HO CHI MINH CITY, UNIVERSITY OF TECHNOLOGY
DEPARTMENT OF COMPUTER SCIENCE AND ENGINEER


# Application Based Internet of Things Report

*Student:* Nguyễn Công Thành - 1915144
Đặng Quốc Thắng - 1912084
Tạ Quang Việt - 1915916

HỒ CHÍ MINH CITY

# Content

**1 Introduction** **2**

**2 Requirements** **2**

2.1 Thingsboard server . . . . . 2

2.2 Python gateway . . . . . 2

2.3 Microbit sensor . . . . . 3

**3 Report** **3**

3.1 Thingsboard dashboard . . . . . 3

3.2 Python source code . . . . . 5

3.2.1 Publish data to Thingsboard . . . . . 5

3.2.2 Command detector from 2 buttons . . . . . 6

3.2.3 Github link . . . . . 8

**4 Microbit sensor** **9**

# 1 Introduction

In this second LAB, students are proposed to develop a simple application having the sensory data shown in a Thingsboard dashboard and a 2 buttons to illustrate the controling functions.

The Microbit board is used in this LAB, connected to your PC via USB connector. In the microbit, 2 sensors can be used in this LAB, including the **TEMPERATURE** and the **LIGHT** sensor.

Every source code of this LAB is also required to publish in your Github. The details are described in the next section of this report.

# 2 Requirements

## 2.1 Thingsboard server

In order to present the sensory data, two UIs are required:

- A graph to display the sensory data
- A label or a Gauge view to display the current data

Students can create 2 different graphs for this requirement. However, one graph to display 2 different sensory data is also allowed.

Finally, two different buttons, labled **LED** and **FAN** are also required in the dashboard.

## 2.2 Python gateway

A python source code, running on your PC to receive the sensory data and publish to the server. Moreover, a command when a button is clicked, is needed to send to the Microbit platform.

A skeleton of the source code is published in this link:

`https://github.com/npnlab-vn/code-manager/blob/ThingsBoard_Lab3/IoT_Lab.py`

Following package are required to install for this source code:

- pip install paho-client
- pip install pyserial

Students are proposed to change the **THINGS_BOARD_ACCESS_TOKEN** and the name of the COM port (line 13 and 14), in the case it is connected to the Microbit.

### 2.3 Microbit sensor

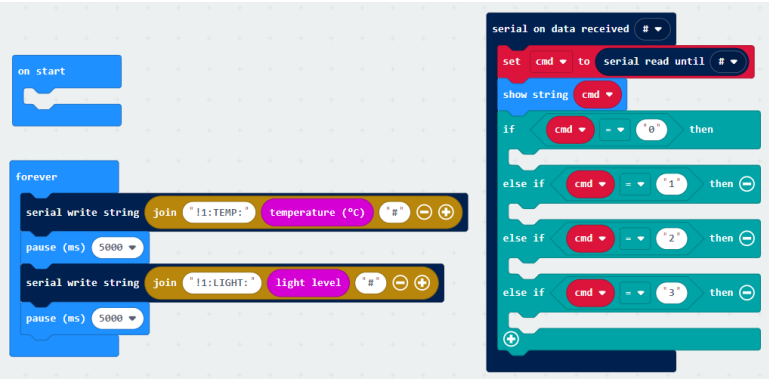
The temperature value of the microbit is sent to the gateway first. After that 5 seconds, the light sensor value is sent. The command is received at the microbit through serial connection. A refference source code can be found at:

`https://makecode.microbit.org/_Cj3JUFXij6bu`

In the case there is an error when downloading the hex file to the microbit platform, following steps are required to fix the error:

- Step 1: Create a new project in MakeCode
- Step 2: Switch to JavaScript view
- Step 3: From the refference source code, switch to JavaScript view as well
- Step 4: Copy the java script source code from the refference link to your project
- Step 5: Switch to Block view

If the error can not be fixed, students can implement the source code from the beginning. The image of the source code is depicted bellow:




