

Project Development Phase

Sprint-2

Date	05 Nov 2022
Team ID	PNT2022TMID46771
Project Name	Smart farmer-IOT Enabled smart farming application

Step-1

import time

import sys

import ibmiotf.application

import ibmiotf.device

import random

#Provide your IBM Watson Device Credentials

organization = "kiebfw"

deviceType = "nodefarmer"

deviceId = "2002"

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
```

```

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

data = { 'temp' : temp, 'Humid': Humid }
#print data
def myOnPublishCallback():
    print ("Published Temperature = %s C" % temp, "Humidity = %s %" % Humid, "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

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

```

OUTPUT:

```
nodefarmer.py - C:\Users\Satellite\Downloads\node\nodefarmer.py (3.7.0)
File Edit Format Run Options Window Help

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

#Provide your IBM Watson Device Credentials
organization = "kiebfw"
deviceType = "nodefarmer"
deviceId = "2002"
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}
    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
deviceCli.connect()
```

```
*Python 3.7.0 Shell*
File Edit Shell Debug Options Window Help

Published Temperature = 78 C Humidity = 26 % to IBM Watson
Published Temperature = 46 C Humidity = 14 % to IBM Watson
Published Temperature = 73 C Humidity = 14 % to IBM Watson
Published Temperature = 10 C Humidity = 71 % to IBM Watson
Published Temperature = 19 C Humidity = 67 % to IBM Watson
Published Temperature = 74 C Humidity = 79 % to IBM Watson
Published Temperature = 62 C Humidity = 81 % to IBM Watson
Published Temperature = 86 C Humidity = 55 % to IBM Watson
Published Temperature = 67 C Humidity = 85 % to IBM Watson
Published Temperature = 64 C Humidity = 24 % to IBM Watson
Published Temperature = 43 C Humidity = 92 % to IBM Watson
Published Temperature = 75 C Humidity = 63 % to IBM Watson
Published Temperature = 17 C Humidity = 19 % to IBM Watson
Published Temperature = 46 C Humidity = 98 % to IBM Watson
Published Temperature = 68 C Humidity = 99 % to IBM Watson
Published Temperature = 96 C Humidity = 93 % to IBM Watson
Published Temperature = 14 C Humidity = 93 % to IBM Watson
Published Temperature = 40 C Humidity = 95 % to IBM Watson
Published Temperature = 17 C Humidity = 20 % to IBM Watson
Published Temperature = 56 C Humidity = 49 % to IBM Watson
Published Temperature = 81 C Humidity = 69 % to IBM Watson
Published Temperature = 99 C Humidity = 53 % to IBM Watson
Published Temperature = 87 C Humidity = 81 % to IBM Watson
Published Temperature = 47 C Humidity = 69 % to IBM Watson
Published Temperature = 66 C Humidity = 34 % to IBM Watson
Published Temperature = 52 C Humidity = 37 % to IBM Watson
Published Temperature = 59 C Humidity = 57 % to IBM Watson
Published Temperature = 51 C Humidity = 95 % to IBM Watson
Published Temperature = 93 C Humidity = 59 % to IBM Watson
Published Temperature = 69 C Humidity = 95 % to IBM Watson
Published Temperature = 58 C Humidity = 30 % to IBM Watson
Published Temperature = 1 C Humidity = 17 % to IBM Watson
Published Temperature = 64 C Humidity = 84 % to IBM Watson
Published Temperature = 86 C Humidity = 67 % to IBM Watson
Published Temperature = 11 C Humidity = 40 % to IBM Watson
Published Temperature = 73 C Humidity = 47 % to IBM Watson
Published Temperature = 19 C Humidity = 25 % to IBM Watson
Published Temperature = 91 C Humidity = 7 % to IBM Watson

Ln: 11 Col: 17
Ln: 35 Col: 57
5:12 PM
11/18/2022
```

<https://node-red-iyokf-2022-11-08.eu-gb.mybluemix.net/red/#flow/a8f261bd1dae67b4>

Get /command

The screenshot shows the Node-RED web interface in a browser. The flow diagram on the left includes nodes for 'catch', 'status', 'link in', 'link call', 'link out', 'comment', 'function', 'switch', 'change', 'range', 'template', and 'delay'. The flow is connected to a 'temp' node, which is connected to a 'Humid' node, which is connected to an 'alarmon' node, which is connected to an 'alarmoff' node, which is connected to a 'mit' node, which is connected to a '[get] /sensor' node. The 'Edit http in node' dialog is open, showing the following properties:

- Method: GET
- URL: /command
- Name: mit

The debug console on the right shows the following messages:

```
11/18/2022, 5:22:03 PM node: ace51a660b648914
iot-2/type/nodefarmerid/2002/evt/IoTSensor/fmt/json :
msg.payload: Object
{ temp: 52, Humid: 96 }

11/18/2022, 5:22:03 PM node: ace51a660b648914
iot-2/type/nodefarmerid/2002/evt/IoTSensor/fmt/json :
msg.payload: number
52

11/18/2022, 5:22:04 PM node: ace51a660b648914
iot-2/type/nodefarmerid/2002/evt/IoTSensor/fmt/json :
msg.payload: number
96

11/18/2022, 5:22:04 PM node: ace51a660b648914
iot-2/type/nodefarmerid/2002/evt/IoTSensor/fmt/json :
msg.payload: Object
{ temp: 89, Humid: 15 }

11/18/2022, 5:22:04 PM node: ace51a660b648914
iot-2/type/nodefarmerid/2002/evt/IoTSensor/fmt/json :
msg.payload: number
89

11/18/2022, 5:22:04 PM node: ace51a660b648914
iot-2/type/nodefarmerid/2002/evt/IoTSensor/fmt/json :
```

Get /sensor

The screenshot displays the Node-RED web interface in a browser. The main workspace shows a flow named 'Flow 1' with several nodes: 'temp', 'Humid', 'alarmon', 'alarmoff', 'mit', and '[get] /sensor'. The '[get] /sensor' node is selected, and its configuration is shown in the 'Edit http in node' panel. The configuration includes:

- Method: GET
- URL: /sensor
- Name: Name

The debug console on the right shows the following log entries:

```
11/18/2022, 5:23:15 PM node: ace51a660b648914  
iot-2/type/nodefarmerid/2002/ev/IoTSensor/fmt/json :  
msg.payload : Object  
  { temp: 76, Humid: 67 }  
11/18/2022, 5:23:16 PM node: ace51a660b648914  
iot-2/type/nodefarmerid/2002/ev/IoTSensor/fmt/json :  
msg.payload : number  
76  
11/18/2022, 5:23:17 PM node: ace51a660b648914  
iot-2/type/nodefarmerid/2002/ev/IoTSensor/fmt/json :  
msg.payload : number  
67  
11/18/2022, 5:23:17 PM node: ace51a660b648914  
iot-2/type/nodefarmerid/2002/ev/IoTSensor/fmt/json :  
msg.payload : Object  
  { temp: 21, Humid: 56 }  
11/18/2022, 5:23:17 PM node: ace51a660b648914  
iot-2/type/nodefarmerid/2002/ev/IoTSensor/fmt/json :  
msg.payload : number  
21  
11/18/2022, 5:23:17 PM node: ace51a660b648914  
iot-2/type/nodefarmerid/2002/ev/IoTSensor/fmt/json :  
msg.payload : number
```