Smarthon Documentation

Release 1.0

Smarthon Limited

Mar 08, 2023

CONTENTS

CHAPTER

ONE

QUICK START GUIDE

1.1 Smarthon Smart City IoT Starter Kit

1.1.1 Introduction

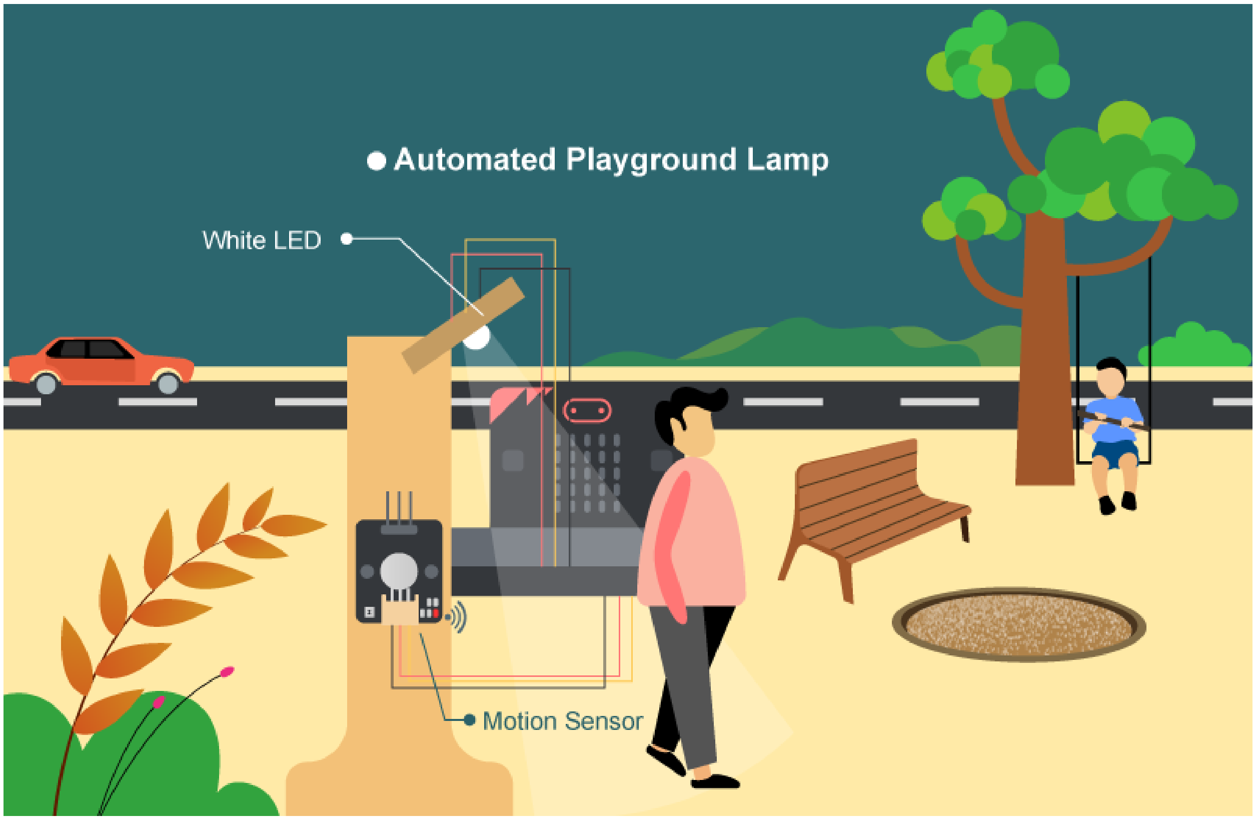
Introduction

Smarthon Smart City IoT Starter kit is designed to introduce Internet of things (IoT). With basic knowledge of computing knowledge and electronics provided in the kit, you can be a city creator and build a unique IoT system in the city. Based on Smarthon IoT board, which is compatible with multiple sensors and actuators, you can design city features; for example: using sensors to detect traffic status and upload city information to the internet.

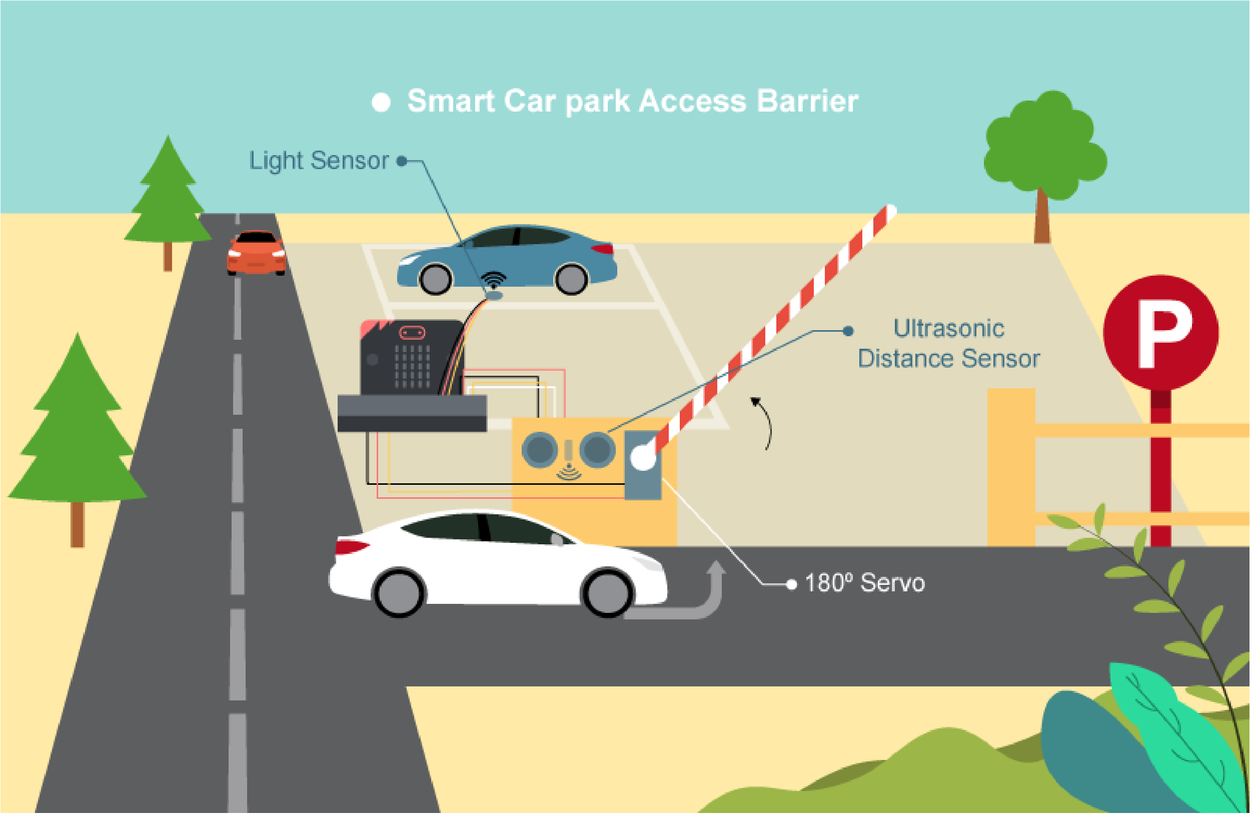
SMARTHON
SMART CITY IOT
STARTER KIT
FOR MICRO:BIT
• CREATE YOUR OWN SMART CITY MODEL
• LEARN TO MAKE 11 IOT PROJECTS OUT OF THE BOX
• MOST POWERFUL ESP32 IOT EXTENSION BOARD
micro:bit
*MICROBIT NON INCLUDED
10 +
AGES
MODEL INCLUDED

Application Scene

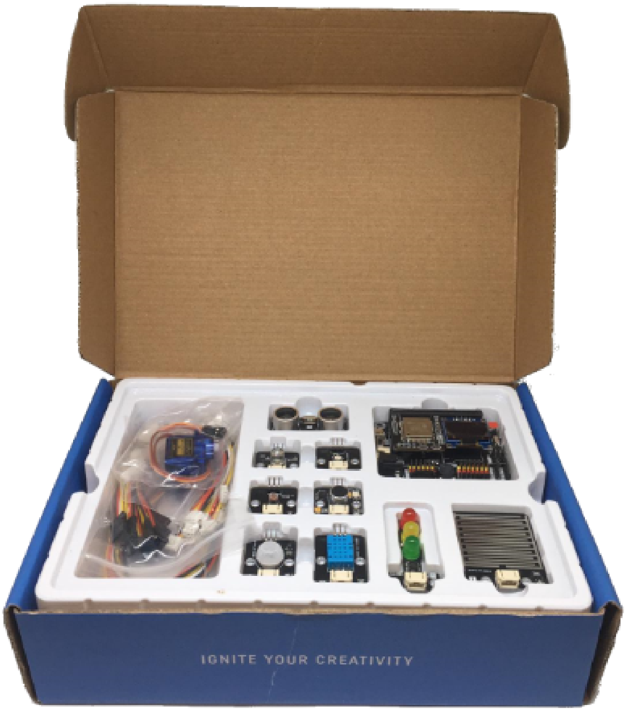
Automated Playground Lamp



Smart Car park Access Barrier



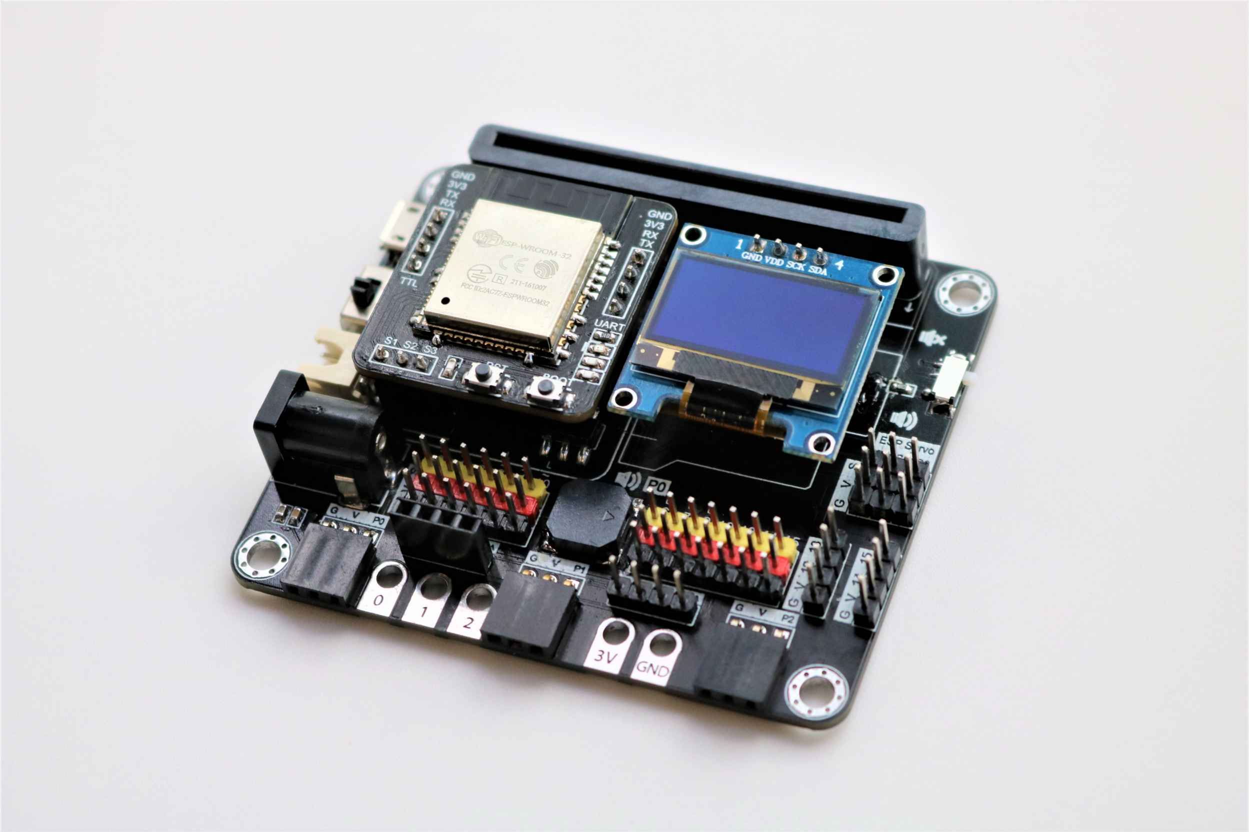
Component List



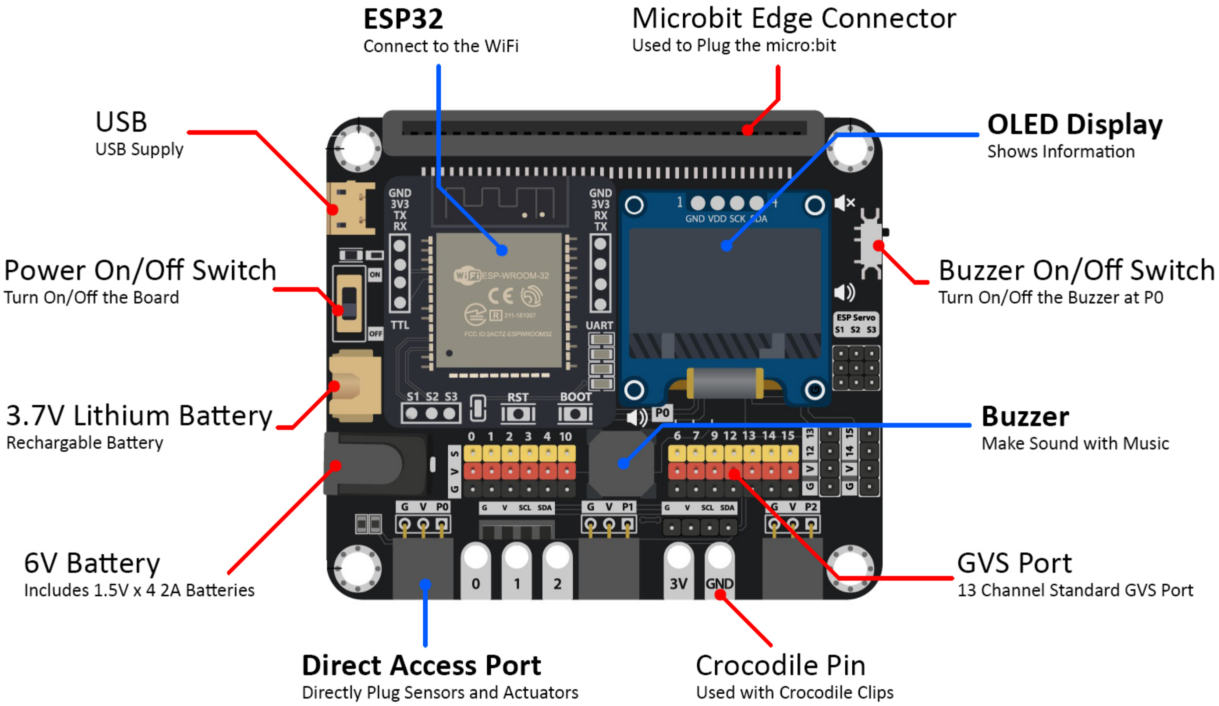
1.1.2 IoT:bit Introduction

Introduction

Smarthon IoT Bit is an IoT orientated expansion board for Micro:bit. With the help of this, Micro:bit able to connect to different sensors or actuators without welding and start to use the internet services.


GND
3V3
TX
RX
TTL
GND
3V3
RX
TX
UART
S1 S2 S3
1
GND VDD SCK SDA
4
P0
G V P0
0
1
2
G V P1
3V
GND
G V P2

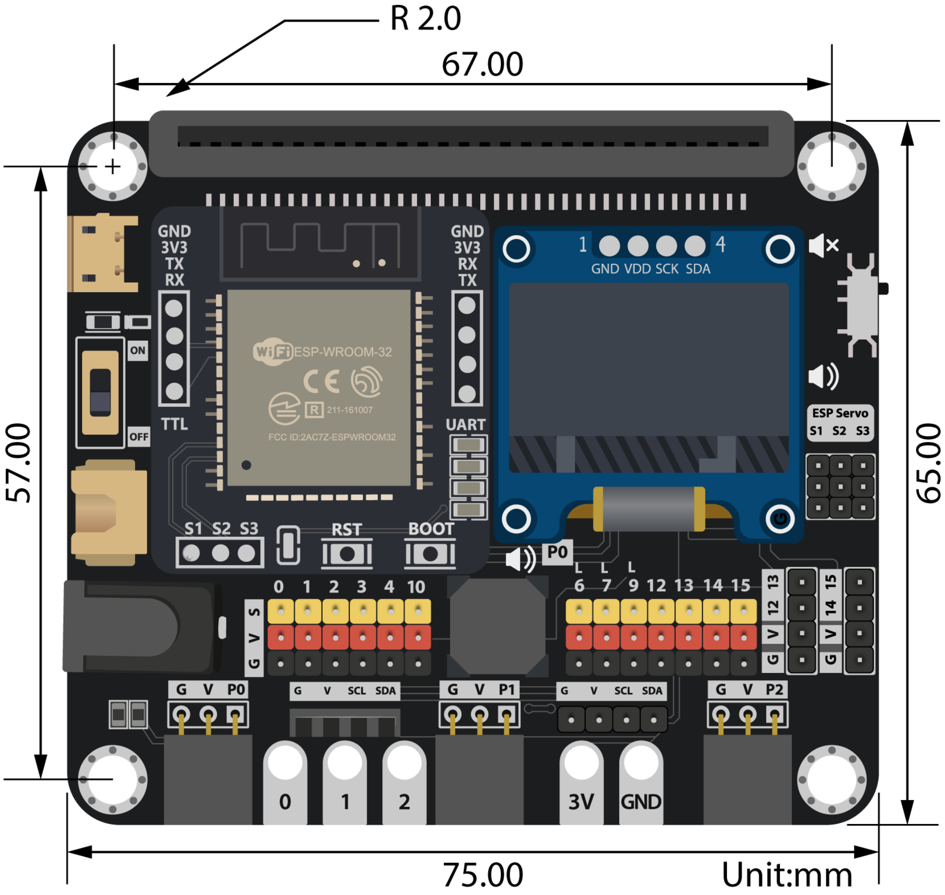
Product Features



- The Most powerful IoT chip ESP32 embedded for micro:bit
- Exchangeable wireless modular design (ESP8266, ESP32, BLE)
- Integrated OLED and Onboard buzzer embedded
- Extended I/O port x13, I2C port x2 from Micro:bit with GVS socket
- Extended Servo port x3 from ESP module
- Various types of power supply (USB, 3.7V lithium battery, 6V Normal battery)
- Extra Port: Crocodile Pin, Quick Access Port and 4-Pin Port for different usage

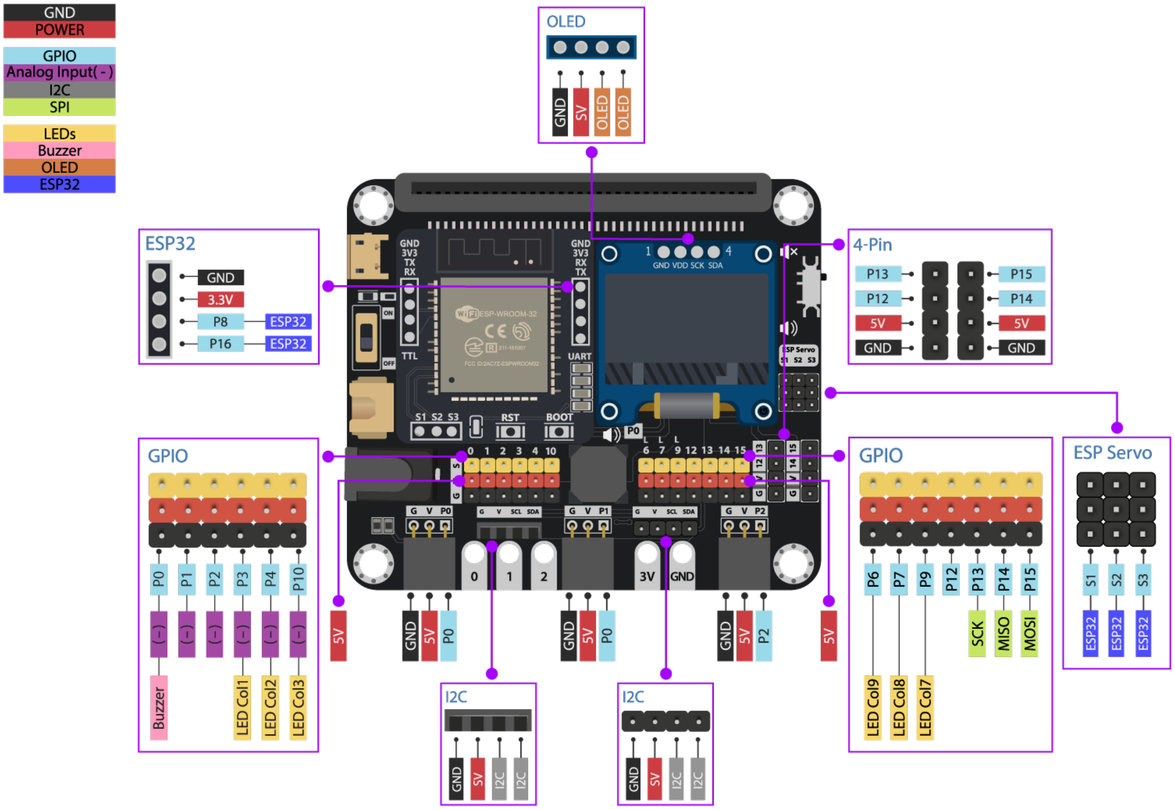
Hardware Size

Product size: 75mm X 65mm X 20mm PCB thinkness: 1.5mm Hole diameter: 4mm

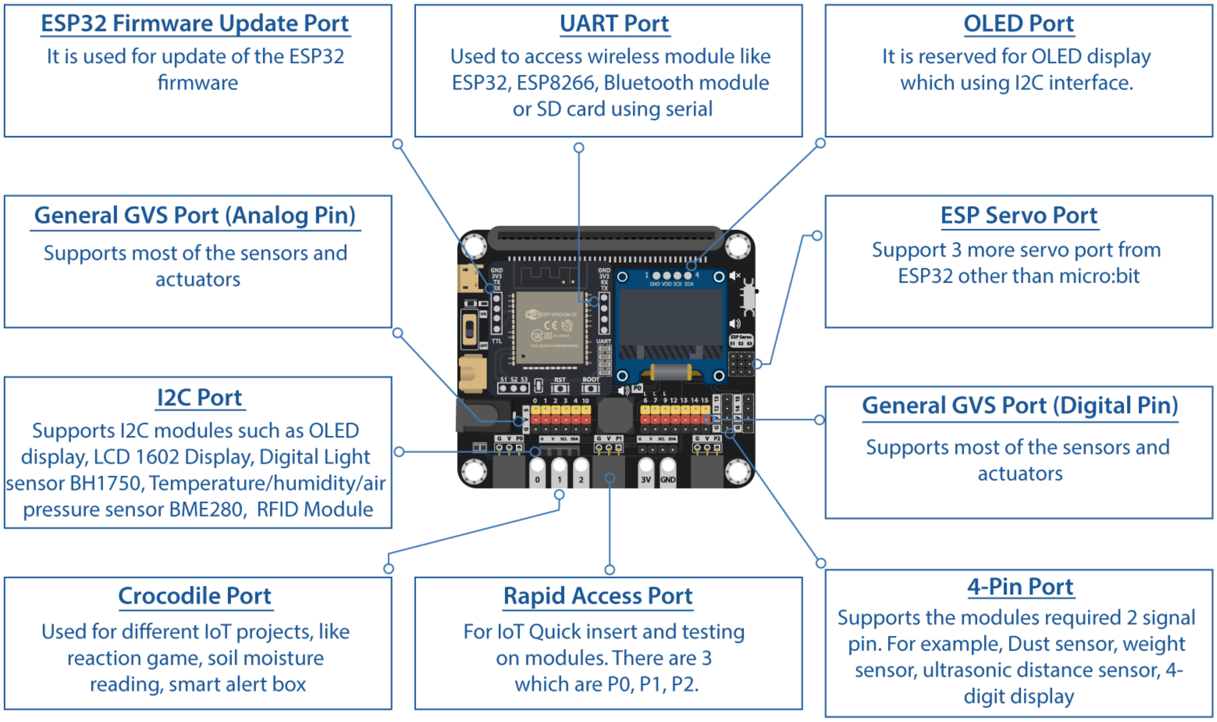


Specification

Pinout Information



Pinout Diagram:



ESP32 Firmware Update Port

It is used for update of the ESP32 firmware

UART Port

Used to access wireless module like ESP32, ESP8266, Bluetooth module or SD card using serial

OLED Port

It is reserved for OLED display which using I2C interface.

General GVS Port (Analog Pin)

Supports most of the sensors and actuators

ESP Servo Port

Support 3 more servo port from ESP32 other than micro:bit

I2C Port

Supports I2C modules such as OLED display, LCD 1602 Display, Digital Light sensor BH1750, Temperature/humidity/air pressure sensor BME280, RFID Module

General GVS Port (Digital Pin)

Supports most of the sensors and actuators

Crocodile Port

Used for different IoT projects, like reaction game, soil moisture reading, smart alert box

Rapid Access Port

For IoT Quick insert and testing on modules. There are 3 which are P0, P1, P2.