Hình 1: *A simple program in BBC Microbit platform*

## 3 Report

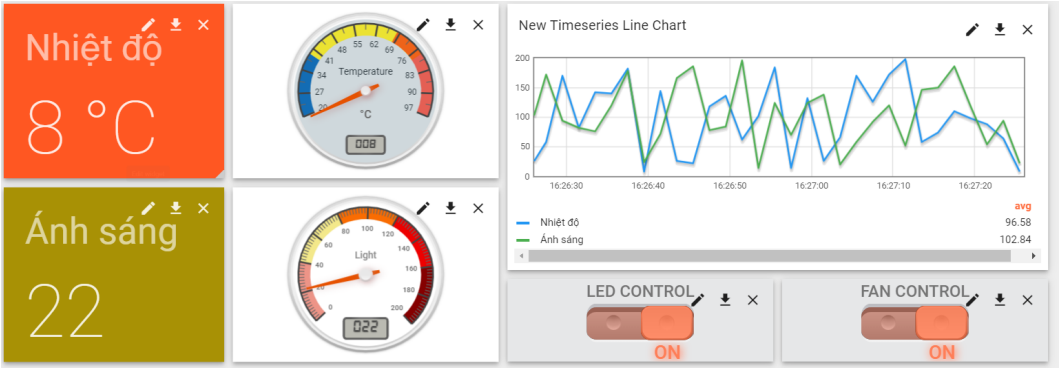
### 3.1 Thingsboard dashboard

A picture as well as the public link your your dashboard are required to present in this report.

In the case an UI is import from third party library (e.g. from github), a brief description to use this UI can be provided here. **These advance UIs can be considered as an extra point for this lab**.

**Nguyễn Công Thành**

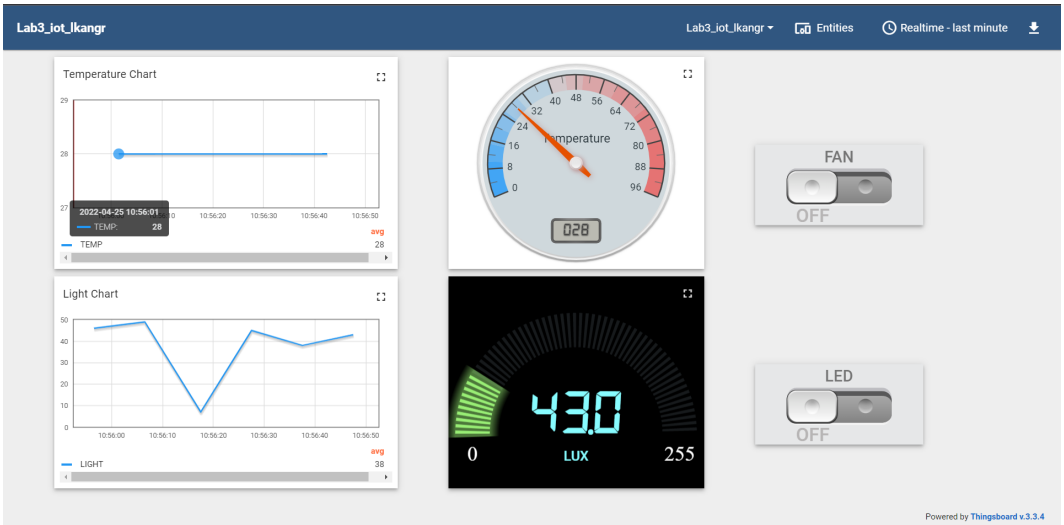
**Link Thingsboard:** https://bit.ly/ThingsboardIoT




