Smart Farmer-IOT Enabled Smart Farming Application

DEVELOP A PYTHON CODE

TEAM ID: PNT2022TMID38090

Connect to the MQTT broker

This article will use the free public MQTT broker provided by EMQX. This service is based on MQTT cloud service - EMQX Cloud to create.

The accessing information of the broker is as follows:

- Broker: **broker.emqx.io**
- TCP Port: **1883**
- Websocket Port: **8083**

Import the Paho MQTT client

```
from paho.mqtt import client as mqtt_client
```

Set the parameter of the MQTT Broker connection

Set the address, port and topic of the MQTT Broker connection. At the same time, we call the Python function `random.randint` to randomly generate the MQTT client id.

```
broker = 'broker.emqx.io' port = 1883 topic

= "python/mqtt" client_id = f'python-mqtt-

{random.randint(0,

1000)}'
# username = 'emqx'

# password = 'public'
```

Write the MQTT connect function

Write the connect callback function on_connect. This function will be called after connecting the client, and we can determine whether the client is connected successfully according to rc in this function. Usually, we will create an MQTT client at the same time and this client will connect to broker.emqx.io.

```
def connect  mqtt():      def

on  connect(client, userdata, flags, rc):
```

```
        if rc == 0:

            print("Connected to MQTT Broker!")

        else:

            print("Failed to connect, return code %d\n",
rc)

    # Set Connecting Client ID     client =
mqttclient.Client(client_id)
client.username_pw_set(username, password)
client.on_connect = on_connect
client.connect(broker, port)
```

```
    return client
```

Publish messages

First, we define a while loop. In this loop, and we will set the MQTT client publish function to send messages to the topic python/mqtt every second.

```
 def publish(client):


msg  count = 0       while


True:


        time.sleep(1)          msg =
f"messages: {msg  count}"
```

```
        result = client.publish(topic, msg)

        # result: [0, 1]

        status = result[0]

        if status == 0:

            print(f"Send `{msg}` to topic `{topic}`")

        else:

            print(f"Failed to send message to topic
{topic}")

        msg_count += 1
```

Subscribe

Write the message callback function on message. This function will be called after the client receives messages from the MQTT Broker. In this function, we will print out the name of subscribed topics and the received messages.

```
def subscribe(client: mqtt client):      def

on   message(client, user data, msg):

       print (f "Received ,{msg .payload .decode()}`

from `{msg. topic}` topic")



   client.subscribe(topic)

   client.on message = on message
```

The full code

The code of publishing messages

```
# python 3.6



import random

import time



from paho.mqtt import client as mqtt_client
```

```
broker =

'broker.emqx.io' port =

1883 topic =

"python/mqtt"

# generate client ID with pub prefix randomly

client_id = f'python-mqtt-{random.randint(0,

1000)}'

# username = 'emqx'
```

```
# password = 'public'
```

```
def connect_mqtt():     def

on_connect(client, userdata, flags, rc):

if rc == 0:

        print("Connected to MQTT Broker!")

    else:

        print("Failed to connect, return code %d\n",

rc)



   client          =          mqtt_client.Client(client_id)
client.username_pw_set(username, password)
```

```
client.on_connect = on_connect
```

```
    client.connect(broker, port)

return client




def publish(client):

msg_count = 0     while

True:

        time.sleep(1)         msg =
f"messages: {msg_count}"
```

```
        result = client.publish(topic, msg)
        # result: [0, 1]
status = result[0]
if status == 0:
            print(f"Send `{msg}` to topic `{topic}`")
        else:
            print(f"Failed to send message to topic
{topic}")
msg_count += 1
```

```
def run():

    client = connect_mqtt()

    client.loop_start()

    publish(client)




if __name__ == '__main__':

    run()
```

The code of subscribing

```
# python3.6
```

```
import random



from paho.mqtt import client as mqtt_client




broker = 'broker.emqx.io'
port = 1883 topic =
"python/mqtt"



# generate client ID with pub prefix randomly
```

```
client_id = f'python-mqtt-{random.randint(0, 100)}'

# username = 'emqx'

# password = 'public'




def connect_mqtt() -> mqtt_client:      def

on_connect(client, userdata, flags, rc):

if rc == 0:

            print("Connected to MQTT Broker!")
```

```
        else:
            print("Failed to connect, return code %d\n",
rc)


    client = mqtt_client.Client(client_id)
client.username_pw_set(username, password)
client.on_connect = on_connect
client.connect(broker, port)     return client
```

```
def subscribe(client: mqtt_client):
    def on_message(client, userdata, msg):
        print(f"Received `{msg.payload.decode()}`
from `{msg.topic}` topic")

    client.subscribe(topic)
client.on_message = on_message
```

```
def run():

    client = connect  mqtt()

subscribe(client)

client.loop  forever()




if __name___== '__main__':

    run()
```