

# Develop the Python Script

(Publish data to IBM cloud)

Date	04 November 2022
Team ID	PNT2022TMIDO5194
Project Name	Industry-specific intelligent fire management system
Maximum Marks	4 Marks

## Industry-specific intelligent fire management system



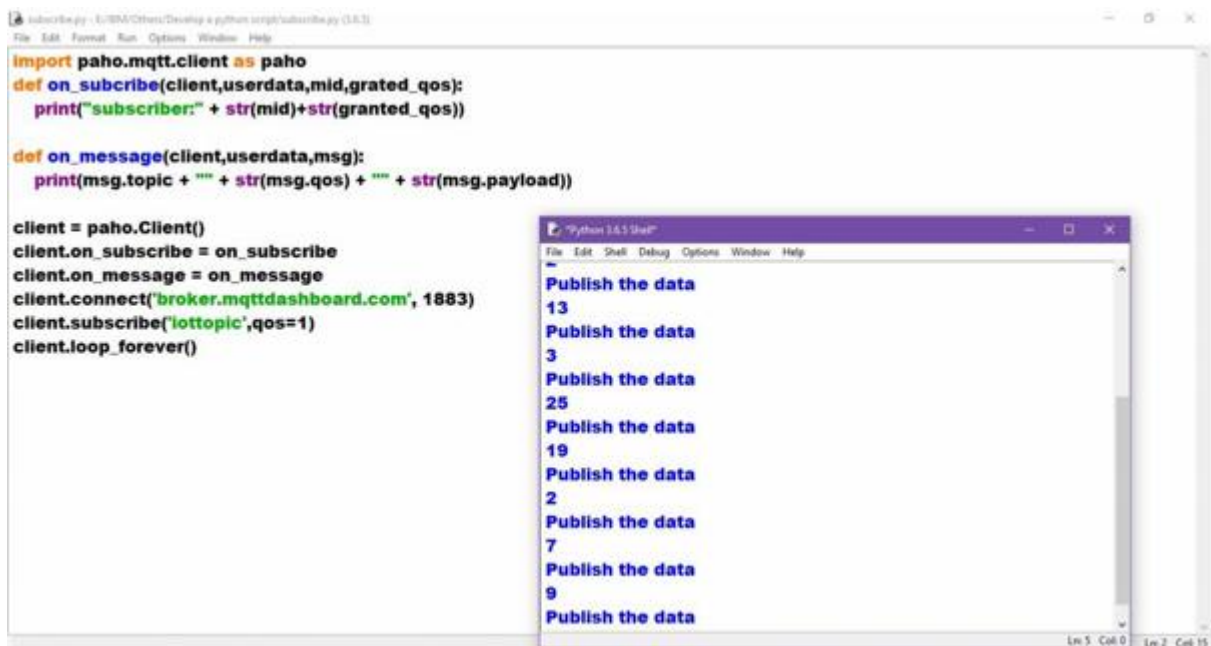
The screenshot shows a Python script named 'publish.py' in a text editor. The script is designed to publish data to an MQTT broker. It includes a comment: '#Through python coding we are going to access the subscriber'. The code imports 'paho.mqtt.client as paho', 'time', and 'random'. It defines a function 'on\_publish' that prints 'Publish the data'. The main logic creates a 'paho.Client', connects to 'broker.mqttdashboard.com' on port 1883, and enters a 'while True' loop. In the loop, it generates a random integer between 1 and 30, publishes it to the 'iotopic' with a QoS of 1, prints the value, and sleeps for 10 seconds. An inset window titled 'Python 3.6.5 Shell' shows the execution output, which includes the Python version and a restart message, followed by the repeated output 'Publish the data' with values 7, 19, 10, and 10.

```
#Through python coding we are going to access the subscriber
import paho.mqtt.client as paho
import time
import random

def on_publish(client, userdata, mid):
    print("Publish the data ")

client = paho.Client()
client.on_publish = on_publish
client.connect('broker.mqttdashboard.com', 1883)
client.loop_start()
while True:
    temp = random.randint(1,30)
    (re,mid) = client.publish('iotopic',str(temp),qos=1)
    print(temp)
    time.sleep(10)
```

