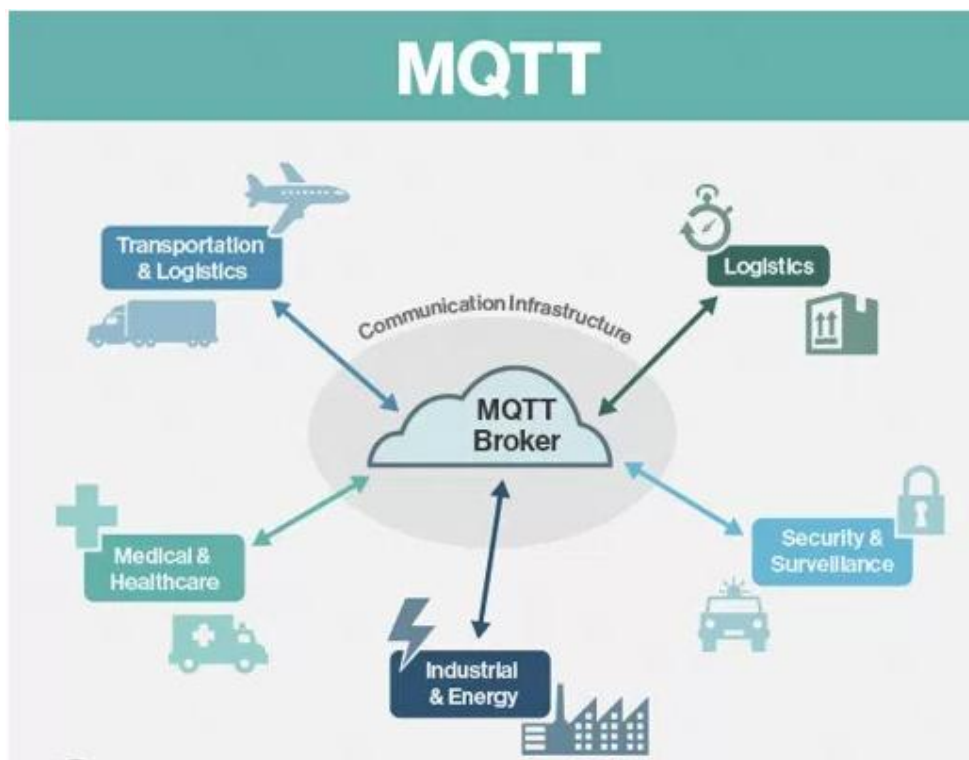
- 📖 Giới thiệu về giao thức MQTT
- 📖 Lập trình về giao thức MQTT



- ❑ MQTT (Message Queuing Telemetry Transport) giao thức gửi dạng publish/subscribe sử dụng cho các thiết bị Internet of Things với băng thông thấp, độ tin cậy cao và khả năng được sử dụng trong mạng lưới không ổn định.

- ❑ Vì giao thức sử dụng băng thông thấp trong môi trường có độ trễ cao nên nó là một giao thức lý tưởng cho các ứng dụng [M2M](#).
- ❑ MQTT cũng là giao thức sử dụng trong [Facebook Messenger](#)

- ❑ MQTT Broker là một trạm trung chuyển, một trung tâm thông tin, có nhiệm vụ nhận thông tin từ rất nhiều các thiết bị khác nhau và chuyển các thông tin này tới từng thiết bị riêng biệt.



□ Chuẩn bị

- ❖ ESP8266 (NodeMCU)
- ❖ Arduino IDE
- ❖ Thư viện Pubsubclient
- ❖ Cách kết nối cũng như dùng Arduino cho ESP8266

❑ Thực hiện

- ❖ Dùng ESP8266 (NodeMCU) làm Client để kết nối lên một dịch vụ MQTT Broker là [CloudMQTT](#) với 2 nhiệm vụ chính.
- ❖ **Nhiệm vụ 1:** Gửi (publish) dữ liệu lên broker, và nhận thông tin (subscribe) từ broker, mục đích chính là kiểm tra publish và subscribe giữa ESP8266 và MQTT Broker.
- ❖ **Nhiệm vụ 2:** Nhận(subscribe) thông tin từ broker, kiểm tra dữ liệu và thực hiện bật tắt LED