Hình 2: *Dashboard - Thành*

**Đặng Quốc Thắng**

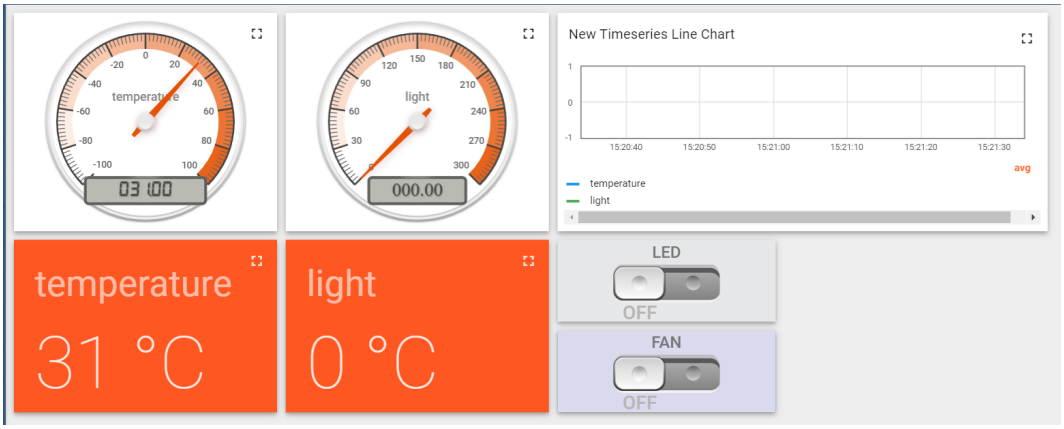
**Link Thingsboard:** https://demo.thingsboard.io/dashboard/1aaafe40-beb3-11ec-942a-51543ec1dbaf?publicId=15496020-6d34-11ec-8159-03103585248e




Hình 3: *Dashboard - Thắng*

**Tạ Quang Việt**

**Link Thingsboard:** https://demo.thingsboard.io/dashboard/7bd61c90-ba12-11ec-9a68-6b50da95566e?publicId=fe4a2170-721a-11ec-9a90-af0223be0666




Hình 4: *Dashboard - Việt*

## 3.2 Python source code

Students are proposed to present some part of the source code, as listed following.

### 3.2.1 Publish data to Thingsboard

In order to publish the data to the **Thingsboard server**, students have to add more source code to the **processData** function. Please fill your implementation in this report.

**Nguyễn Công Thành**

Listing 1: Publishing data to Thingsboard

```
1  def processData(data):
2      data = data.replace("!", "")
3      data = data.replace("#", "")
4      splitData = data.split(":")
5      print(splitData)
6      #TODO: Add your source code to publish data to Thingsboard
7      # Example demo hercules: !1:TEMP:35# !2:LIGHT:33##
8      if splitData[1] == "TEMP":
9          splitData[1] = "temperature"
10     elif splitData[1] == "LIGHT":
11         splitData[1] = "light"
12     collect_data = {splitData[1]: int(splitData[2])}
13     client.publish("v1/devices/me/telemetry", json.dumps(collect_data)↩
           , 1)
```

**Đặng Quốc Thắng**

Listing 2: Publishing data to Thingsboard - Thắng

```
def processData(data):
    data = data.replace("!", "")
    data = data.replace("#", "")
    splitData = data.split(":")
    print(splitData)
    #TODO: Add your source code to publish data to Thingsboard
    collect_data = {splitData[1]: splitData[2]}
    client.publish('v1/devices/me/telemetry', json.dumps(collect_data)
        , 1)
```

**Tạ Quang Việt**

Listing 3: Publishing data to Thingsboard - Việt

```
def processData(data):
    data = data.replace("!", "")
    data = data.replace("#", "")
    splitData = data.split(":")
    print(splitData)
    # TODO: Add your source code to publish data to the server
    if splitData[1] == "TEMP":
        splitData[1] = "temperature"
    elif splitData[1] == "LIGHT":
        splitData[1] = "light"
    collect_data = {splitData[1]: int(splitData[2])}
    client.publish("v1/devices/me/telemetry", json.dumps(collect_data)
        , 1)
```

### 3.2.2 Command detector from 2 buttons

In this part, the python gateway needs to send 4 different values to the microbit. Please add your implementation to finallize this part.

