

Sprint-2

DATE	18 November 2022
TEAM ID	PNT2022TMID13267
PROJECT NAME	SMART WASTE MANAGEMENT FOR METROPOLITAN CITIES

Python code to connect ibm watson

```
import time
import sys
import ibmiotf.application
import ibmiotf.device
import random
#Provide your IBM Watson Device Credentials
organization = "vi4esk"
deviceType = "sudhan"
deviceId = "12345"
authMethod = "token"
authToken = "12345678"
# Initialize GPIO
def myCommandCallback(cmd):
    print("Command received: %s" % cmd.data['command'])
    status=cmd.data['command']
    if status=="alarmon":
        print ("Alarm is on")
    else:
        print ("Alarm 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()
# Connect and send a datapoint "hello" with value "world" into the cloud as an event of type "greeting" 10 times
deviceCli.connect()
while True:
    #Get Sensor Data from DHT11

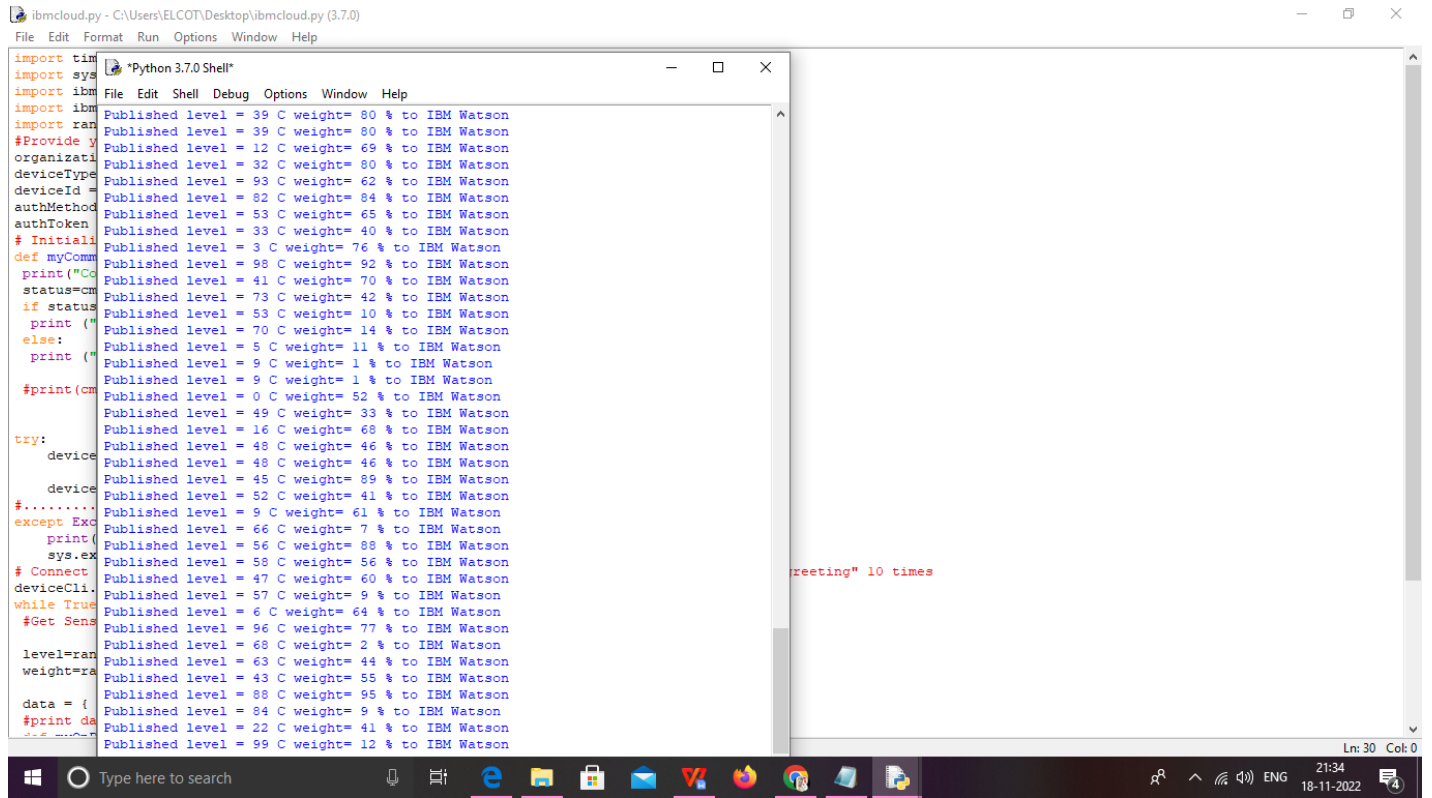
    level=random.randint(0,100)
    weight=random.randint(0,100)

    data = { 'level' : level, 'weight': weight }
    #print data
    def myOnPublishCallback():
        print ("Published level = %s C" % level, "weight= %s %" % weight, "to IBM Watson")
    success = deviceCli.publishEvent("IoTSensor", "json", data, qos=0,
    on_publish=myOnPublishCallback)
    if not success:
```

```
print("Not connected to IoT")
time.sleep(10)
```

