

## DEVELOP THE PYTHON SCRIPT(PUBLISH DATA TO IBM CLOUD)

IBM ID	IBM-Project-6081-1658823192
Team ID	PNT2022TMID10960
Project Name	Industry-Specific Intelligent Fire Management System

```
# 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 (v3.6.5:f59c0932b4, Mar 28 2018, 17:00:18) [MSC 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

```
import paho.mqtt.client as paho

def on_subscribe(client,userdata,mid,granted_qos):
    print("subscriber:" + str(mid)+str(granted_qos))

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

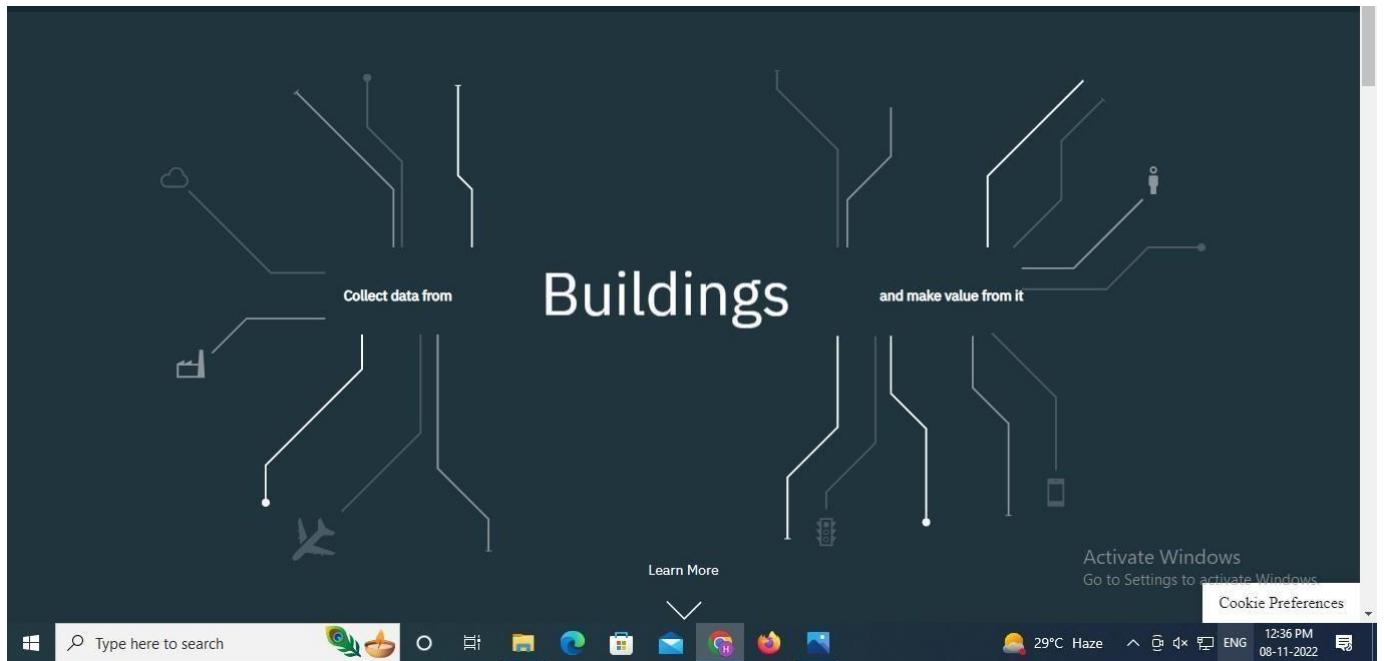
client = paho.Client()
client.on_subscribe = on_subscribe
client.on_message = on_message
client.connect('broker.mqttdashboard.com', 1883)
client.subscribe('iottopic',qos=1)
client.loop_forever()
```

Python 3.6.5 (v3.6.5:f59c0932b4, Mar 28 2018, 17:00:18) [MSC v.1900 64 bit (AMD64)] on win32  
Type "copyright", "credits" or "license()" for more information.  
>>>  
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 a web-based device management application. On the left is a vertical sidebar with icons for search, browse, action, device types, interfaces, and add device. The main header includes a search bar, navigation links (Browse, Action, Device Types, Interfaces), and a prominent blue "Add Device" button. Below the header is a table with columns: Device ID, Status, Device Type, Class ID, Date Acquired, and a more options icon. A single row is selected, showing "abed" as the Device ID, "Disconnected" as the Status, "123" as the Device Type, and "Nov 3, 2022 12:13 PM" as the Date Acquired. The table has a dark header and light rows. The main content area displays a tabbed view with "Identity" (selected), "Device Information", "Recent Events", "State", and "Logs". A message below the tabs states: "The recent events listed show the live stream of data that is coming and going from this device." Below this is a table of recent events:

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

A message box at the bottom right indicates "1 Simulation running". The bottom of the screen shows a Windows taskbar with a search bar containing "Type here to search", several pinned application icons (IBM, Screen, IBM, Node.js, What's New, node.js, IBM Cloud, and others), and system status indicators (Wi-Fi, battery, volume, and date/time: 04-11-2022 00:10).



## Program :

```
#IBM Watson IOT
Platform

#pip install wiotp-sdk
import wiotp.sdk.device
import time
import random

myConfig = { "identity": {
    "orgId": "f68qgi",
    "typeId": "NodeMCU",
    "deviceId": "2345",
    "auth": { "token": "12345678" }
}
def myCommandCallback(cmd): print ("Message received from IBM
IoTPlatform: %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()
```