**Nguyễn Công Thành**

Listing 4: Command processing at the gateway

```

def recv_message(client, userdata, message):
    print("Received: ", message.payload.decode("utf-8"))
    temp_data = {"value": True}
```

```
    cmd = 0
    # TODO: Update the cmd to control 2 devices
    try:
        jsonobj = json.loads(message.payload)
        if jsonobj["method"] == "setLED":
            temp_data["value"] = jsonobj["params"]
            client.publish("v1/devices/me/LED_SWITCH", json.dumps(↩
                temp_data), 1)
            cmd = 1 if jsonobj["params"] == True else 0
        if jsonobj["method"] == "setFAN":
            temp_data["value"] = jsonobj["params"]
            client.publish("v1/devices/me/FAN_SWITCH", json.dumps(↩
                temp_data), 1)
            cmd = 3 if jsonobj["params"] == True else 2
    except:
        pass

    if len(bbc_port) > 0:
        ser.write((str(cmd) + "#").encode())
```

**Đặng Quốc Thắng**

Listing 5: Command processing at the gateway - Thắng

```
def recv_message(client, userdata, message):
    print("Received: ", message.payload.decode("utf-8"))
    temp_data = {'value': True}
    cmd = 1
    #TODO: Update the cmd to control 2 devices
    try:
        jsonobj = json.loads(message.payload)
        if jsonobj['method'] == "setLED":
            temp_data['valueLED'] = jsonobj['params']
            client.publish('v1/devices/me/attributes',json.dumps(↩
                temp_data), 1)
            cmd = 1 if jsonobj['params'] == True else 0
        if jsonobj['method'] == "setFAN":
            temp_data['valueFAN'] = jsonobj['params']
            client.publish('v1/devices/me/attributes',json.dumps(↩
                temp_data), 1)
            cmd = 3 if jsonobj['params'] == True else 2
    except:
        pass

    if len(bbc_port) > 0:
```

```
20          ser.write((str(cmd) + "#").encode())
```

**Tạ Quang Việt**

Listing 6: Command processing at the gateway - Việt

```
1  def recv_message(client, userdata, message):
2      print("Received: ", message.payload.decode("utf-8"))
3      temp_data = {'value': True}
4      cmd = 0
5      #TODO: Update the cmd to control 2 devices
6      try:
7          jsonobj = json.loads(message.payload)
8          if jsonobj['method'] == "setLED":
9              temp_data['value'] = jsonobj['params']
10             client.publish('v1/devices/me/LED', json.dumps(temp_data),↩
                   1)
11             if(temp_data['value'] == True):
12                 cmd =1
13             else:
14                 cmd = 0
15         if jsonobj['method'] == "setFAN":
16             temp_data['value'] = jsonobj['params']
17             client.publish('v1/devices/me/FAN', json.dumps(temp_data),↩
                   1)
18             if(temp_data['value'] == True):
19                 cmd = 3
20             else:
21                 cmd = 2
22     except:
23         pass
24
25     if len(bbc_port) > 0:
26         ser.write((str(cmd) + "#").encode())
```

Your source code is mostly in the **try except** block. The **cmd** variable is required to update here.

