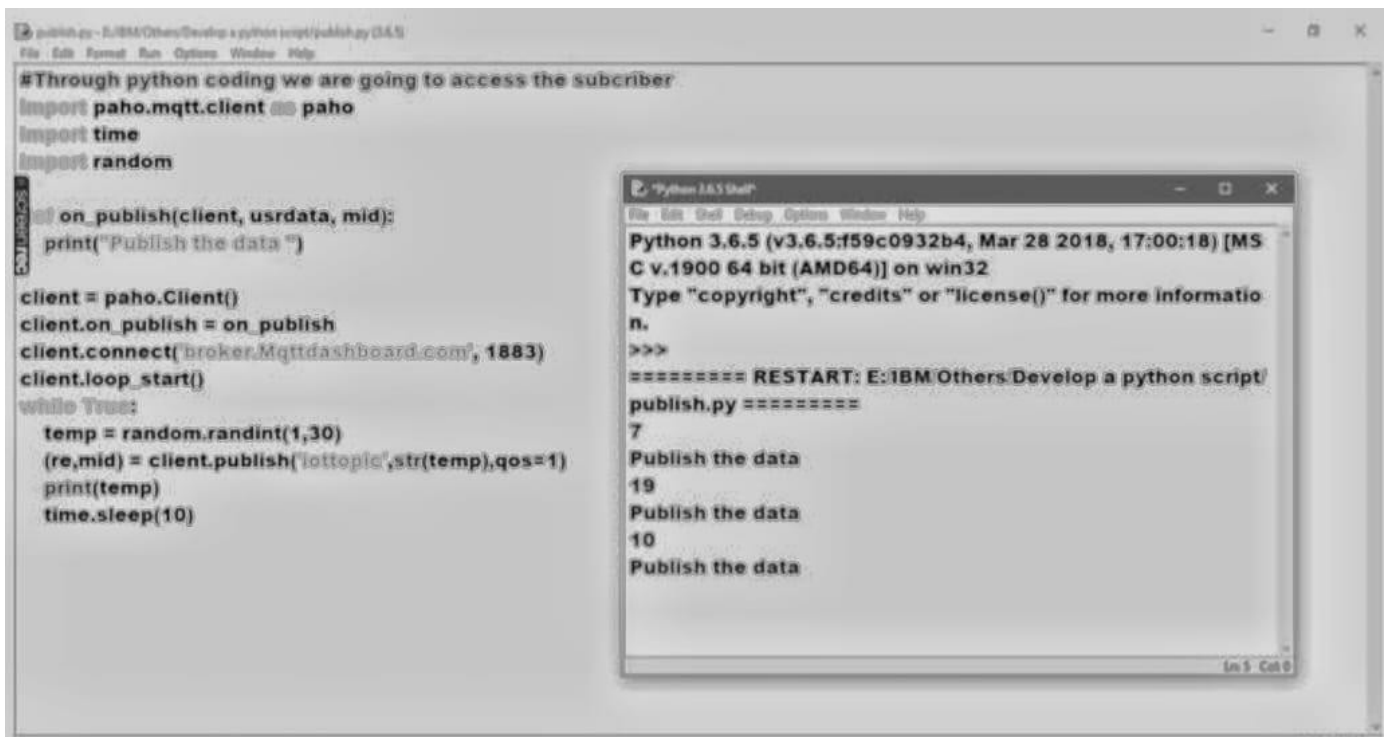


Develop the Python Script

(Publish data to IBM cloud)

Date	13 November 2022
Team ID	PNT2022TMID26584
Project Name	Industry-specific intelligent fire management system
Maximum Marks	4 Marks

Industry-specific intelligent fire management system



The screenshot shows a Python script in a text editor and its execution output in a terminal window. The script, named `publish.py`, is designed to publish data to an MQTT broker. It imports the `paho.mqtt.client` module, `time`, and `random`. It defines an `on_publish` callback function that prints the data being published. The main logic connects to the broker `broker.Mqtttdashboard.com` on port `1883`, starts the loop, and enters a `while True` loop where it generates a random number between 1 and 30, publishes it to the topic `iottopic`, and sleeps for 10 seconds.