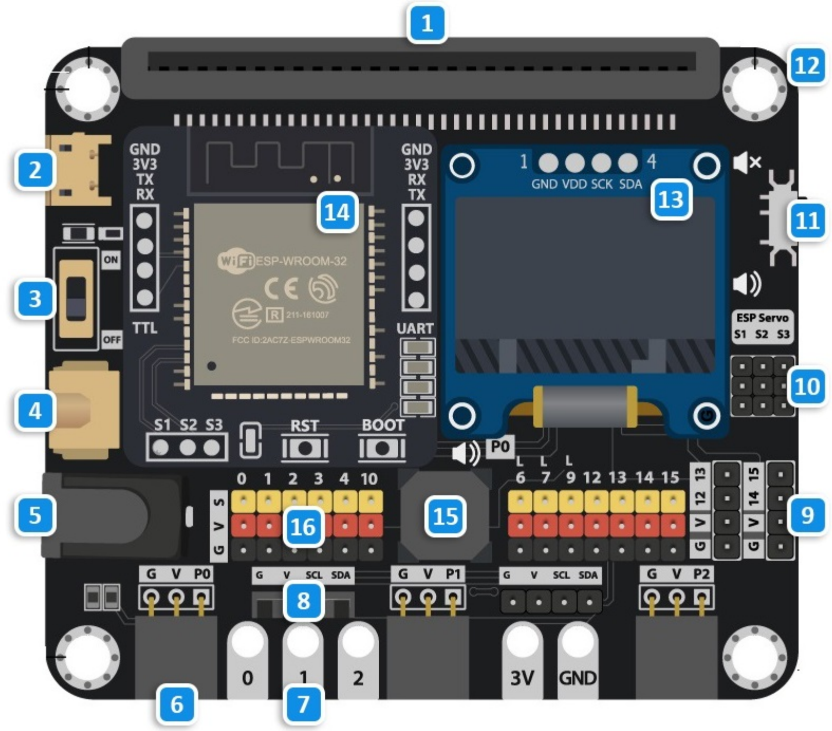
4-Pin Port

Supports the modules required 2 signal pin. For example, Dust sensor, weight sensor, ultrasonic distance sensor, 4-digit display

Port Diagram:

.tg {border-collapse:collapse;border-spacing:0;} .tg td{border-color:black;border-style:solid;border-width:1px;font-family:Arial, sans-serif;font-size:14px; overflow:hidden;padding:10px 5px;word-break:normal;} .tg th{border-color:black;border-style:solid;border-width:1px;font-family:Arial, sans-serif;font-size:14px; font-weight:normal;overflow:hidden;padding:10px 5px;word-break:normal;} .tg .tg-ux7d{background-color:#c0c0c0;border-color:#000000;font-family:Arial, Helvetica, sans-serif !important;;text-align:center; vertical-align:middle} .tg .tg-l89d{border-color:#000000;font-family:Arial, Helvetica, sans-serif !important;;text-align:center;vertical-align:middle} .tg .tg-j1bp{background-color:#efefef;border-color:#000000;font-family:Arial, Helvetica, sans-serif !important;;text-align:center; vertical-align:middle}

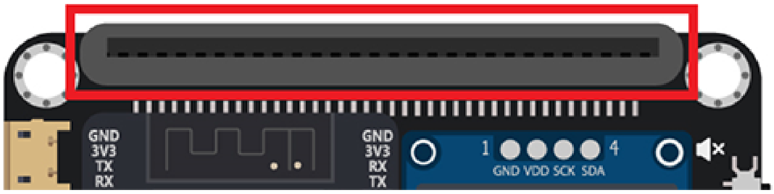
Hardware Interface



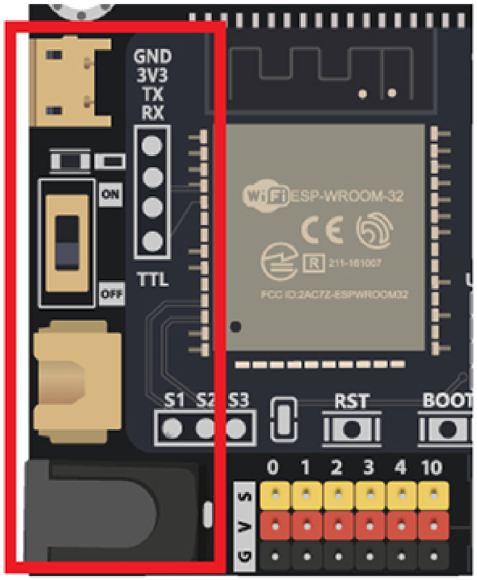
1. Micro:bit edge connecter
2. USB Port
3. Power Switch
4. 3.7V Lithium battery Port [XH2.54]
5. 6V 4*AA(1.5V) battery Port [DC 5.5mm]
6. Rapid Access Port
7. Crocodile Pin
8. I2C Port
9. 4-Pins Port
10. ESP Servo Port
11. Buzzer switch
12. 4mm Hole

13. OLED Module (128*64 pixel)
14. ESP32 Wireless Module
15. Buzzer
16. General GVS Port

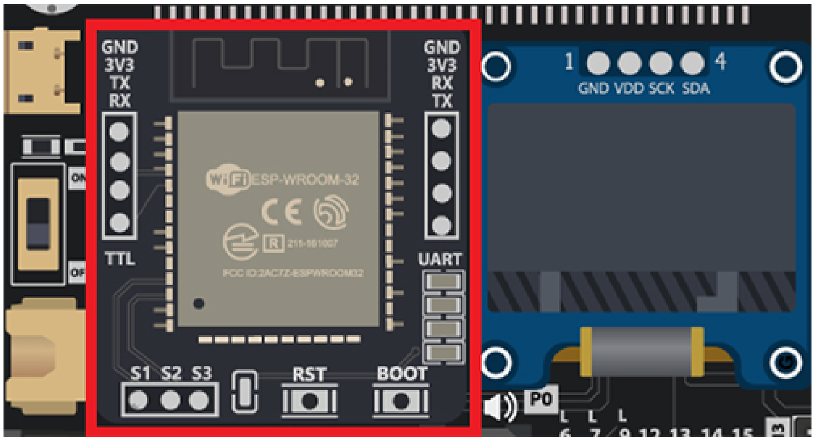
Detail Hardware Description



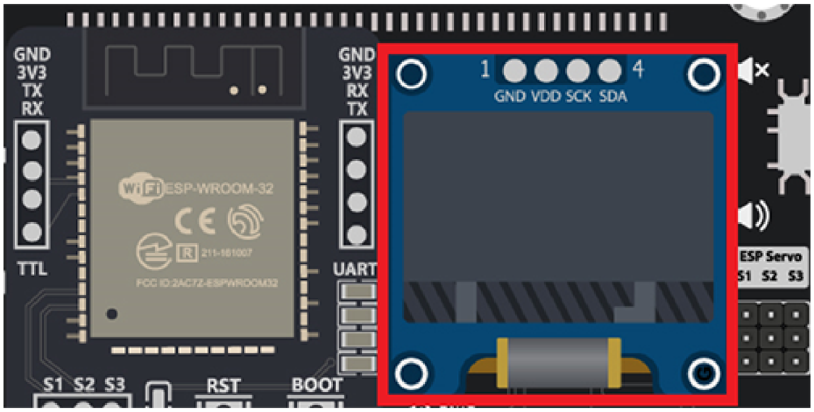
Micro:bit edge connecter

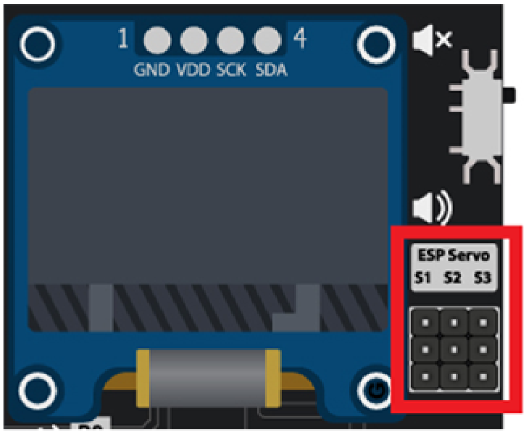


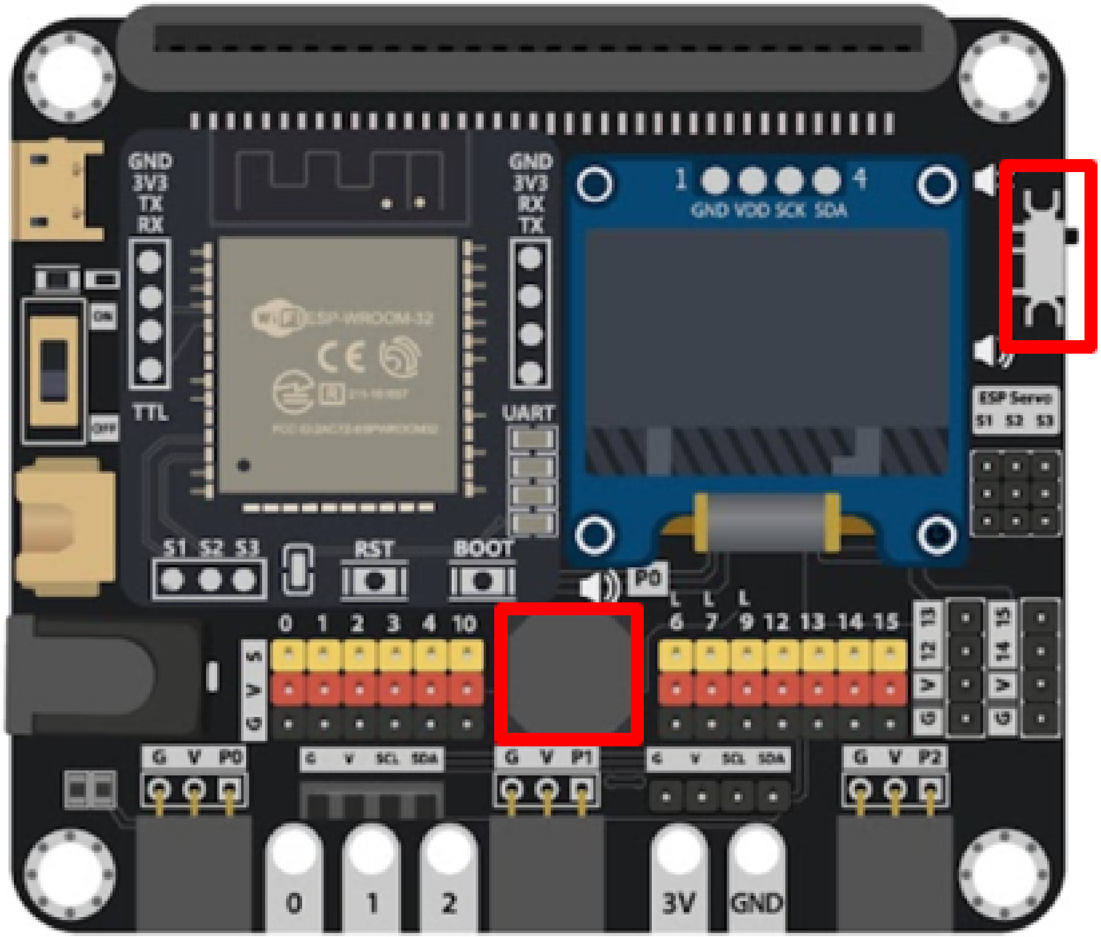
Power port(USB, Battery, DC) & Switch



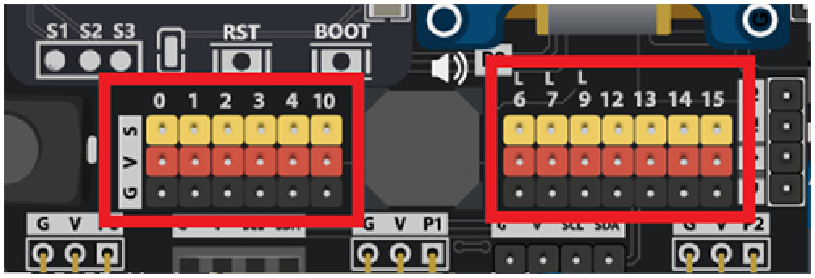
Wireless Module

OLED Module 

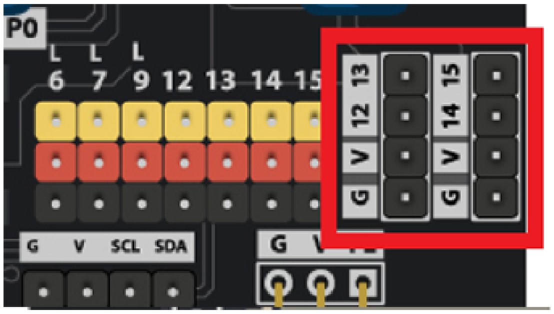
ESP Servo port 



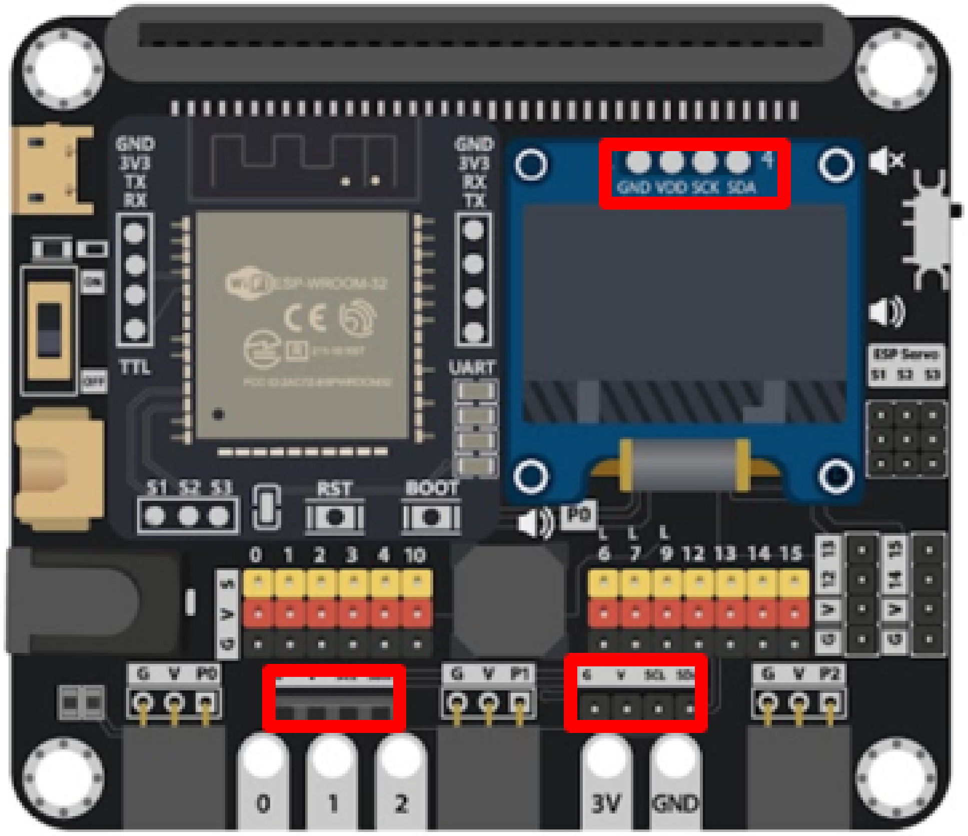
Buzzer & Switch



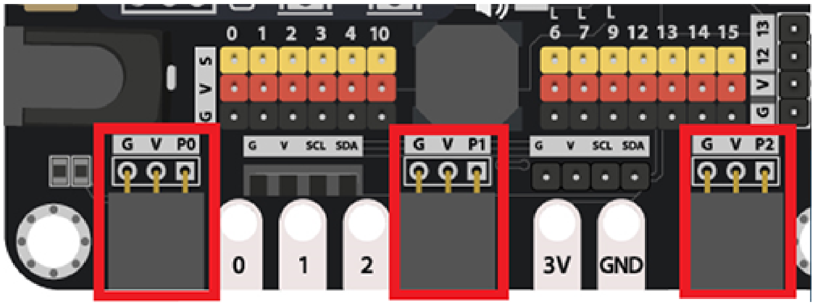
Micro:bit lead out Pins



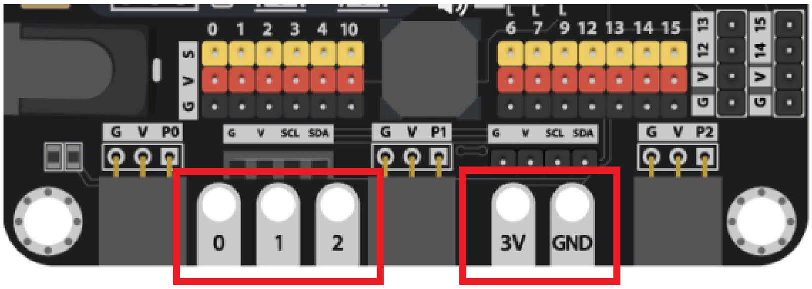
4-Pins Module Port



I2C Port



Rapid Access Port

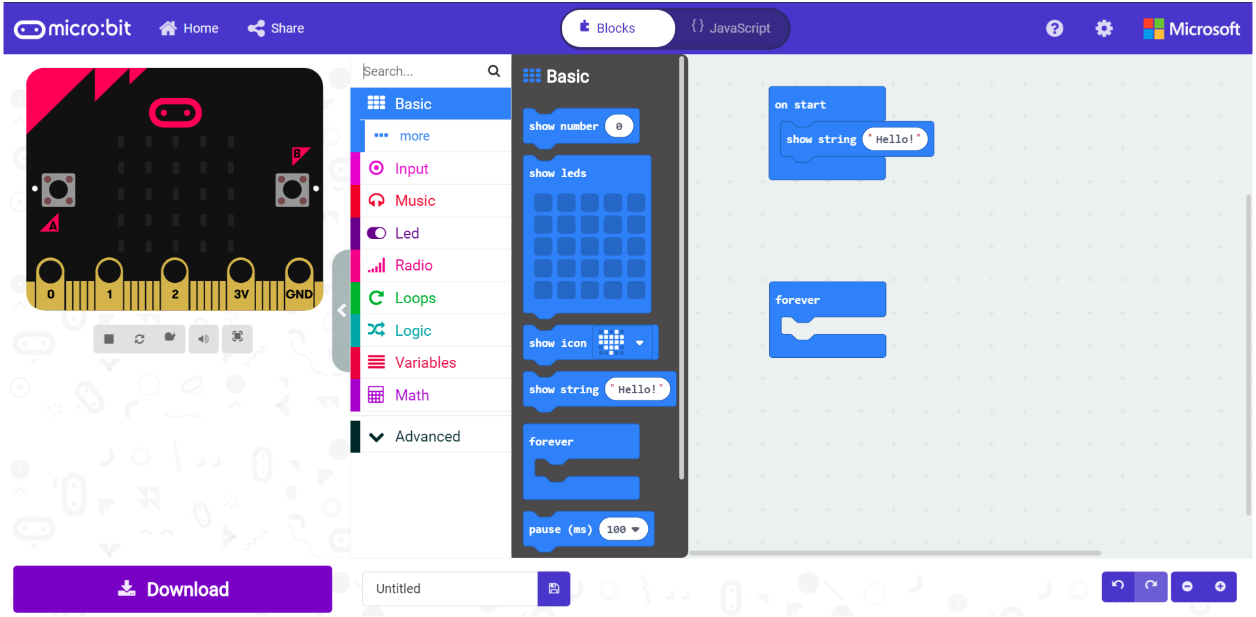


Crocodile Pin

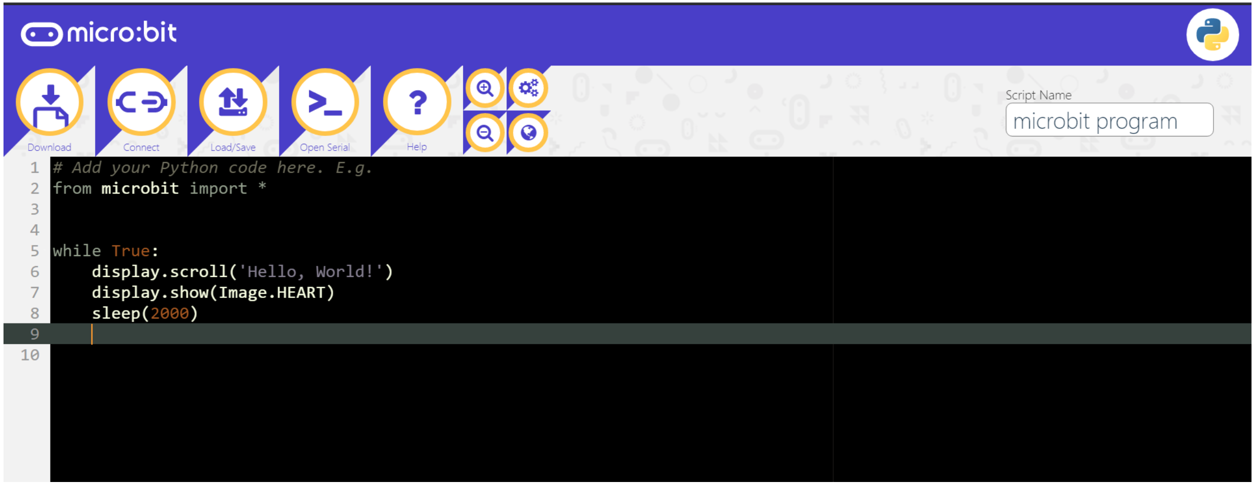
Software Support

MakeCode editor

The MakeCode editor is the official cross-platform editor designed for BBC Micro:bit, it is available on Web Browser, Mobile and tablets Apps. The Colorful coding block is familiar to who previously used Scratch, providing a simple and clear way to programming on the Micro:bit. The editor offer a text-based mode for user to programming using JavaScript rather than drag the blocks MakeCode



Python editor Python is one of the most famous programming language, widely used in school and industry. Micro:bit provided an online python editor for user to study and exercise python on Micro:bit. User just need to connect Micro:bit with USB and click the Flash button, the compile and upload progress will be finish automatically. Python editor



```
# Add your Python code here. E.g.
from microbit import *


while True:
    display.scroll('Hello, World!')
    display.show(Image.HEART)
    sleep(2000)

```

Quick Start

1.Plug the Micro:bit(with Program) into the the slot

2.Get the power supply and turn on

Method 1:

Plug the USB cable into the USB port and turn on the switch

Method 2:

Plug the battery case(1.5V AA*4) into the battery port and turn on the switch

Method 3:

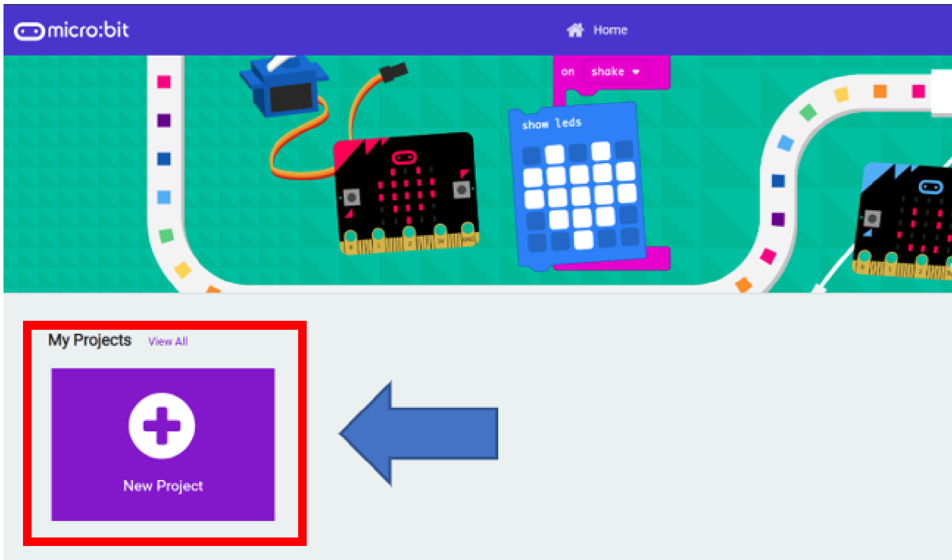
Plug the lithium battery into the lithium battery port and turn on the switch

*18650 battery holder and the battery are not included in the kit set

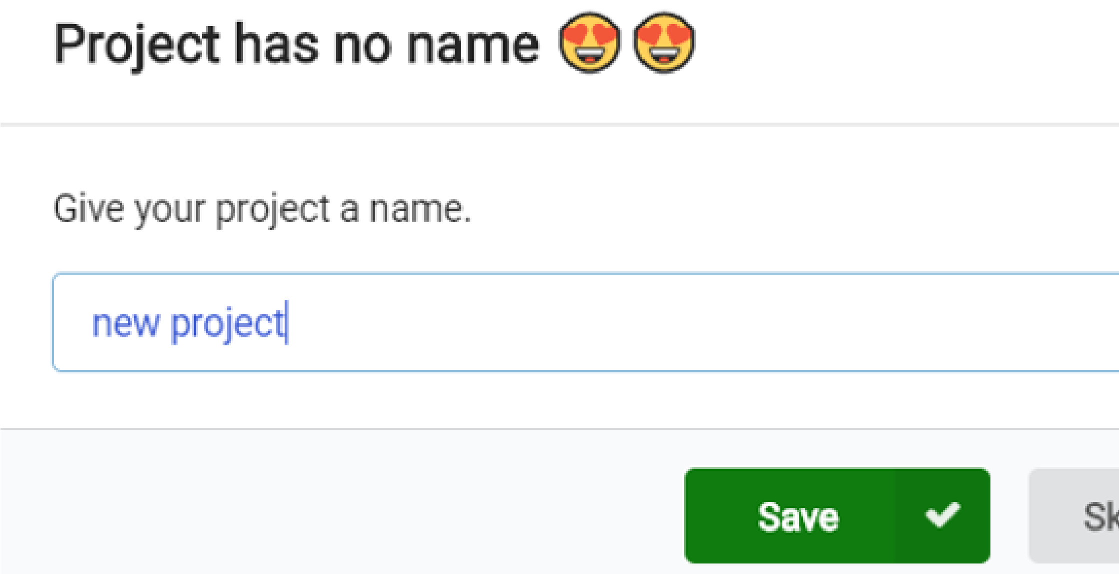
Best Practice

1. Load the program into the micro:bit (Micro:bit not connected to iot:bit)
2. Take the USB out from micro:bit
3. Insert the micro:bit to iot:bit
4. Switch on the iot:bit

1.1.3 Getting started: Add the extension



Step 1 Go to MakeCode, create a new project

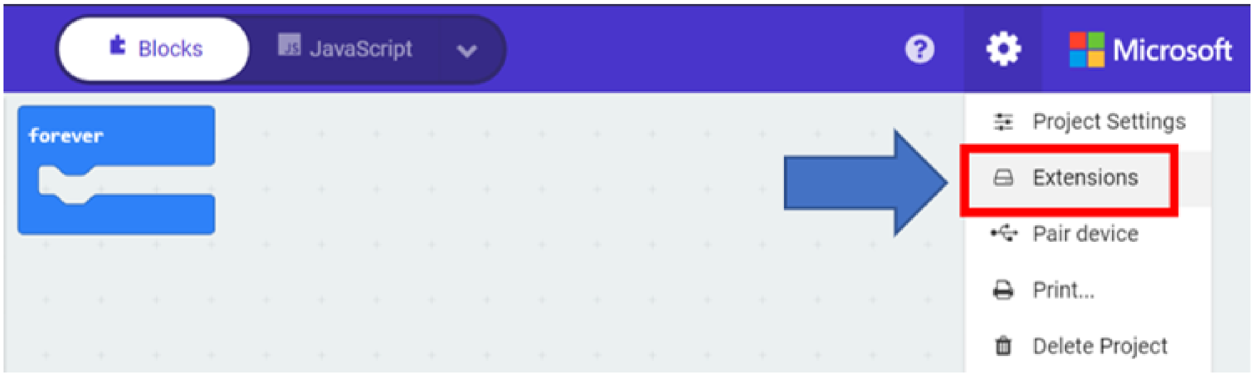

Project has no name 😍😍

Give your project a name.