### 3.2.3 Github link

**Link Github:**

Nguyễn Công Thành: https://github.com/nct74/IoT_Lab/tree/IOT_Lab3

Đặng Quốc Thắng: https://github.com/lkangr/lab3-iot-lkangr.git

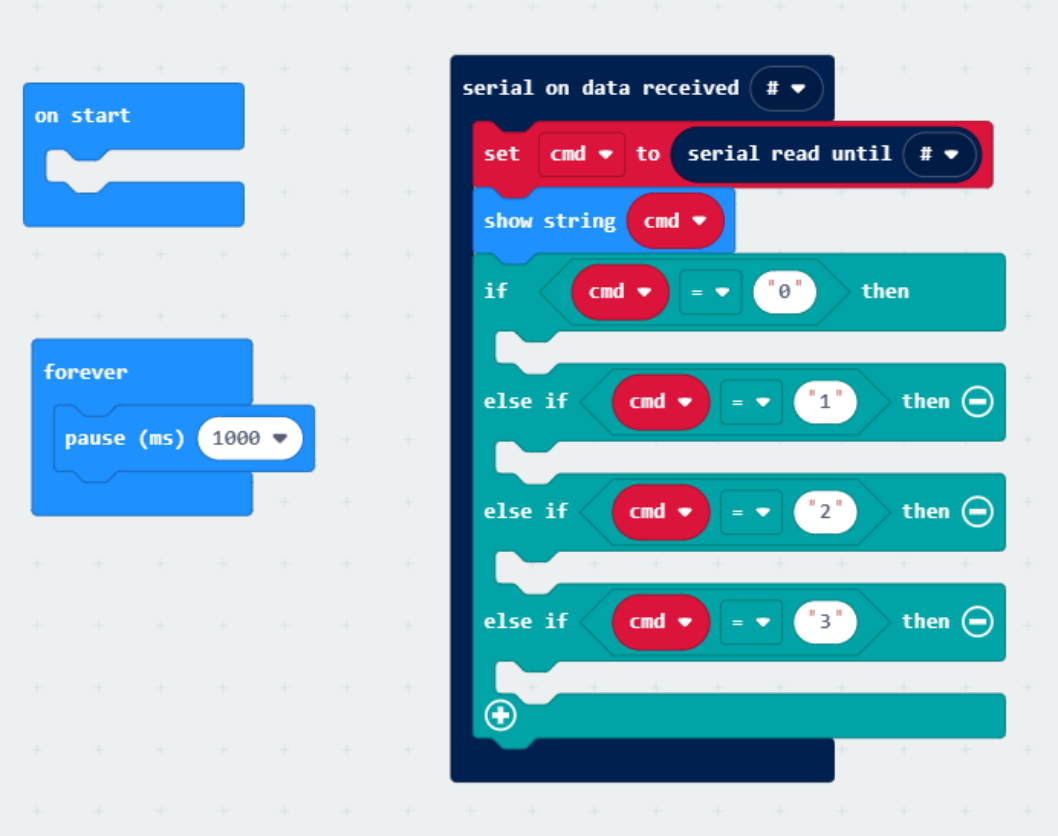
Tạ Quang Việt: https://github.com/viet-123/IoT_Lab3.git

# 4 Microbit sensor

Finallize your source code and display 4 different images (using show icons block, in the Basic group). Then, capture your source code in MakeCode and present it in this report.

An extra point for this part which is described as follows:

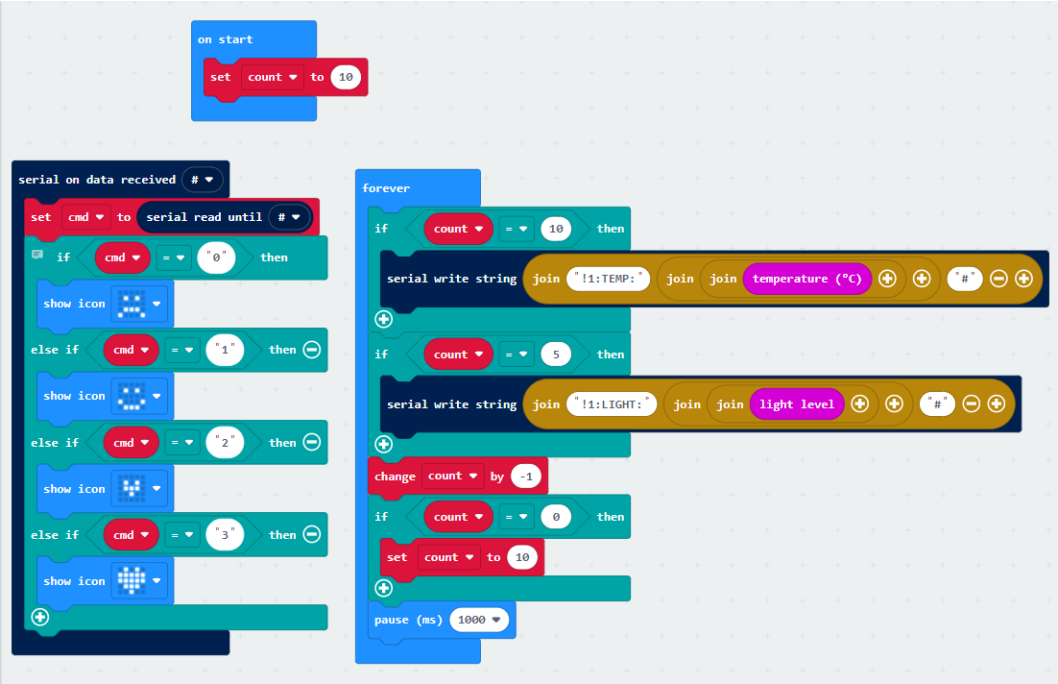
- Modify the source code in forever, which is used to send sensory data to the gateway.
- There is only one delay instruction (pause(1000)) at the end of the forever
- The temperature is sent first. After that 5 seconds, the light is sent. This process is repeated again after 5 seconds




