

TEAM ID: PNT2022TMID18367

Code

```
import
random

import ibmiotf.application
import ibmiotf.device
from time import sleep
import sys
#IBM Watson Device Credentials.
organization = " rqs1m "
deviceType = "archana"
deviceId = "archana15"
authMethod = " use-token-auth "
authToken = "123456789"
def myCommandCallback(cmd):
    print("Command received: %s" % cmd.data['command'])
    status=cmd.data['command']
    if status=="sprinkler_on":
        print ("sprinkler is ON")
    else :
        print ("sprinkler is OFF")
    #print(cmd)

try:
    deviceOptions = {"org": organization, "type": deviceType, "id": deviceId,
"auth-method": authMethod, "auth-token": authToken}
    deviceCli = ibmiotf.device.Client(deviceOptions)
except Exception as e:
    print("Caught exception connecting device: %s" % str(e))
sys.exit()
#Connecting to IBM watson.
deviceCli.connect()
while True:
    #Getting values from sensors.
    temp_sensor = round( random.uniform(0,80),2)
    PH_sensor = round(random.uniform(1,14),3)
    camera = ["Detected", "Not Detected", "Not Detected", "Not Detected", "Not
Detected", "Not Detected",]
    camera_reading = random.choice(camera)
    flame = ["Detected", "Not Detected", "Not Detected", "Not Detected", "Not
Detected", "Not Detected",]
    flame_reading = random.choice(flame)
    moist_level = round(random.uniform(0,100),2)
```

```
water_level = round(random.uniform(0,30),2)
```

```
#storing the sensor data to send in json format to cloud.
```

```
temp_data = { 'Temperature' : temp_sensor }  
PH_data = { 'PH Level' : PH_sensor }  
camera_data = { 'Animal attack' : camera_reading}  
flame_data = { 'Flame' : flame_reading }  
moist_data = { 'Moisture Level' : moist_level}  
water_data = { 'Water Level' : water_level}
```

```
# publishing Sensor data to IBM Watson for every 5-10 seconds.
```

```
success = deviceCli.publishEvent("Temperature sensor", "json", temp_data,  
qos=0)  
sleep(1)  
if success:  
    print (" .....publish ok..... ")  
    print ("Published Temperature = %s C" % temp_sensor, "to IBM Watson")
```

```
success = deviceCli.publishEvent("PH sensor", "json", PH_data, qos=0)  
sleep(1)  
if success:  
    print ("Published PH Level = %s" % PH_sensor, "to IBM Watson")
```

```
success = deviceCli.publishEvent("camera", "json", camera_data, qos=0)  
sleep(1)  
if success:  
    print ("Published Animal attack %s " % camera_reading, "to IBM Watson")  
success = deviceCli.publishEvent("Flame sensor", "json", flame_data, qos=0)  
sleep(1)  
if success:  
    print ("Published Flame %s " % flame_reading, "to IBM Watson")
```

```
success = deviceCli.publishEvent("Moisture sensor", "json", moist_data,  
qos=0)  
sleep(1)  
if success:  
    print ("Published Moisture Level = %s " % moist_level, "to IBM Watson")
```

```
success = deviceCli.publishEvent("Water sensor", "json", water_data, qos=0)  
sleep(1)  
if success:  
    print ("Published Water Level = %s cm" % water_level, "to IBM Watson")  
print ("")
```

```
#Automation to control sprinklers by present temperature an to send alert  
message to IBM Watson.
```

```

if (temp_sensor > 35):
    print("sprinkler-1 is ON")
    success = deviceCli.publishEvent("Alert1", "json", { 'alert1' :
"Temperature(%s) is high, sprinklerlers are turned ON" %temp_sensor }
, qos=0)
    sleep(1)
    if success:
        print( 'Published alert1 : ', "Temperature(%s) is high, sprinklerlers are turned
ON" %temp_sensor,"to IBM Watson")
        print("")
    else:
        print("sprinkler-1 is OFF")
        print("")

```

#To send alert message if farmer uses the unsafe fertilizer to crops.

```

if (PH_sensor > 7.5 or PH_sensor < 5.5):
    success = deviceCli.publishEvent("Alert2", "json", { 'alert2' : "Fertilizer PH
level(%s) is not safe,use other fertilizer" %PH_sensor } ,
qos=0)
    sleep(1)
    if success:
        print('Published alert2 : ', "Fertilizer PH level(%s) is not safe,use other
fertilizer" %PH_sensor,"to IBM Watson")
        print("")

