DEVELOP A PYTHON SCRIPT TO PUBLISH AND SUBSCRIBE TO IBM IOT PLATFORM

Date	19 Nov 2022
Team ID	PNT2022TMID11377
Project Name	IOT enabled smart farmer application

PYTHON CODE FOR HUMDITY, SOIL MOISTURE AND TEMPERATURE:

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


#Provide your IBM Watson Device Credentials
organization = "z8hv0n"
deviceType = "Sprint1"
deviceId = "12345678"
authMethod = "token"
authToken = "12345678"

# Initialize GPIO


def myCommandCallback(cmd):
   print("Command received: %s" % cmd.data['command'])
   status=cmd.data['command']
   if status=="motoron":
      print ("led is on")
   elif status == "motoroff":
      print("led is off")
   else :
      print ("please send proper 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:
       #Get Sensor Data from DHT11

       temp=random.randint(0,100)
       Hum=random.randint(0,100)

       data = { 'temp' : temp, 'Hum': Hum }
       #print data

       def myOnPublishCallback():
          print ("Published Temperature = %s C" % temp, "Hum = %s %%" % Hum, "to IBM
Watson")
       success = deviceCli.publishEvent("IoTSensor", "json", data, qos=0,
on_publish=myOnPublishCallback)
       if not success:

          print("Not connected to IoTF")


       time.sleep(1)


       deviceCli.commandCallback = myCommandCallback

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

PYTHON CODE FOR PIR SENSOR:

```
#include  &lt;WiFi.h&gt;
#include &lt;PubSubClient.h&gt;
WiFiClient wifiClient;
String data3;
#define ORG &quot;oezexmo&quot;
#define DEVICE_TYPE &quot;resberypi&quot;
#define DEVICE_ID &quot;12345&quot;
#define TOKEN &quot;y6Lb7lznmBD&amp;Iv9euq&quot;
int ledPin = 12; // choose the pin for the LED
int inputPin = 2; // choose the input pin (for PIR sensor)
int pirState = LOW; // we start, assuming no motion detected
int val = 0; // variable for reading the pin status
void setup() {
pinMode(ledPin, OUTPUT); // declare LED as output
pinMode(inputPin, INPUT); // declare sensor as input
Serial.begin(9600);
}
void loop() {
val = digitalRead(inputPin); // read input value
if (val == HIGH) { // check if the input is HIGH
digitalWrite(ledPin, HIGH); // turn LED ON
//void publishData();
if (pirState == LOW) {
// we have just turned on
Serial.println(&quot;Motion detected!&quot;);
Serial.println(&quot;Camera activated!&quot;);
delay(1000);
Serial.println(&quot;Pictures taken!&quot;);
// We only want to print on the output change, not state
pirState = HIGH;
}
}
else {
digitalWrite(ledPin, LOW); // turn LED OFF
//void publishData();
if (pirState == HIGH) {
// we have just turned of
Serial.println(&quot;Motion ended!&quot;);
// We only want to print on the output change, not state
pirState = LOW;
}}}
```