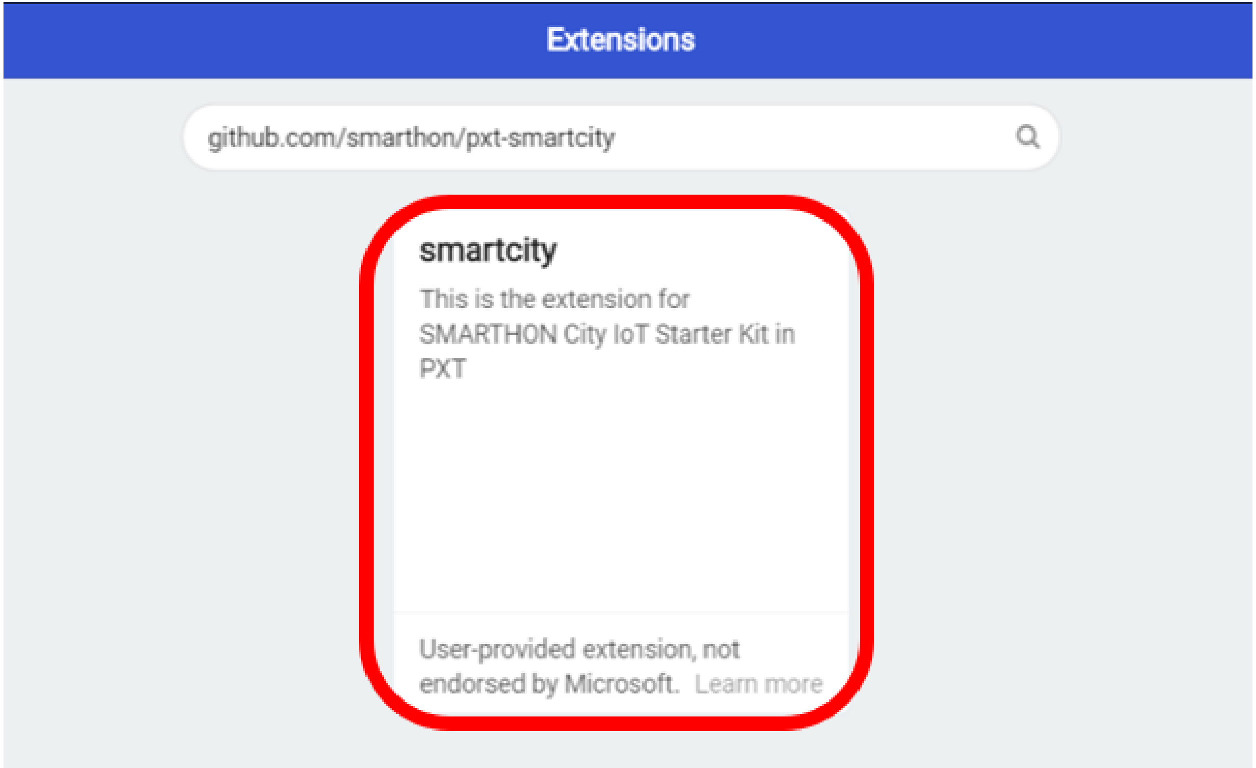
new project

Step 2 Give the project a name

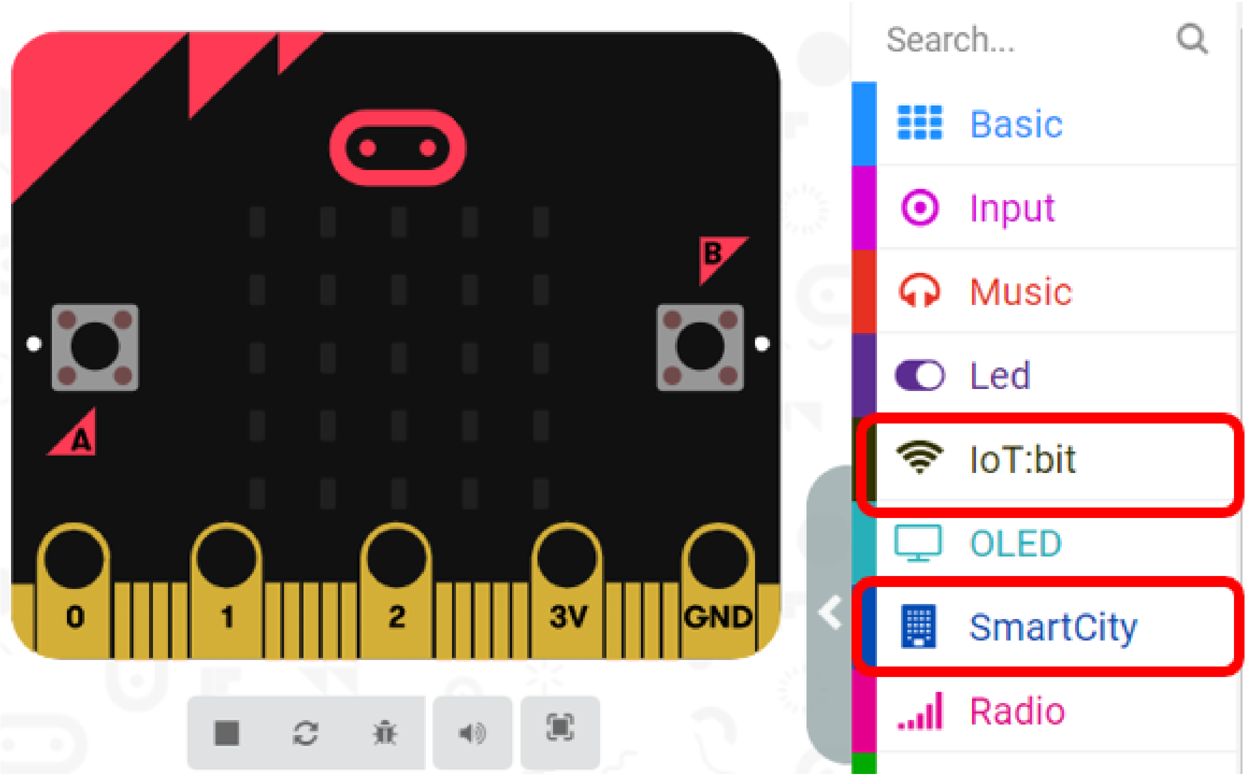
Step 3 Click "⚙" and choose "Extensions" in the pull-down menu



Step 4 Search "https://github.com/smarthon/pxt-smartcity" and click "smartcity"

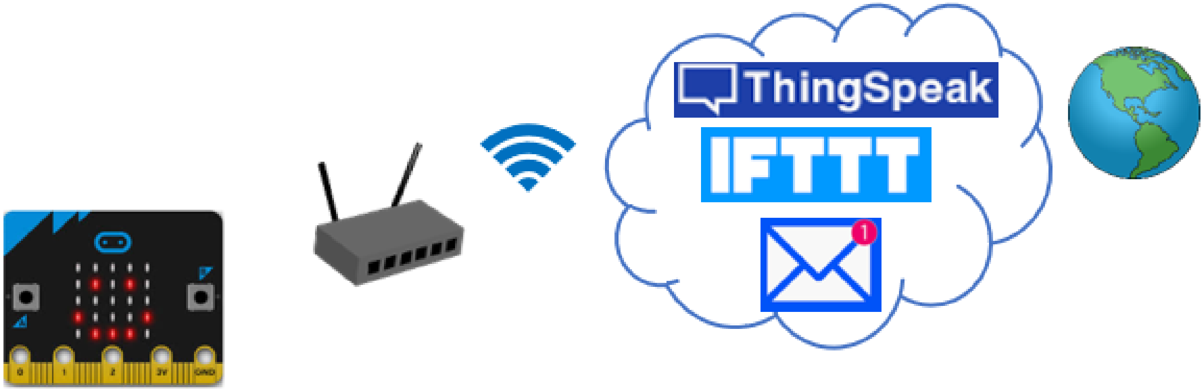

Extensions
github.com/smarthon/pxt-smartcity
smartcity
This is the extension for SMARTHON City IoT Starter Kit in PXT
User-provided extension, not endorsed by Microsoft. Learn more

Step 5 Extension `SmartCity` & `WiFi IoT:bit` package has been added successfully.



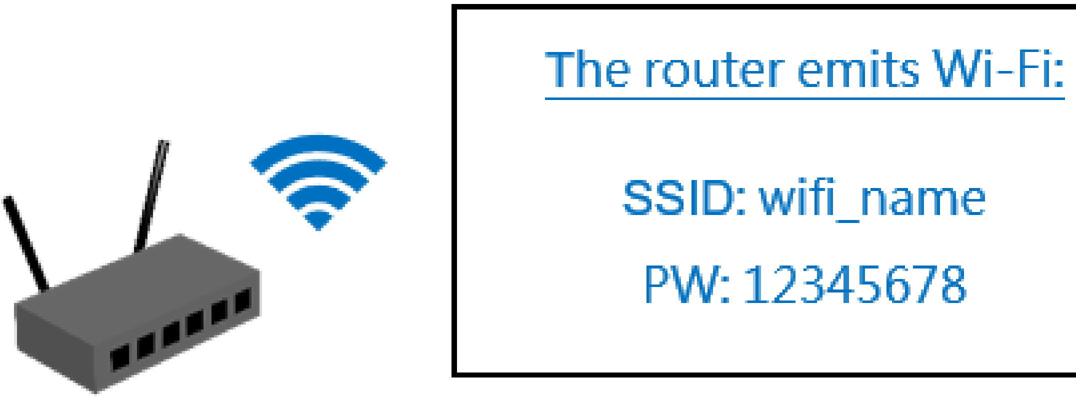
1.1.4 First step: Connect the micro:bit to the Wi-Fi

Connecting Wi-Fi is the very first step to access internet services, including ThingSpeak, IFTTT, email, etc. Before you start, make sure your micro:bit device is connected to the Internet.



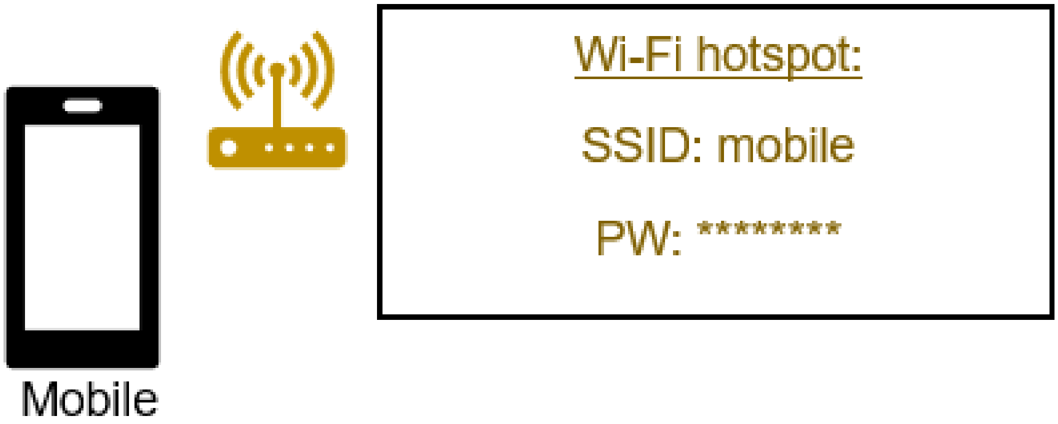
Open the Home Router or Mobile Phone Hotspot

Please sure there is a Wi-Fi signal (with 2.4GHz) available in your area. *5GHz Wi-Fi signal is not supported You can either setup a home router or a Wi-Fi hotspot from your mobile phone.

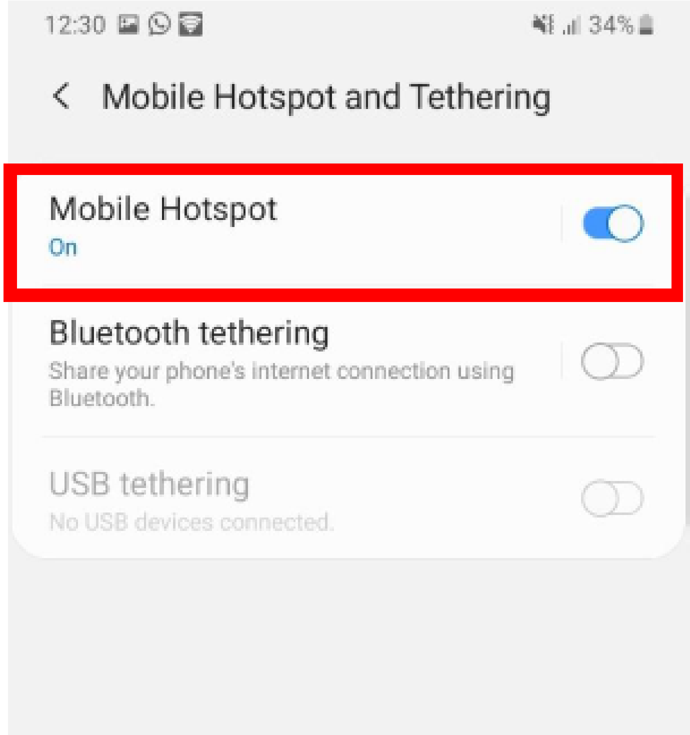


a) Setup your home/office Wi-Fi

b) Setup your mobile phone Hotspot * make sure your phone has internet service.



For android:

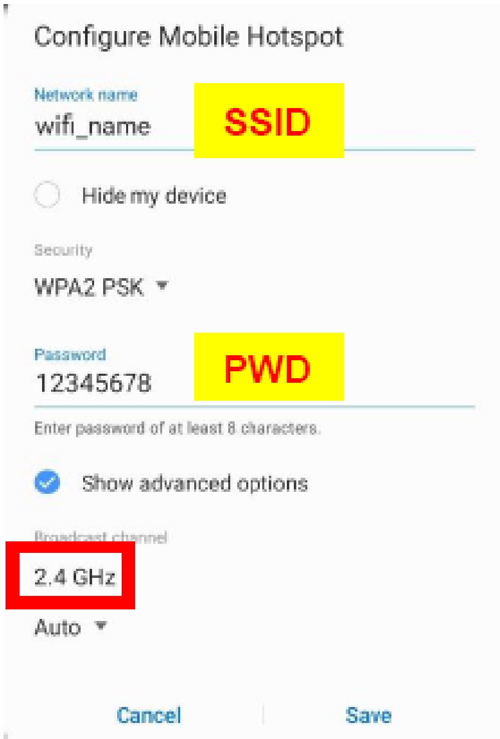

12:30 34%

Mobile Hotspot and Tethering

Mobile Hotspot
On

Bluetooth tethering
Share your phone's internet connection using Bluetooth.

USB tethering
No USB devices connected.


Configure Mobile Hotspot

Network name
wifi_name

SSID

○ Hide my device

Security
WPA2 PSK ▾

Password
12345678

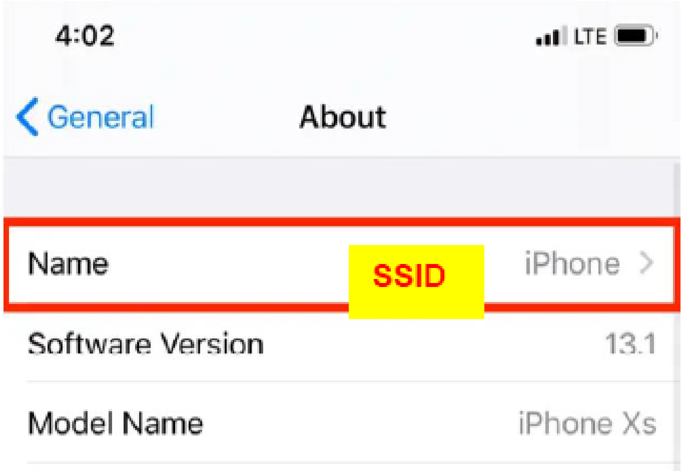
PWD

Enter password of at least 8 characters.

● Show advanced options

Broadcast channel
2.4 GHz
Auto ▾

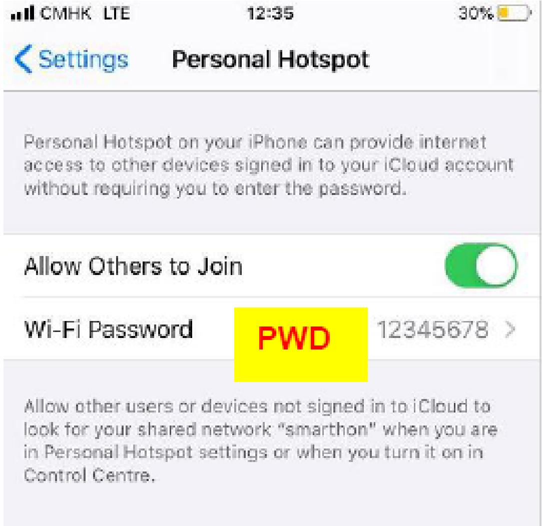
For iPhone:


4:02 LTE

< General About

Name	iPhone >
Software Version	13.1
Model Name	iPhone Xs

SSID


CMHK LTE 12:35 30%

< Settings Personal Hotspot

Personal Hotspot on your iPhone can provide internet access to other devices signed in to your iCloud account without requiring you to enter the password.

Allow Others to Join	On
Wi-Fi Password	12345678 >

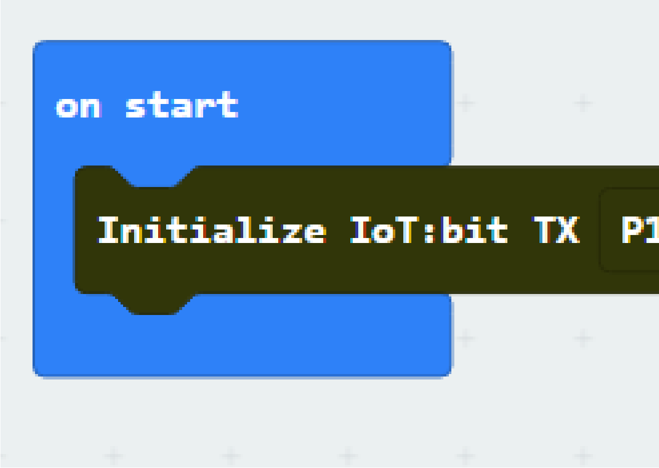
PWD

Allow other users or devices not signed in to iCloud to look for your shared network "smarthon" when you are in Personal Hotspot settings or when you turn it on in Control Centre.

Programming (Makecode)

Step 1: Initialize the IoT:bit Before you start, you need to initialize the IoT:bit.

- Go to IoT:bit tab.

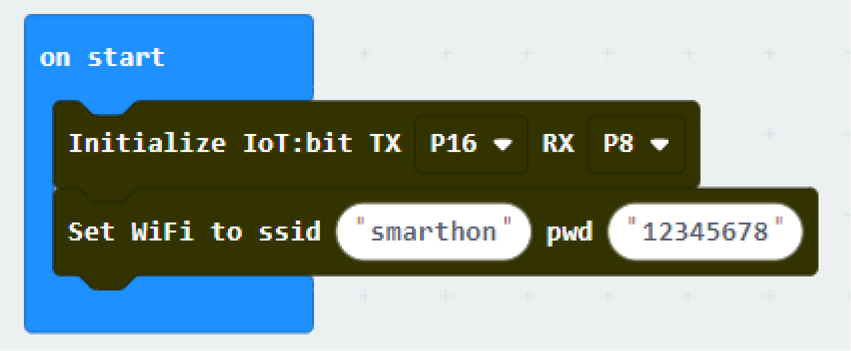


```
on start
Initialize IoT:bit TX P1
```

- Snap `Initialize IoT:bit TX P16 RX P8` to `on start`

Step 2: Connect to the Wi-Fi After initializing, you can connect to the Wi-Fi

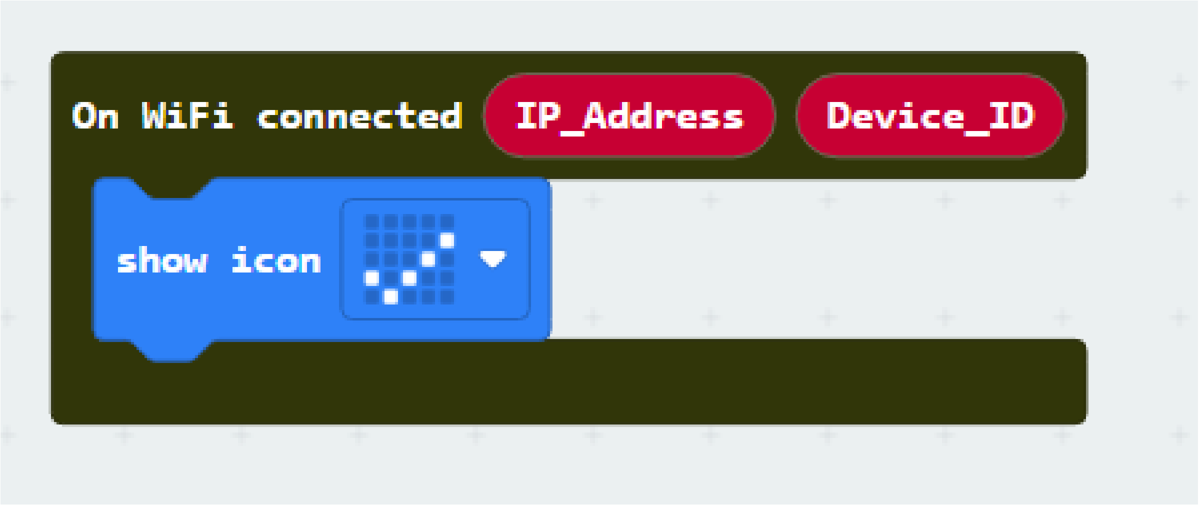
- Go to IoT:bit tab
- Snap `Set Wi-Fi to ssid pwd` to `on start` following initialize
- Enter your Wi-Fi name and password. Here we set `smarthon` as `SSID` and "12345678" as `password`.



```
on start
Initialize IoT:bit TX P16 ▾ RX P8 ▾
Set WiFi to ssid "smarthon" pwd "12345678"
```

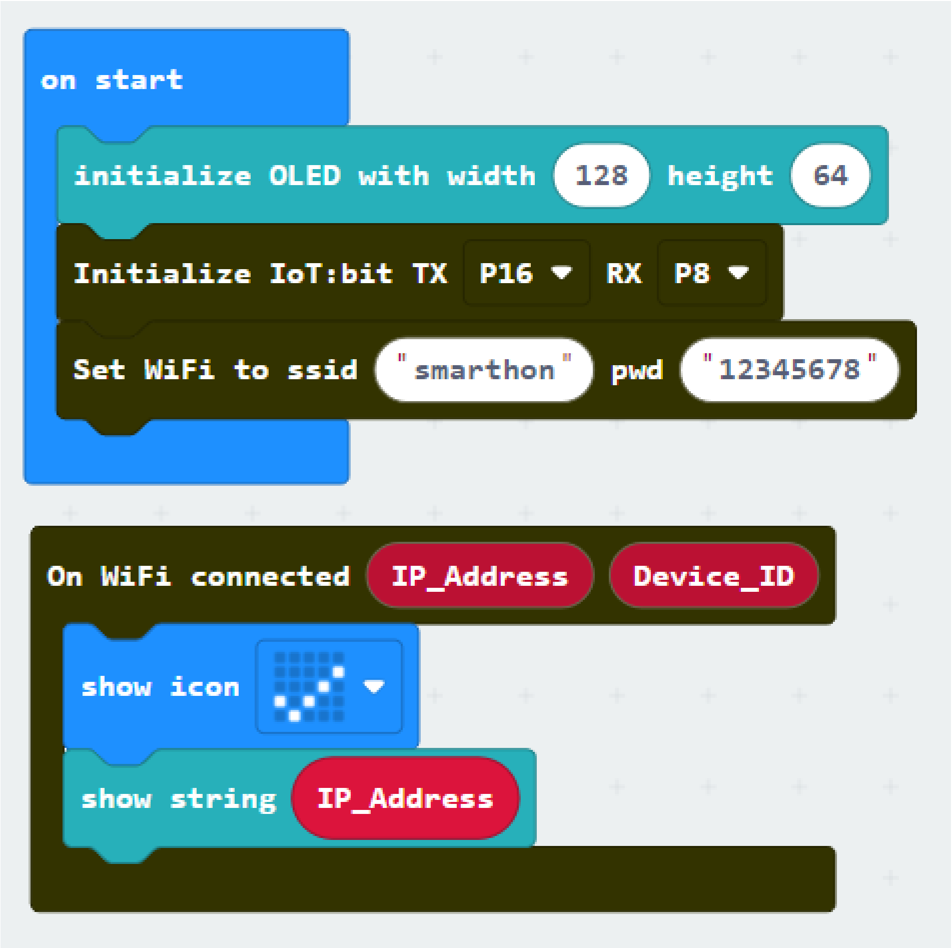
Step 3: Show the Tick if it is connected `On WiFi connected` is an event handler. It will be triggered once after connected with WiFi. After the WiFi is connected, it provide two variable `IP_Address` and `Device_ID` for you to control the IoT:bit though network.

- Snap `On WiFi connected` to the stage
- On `Basic` tab, snap `show icon heart` inside `On WiFi connected`
- Swtich the heart to tick



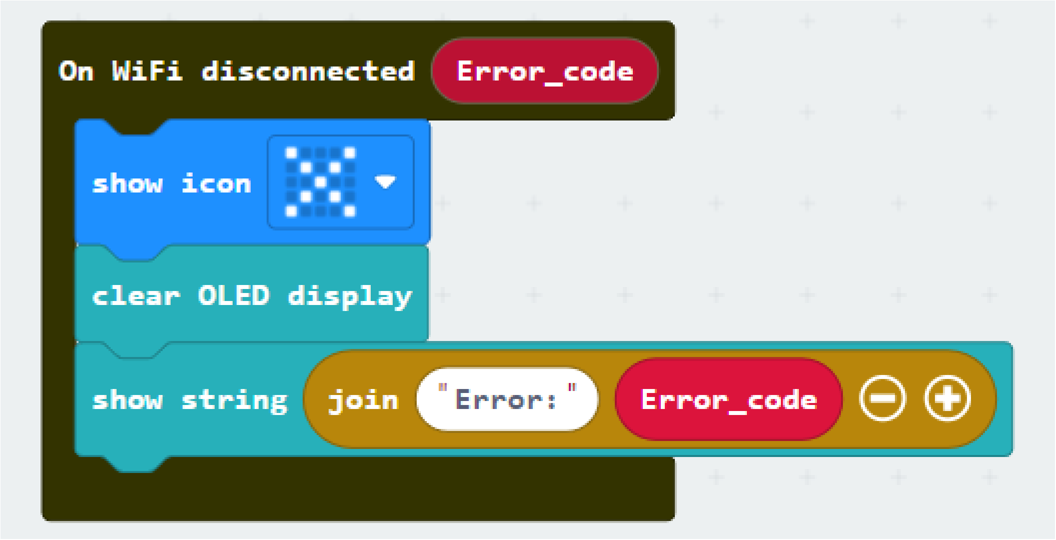
Step 4: Show you IP If you want to see the IP address, you may show it on the OLED display.

- Go to OLED tab
- Snap "initialize OLED with width. . . height.." to on start
- Snap the "show string" inside the On WiFi connected
- Draw the variable from On WiFi connected to the "show string" block placeholder
- You may control the text format by the function in Text tab



```
on start
    initialize OLED with width 128 height 64
    Initialize IoT:bit TX P16 RX P8
    Set WiFi to ssid "smarthon" pwd "12345678"

On WiFi connected IP_Address Device_ID
    show icon
    show string IP_Address
```

Optional: On WiFi disconnected handler will be triggered when the WiFi connection is loss, it may useful when you get disconnection to the WiFi Station during the application running.