```

#To send alert message to farmer that animal attack on crops.

```

if (camera_reading == "Detected"):
    success = deviceCli.publishEvent("Alert3", "json", { 'alert3' : "Animal attack
on crops detected" }, qos=0)
    sleep(1)
    if success:
        print('Published alert3 : ', "Animal attack on crops detected","to IBM
Watson","to IBM Watson")
        print("")

```

#To send alert message if flame detected on crop land and turn ON the splinkers to take immediate action.

```

if (flame_reading == "Detected"):
    print("sprinkler-2 is ON")
    success = deviceCli.publishEvent("Alert4", "json", { 'alert4' : "Flame is
detected crops are in danger,sprinklers turned ON" }, qos=0)
    sleep(1)
    if success:

```

```

    print( 'Published alert4 : ' , "Flame is detected crops are in danger,sprinklers
turned ON","to IBM Watson")

#To send alert message if Moisture level is LOW and to Turn ON Motor-1 for
irrigation.
if (moist_level < 20):
    print("Motor-1 is ON")
    success = deviceCli.publishEvent("Alert5", "json", { 'alert5' : "Moisture
level(%s) is low, Irrigation started" %moist_level }, qos=0)
    sleep(1)
    if success:
        print('Published alert5 : ' , "Moisture level(%s) is low, Irrigation started"
%moist_level,"to IBM Watson" )
        print("")
#To send alert message if Water level is HIGH and to Turn ON Motor-2 to
take water out.
if (water_level > 20):
    print("Motor-2 is ON")
    success = deviceCli.publishEvent("Alert6", "json", { 'alert6' : "Water level(%s)
is high, so motor is ON to take water out "
% water_level }, qos=0)
    sleep(1)
    if success:
        print('Published alert6 : ' , "water level(%s) is high, so motor is ON to take
water out " %water_level,"to IBM Watson" )
        print("")
#command recived by farmer
deviceCli.commandCallback = myCommandCallback
# Disconnect the device and application from the cloud
deviceCli.disconnect()

```

Code 1

```

def
myCommandCallback(cmd):
# function for Callback if
cm.data['command'] ==
'motoron':

    print("MOTOR ON IS RECEIVED")
    elif cmd.data['command'] == 'motoroff': print("MOTOR
OFF IS RECEIVED")
    if cmd.command == "setInterval":

    else:
    if 'interval' not in cmd.data:

```

```
print("Error - command is missing required information:  
'interval'")
```

```
interval = cmd.data['interval']
```

```
elif cmd.command == "print":  
    if 'message' not in cmd.data:  
        print("Error - command is missing required information:  
'message'")  
    else: output = cmd.data['message']  
    print(output)
```

```
try:
```

```
    deviceOptions = {"org": organization, "type":  
deviceType, "id": deviceId, "authmethod": authMethod,  
"auth-token": authToken} deviceCli  
    = ibmiotf.device.Client(deviceOptions) #  
    .....
```

```
except Exception as e:  
    print("Caught exception connecting device: %s" % str(e))  
    sys.exit()
```

```
# Connect and send a datapoint "hello" with value "world"  
into the cloud as an event of type "greeting" 10 times  
deviceCli.connect()
```

```
while True:  
    deviceCli.commandCallback = myCommandCallback
```

```
# Disconnect the device and application from the cloud  
deviceCli.disconnect()
```

SENSOR.PY

```
import time  
import sys  
import ibmiotf.application  
import ibmiotf.device  
import random
```

```
# Provide your IBM Watson Device Credentials  
organization = "8gyz7t" # replace the ORG ID  
deviceType = "weather_monitor" # replace the Device  
type deviceId = "b827ebd607b5" # replace Device ID
```

```

authMethod = "token" authToken =
"LWVpQPpVQ166HWN48f" # Replace the authtoken

def myCommandCallback(cmd):

    print("Command received: %s" % cmd.data['command'])
    print(cmd)

    try:
        deviceOptions = {"org": organization, "type":
        deviceType, "id": deviceId,
        "auth-method": authMethod, "auth-token": authToken}
        deviceCli = ibmiotf.device.Client(deviceOptions)
        #.....

    except Exception as e:
        print("Caught exception connecting device: %s" % str(e))
        sys.exit()

    # Connect and send a datapoint "hello" with value "world"
    into the cloud as an event of type "greeting" 10 times
    deviceCli.connect()

    while True:
        temp=random.randint(0,100)
        pulse=random.randint(0,100)
        soil=random.randint(0,100)

        data = { 'temp' : temp, 'pulse': pulse ,'soil':soil} #print data
        def
        myOnPublishCallback():
            print ("Published Temperature = %s C" % temp,
            "Humidity = %s %%" % pulse,"Soil Moisture = %s %%"
            % soil,"to IBM Watson")

        success = deviceCli.publishEvent("IoTSensor", "json",
        data, qos=0, on_publish=myOnPublishCallback)
        if not success:
            print("Not connected to IoT") time.sleep(1)

    deviceCli.commandCallback = myCommandCallback