Hình 5: *Structure of the program in Microbit*

The picture of your source code is also required to present in this report.

**Nguyễn Công Thành**

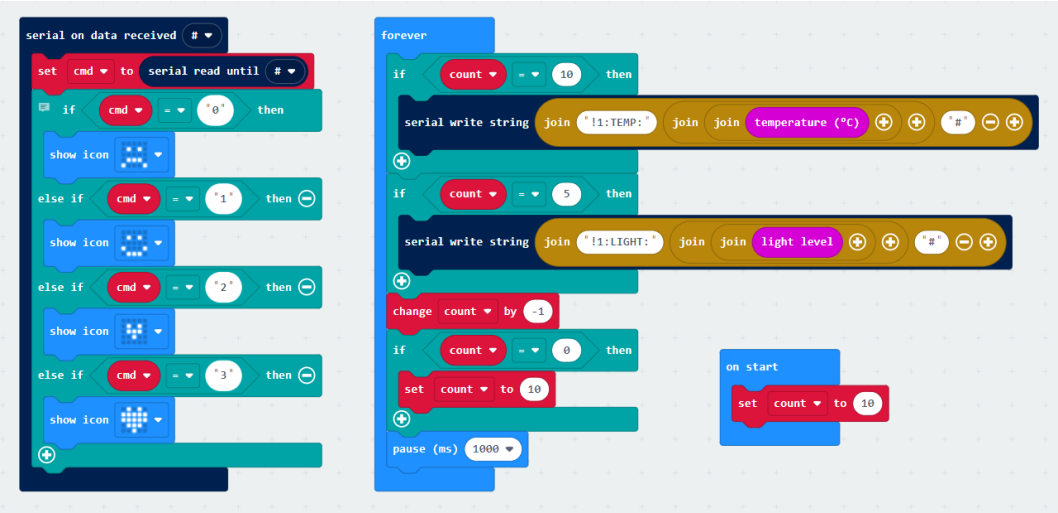



Hình 6: *Structure of the program in Microbit*

```python
def on_data_received():
    global cmd
    cmd = serial.read_until(serial.delimiters(Delimiters.HASH))
    # basic.show_string(cmd)
    if cmd == "0":
        basic.show_icon(IconNames.SAD)
    elif cmd == "1":
        basic.show_icon(IconNames.HAPPY)
    elif cmd == "2":
        basic.show_icon(IconNames.SMALL_HEART)
    elif cmd == "3":
        basic.show_icon(IconNames.HEART)
serial.on_data_received(serial.delimiters(Delimiters.HASH), on_data_received)

cmd = ""
count = 10

def on_forever():
    global count
    if count == 10:
        serial.write_string("!1:TEMP:" + ("" + str(input.temperature())) + "#")
    if count == 5:
        serial.write_string("!1:LIGHT:" + ("" + str(input.light_level())) + "#")
    count += -1
    if count == 0:
        count = 10
    basic.pause(1000)
basic.forever(on_forever)