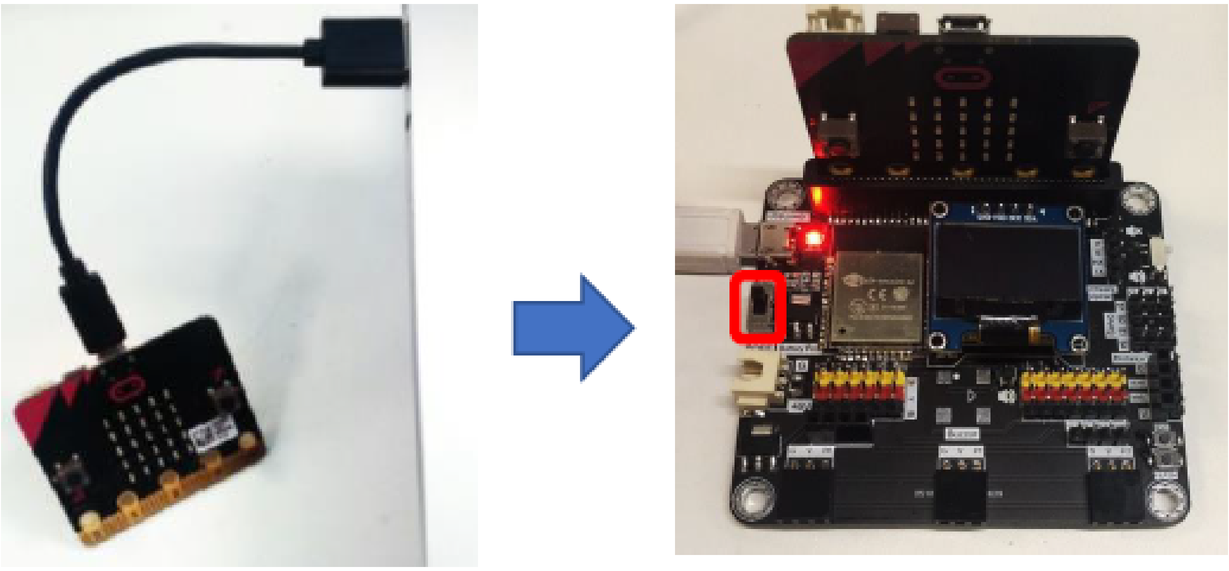


```
On WiFi disconnected Error_code
    show icon
    clear OLED display
    show string join "Error:" Error_code
```

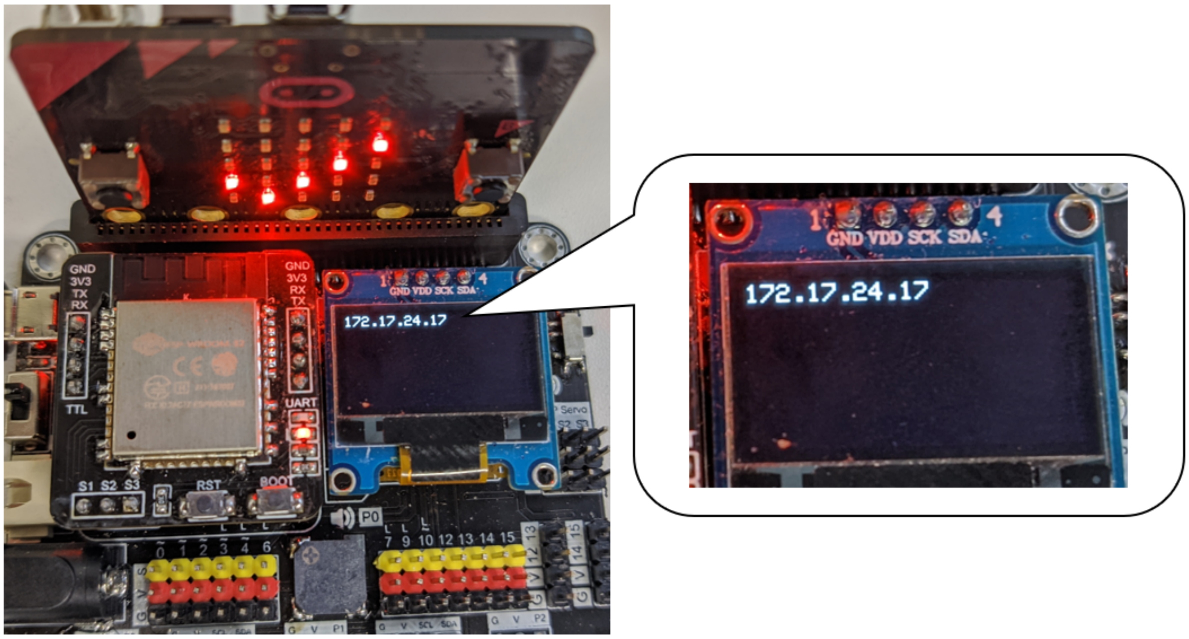
Full Solution MakeCode: https://makecode.microbit.org/_53Yf8qcx521m You could also download the program from the following website:

Result

Step 1 Load the program to the micro:bit and turn on the IoT:bit by using USB.



Step 2 Wi-Fi IoT:bit will start to connect to internet, when the connection is successful, the IP Address would be shown. AND the tick icon will be shown.

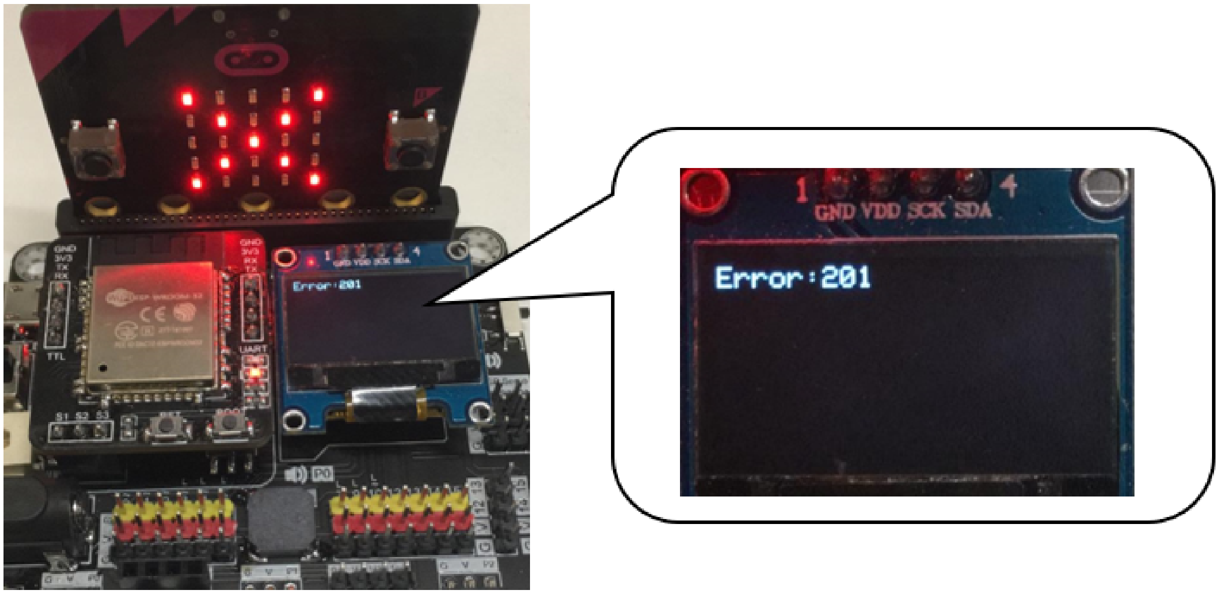


After the IoT:bit is connected to the wifi, when disconnect to the router. It will show "cross" symbol and the error code will be shown. Normally, the code `201` means "No AP Found".

Reasons might cause unable to connect or disconnect:

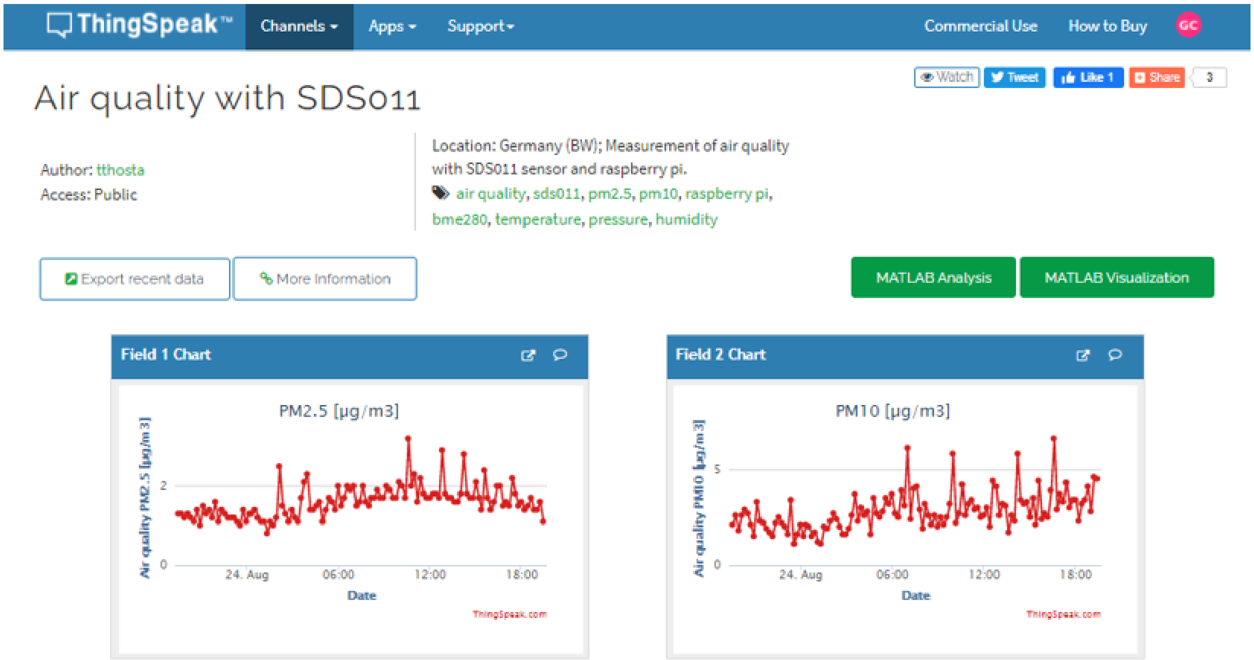
1. Router is too far away
2. No that AP actually
3. The typo on SSID/PWD
4. The battery have run out, please change to new battery
5. Not using 2.4Ghz frequency Wi-Fi channel
6. Follow the best practice on using the iot:bit

You may get more detailed information on error codes from NodeMCU Documentation



1.1.5 Chapter 1: Upload Data to ThingSpeak

ThingSpeak is IoT platform for user to gather real-time data; for instance, climate information, location data and other device data. In different channels in ThingSpeak, you can summarize information and visualize data online in charts and analyze useful information. ThingSpeak can integrate IoT:bit (micro:bit) and other software/ hardware platforms. Through IoT:bit, you can upload sensors data to ThingSpeak (e.g. temperature, humidity, light intensity, noise, motion, raindrop, distance and other device information).



Air quality with SDS011

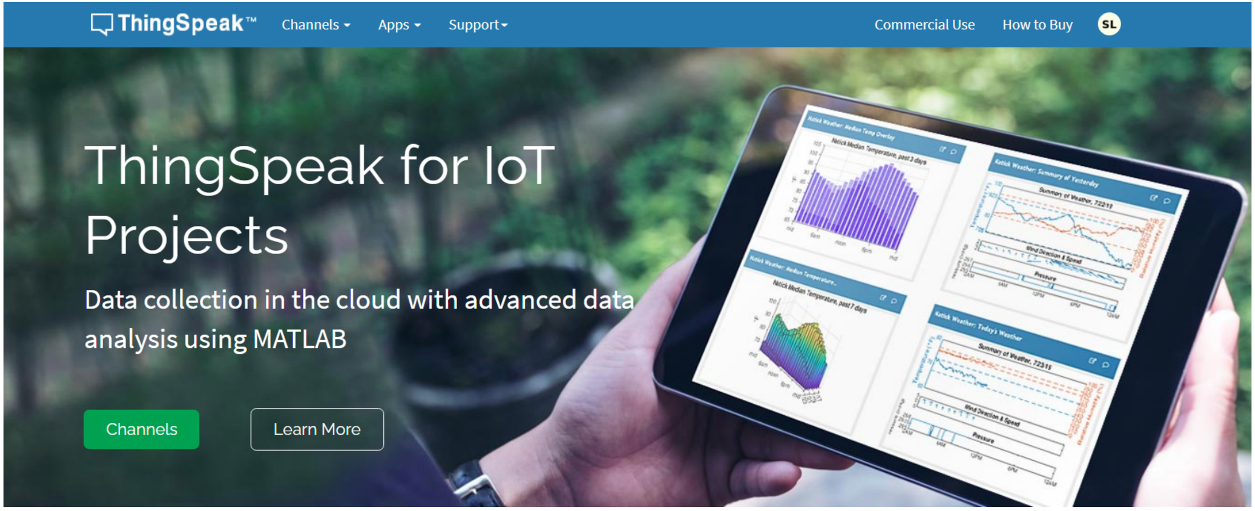
Author: tthosta
Access: Public

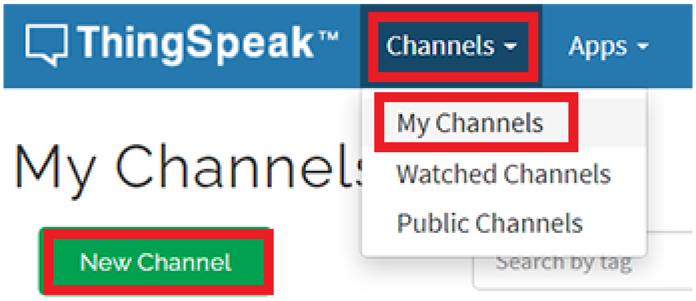
Location: Germany (BW); Measurement of air quality with SDS011 sensor and raspberry pi.
air quality, sds011, pm2.5, pm10, raspberry pi, bme280, temperature, pressure, humidity

Thingspeak Configuration

Goal: we need to create the thingspeak channel and get the key

Step 1 Go to https://thingspeak.com/, register an account and login to the platform

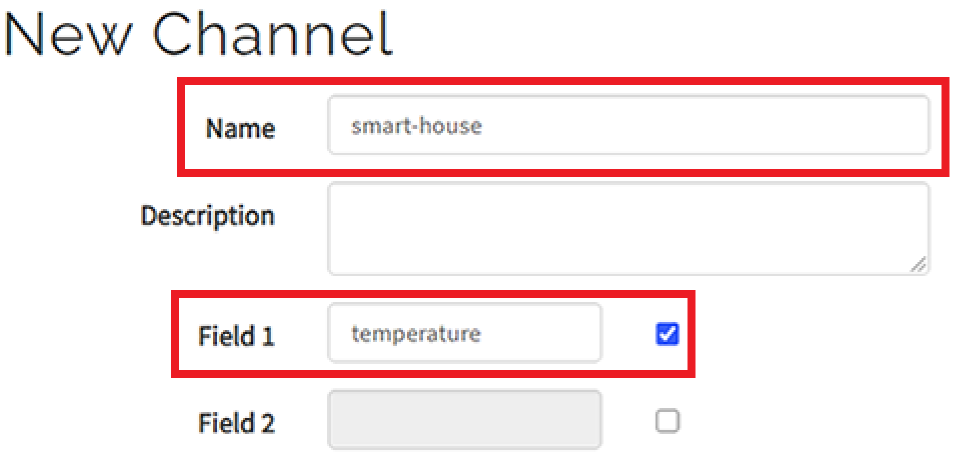

ThingSpeak™
Channels
Apps
Support
Commercial Use
How to Buy
SL
ThingSpeak for IoT Projects
Data collection in the cloud with advanced data analysis using MATLAB
Channels
Learn More



Step 2 Choose Channels -> My Channels -> New Channel

Step 3 Input Channel name, Field1 , then click "Save Channel"

- Channel name: smart-house



New Channel

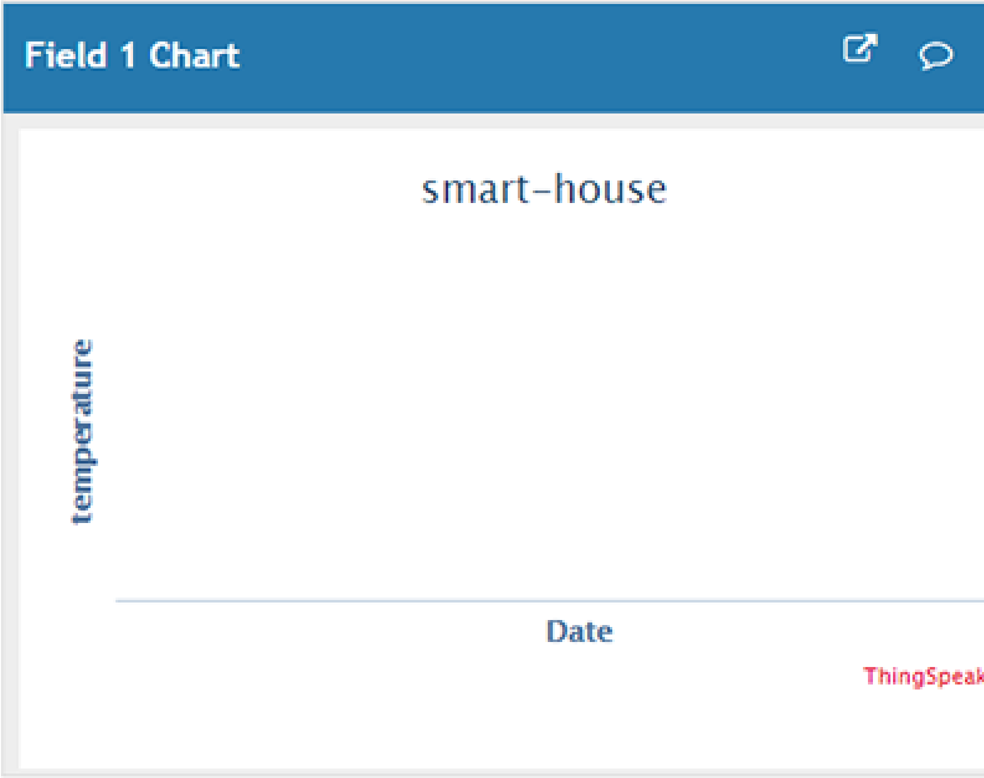
Name: smart-house

Description:

Field 1: temperature ☑

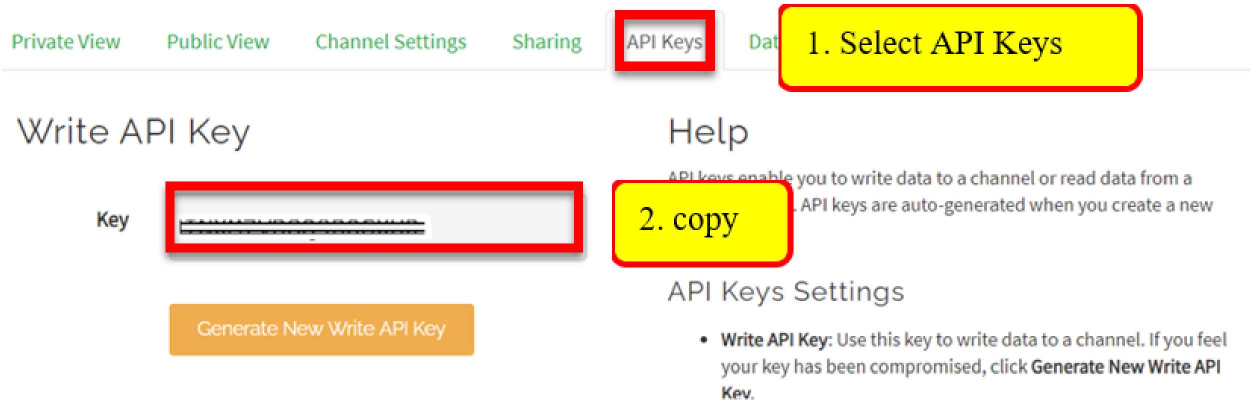
Field 2: □

- Field 1: Temperature



Step 4 You will see a chat for data field1

Step 5 Open your web browser, go to https://thingspeak.com , select your channel > "API Keys" copy the API key as



Write API Key

Key [redacted]

Help

API keys enable you to write data to a channel or read data from a ... API keys are auto-generated when you create a new

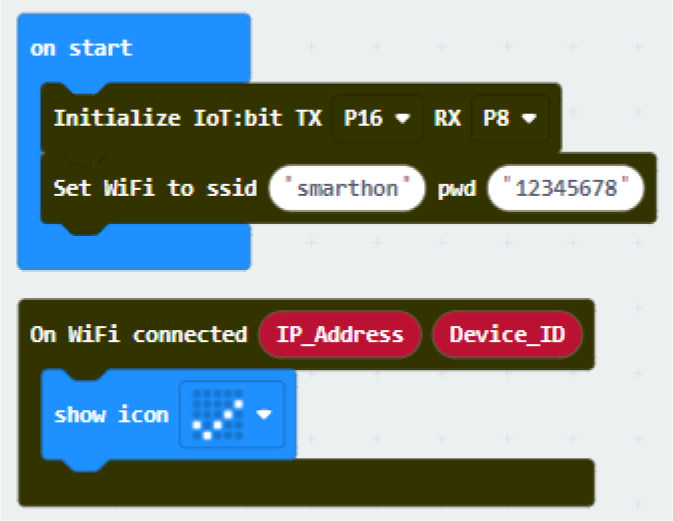
API Keys Settings

- **Write API Key:** Use this key to write data to a channel. If you feel your key has been compromised, click **Generate New Write API Key.**

follows:

Coding (Makecode)

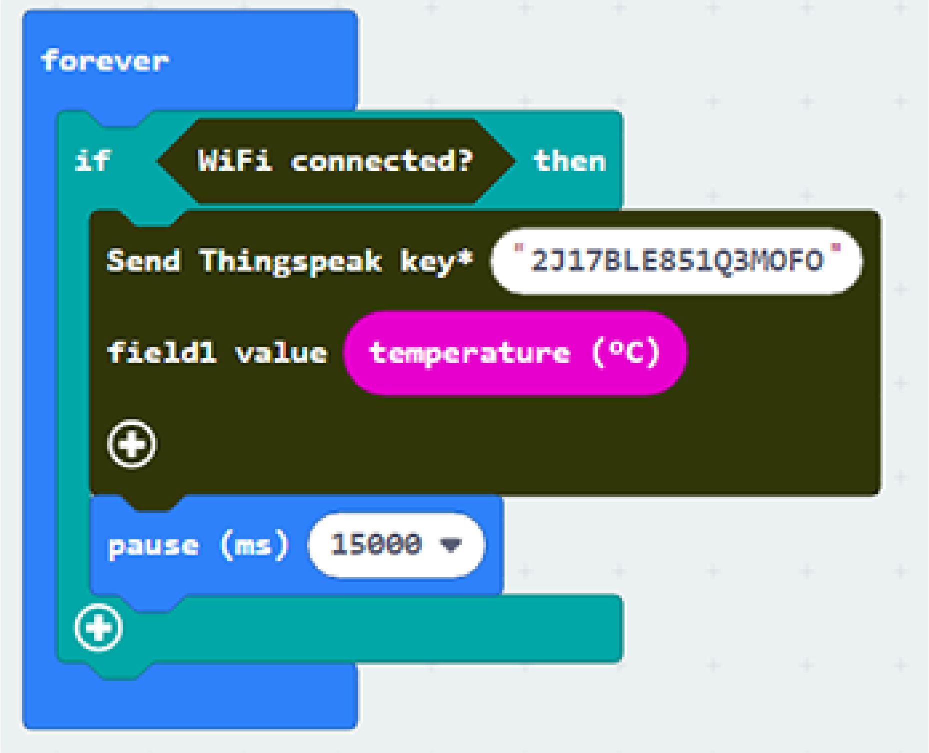
Step 1 Before we do the thingspeak uploading part, we already know how to connect to the WiFi on the first chapter.



```
on start
    Initialize IoT:bit TX P16 RX P8
    Set WiFi to ssid "smarthon" pwd "12345678"

On WiFi connected IP_Address Device_ID
    show icon
```

Step 2 On every 15 seconds, if the WiFi is connected, it will send data to ThingSpeak.

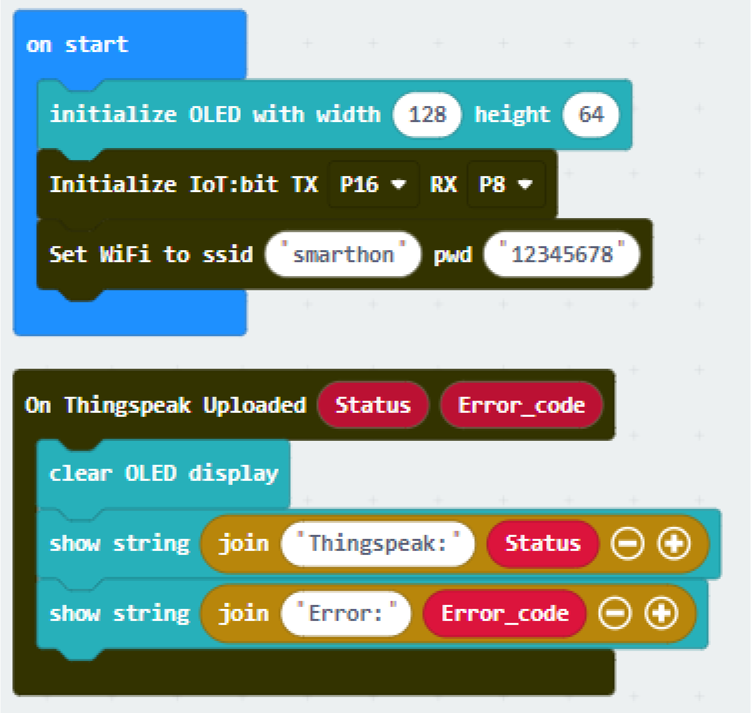
- Use `if-statement` inside the `forever` loop to check `WiFi connected?` status
- Snap the `Send Thingspeak key.....` block inside the `if-statement`
- Fill in the `api key` from Thingspeak and the `temperature` values
- Snap `pause` to the loop to wait 15 second



```
forever
    if WiFi connected? then
        Send Thingspeak key* "2J17BLE851Q3MOFO"
        field1 value temperature (°C)
    pause (ms) 15000
```

Step 3 If you want to show the thingspeak status, you can use the on Thingspeak Uploaded handler to get the variable for status and error code. You may use the OLED to display the status and error code.

- Go to OLED Tab
- Snap `initialize OLED with width 128 height 64` to `on start`
- Snap the `show string` inside the `On Thingspeak Uploaded`
- Draw the variable from `On Thingspeak Uploaded` to the `show string` block placeholder.