```

Code 3

```

[
    {

```

```
"id":"625574ead9839b34",
{
  "type":"ibmiotout", "z":"630c8601c5ac3295",
  "authentication":"apiKey",
  "apiKey":"ef745d48e395ccc0",
  "outputType":"cmd",
  "deviceId":"b827ebd607b5",
  "deviceType":"weather_monitor",
  "eventCommandType":"data",
  "format":"json",
  "data":"data",
  "qos":0,
  "name":"IBM IoT",
  "service":"registere
d", "x":680,
  "y":220,
  "wires":[]
},
{
  "id":"4cff18c3274cccc4", "type":"ui_button",
  "z":"630c8601c5ac3295",
  "name": "",
  "group":"716e956.00eed6c",
  "order":2,
  "width":0,
  "height":0,
  "passthru":false,
  "label":"MotorON",
  "tooltip": "",
  "color": "",
  "bgcolor": "",
  "className": "",
  "icon": "",
  "payload": "{ \"command\": \"motoron\" }",
  "payloadType": "str",
  "topic": "motoron",
  "topicType": "str",
  "x":360,
  "y":160, "wires":[["625574ead9839b34"]],
  {
    "id":"659589baceb4e0b0",
    "type":"ui_button", "z":"630c8601c5ac3295",
    "name": "",
    "group":"716e956.00eed6c",
    "order":3,
    "width":0,
```

```
"height": "0",
"passthru": true,
"label": "MotorOFF",
"tooltip": "",
"color": "",
"bgcolor": "",
"className": "",
"icon": "",
"payload": "{\"command\": \"motoroff\"}",
"payloadType": "str",
"topic": "motoroff",
"topicType": "str",
"x": 350,
"y": 220, "wires": [{"id": "625574ead9839b34"}],
{"id": "ef745d48e395ccc0", "type": "ibmiot",
"name": "weather_monitor", "keepalive": "60",
"serverName": "",
"cleansession": true,
"appId": "",
"shared": false},
{"id": "716e956.00eed6c",
"type": "ui_group",
"name": "Form",
"tab": "7e62365e.b7e6b8",
"order": 1,
"disp": true,
"width": "6",
"collapse": false},
{"id": "7e62365e.b7e6b8",
"type": "ui_tab",
"name": "control",
"icon": "dashboard",
"order": 1,
"disabled": false,
"hidden": false}
]
[
{
"id": "b42b5519fee73ee2", "type": "ibmiotin",
"z": "03acb6ae05a0c712",
"authentication": "apiKey",
"apiKey": "ef745d48e395ccc0",
"inputType": "evt",
"logicalInterface": ""
```



```
"ruleId": "",
"deviceId": "b827ebd607b5",
"applicationId": "",
"deviceType": "weather_monitor",
"eventType": "+",
"commandType": "",
"format": "json",
"name": "IBMIoT",
"service": "registered",
"allDevices": "",
"allApplications": "",
"allDeviceTypes": "",
"allLogicalInterfaces": "",
"allEvents": true,
"allCommands": "",
"allFormats
": "",
"qos": 0,
"x": 270,
"y": 180,
"wires": [
["50b13e02170d73fc", "d7da6c2f5302ffaf", "a949797028158f3f", "a71f164
bc3 78bcf1"]
],
{
"id": "50b13e02170d73fc",
",
"type": "function",
"z": "03acb6ae05a0c712",
", "name": "Soil
Moisture",
"func": "msg.payload = msg.payload.soil;\nglobal.set('s',msg.payload);\nreturn
msg;",
"outputs": 1,
"noerr":
0,
"initialize
": "",
"finalize": "",
"libs": [],
"x": 490,
"y": 120,
"wires": [
["a949797028158f3f", "ba98e701f55f04fe"]
],
{
"id": "d7da6c2f5302ffaf", "type": "function",
"z": "03acb6ae05a0c712",
```

```
"name":"Humidity",
"func":"msg.payload = msg.payload.pulse;\nglobal.set('p',msg.payload)\nreturn
msg;",
"outputs":1,
"noerr":
0,
"initialize
": "",
"finalize": "",
"li
bs
": [
],
"x
":
48
0,