❑ Tạo tài khoản và cấu hình CloudMQTT



- ❑ **Bước 1:** Vào trang <https://www.cloudmqtt.com/> để tạo một tài khoản.
- ❑ **Bước 2:** Đăng ký tài khoản

The screenshot shows the registration process on the CloudMQTT website. At the top, a progress bar indicates four steps: Plan (active), Region, Configure (Dedicated plans only), and Confirm. Below the progress bar, the heading "Select a plan and name - Step 1 of 4" is displayed. The form contains three input fields: "Name" with the value "diennt", "Plan" with a dropdown menu showing "Cute Cat (Free)", and "Tags" with the value "iot; webservice". Below the "Tags" field, there is explanatory text: "Tags are used to separate your instances between projects. This is primarily used in the project listing view for easier navigation and access control." and "Tags allow admins to manage team members access to different groups of instances." To the right of the form, there is a "Plan" section featuring a cute cat illustration and the text "Cute Cat". Below this, a link says "See the plan page to learn about the different plans." At the bottom right, there are two buttons: "Cancel" and "Select Region".

Plan

Region

Configure
(Dedicated plans only)

Confirm

Select a plan and name - Step 1 of 4

Name: diennt

Plan: Cute Cat (Free)

Tags: iot; webservice

Tags are used to separate your instances between projects. This is primarily used in the project listing view for easier navigation and access control.

Tags allow admins to manage team members access to different groups of instances.

Plan

Cute Cat

See the plan page to learn about the different plans.

Cancel Select Region

❑ Bước 3: Chọn vùng

No credit card Please [add a credit card](#) if you want to subscribe to a paid plan

Plan

Region

Configure
(Dedicated plans only)

Confirm

Select a region and data center - Step 2 of 4

Data center EU-West-1 (Ireland) ▼


aws

« Back


Cancel Confirm

❑ Bước 4: Xác thực thông tin


Confirm new instance - Step 4 of 4

Plan	
	
Cute Cat	
Total: \$0/month	
Name:	diennt
Provider:	Amazon Web Services
Region:	EU-West-1 (Ireland)
Tags:	lot; webservice

❑ Bước 5: Hoàn tất thông tin

 CloudMQTT

List all instances ▾

 diennt@fpt.edu.vn ▾

Instances

lot; webservice

✓ Instance successfully created

Name	Plan	Datacenter	Actions
diennt	Cat	Amazon Web Services EU-West-1 (Ireland)	<div>Edit</div>

- ❑ **Bước 6:** Trong **User & ACL** điền tên user là esp8266, password là 123456, sau đó **Add**

DETAILS

SETTINGS

CERTIFICATES

USERS & ACL

BRIDGES


AMAZON KINESIS STREAM

WEBSOCKET UI

CONNECTIONS

LOG

Users and ACL

 API Access

Users

Name

- ❑ **Bước 7:** Trong New Rule chọn user như hình, tên topic đặt demo, tick chọn Read, Write

ACLs

Note:

- There are two types of ACL rules, topic and pattern. Topic ACLs is applied to a given user. Pattern ACLs is applied to all users.
- Use # for multi level wildcard acl.
- Use + for single level wildcard acl.
- Creating and deleting users and ACLs are asynchronous tasks and may take up to a minute. Poll list APIs to see when ready.

For API docs look at HTTP API

Type	Pattern		Read/Write	
<div>PatternTopic</div>	<div>esp8266</div>	<div>demo</div>	<input checked="" type="checkbox"/> Read Access?	<div>+ Add</div>
			<input checked="" type="checkbox"/> Write Access?	

❑ Bước 8: Chú ý thông tin quan trọng

DETAILS

SETTINGS

CERTIFICATES

USERS & ACL

BRIDGES

AMAZON KINESIS STREAM

WEBSOCKET UI

CONNECTIONS

LOG

Details

Instance info

Server

m21.cloudmqtt.com

User

eyfjrueu

Restart

Password

ievLvfyimNC

Rotate

Port

11767

SSL Port

21767


Websockets Port (TLS only)

31767

Connection limit

5

Active Plan



Cute Cat

Upgrade Instance



LẬP TRÌNH IOT CƠ BẢN

BÀI 7.2 LẬP TRÌNH GIAO THỨC KẾT NỐI MQTT(P2)