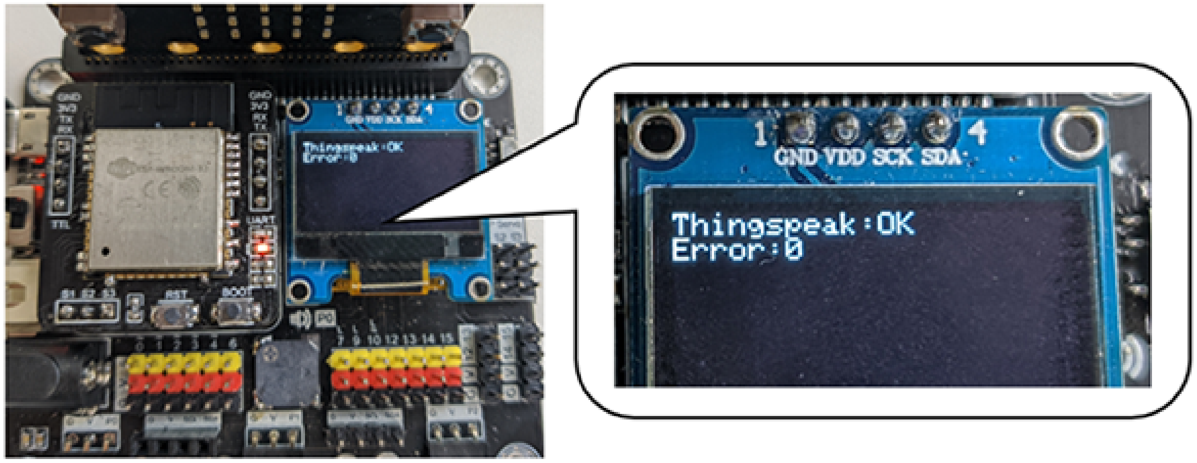


```
on start
    initialize OLED with width 128 height 64
    Initialize IoT:bit TX P16 ▾ RX P8 ▾
    Set WiFi to ssid "smarthon" pwd "12345678"

On Thingspeak Uploaded Status Error_code
    clear OLED display
    show string join "Thingspeak:" Status ⊖ ⊕
    show string join "Error:" Error_code ⊖ ⊕
```

Full Solution MakeCode: https://makecode.microbit.org/_PHU0oJLwHAYD You could also download the program from the following website:

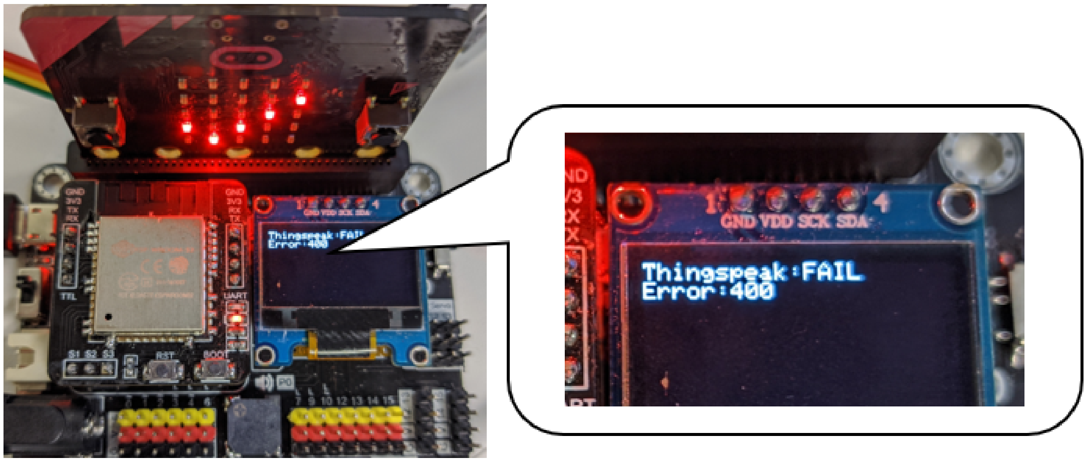
Result

After Connected to Internet, the program start upload the data to Thingspeak. When the Upload is finish, the it will shown the status `OK` and Error Code `0`.

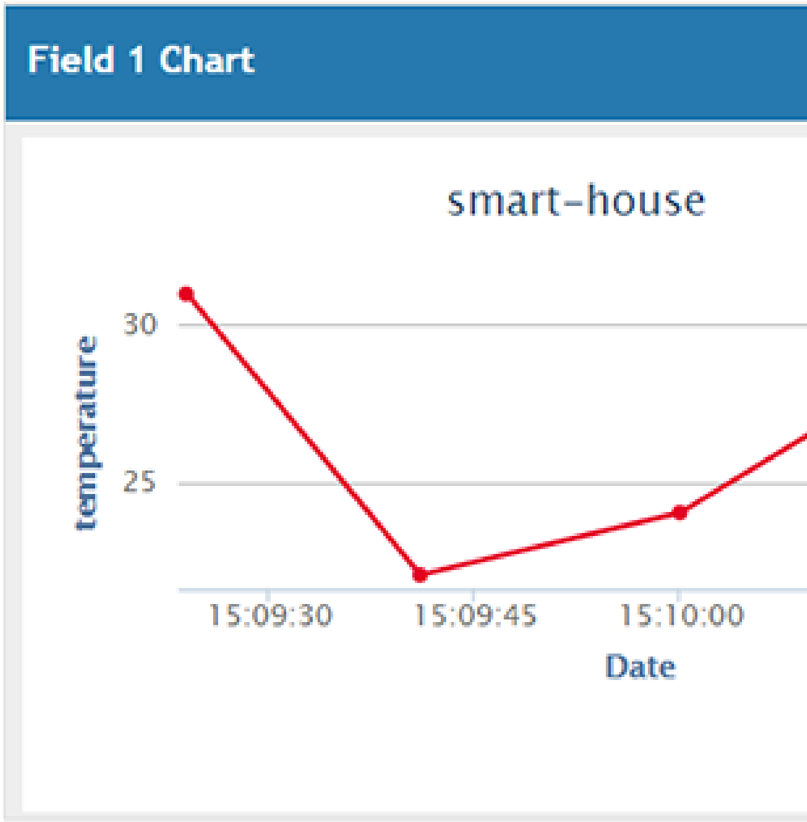


```
Thingspeak:OK
Error:0
```

If the thingspeak cannot be uploaded successfully, the status will return `FAIL` with `error code`. For the error 400, the user inputted the wrong API Key. For the error code -28674, there is no internet connection.

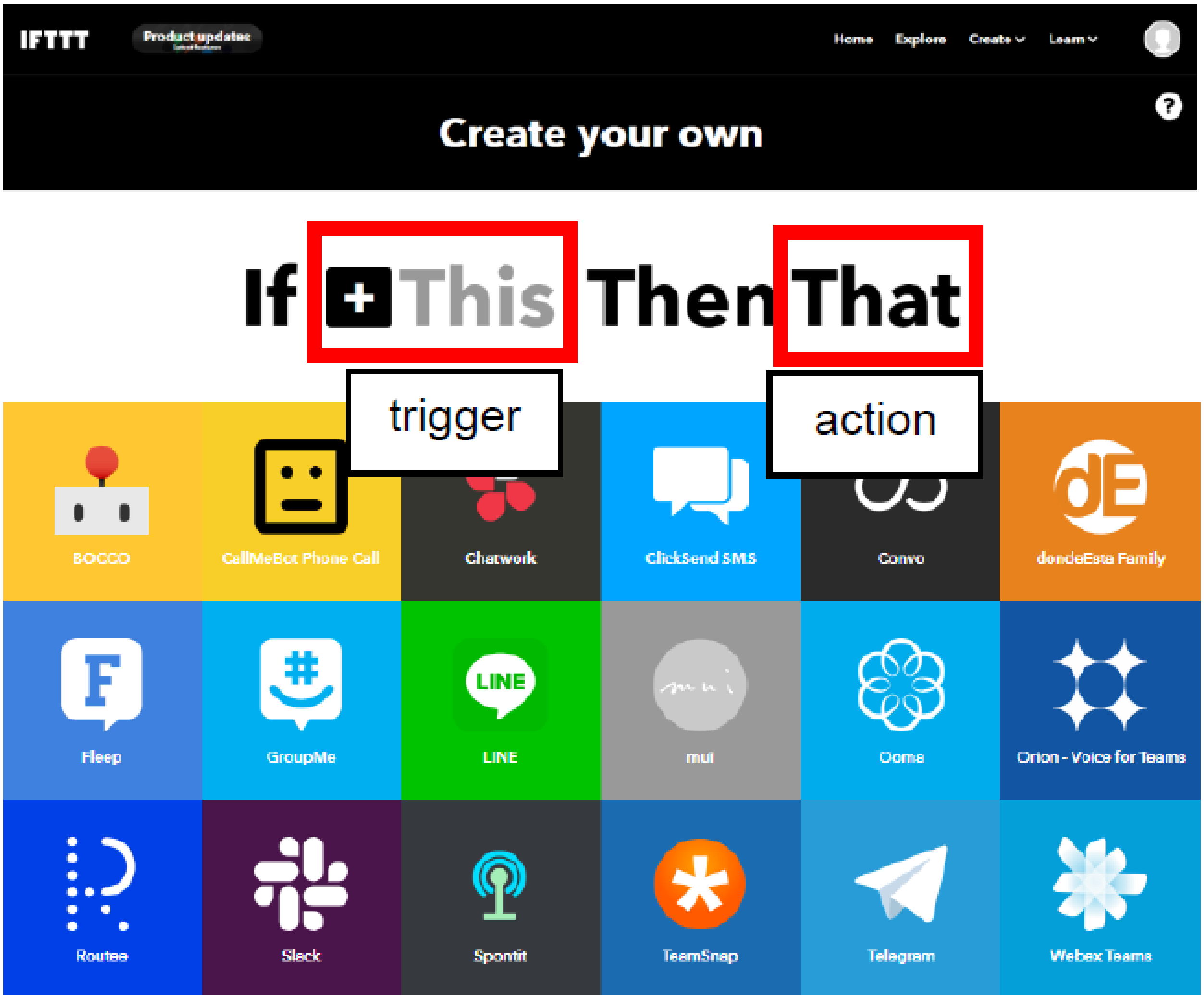

Thingspeak:FAIL
Error:400

The charts in your channel in ThingSpeak will be updated.



1.1.6 Chapter 2: Send Email by IFTTT

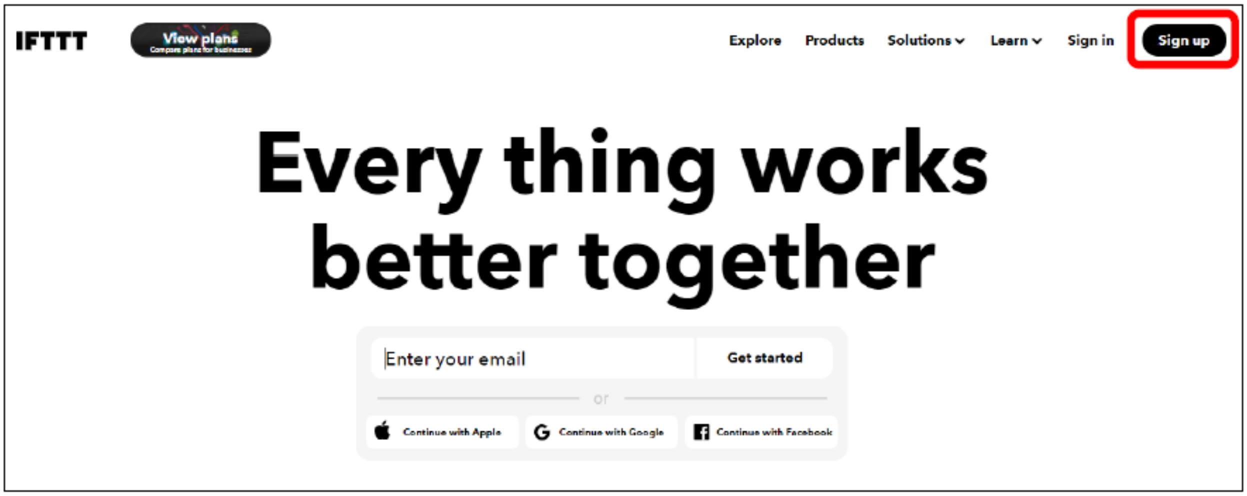
IFTTT is the way to connect hundreds of the apps and devices, including Twitter, Dropbox, Email, Google Assistant, etc. By the basic programming logic "if this then that", it connects IoT services through their API (application programming interface). In IFTTT, you can make your own applet and link services together by "if this(trigger) then that(action)". For example, if we select webhooks (micro:bit button A) as trigger, Email (send email to your mailbox) as action. When micro:bit button A is pressed, it triggers (webhooks) and implement the action (send email) in IFTTT.

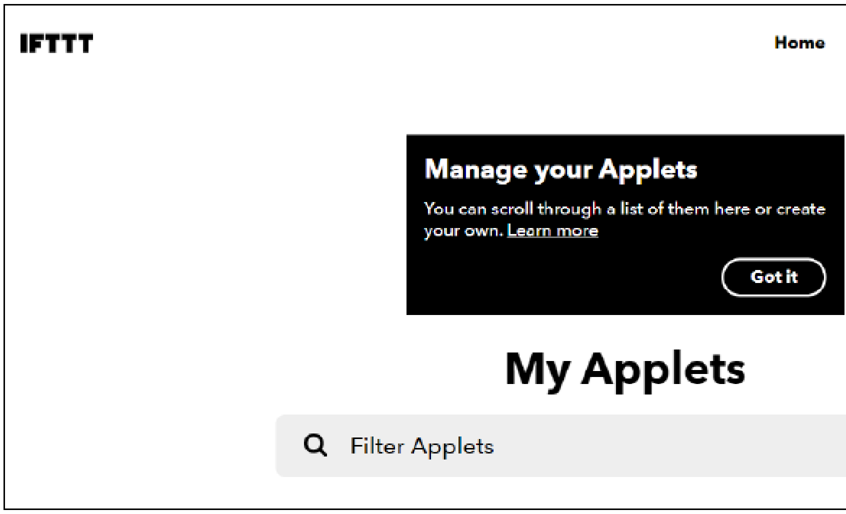


In this chapter, you will learn using micro:bit to activate trigger (webhooks). After that, it can implement other services (e.g. email)

IFTTT Configuration

Step 1 Go to http://www.ifttt.com, register an account and login to the platform



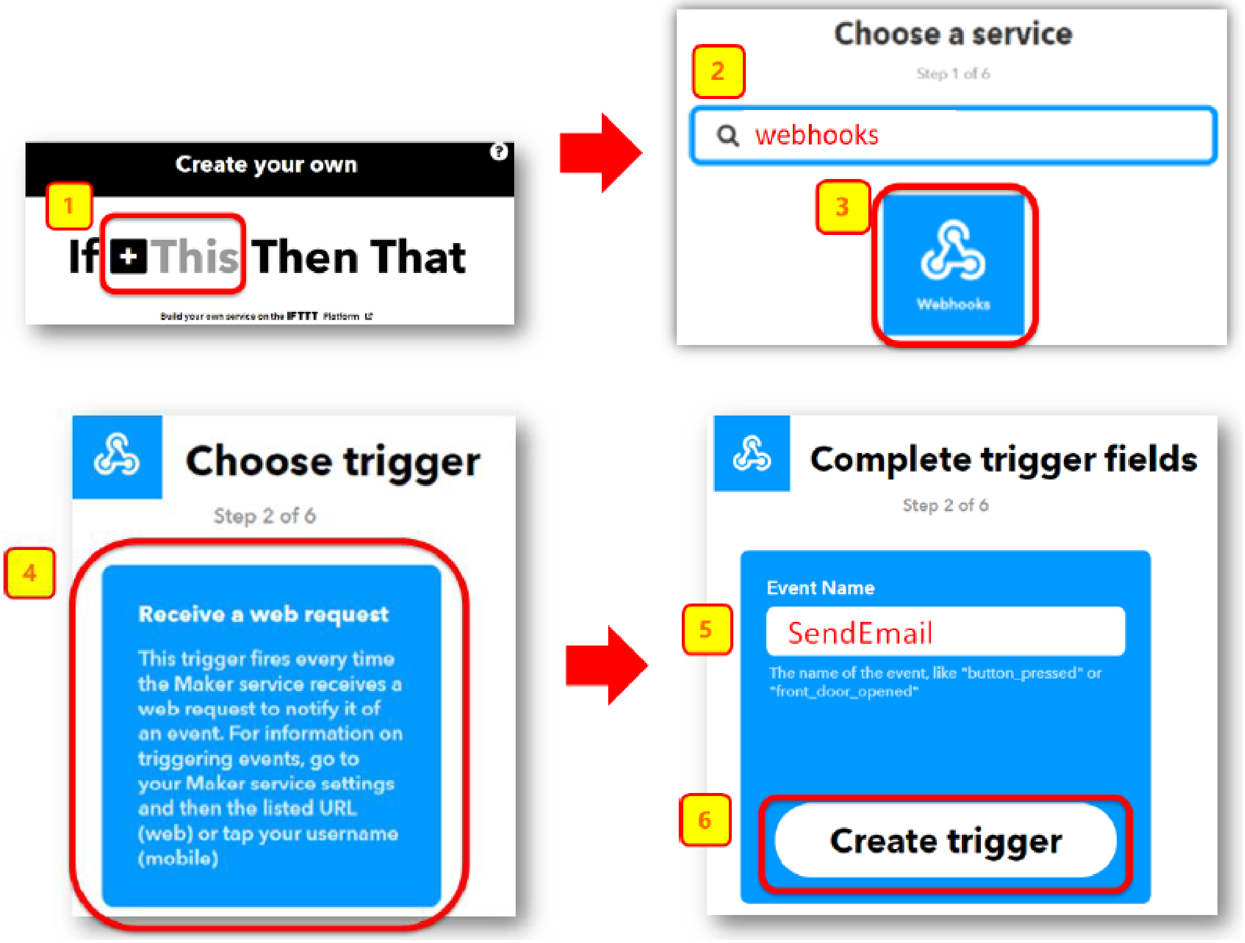


Manage your Applets

You can scroll through a list of them here or create your own. Learn more

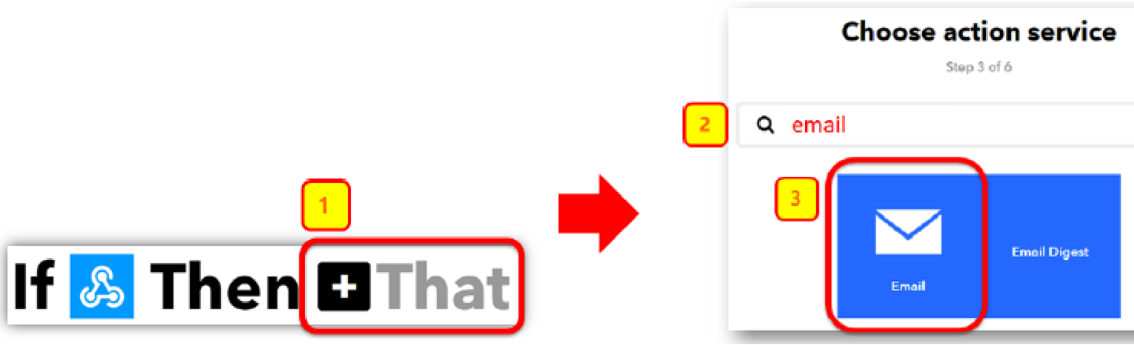
Step 2 On the top right menu, click “Create” > “Applets”

Step 3 Select this -> select webhooks -> input Event Name (eg. Event Name: SendEmail)then click “Create trigger”



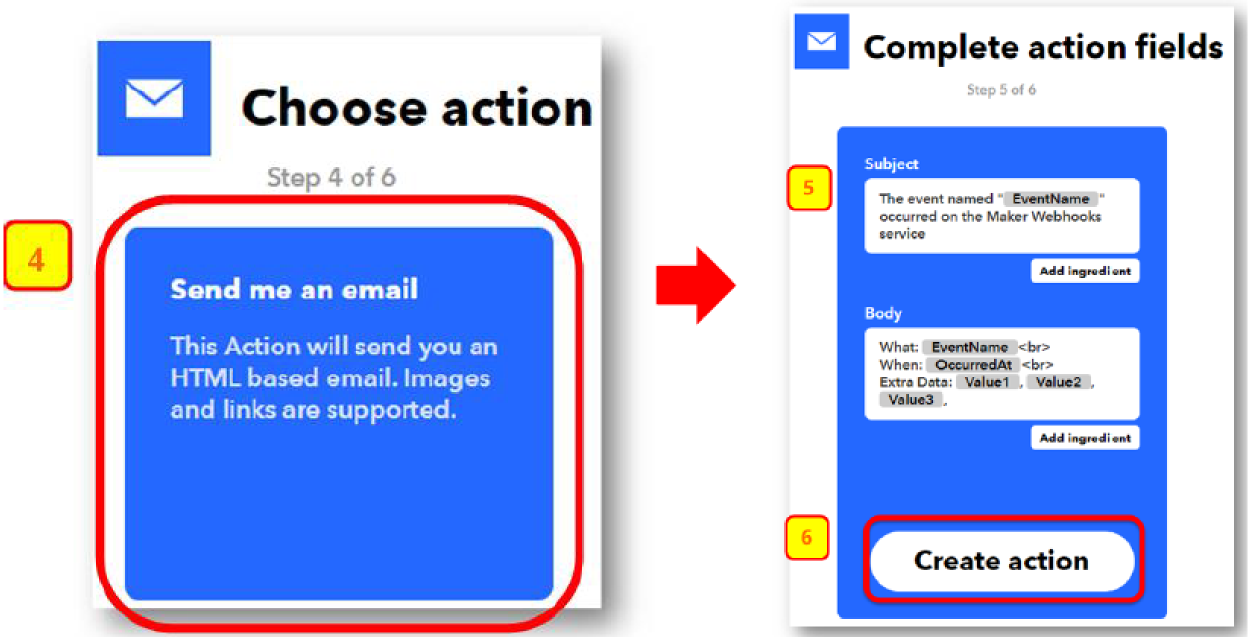
This trigger fires every time the Maker service receives a web request to notify it of an event. For information on triggering events, go to your Maker service settings and then the listed URL (web) or tap your username (mobile)

The name of the event, like "button_pressed" or "front_door_opened"



Step 4 Select “That” > Email

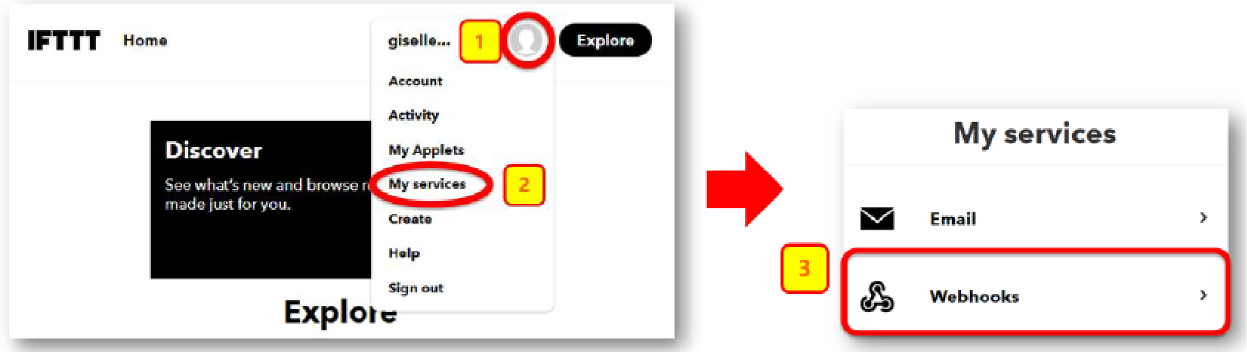
Step 5 Select “Send me an email” , Input email title and body, then click “Create action”



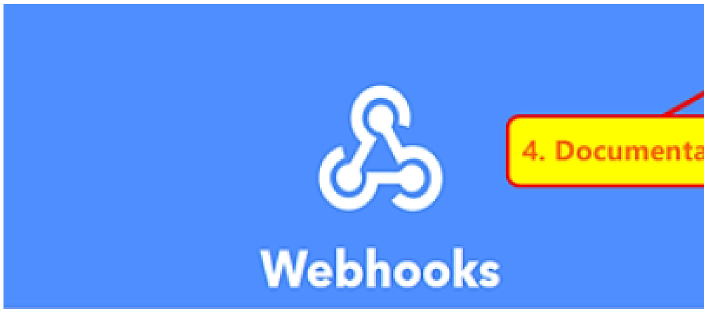
Subject: The event named " EventName " occurred on the Maker Webhooks service

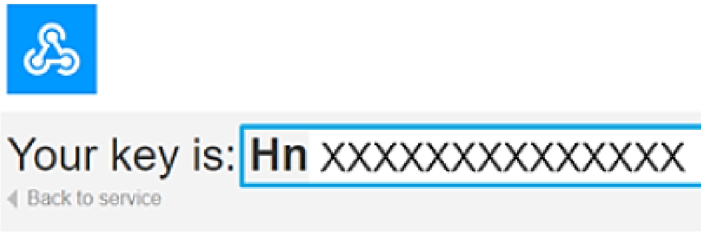
Body: What: EventName
 When: OccurredAt
 Extra Data: Value1 , Value2 , Value3 .