"y":260, "wires":[["a949797028158f3f","70a5b076eeb80b70"]]
},
{
"id":"a949797028158f3f
",
"type":"debug",
"z":"03acb6ae05a0c712
", "name":"IBMo/p",
"active":true,
"tosidebar":true,
"console":false,
"tostatus":false,
"complete":"payload",
"targetType":"msg",
"statusVal":"",
"statusType":"auto",
"x":780,
"y":180,
"wires":[
],
},
{
"id":"70a5b076eeb80b70",
"type":"ui_gauge",
"z":"03acb6ae05a0c712",
"name":"",
"group":"f4cb8513b95c98a4",
"order":6,
"width":"0",
"height":"0",
```

```
"gtype":"gage",
"title":"Humidity",
"label":"Percentage(%)",
"format":"{{ value }}"
,"min":0,
"max":100,
"colors":["#00b500","#e6e600","#ca3838"], "seg1":"","
"seg2":"","
"classname
":"","x":86
0,
"y":260,
"wires":[]
},
{
"id":"a71f164bc378bcf1","type":"function",
"z":"03acb6ae05a0c712",
"name":"Temperature",
"func":"msg.payload=msg.payload.temp;\nglobal.set('t',msg.payload);\nreturn
msg;","outputs":1,
"noerr":
0,
"initialize
":"","
"finalize":"","
"li
bs
":["
],
"x
":
49
0,
"y":360,
"wires":[["8e8b63b110c5ec2d","a949797028158f3f"]]
},
{
"id":"8e8b63b110c5ec2d",
"type":"ui_gauge",
"z":"03acb6ae05a0c712",
"name":"","
"group":"f4cb8513b95c98a4",
"order":11,
"width":0,
"height":0,
"gtype":"gage",
```

```
"title":"Temperature",
"label":"DegreeCelcius",
"format":"{{ value }}",
"min":0,
"max":"100",
"colors":["#00b500","#e6e600","#ca3838"],"seg1":"","
"seg2":"","
"className
":"",
"x":790,
"y":360,
"wires":[]
},
{
"id":"ba98e701f55f04fe",
"type":"ui_gauge",
"z":"03acb6ae05a0c712",
"name":"",
"group":"f4cb8513b95c98a4",
"order":1,
"width":"0",
"height":"0",
"ctype":"gage",
"title":"Soil Moisture",
"label":"Percentage(%)",
"format":"{{ value }}"
,"min":0,
"max":"100",
"colors":["#00b500","#e6e600","#ca3838"],"seg1":"","
"seg2":"","
"className
":"",
"x":790,
"y":120,
"wires":[]
},
{
"id":"a259673baf5f0f98",
"type":"httpin",
"z":"03acb6ae05a0c712",
"name":"",
"url":"/sensor",
"method":"get",
"upload":false,
e,
```

```
"swaggerDoc"
:"","x":370,
"y":500,
"wires":[["18a8cdbf7943d27a"]]
},
{
  "id":"18a8cdbf7943d27a","type":"function",
  "z":"03acb6ae05a0c712",
  "name":"httpfunction",
  "func":"msg.payload{\"pulse\":global.get('p'),\"temp\":global.get('t'),\"soil\":global.get('s')};\nreturn msg;",
  "outputs":1,
  "noerr":0,
  "initialize":"","",
  "finalize":"","",
  "labeli
bs
":[
],
"x
":
63
0,
"y":500, "wires":[["5c7996d53a445412"]]
},
{
  "id":"5c7996d53a445412",
  "type":"httpresponse",
  "z":"03acb6ae05a0c712",
  "name":"","",
  "statusCode":"","",
  "headers":{ },
  "x":870,
  "y":500,
  "wires":[]
},
{
  "id":"ef745d48e395ccc0",
  "type":"ibmiot",
  "name":"weather_monitor",
  "keepalive":"60",
  "serverName":"","",
  "cleansession":true,
```

```
"appId":"","  
"shared":false},  
{  
"id":"f4cb8513b95c98a4","type":"ui_group",  
"name":"monitor",  
"tab":"1f4cb829.2fdee8  
", "order":2,  
"disp":  
true,  
"width  
":"6",  
"collapse":f  
alse,  
"className  
":"  
"},  
{  
"id":"1f4cb829.2fdee8",  
"type":"ui_tab",  
"name":"Home",  
"icon":"dashboard  
", "order":3,  
"disabled":false,  
"hidden":false }
```