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

Output image:



```
import time
import sys
import ibm
import random
# Provide your organization details
organizationId = "1234567890"
deviceType = "IBM Watson"
deviceId = "1234567890"
authMethod = "API Key"
authToken = "1234567890"
# Initial setup
def myCommandCallback(deviceId, data):
    print("Command received for device %s" % deviceId)
    status = cm.get_status(deviceId)
    if status:
        print("Device status: %s" % status)
    else:
        print("Device status not found")
    # Print command details
    #print("Command details: %s" % data)
try:
    device = cm.connect(deviceId, authMethod, authToken)
    device.commandCallback = myCommandCallback
    # Connect to the cloud
    deviceCli = cm.deviceCli(deviceId, authMethod, authToken)
    while True:
        # Get sensor data
        level = random.randint(1, 100)
        weight = random.randint(1, 100)
        data = {
            "level": level,
            "weight": weight
        }
        # Print data
        print("Data: %s" % data)
```

Published level = 39 C weight= 80 % to IBM Watson
Published level = 39 C weight= 80 % to IBM Watson
Published level = 12 C weight= 69 % to IBM Watson
Published level = 32 C weight= 80 % to IBM Watson
Published level = 93 C weight= 62 % to IBM Watson
Published level = 82 C weight= 84 % to IBM Watson
Published level = 53 C weight= 65 % to IBM Watson
Published level = 33 C weight= 40 % to IBM Watson
Published level = 3 C weight= 76 % to IBM Watson
Published level = 98 C weight= 92 % to IBM Watson
Published level = 41 C weight= 70 % to IBM Watson
Published level = 73 C weight= 42 % to IBM Watson
Published level = 53 C weight= 10 % to IBM Watson
Published level = 70 C weight= 14 % to IBM Watson
Published level = 5 C weight= 11 % to IBM Watson
Published level = 9 C weight= 1 % to IBM Watson
Published level = 9 C weight= 1 % to IBM Watson
Published level = 0 C weight= 52 % to IBM Watson
Published level = 49 C weight= 33 % to IBM Watson
Published level = 16 C weight= 69 % to IBM Watson
Published level = 48 C weight= 46 % to IBM Watson
Published level = 48 C weight= 46 % to IBM Watson
Published level = 45 C weight= 89 % to IBM Watson
Published level = 52 C weight= 41 % to IBM Watson
Published level = 9 C weight= 61 % to IBM Watson
Published level = 66 C weight= 7 % to IBM Watson
Published level = 56 C weight= 88 % to IBM Watson
Published level = 58 C weight= 56 % to IBM Watson
Published level = 47 C weight= 60 % to IBM Watson
Published level = 57 C weight= 9 % to IBM Watson
Published level = 6 C weight= 64 % to IBM Watson
Published level = 96 C weight= 77 % to IBM Watson
Published level = 68 C weight= 2 % to IBM Watson
Published level = 63 C weight= 44 % to IBM Watson
Published level = 43 C weight= 55 % to IBM Watson
Published level = 88 C weight= 95 % to IBM Watson
Published level = 84 C weight= 9 % to IBM Watson
Published level = 22 C weight= 41 % to IBM Watson
Published level = 99 C weight= 12 % to IBM Watson

greeting" 10 times

Ln: 30 Col: 0

Ibm watson image:

The screenshot displays the IBM Watson IoT Platform dashboard in a web browser. The browser's address bar shows the URL `vi4esk.internetofthings.ibmcloud.com/dashboard/devices/browse`. The page header includes the IBM Watson IoT Platform logo and a user profile section with the email `621319106082@smartinternz.com` and ID `vi4esk`. The dashboard has a sidebar with navigation icons and a main content area with tabs for `Browse`, `Action`, `Device Types`, and `Interfaces`. The `Browse` tab is active, showing a message: "The recent events listed show the live stream of data that is coming and going from this device." Below this message is a table of recent events. The table has four columns: `Event`, `Value`, `Format`, and `Last Received`. It lists five events, all from `IoTSensor` devices, with values in JSON format and timestamps of "a few seconds ago". At the bottom of the dashboard, a status box indicates "0 Simulations running". The Windows taskbar at the bottom shows the search bar and various application icons, with the system clock displaying 21:36 on 18-11-2022.

Event	Value	Format	Last Received
IoTSensor	<code>{"level":86,"weight":73}</code>	json	a few seconds ago
IoTSensor	<code>{"level":85,"weight":14}</code>	json	a few seconds ago
IoTSensor	<code>{"level":78,"weight":21}</code>	json	a few seconds ago
IoTSensor	<code>{"level":78,"weight":88}</code>	json	a few seconds ago
IoTSensor	<code>{"level":88,"weight":80}</code>	json	a few seconds ago