Step 6 Open your web browser, open the top right menu, select “My services” > “Webhooks”



Step 7 Select “Documentation” Copy your Webhooks Key as follows:

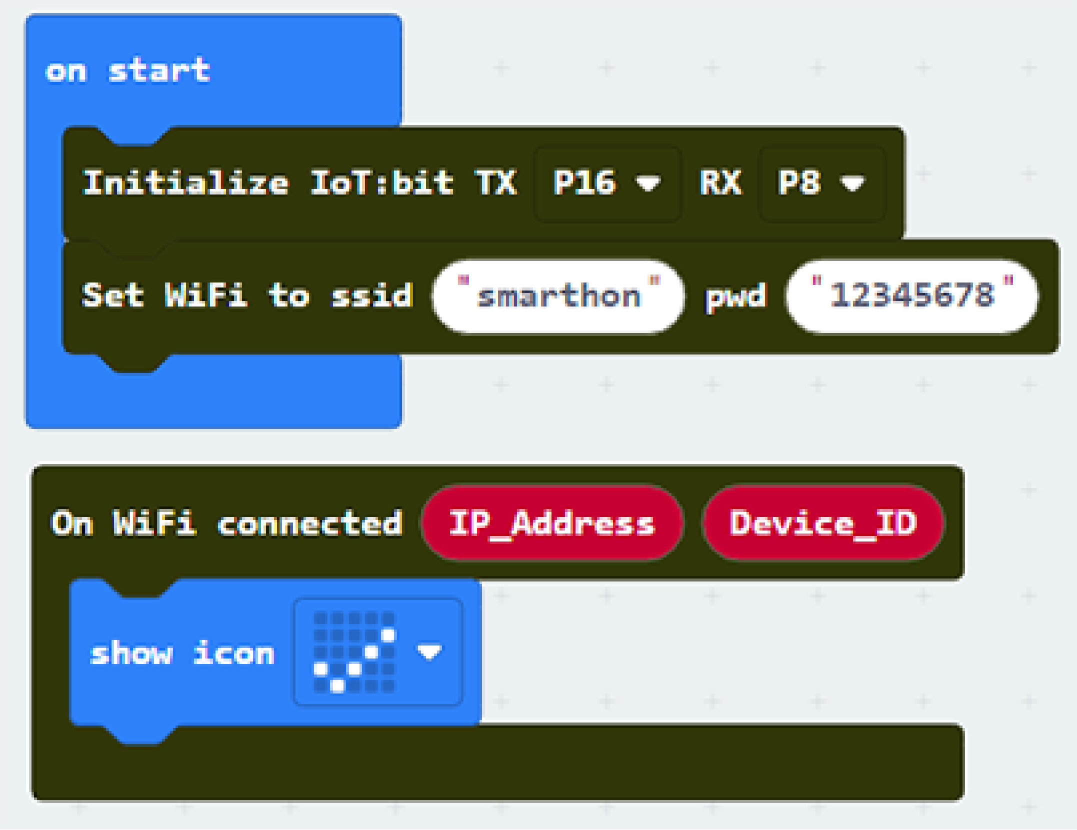




Your key is: Hn XXXXXXXXXXXXXX

Coding (Makecode)

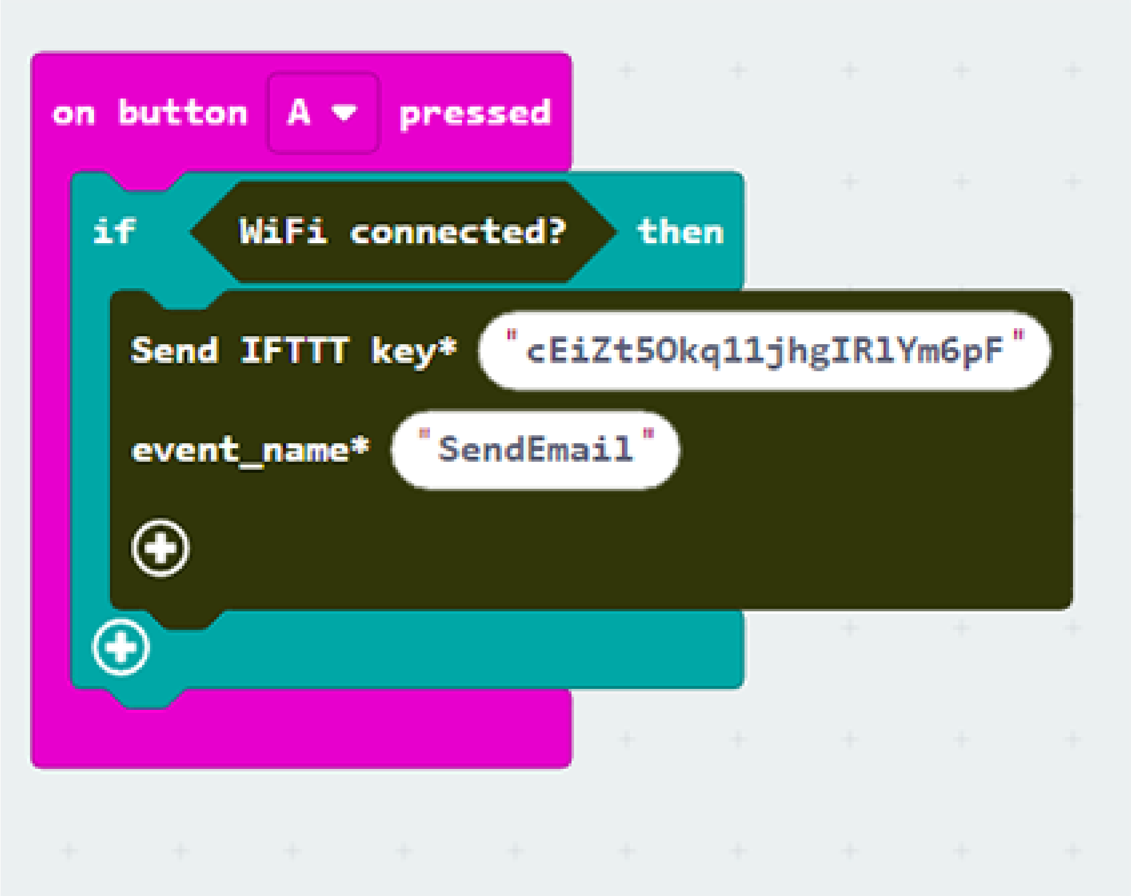
Step 1 Before we do the IFTTT uploading part, we have already know how to connect to the WiFi on the first chapter.



```
on start
    Initialize IoT:bit TX P16 ▾ RX P8 ▾
    Set WiFi to ssid "smarthon" pwd "12345678"

On WiFi connected IP_Address Device_ID
    show icon ▾
```

Step 2 When need to send ITFFF request, if `WiFi connected`, send data to IFTTT. Input the following data of the IFTTT Application to the `Send IFTTT key...` block

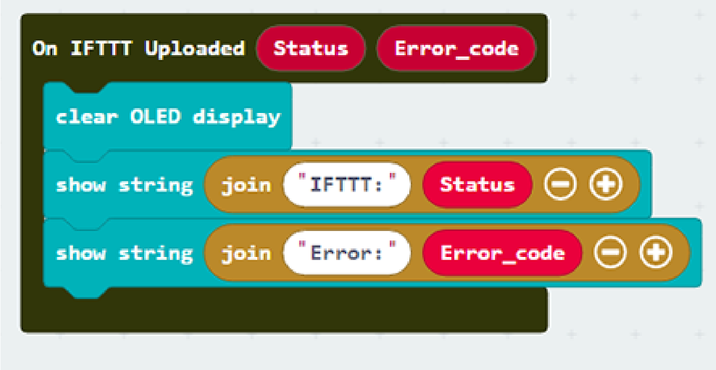
- `IFTTT API key`: XXXXXXXXXXXXXXXX.
- `event_name`: event name to trigger in Webhooks (eg. SendEmail)
- `value 1-3`: if user want to add the value inside the email, click the add button and input value1, value2, value3



```
on button A ▾ pressed
    if WiFi connected? then
        Send IFTTT key* "cEiZt5OkqlljhgIRlYm6pF"
        event_name* "SendEmail"
        ⊕
    ⊕
```

Step 3 If you want to show the IFTTT upload status, you can use the "on IFTTT Uploaded" handler to check the variable. You may use the OLED to display the status and error code.

- Go to OLED Tab
- Snap `initialize OLED with width...height..` to `on start`
- Snap the `show string` inside the `On IFTTT Uploaded`
- Draw the variable from `On IFTTT Uploaded` to the `show string` block placeholder

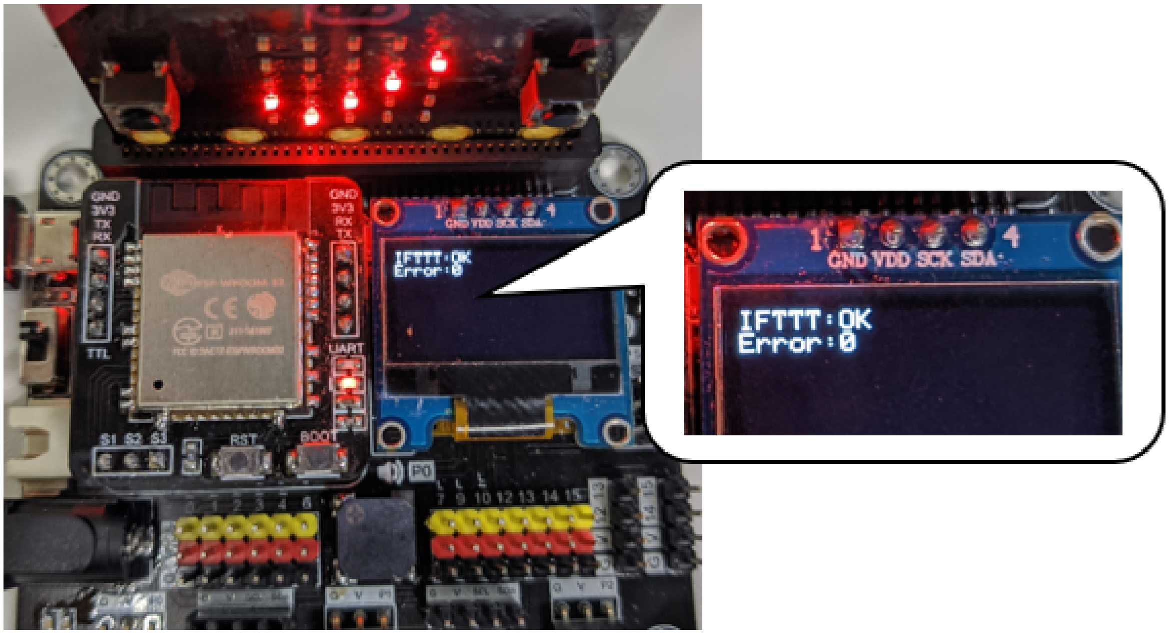


```
On IFTTT Uploaded Status Error_code
    clear OLED display
    show string join "IFTTT:" Status ⊖ ⊕
    show string join "Error:" Error_code ⊖ ⊕
```

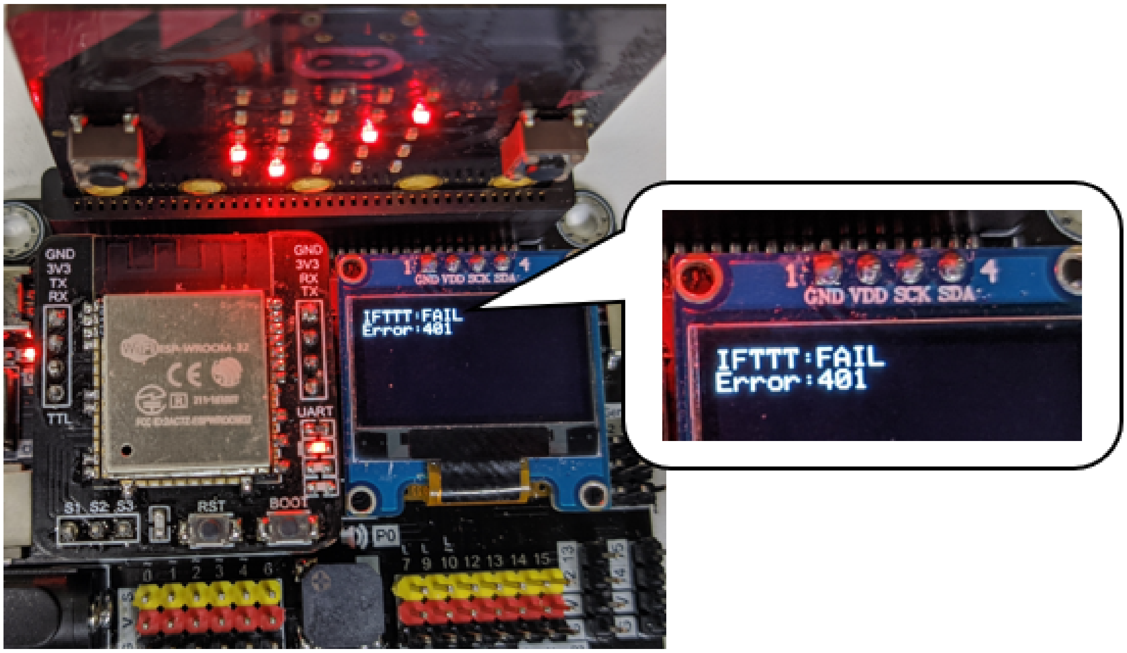
Full Solution MakeCode: https://makecode.microbit.org/_CwbdXV0ascyx You could also download the program from the following website:

Result

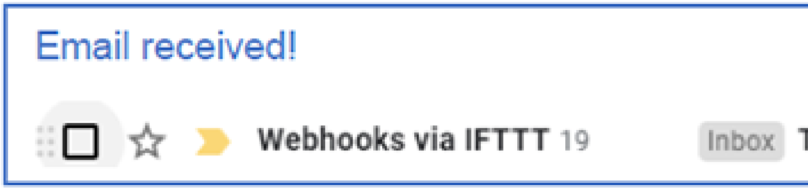
After micro:bit is connected to WiFi and click button A, it will upload data to IFTTT. If the upload is success, it will show `OK` with error code `0`



If the upload is fail (such input wrong API key in this case), it will show `FAIL` with `error code`. For the error 401, the user inputted the wrong API Key. For the error code -28674, there is no internet connection.



When success, email will be sent to your mailbox by IFTTT.



Email received!

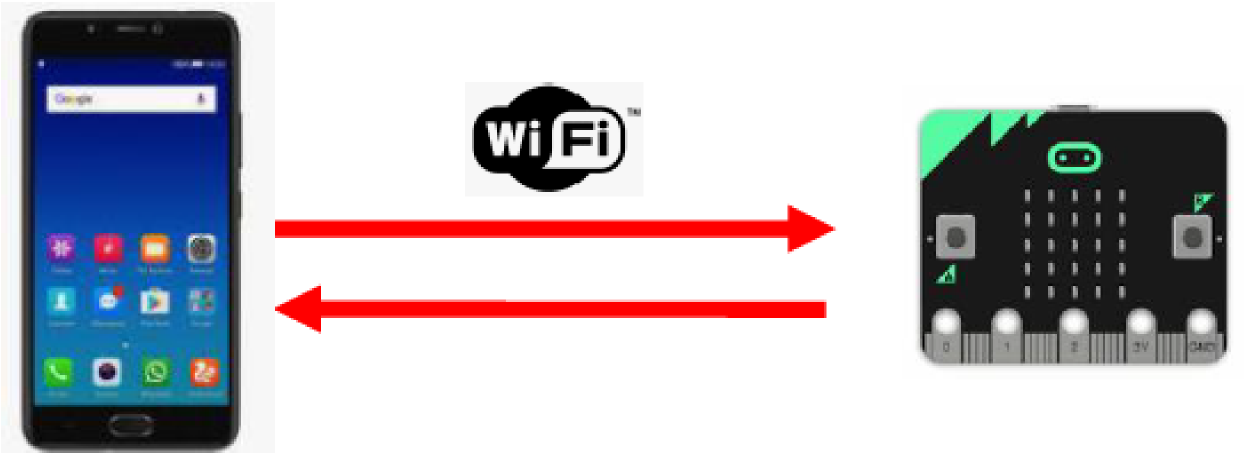
□ ☆ Webhooks via IFTTT 19 Inbox T

1.1.7 Chapter 3: Direct Control micro:bit by App Inventor 2

Introduction

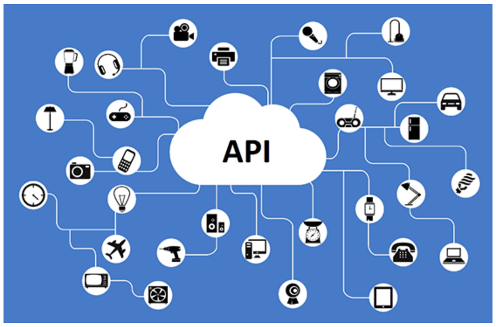
App inventor 2 is a blocks-based programming tool, which help beginners to build fully functional app in Android or IOS device. The interface is graphical base and support drag and drop operations. It consists of different components, such as button, slider, date pickers, image, camera, sensors including Accelerometer and connectivity to web, etc. The app can be exported or published to Play store for our daily life.

For creating IoT application, you can use App Inventor 2 to create web connection with micro:bit. In WAN control, after micro:bit is connected to internet, it keep listening and execute customized operation when WAN command is received. The customized operation could be Pin On/off, such as “open/close LED”, or set value such as “set the light intensity” and “set current time”.



Know the API (control command)

What is API? API is one way to communicate between the objects in the Internet world. API is just like an "URL" which is website link. In App inventor, we use "API" to communicate with micro:bit. Normally, we need to know the (1) device ID (2) the message need to be sent to the micro:bit and optionally (3) value if needed.



(1) Basic API:

```
https://control.smarthon.cc/publish?id=DeviceID&msg=ControlCommand
```

`id`: The unique ID of device, used to identify the target . `msg`: The command needs to send. Example: "https://control.smarthon.cc/publish?id=0x123456781&msg=lighton". In this case, we send the msg command "lighton" to device id "0x123456781".

(2) Advanced API:

```
https://control.smarthon.cc/publish?id=DeviceID&msg=ControlCommand&value=Value
```

`id`: The unique ID of device, used to identify the target . `msg`: The command needs to send. `value`(optional): Used when need for the command. Example: "https://control.smarthon.cc/publish?id=0x123456781&msg=lighton&value=500" . In this case, we send the msg command "lighton" with intensity value "500" to device id "0x123456781".

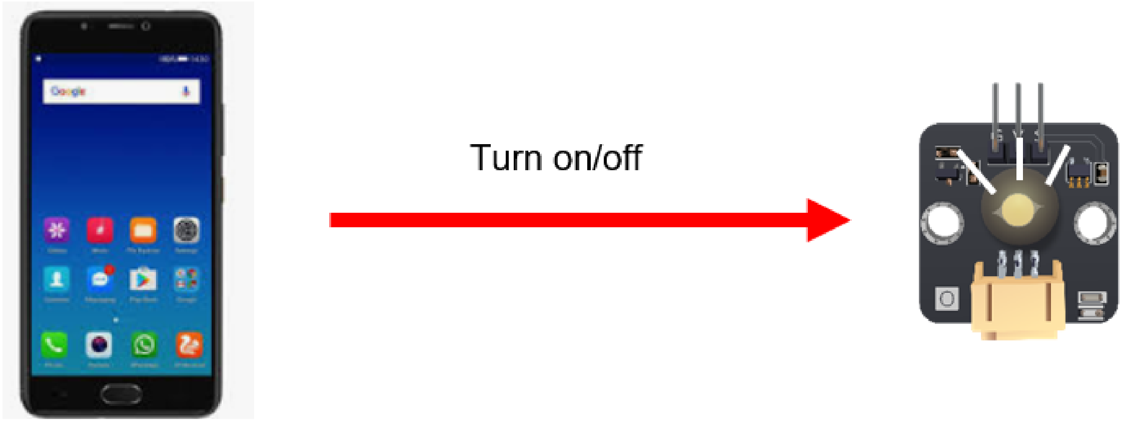
Scenario Example

This example is to turn on/off micro:bit LED by using app inventor 2.

Description:

In this example, there are 2 parts involved.

- In part 1, we need to connect the micro:bit to the internet and get the device ID.
- In part 2, design the mobile app and set the API with the device ID from part 1.

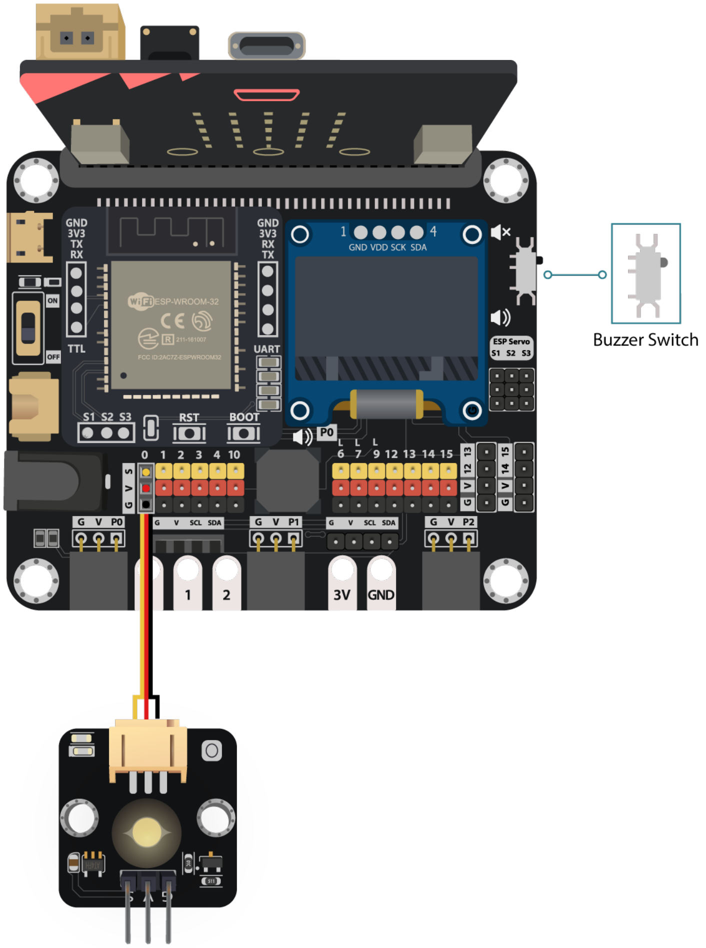


Part 1: Coding

Goal: We need to get the Device ID and set the corresponding action.

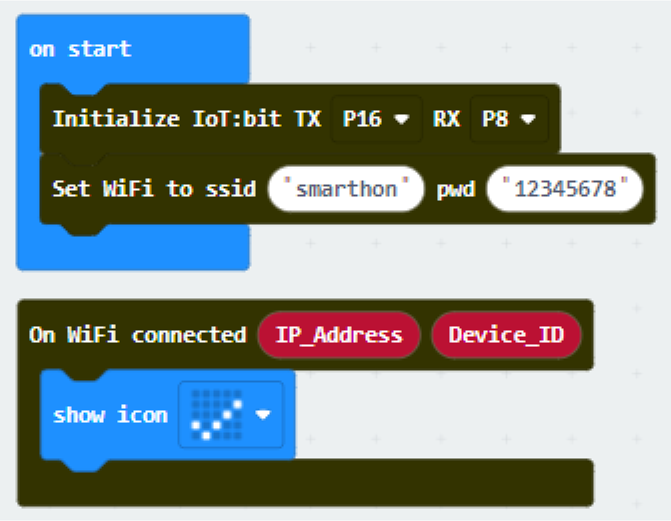
Connection Diagram:

- Connect LED to P0



Pull the buzzer switch 'up' to disconnect the buzzer in this execrise

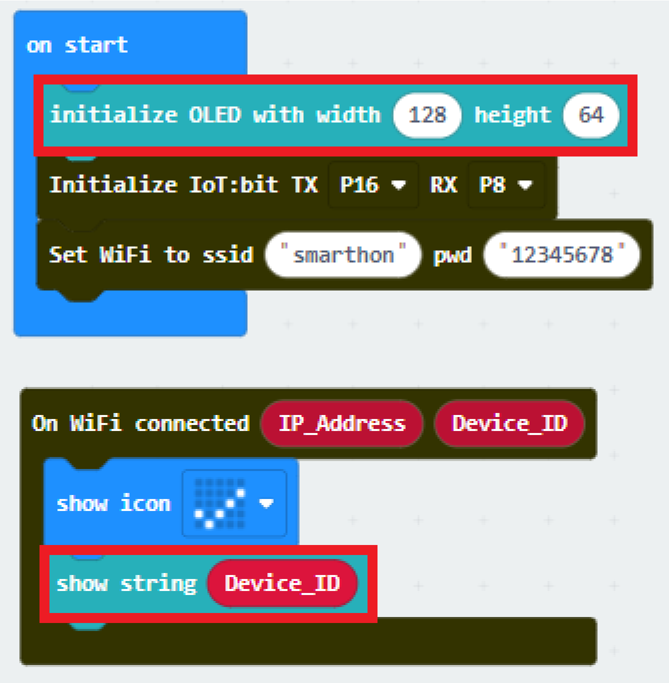
Step 1: Connect WiFI Before we try to use WiFi Control function, we need to connect to the network, we have already know how to connect to the WiFi on the first chapter.



```
on start
  Initialize IoT:bit TX P16 RX P8
  Set WiFi to ssid "smarthon" pwd "12345678"

On WiFi connected IP_Address Device_ID
  show icon
```

Step 2: Get Device ID `On WiFi connected` is an event handler. It will be triggered once after connected with WiFi. The handler will provide the `Device ID` variable which used to identify and control the Microbit.

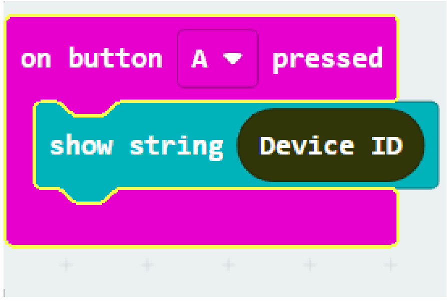
- Go to OLED Tab
- Snap `initialize OLED with width...height..` to `on start`
- Snap the `show string` inside the `On WiFi connected`
- Draw the `Device ID` variable from `On WiFi connected` to the `show tring` block placeholder



```
on start
  initialize OLED with width 128 height 64
  Initialize IoT:bit TX P16 RX P8
  Set WiFi to ssid "smarthon" pwd "12345678"

On WiFi connected IP_Address Device_ID
  show icon
  show string Device_ID
```

*If you worried about forget the `Device ID` during program running, you may access it by the variable under Control tab

- Go to Control tab
- Snap the `Device ID` variable to the placeholder

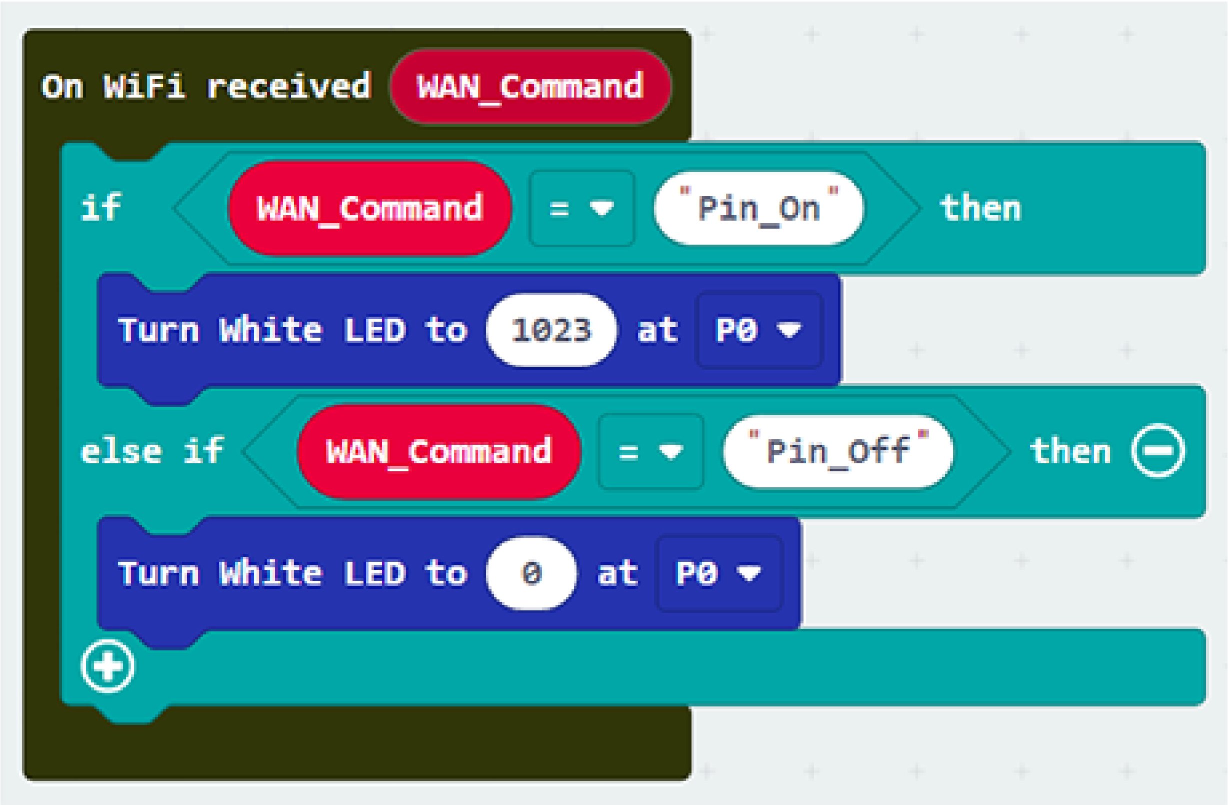


```
on button A pressed
    show string Device ID
```

Step 3: Control with Command After connected to the WiFI, the connection to the server will be done automatically, it is ready to receive command though network. To get the command, we can use the `on Wi-Fi received` handler in WAN control tab.