- ❑ Lập trình ESP8266
- ❑ Kiểm tra chức năng subscribe và publish
 - ❖ Để ESP8266 có thể publish và subscribe dữ liệu lên MQTT broker thì cần phải có thư viện MQTT

❑ Code tham khảo

```
1  #include <ESP8266WiFi.h>
2  #include <PubSubClient.h>
3  // Cập nhật thông tin
4  // Thông tin về wifi
5  #define ssid "ten_wifi"
6  #define password "password"
7  // Thông tin về MQTT Broker
8  #define mqtt_server "m12.cloudmqtt.com" // Thay bằng thông tin của bạn
9  #define mqtt_topic_pub "demo" //Giữ nguyên nếu bạn tạo topic tên là dem
10 #define mqtt_topic_sub "demo"
11 #define mqtt_user "esp8266" //Giữ nguyên nếu bạn tạo user là esp8266 v
12 #define mqtt_pwd "123456"
13
14 const uint16_t mqtt_port = 10769; //Port của CloudMQTT
```

```
15
16  WiFiClient espClient;
17  PubSubClient client(espClient);
18
19  long lastMsg = 0;
20  char msg[50];
21  int value = 0;
22
23  void setup() {
24    Serial.begin(115200);
25    setup_wifi();
26    client.setServer(mqtt_server, mqtt_port);
27    client.setCallback(callback);
28  }
29  // Hàm kết nối wifi
30  void setup_wifi() {
```

```
31     delay(10);
32     Serial.println();
33     Serial.print("Connecting to ");
34     Serial.println(ssid);
35     WiFi.begin(ssid, password);
36     while (WiFi.status() != WL_CONNECTED) {
37         delay(500);
38         Serial.print(".");
39     }
40     Serial.println("");
41     Serial.println("WiFi connected");
42     Serial.println("IP address: ");
43     Serial.println(WiFi.localIP());
44 }
```

