

Develop the Python Script

(Publish data to IBM cloud)

Date	04 November 2022
Team ID	PNT2022TMID25653
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 in a Python 3.6.5 Shell. The script, named 'publish.py', imports 'paho.mqtt.client as paho', 'time', and 'random'. It defines a function 'on_publish' that prints 'Publish the data'. The main code creates a 'paho.Client()', sets 'on_publish' as the callback, connects to 'broker.mqttdashboard.com' on port 1883, and starts a loop that publishes random data to 'iottopic' every 10 seconds. The shell output shows the script running successfully, with the message 'Publish the data' printed multiple times.

```
#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('iottopic',str(temp),qos=1)
    print(temp)
    time.sleep(10)
```

```
Python 3.6.5 Shell
File Edit Shell Debug Options Window Help
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 informati
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 in a Python 3.6.5 Shell. The script, named 'subscribe.py', imports 'paho.mqtt.client as paho'. It defines two functions: 'on_subscribe' that prints subscription details, and 'on_message' that prints the received message. The main code creates a 'paho.Client()', sets the callbacks, connects to 'broker.mqttdashboard.com' on port 1883, subscribes to 'iottopic', and starts a loop that prints the received data. The shell output shows the script running successfully, with the message 'Publish the data' printed multiple times.

```
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('iottopic',qos=1)
client.loop_forever()
```

```
Python 3.6.5 Shell
File Edit Shell Debug Options Window Help
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
```

The screenshot shows the IBM Watson IoT Platform dashboard. The top navigation bar includes 'Browse', 'Action', 'Device Types', and 'Interfaces'. A sidebar on the left contains various icons for navigation. The main content area displays a device named 'abcd' with a status of 'Disconnected' and a device ID of '123'. Below this, there is a section for 'Recent Events' with a table showing a live stream of data.

Event	Value	Format	Last Received
event_1	{"randomNumber":74}	json	a few seconds ago
event_1	{"randomNumber":47}	json	a few seconds ago
event_1	{"randomNumber":45}	json	a minute ago
event_1	{"randomNumber":19}	json	a minute ago
event_1	{"randomNumber":79}	json	a minute ago

Below the table, it indicates '1 Simulation running'.

The screenshot shows the IBM Watson IoT Platform landing page. The main heading is 'Buildings'. Below it, there are two phrases: 'Collect data from' and 'and make value from it'. The background features a stylized circuit-like pattern. At the bottom, there is a 'Learn More' link.

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()

```