- Snap the `on WiFi received` handler to stage
- Do the `if-condition statement` to the variable `WAN_Command`
- If `Wan_command` "Pin_On" is received, white LED will be turned on (intensity:1023)
- If `Wan_command` "Pin_Off" is received, white LED will be turned off (intensity:0)

Attention: Please be aware that the **P** is in capital letter.

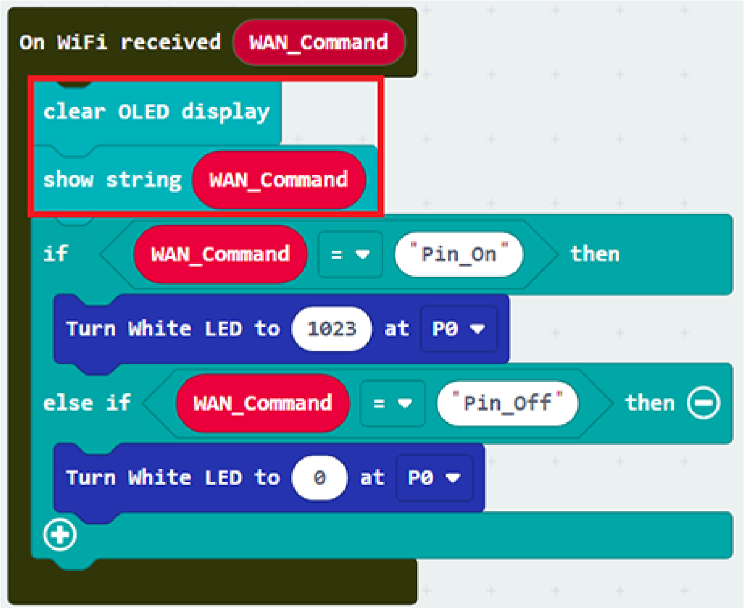


```
On WiFi received WAN_Command
    if WAN_Command = "Pin_On" then
        Turn White LED to 1023 at P0
    else if WAN_Command = "Pin_Off" then
        Turn White LED to 0 at P0
```

Step 4: Show the Command Sometimes you may need to show the recevied command for debugging, so if you need that, you can use the OLED `show string` to display the command on the OLED.

- Go to OLED
- Snap the `clear OLED display` to `On WiFi received` to avoid overlap
- Snap the `show string` to `On WiFi received`

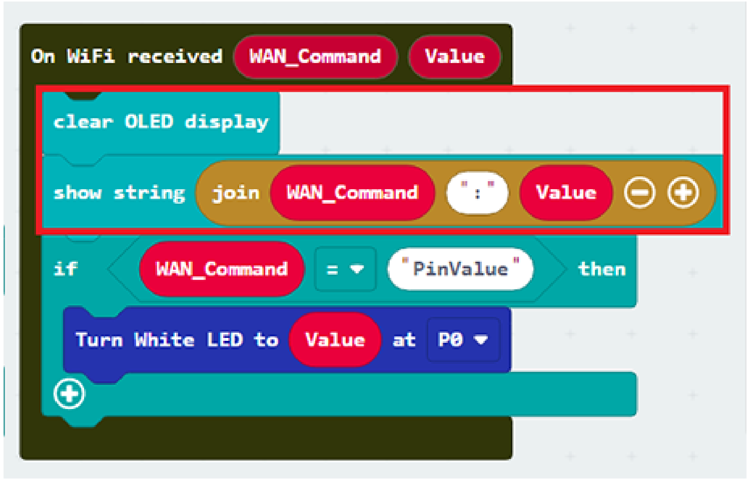
- Draw the `WAN_Command` variable to `show string` placeholder



```
On WiFi received WAN_Command
    clear OLED display
    show string WAN_Command
    if WAN_Command = ▾ "Pin_On" then
        Turn White LED to 1023 at P0 ▾
    else if WAN_Command = ▾ "Pin_Off" then ⊖
        Turn White LED to 0 at P0 ▾
    ⊕
```

Advanced Usage: Command with value If you want to control the module with value, you can use the another block which contain `value` variable. If `Wan command` "PinValue" is received, white LED will be turned on with the given intensity `value`.

You may also show the `WAN_Command` and `value` by `show string`.



```
On WiFi received WAN_Command Value
    clear OLED display
    show string join WAN_Command ":" Value ⊖ ⊕
    if WAN_Command = ▾ "PinValue" then
        Turn White LED to Value at P0 ▾
    ⊕
```

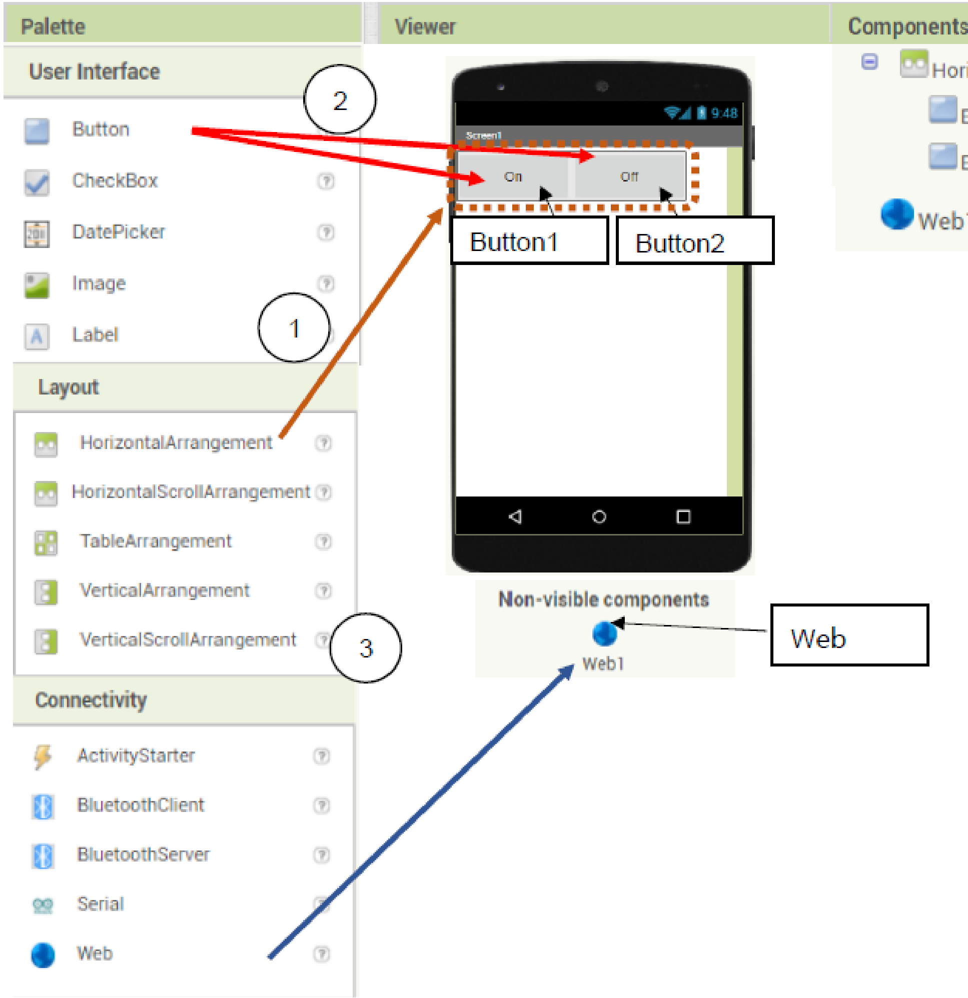
Full Solution MakeCode: https://makecode.microbit.org/_5i7gKJ4KVK9X You could also download the program from the following website:

Part 2: App Inventor 2 configuration

Goal: We need to create mobile app to control the IoT:bit by using API.

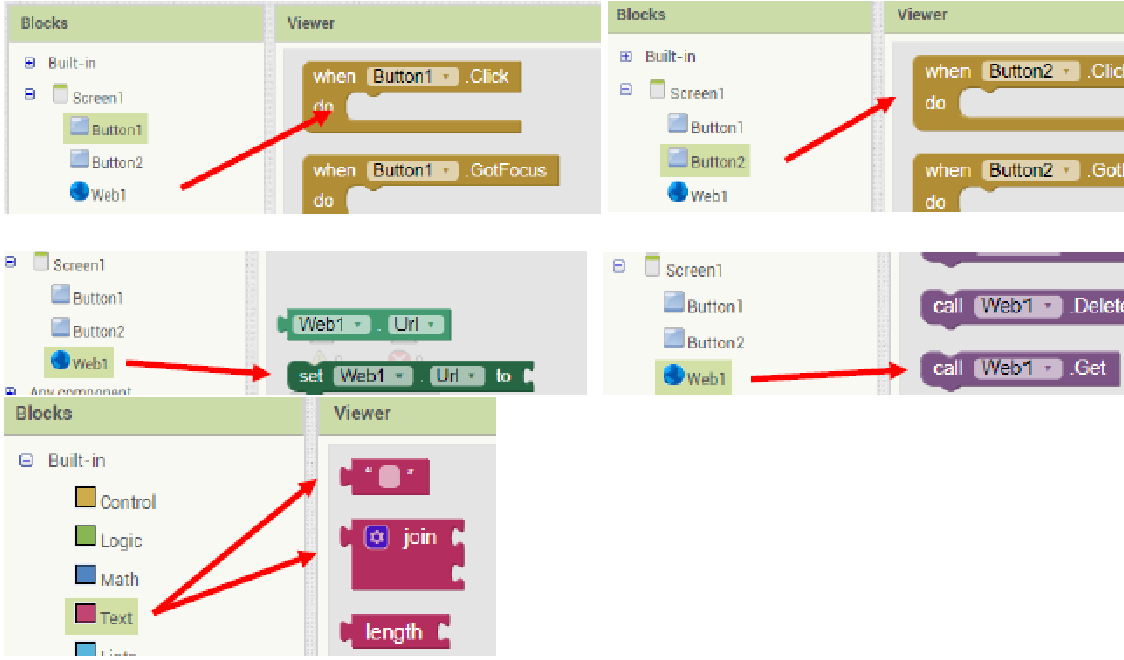
Step 1: Create Layout Create a new project in App Inventor 2, on "Designer"

- Drag "Layout">"HorizontalArragement" to the page.
- Drag "User Interface" > "Button" twice to the place inside the HorizontalArrangement

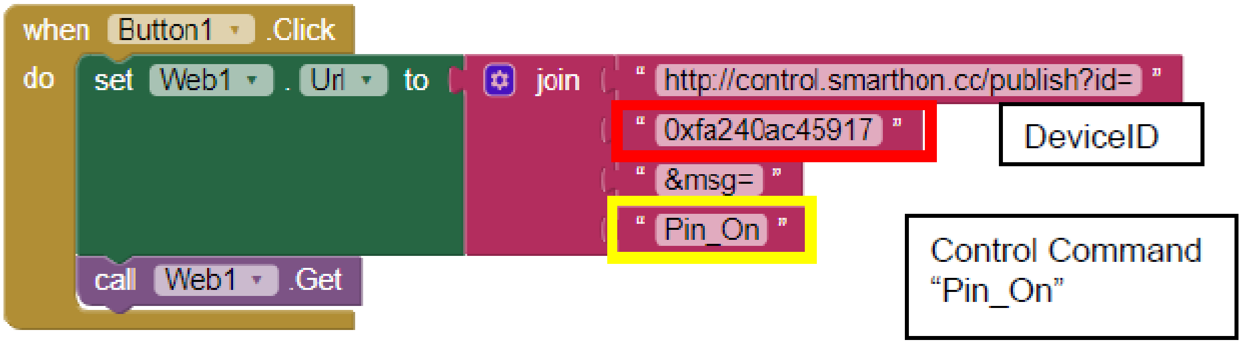


- Drag Connectivity> "Web" to the page

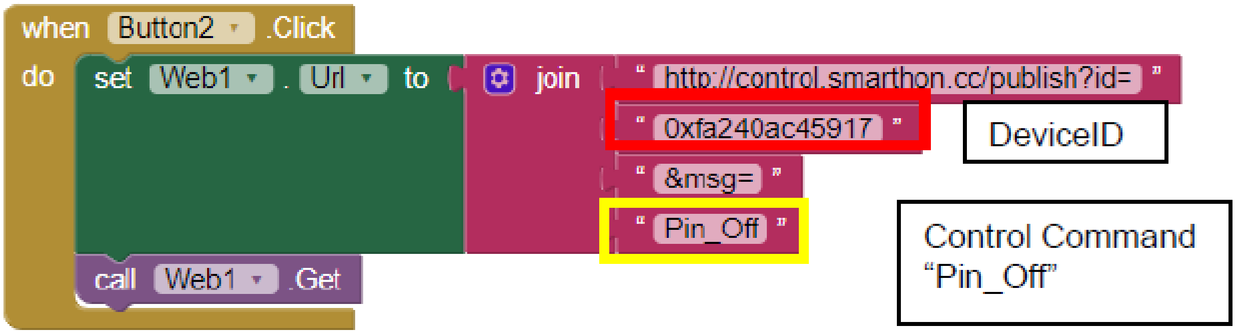
Step 2: Define Function Set the program in app inventor 2, on "Blocks"



- Take the block as below
- When Button 1 and 2 is pressed, it calls the WAN control API respectively http://control.smarthon.cc/publish?id=`DeviceID`&msg=`ControlCommand`

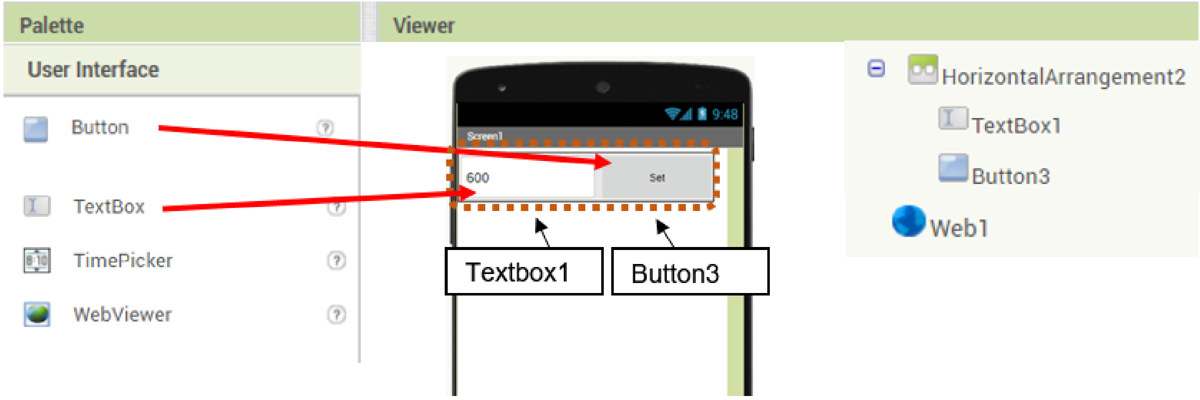


```
when Button1 .Click
do  set Web1 . Url to  join  " http://control.smarthon.cc/publish?id= "
                              " 0xfa240ac45917 "
                              " &msg= "
                              " Pin_On "
    call Web1 .Get
```



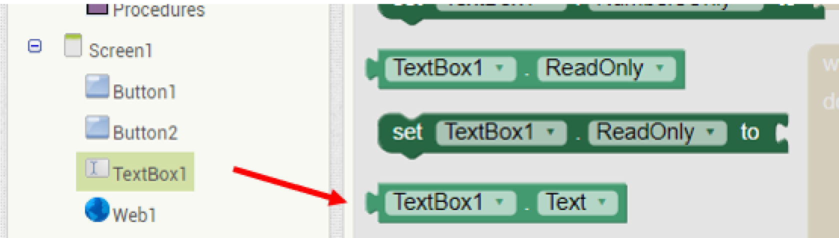
```
when Button2 .Click
do  set Web1 . Url to  join  " http://control.smarthon.cc/publish?id= "
                              " 0xfa240ac45917 "
                              " &msg= "
                              " Pin_Off "
    call Web1 .Get
```

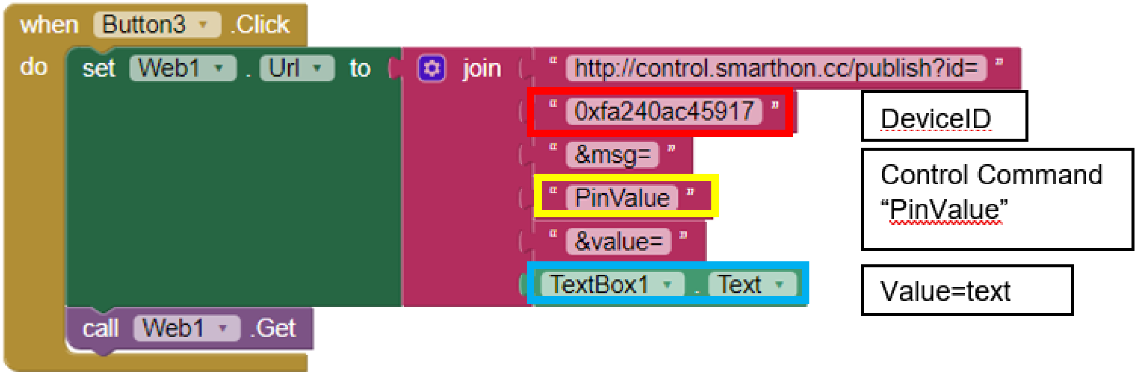
Advanced Usage: Control with value If you want to control the module with value, you can create in textbox with the button let the user to input the value.



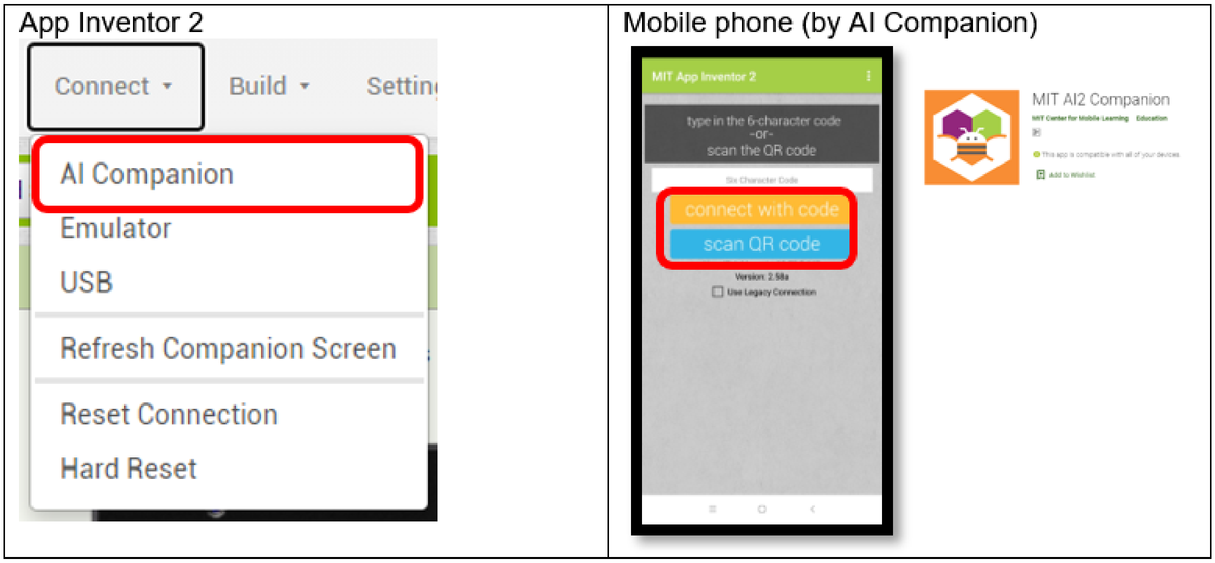
Layout:

Programming: When Button 3 is pressed, it calls the WAN control API respectively http://control.smarthon.cc/publish?id=`0xfa240ac45917`&msg=`PinValue`&value=`600`



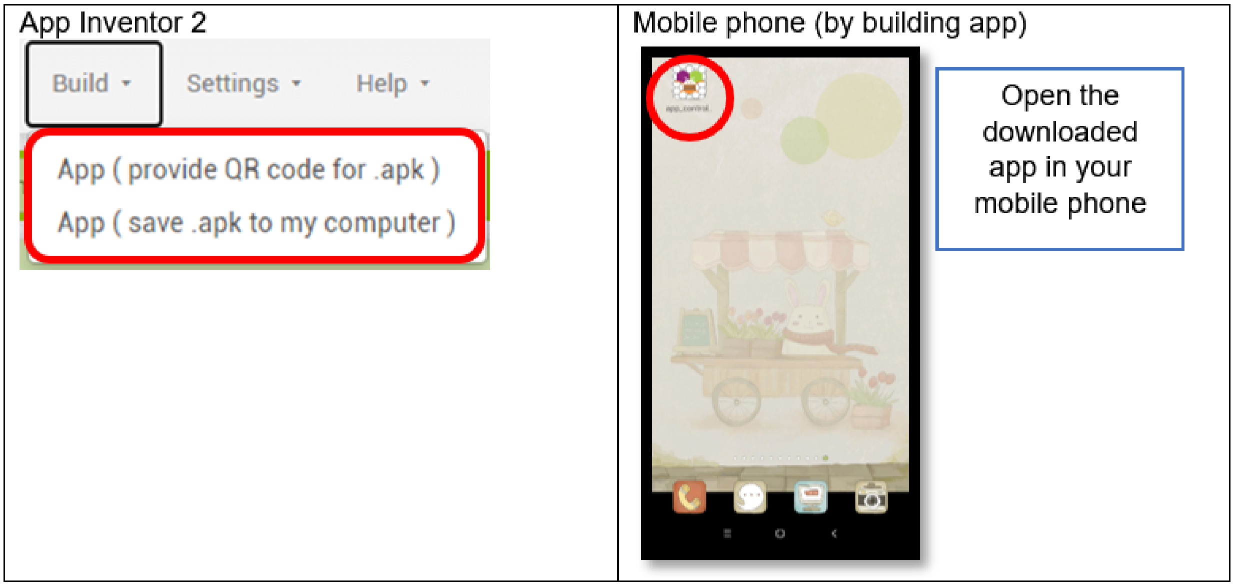


Step 3: Build the apps Build and test the mobile app. In App Inventor 2, you can perform real-time testing in your mobile phone by AI Companion


App Inventor 2

Mobile phone (by AI Companion)

Also, you can also build and download the android app and open it in your mobile phone.


App Inventor 2

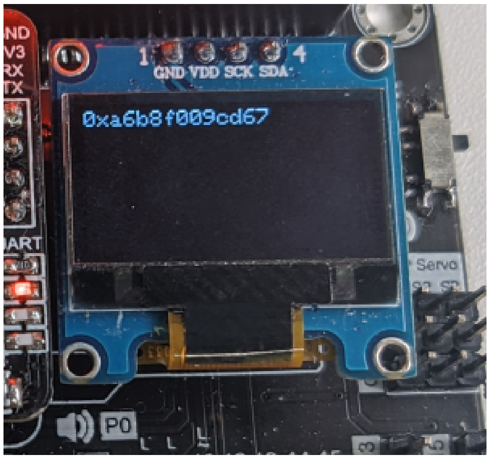
Mobile phone (by building app)

Build
Settings
Help
App (provide QR code for .apk)
App (save .apk to my computer)
app_control

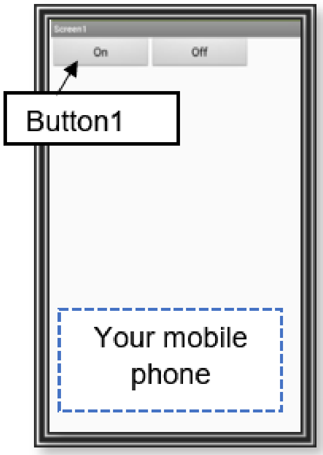
Open the downloaded app in your mobile phone

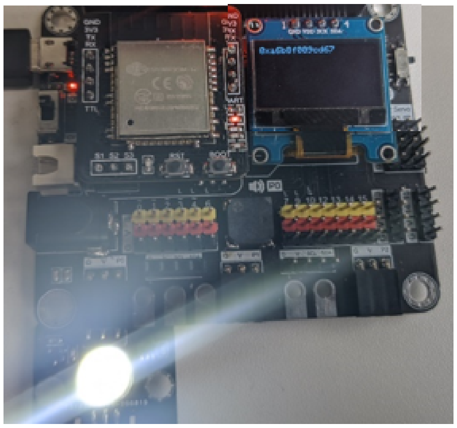
Result

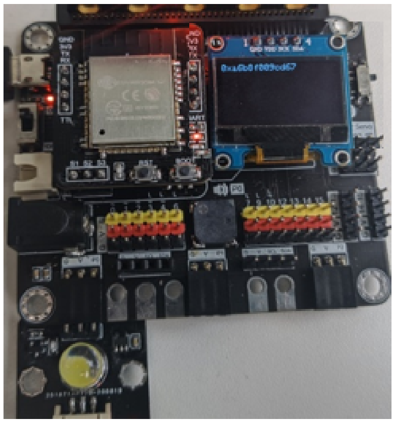
After the WiFI Connected, the Device ID will show



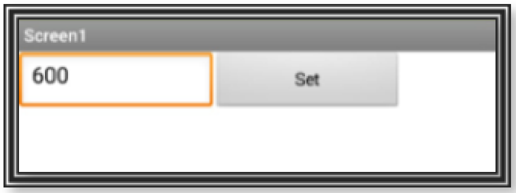
Normal Case: When Button1 is clicked, it will send WAN command "Pin_On" to the micro:bit with provided Device ID, the LED on P0 will be turned on. When button 2 is clicked, the LED will be turned off.





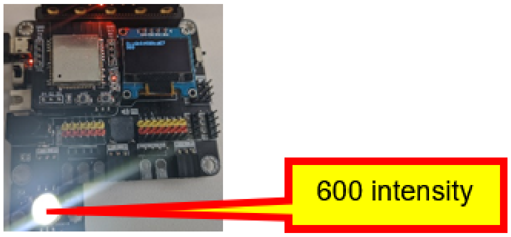


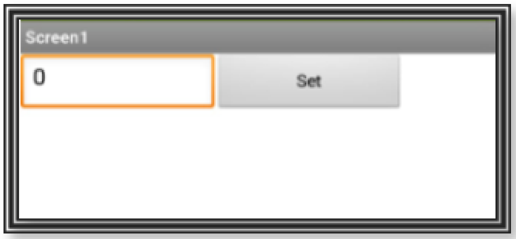
Control the value(advanced): When the button 3 is pressed with the intensity of 0 or 600, it will be turned on accordingly.