Python 3.6.5 (v3.6.5:f59c0932b4, Mar 28 2018, 17:00:18) [MS C v.1900 64 bit (AMD64)] on win32  
Type "copyright", "credits" or "license()" for more information.  
>>>  
===== RESTART: E:\IBM\Others\Develop a python script\publish.py =====  
7  
Publish the data  
19  
Publish the data  
10  
Publish the data



The screenshot shows a Python script named 'subscribe.py' in a text editor. The script is designed to subscribe to an MQTT topic. It imports 'paho.mqtt.client as paho' and defines two functions: 'on\_subscribe' which prints the subscriber ID and granted QoS, and 'on\_message' which prints the topic, QoS, and payload. The main logic creates a 'paho.Client', connects to 'broker.mqttdashboard.com' on port 1883, subscribes to the 'iotopic' with a QoS of 1, and enters a 'loop\_forever()' loop. An inset window titled 'Python 3.6.5 Shell' shows the execution output, which includes the Python version and a restart message, followed by the repeated output 'Publish the data' with various values: 13, 3, 25, 19, 2, 7, 9, and 15.

```
import paho.mqtt.client as paho
def on_subscribe(client,userdata,mid,grated_qos):
    print("subscriber:" + str(mid)+str(granted_qos))

def on_message(client,userdata,msg):
    print(msg.topic + "" + str(msg.qos) + "" + str(msg.payload))

client = paho.Client()
client.on_subscribe = on_subscribe
client.on_message = on_message
client.connect('broker.mqttdashboard.com', 1883)
client.subscribe('iotopic',qos=1)
client.loop_forever()
```

Python 3.6.5 Shell  
Publish the data  
13  
Publish the data  
3  
Publish the data  
25  
Publish the data  
19  
Publish the data  
2  
Publish the data  
7  
Publish the data  
9  
Publish the data

IBM Watson IoT Platform

hariharan07ananth@psnacet.edu.in  
ID: kkfe0q

Browse Action Device Types Interfaces

Add Device +

Identity Device Information **Recent Events** State Logs

The recent events listed show the live stream of data that is coming and going from this device.

Event	Value	Format	Last Received
event_1	{"Temperature":94,"Humidity":95}	json	a few seconds ago
event_1	{"Temperature":73,"Humidity":43}	json	a few seconds ago
event_1	{"Temperature":72,"Humidity":22}	json	a few seconds ago
event_1	{"Temperature":57,"Humidity":55}	json	a few seconds ago
event_1	{"Temperature":64,"Humidity":53}	json	a few seconds ago

1 Simulation running

IBM Watson IoT Platform

hariharan07ananth@psna...  
ID: (select org)

Collect data from **Buildings** and make value from it

Learn More

Cookie Preferences

Show all

## Program :

```
#IBM Watson IOT Platform
#pip install wiotp-sdk
import wiotp.sdk.device
import time
import random
```

```

myConfig = {"identity":
{
    "orgId": "hj5fmy",
    "typeId": "NodeMCU",
    "deviceId": "12345" },
    "auth": { "token": "12345678" }
}

def myCommandCallback(cmd):
    print("Message received from IBM IoT Platform: %s" % cmd.data['command'])
    m=cmd.data['command']

client = wiotp.sdk.device.DeviceClient(config=myConfig, logHandlers=None)
client.connect()

while True:
    temp=random.randint(-20,125)
    hum=random.randint(0,100)
    myData={'temperature':temp, 'humidity':hum}
    client.publishEvent(eventId="status", msgFormat="json", data=myData, qos=0,
onPublish=None)
    print("Published data Successfully: %s", myData)
    client.commandCallback = myCommandCallback
    time.sleep(2)
    client.disconnect()

```