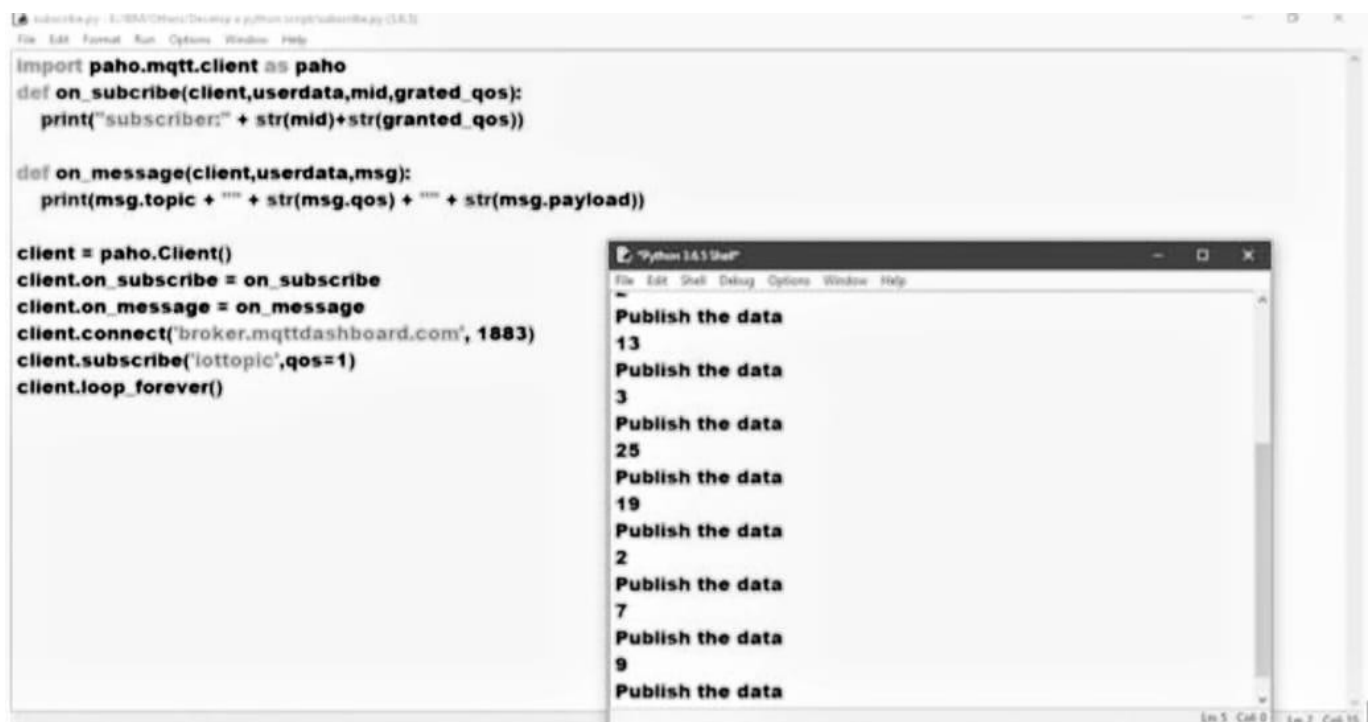
```
#Through python coding we are going to access the subscriber
import paho.mqtt.client as paho
import time
import random

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

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

The terminal window shows the output of the script, which is a series of random numbers published to the topic `iottopic`:

```
Python 3.6.5 Shell
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 informatio
n.
>>>
===== 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 in a text editor and its execution output in a terminal window. The script, named `subscribe.py`, is designed to subscribe to an MQTT topic and print the received messages. It imports the `paho.mqtt.client` module and defines two callback functions: `on_subscribe` and `on_message`. The `on_subscribe` function prints the subscription status, and the `on_message` function prints the received message. The main logic connects to the broker `broker.mqtttdashboard.com` on port `1883`, subscribes to the topic `iottopic`, and starts the loop.

```
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.mqtttdashboard.com', 1883)
client.subscribe('iottopic',qos=1)
client.loop_forever()
```

The terminal window shows the output of the script, which is a series of messages received from the topic `iottopic`:

```
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

211719106007@smartinternz.com
ID: dvo306

Browse Action Device Types Interfaces

Search by Device ID

Device Simulator

Device ID	Status	Device Type	Class ID	Date Added	Descriptive Location
sona22	Disconnected	sona22devicetype	Device	Nov 12, 2022 4:06 PM	

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":90,"Humidity":68}	json	a few seconds ago
event_1	{"Temperature":80,"Humidity":49}	json	a few seconds ago
event_1	{"Temperature":29,"Humidity":96}	json	a few seconds ago
event_1	{"Temperature":81,"Humidity":70}	json	a few seconds ago
event_1	{"Temperature":12,"Humidity":10}	json	a few seconds ago

1 Simulation running

IBM Watson IoT Platform

211719106039@smartinternz.com
ID: (select org)

Collect data from

Buildings

and make value from it

Learn More

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()

```