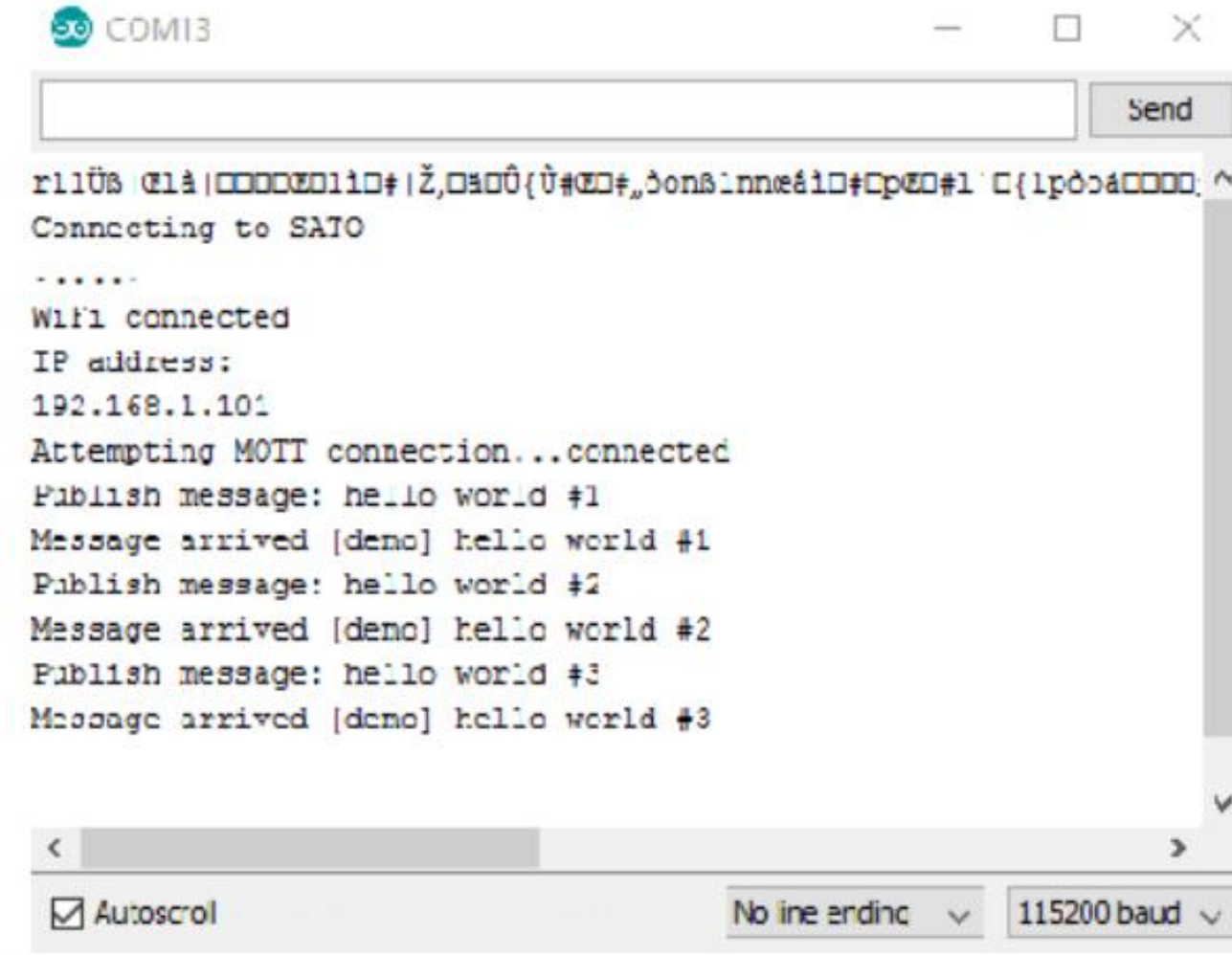
```
45 // Hàm call back để nhận dữ liệu
46 void callback(char* topic, byte* payload, unsigned int length) {
47     Serial.print("Message arrived [");
48     Serial.print(topic);
49     Serial.print("] ");
50     for (int i = 0; i < length; i++) {
51         Serial.print((char)payload[i]);
52     }
53     Serial.println();
54 }
55 // Hàm reconnect thực hiện kết nối lại khi mất kết nối với MQTT Broker
56 void reconnect() {
57     // Chờ tới khi kết nối
58     while (!client.connected()) {
59         Serial.print("Attempting MQTT connection...");
60         // Thực hiện kết nối với mqtt user và pass
61         if (client.connect("ESP8266Client",mqtt_user, mqtt_pwd)) {
```

```
62         Serial.println("connected");
63         // Khi kết nối sẽ publish thông báo
64         client.publish(mqtt_topic_pub, "ESP_reconnected");
65         // ... và nhận lại thông tin này
66         client.subscribe(mqtt_topic_sub);
67     } else {
68         Serial.print("failed, rc=");
69         Serial.print(client.state());
70         Serial.println(" try again in 5 seconds");
71         // Đợi 5s
72         delay(5000);
73     }
74 }
75 }
```

```
76 void loop() {
77     // Kiểm tra kết nối
78     if (!client.connected()) {
79         reconnect();
80     }
81     client.loop();
82     // Sau mỗi 2s sẽ thực hiện publish dòng hello world lên MQTT broker
83     long now = millis();
84     if (now - lastMsg > 2000) {
85         lastMsg = now;
86         ++value;
87         snprintf (msg, 75, "hello world #%ld", value);
88         Serial.print("Publish message: ");
89         Serial.println(msg);
90         client.publish(mqtt_topic_pub, msg);
91     }
92 }
```


- ❑ Chương trình này sẽ thực hiện kết nối với mạng wifi trước để có kết nối tới internet.
- ❑ Sau đó thực hiện gửi thông tin lên broker bằng lệnh publish, sau đó lắng nghe thông tin có trên Broker và in thông tin này ra terminal của Arduino thông qua hàm callback.

❑ Kết quả



```

rllÜS Qlâ | 000000110# | Ž, 0300 { 0# 00 # „ 0onB1nnæá10# 0p00#1 0 { 1p00á0000: ^
Connecting to SATO
.....
Wifi connected
IP address:
192.168.1.101
Attempting MQTT connection...connected
Publish message: hello world #1
Message arrived [demo] hello world #1
Publish message: hello world #2
Message arrived [demo] hello world #2
Publish message: hello world #3
Message arrived [demo] hello world #3
  
```

☒ Autoscroll
 No line ending ▾
115200 baud ▾

☑ Lập trình giao thức kết nối MQTT.





Cảm ơn