```

Hình 7: *Structure of the program in Microbit*

**Đặng Quốc Thắng**

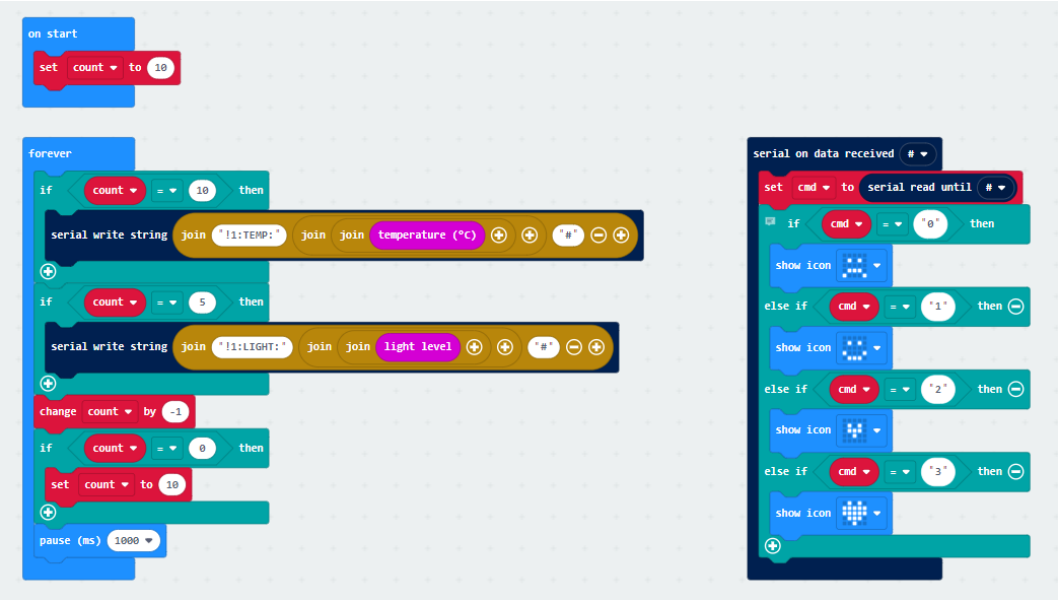



Hình 8: *Structure of the program in Microbit - Thắng*

```
def on_data_received():
    global cmd
    cmd = serial.read_until(serial.delimiters(Delimiters.HASH))
    # basic.show_string(cmd)
    if cmd == "0":
        basic.show_icon(IconNames.SAD)
    elif cmd == "1":
        basic.show_icon(IconNames.HAPPY)
    elif cmd == "2":
        basic.show_icon(IconNames.SMALL_HEART)
    elif cmd == "3":
        basic.show_icon(IconNames.HEART)
serial.on_data_received(serial.delimiters(Delimiters.HASH), on_data_received)

cmd = ""
count = 10

def on_forever():
    global count
    if count == 10:
        serial.write_string("!1:TEMP:" + ("" + str(input.temperature())) + "#")
    if count == 5:
        serial.write_string("!1:LIGHT:" + ("" + str(input.light_level())) + "#")
    count += -1
    if count == 0:
        count = 10
    basic.pause(1000)
basic.forever(on_forever)
```

Hình 9: *Structure of the program in Microbit - Thắng*

**Tạ Quang Việt**




Hình 10: *Structure of the program in Microbit - Việt*

```python
def on_data_received():
    global cmd
    cmd = serial.read_until(serial.delimiters(Delimiters.HASH))
    # basic.show_string(cmd)
    if cmd == "0":
        basic.show_icon(IconNames.SAD)
    elif cmd == "1":
        basic.show_icon(IconNames.HAPPY)
    elif cmd == "2":
        basic.show_icon(IconNames.SMALL_HEART)
    elif cmd == "3":
        basic.show_icon(IconNames.HEART)
serial.on_data_received(serial.delimiters(Delimiters.HASH), on_data_received)

cmd = ""
count = 10

def on_forever():
    global count
    if count == 10:
        serial.write_string("!1:TEMP:" + ("" + str(input.temperature())) + "#")
    if count == 5:
        serial.write_string("!1:LIGHT:" + ("" + str(input.light_level())) + "#")
    count += -1
    if count == 0:
        count = 10
    basic.pause(1000)
basic.forever(on_forever)
```

Hình 11: *Structure of the program in Microbit - Việt*