Screen1

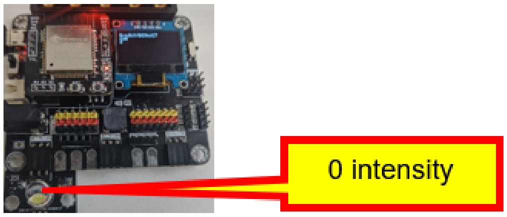
600





Screen1

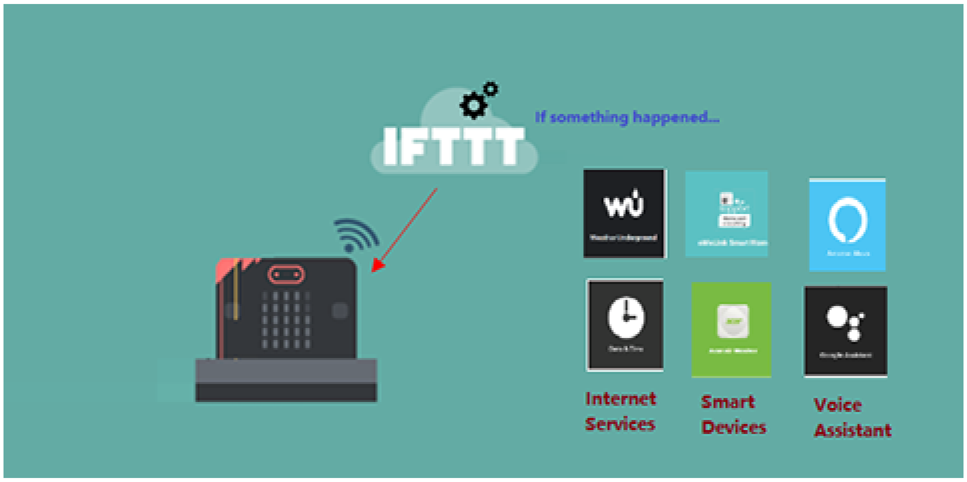
0



1.1.8 Chapter 4: Cloud Control micro:bit by IFTTT

Introduction

In this chapter, you will know how to control micro:bit from the IFTTT cloud. Once there are some changes on Internet services like weather, time or different connected devices like voice assistant and some smart home device, you can send the command to micro:bit to do the action.



In practice, the below example you may apply for:

- Turn the street light on at 6pm and off after 6am everyday
- Turn on the water pump every day at 06:00 for 2 second
- If the forecast weather is rain, open the umbrella
- Use voice command to control robot actions
- When the door is closed, LED is turned on.

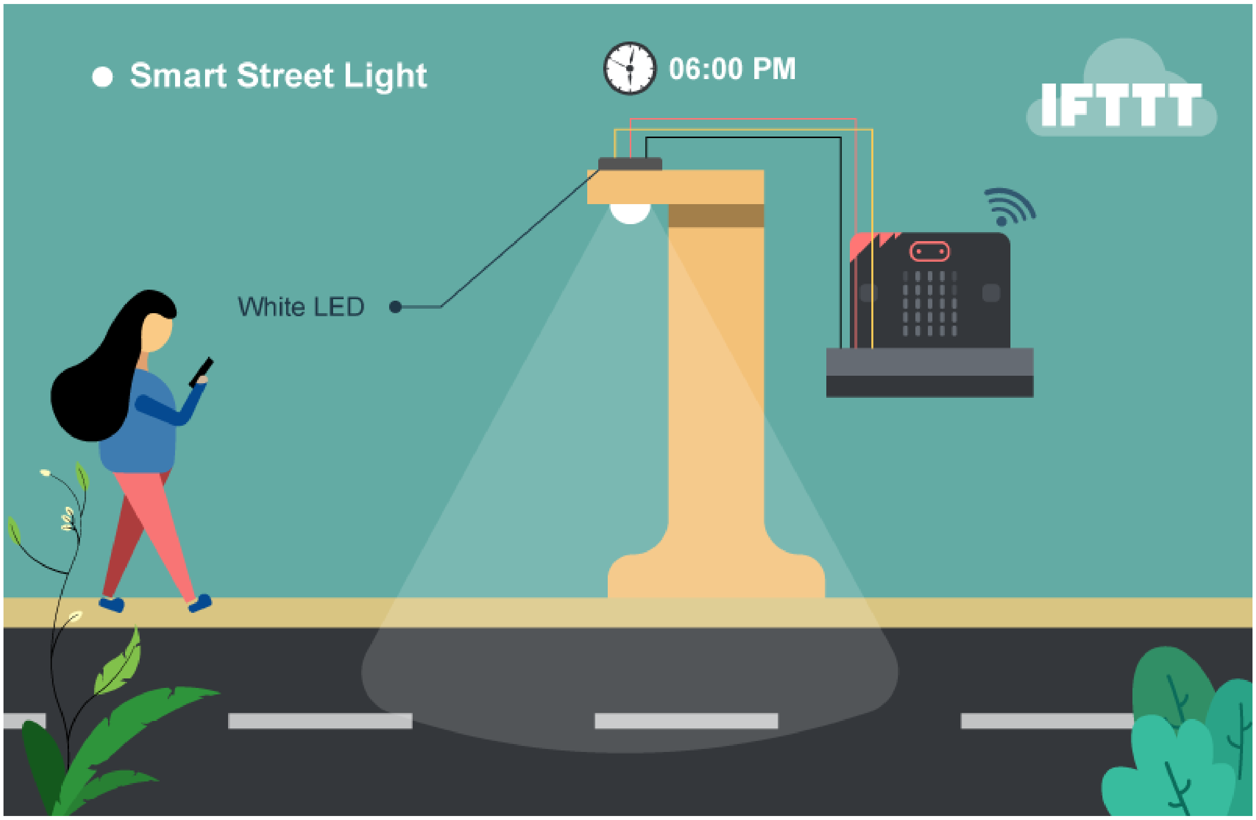
Scenario Example

The micro:bit is connected to the Internet. Once the time is changed to 6 pm, the IFTTT cloud will send the command to the micro:bit to turn on the LED.

Description:

In this example, there are 2 parts involved.

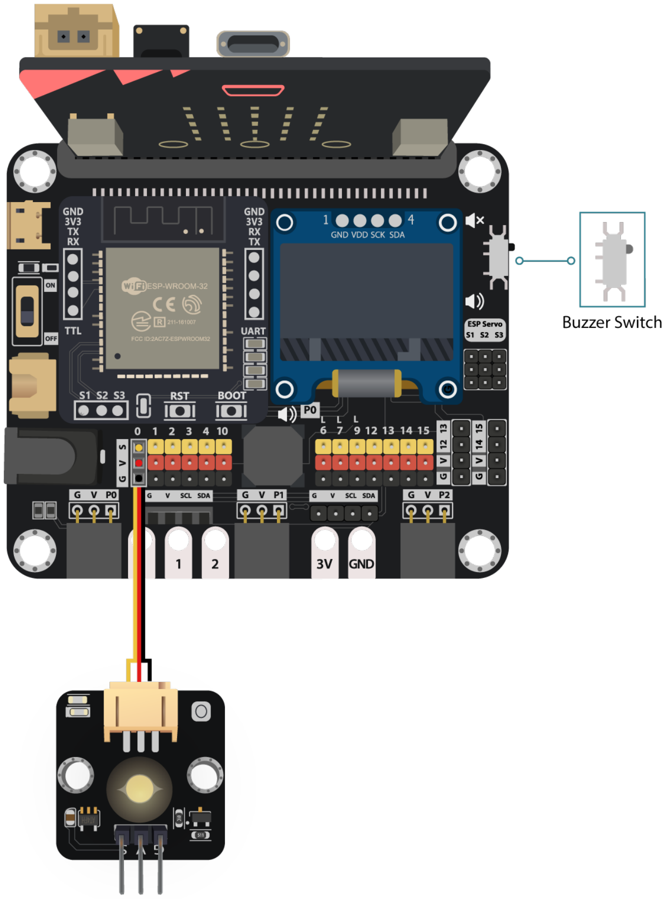
- In part 1, we need to connect the micro:bit to the internet and get the device ID.
- In part 2, set the rule on IFTTT applet. If the time is 6 pm, set the command "light_on"



Part 1: Coding

Goal: We need to get the Device ID and set the corresponding action.

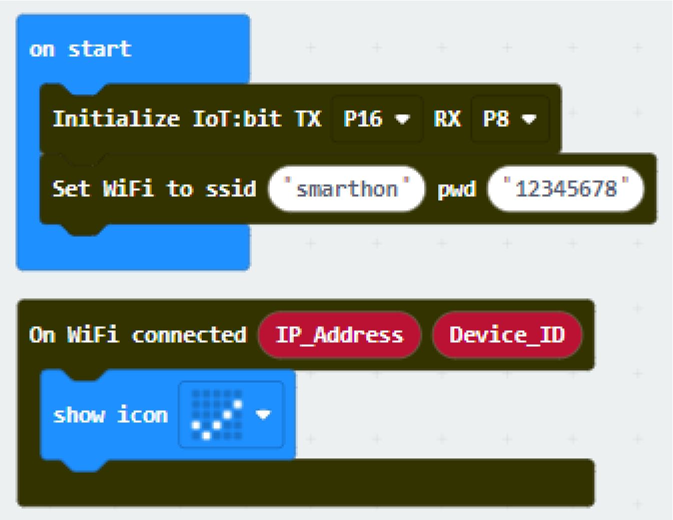
Connection Diagram:



- Connect LED to P0

Pull the buzzer switch 'up' to disconnect the buzzer in this execrise

Step 1: Connect WiFi Before we try to use WiFi Control function, we need to connect to the network, we have already

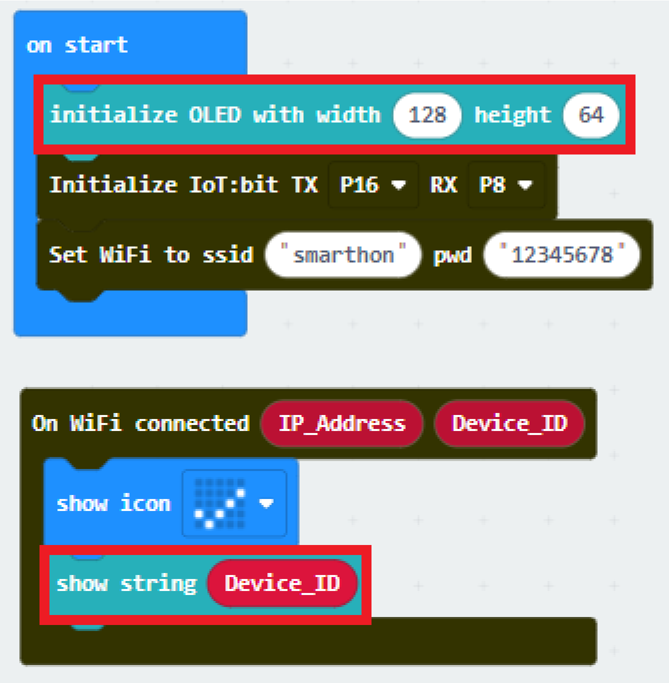


```
on start
    Initialize IoT:bit TX P16 ▾ RX P8 ▾
    Set WiFi to ssid "smarthon" pwd "12345678"

On WiFi connected IP_Address Device_ID
    show icon ▾
```

know how to connect to the WiFi on the first chapter.

Step 2: Get Device ID `On WiFi connected` is an event handler. It will be triggered once after connected with WiFi. The handler will provide the `Device ID` variable which used to identify and control the Microbit.

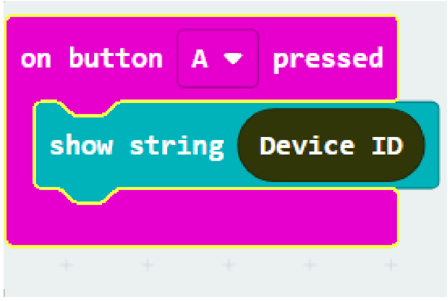
- Go to OLED Tab
- Snap `initialize OLED with width...height..` to `on start`
- Snap the `show string` inside the `On WiFi connected`
- Draw the `Device ID` variable from `On WiFi connected` to the `show string` block placeholder



```
on start
    initialize OLED with width 128 height 64
    Initialize IoT:bit TX P16 ▾ RX P8 ▾
    Set WiFi to ssid "smarthon" pwd "12345678"

On WiFi connected IP_Address Device_ID
    show icon ▾
    show string Device_ID
```

*If you worried about forget the `Device ID` during program running, you may access it by the variable under Control tab

- Go to Control tab



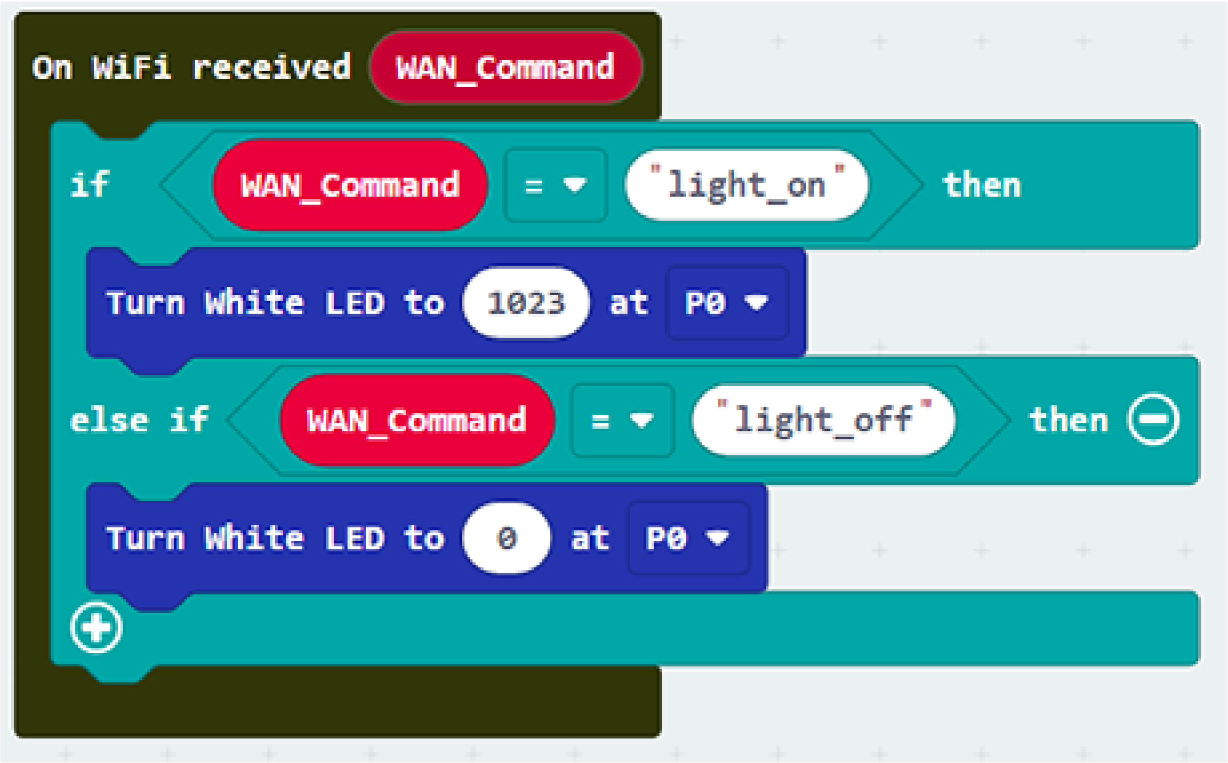
```
on button A ▾ pressed
    show string Device ID
```

- Snap the `Device ID` variable to the placeholder

Step 3: Control with Command

After connected to the WiFI, the connection to the server will be done automatically, it is ready to receive command though network. To get the command, we can use the `on Wi-Fi received` handler in WAN control tab.

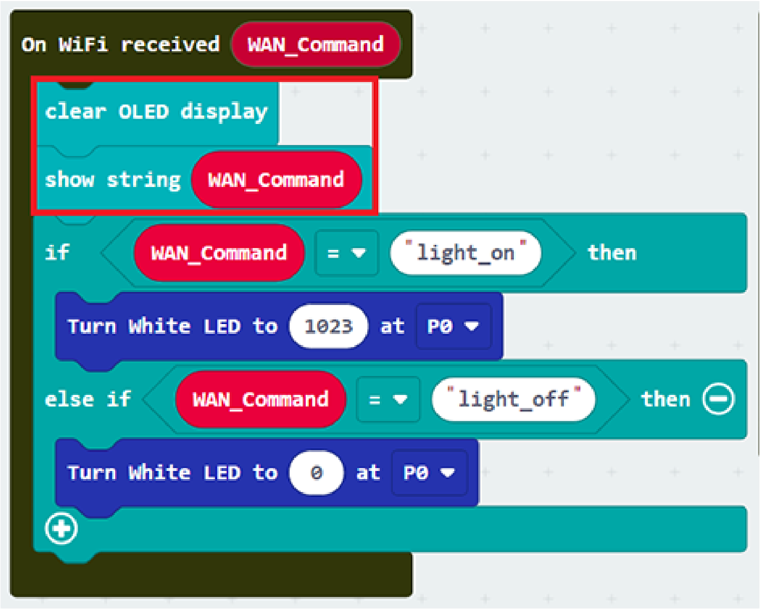
- Snap the `on WiFi received` handler to stage
- Do the `if-condition statement` to the variable `WAN_Command`
- If `Wan_command` "light_on" is received, white LED will be turned on (intensity:1023)
- If `Wan_command` "light_off" is received, white LED will be turned off (intensity:0)



```
On WiFi received WAN_Command
    if WAN_Command = ▾ "light_on" then
        Turn White LED to 1023 at P0 ▾
    else if WAN_Command = ▾ "light_off" then ⊖
        Turn White LED to 0 at P0 ▾
    ⊕
```

Step 4: Show the Command Sometimes you may need to show the recevied command for debugging, so if you need that, you can use the OLED `show string` to display the command on the OLED.

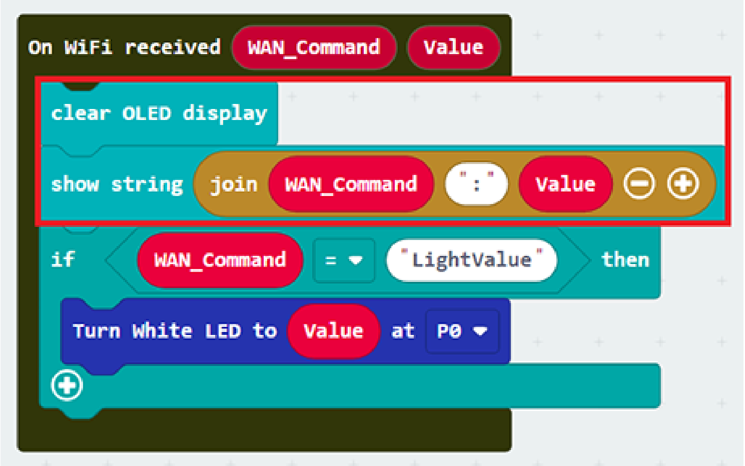
- Go to OLED
- Snap the `clear OLED display` to `On WiFi received` to avoid overlap
- Snap the `show string` to `On WiFi received`
- Draw the `WAN_Command` variable to `show string` placeholder



```
on WiFi received WAN_Command
    clear OLED display
    show string WAN_Command
    if WAN_Command = "light_on" then
        Turn White LED to 1023 at P0
    else if WAN_Command = "light_off" then
        Turn White LED to 0 at P0
```

Advanced Usage(with value):

If you want to control the module with value, you can use the another block which contain `value` variable. If `Wan command` "LightValue" is received, white LED will be turned on with the given intensity `value`. You may also show the `WAN_Command` and `value` by `show string`.



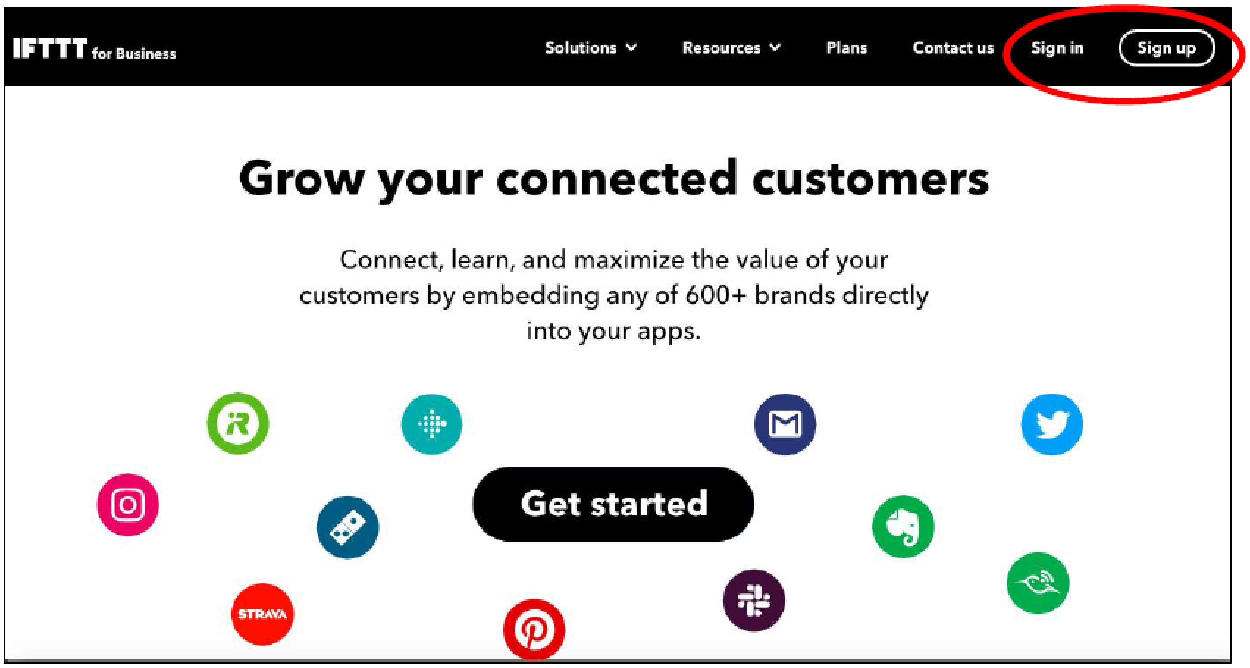
```
on WiFi received WAN_Command Value
    clear OLED display
    show string join WAN_Command ":" Value
    if WAN_Command = "LightValue" then
        Turn White LED to Value at P0
```

Full Solution MakeCode: https://makecode.microbit.org/_Jdb8fA66r77Y You could also download the program from the following website:

Part 2: IFTTT Applet Configuration

Goal: We need to setup the IFTTT Applet to control the Micro:bit

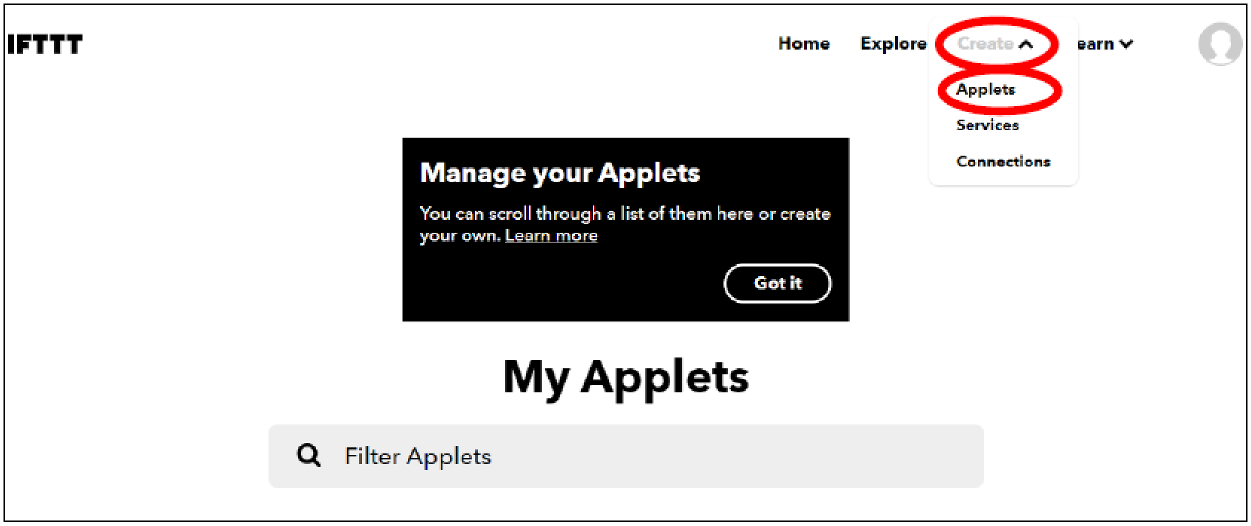
Step 1: Create or login to IFTTT account Open your browser , go to https://ifttt.com/. Register your IFTTT account and once completed, log in to your IFTTT account.



Grow your connected customers

Connect, learn, and maximize the value of your customers by embedding any of 600+ brands directly into your apps.

Step 2: Create Applets On the top right menu, click “Create” > “Applets”



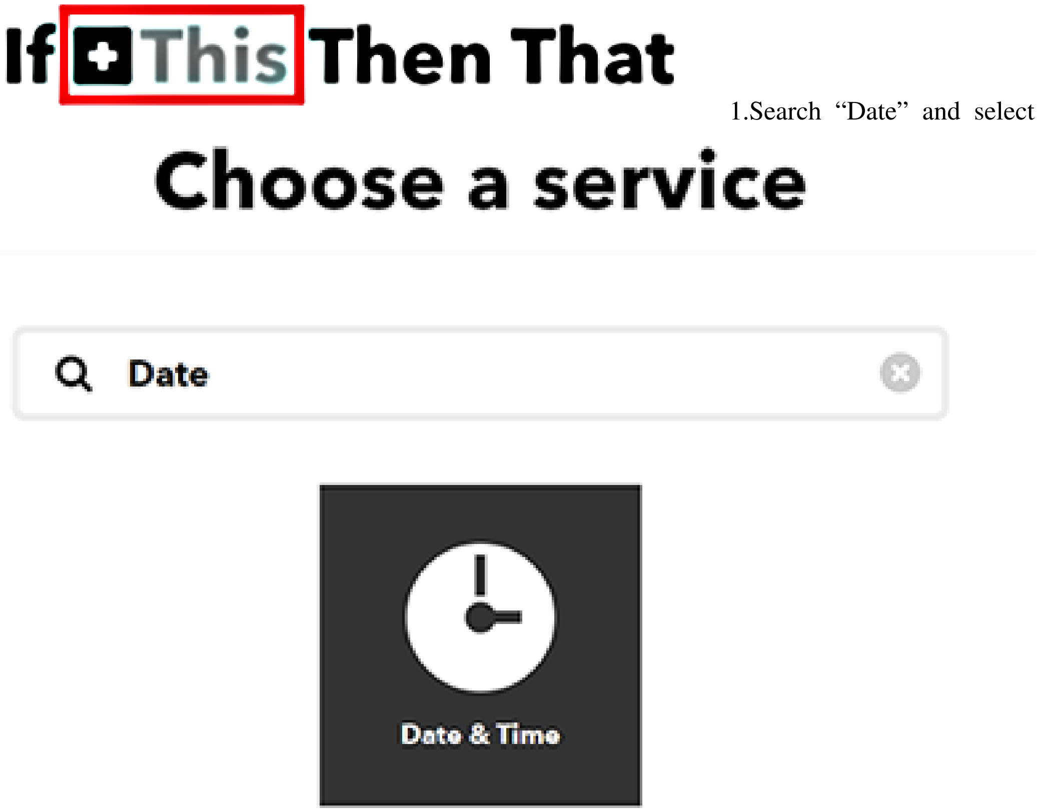
Manage your Applets

You can scroll through a list of them here or create your own. Learn more

My Applets

Step 3: Set the Rules On IFTTT configuration, set the below. We can set the rule on the IFTTT, if something happened like time at 6:00 p.m., we can do the corresponding action.

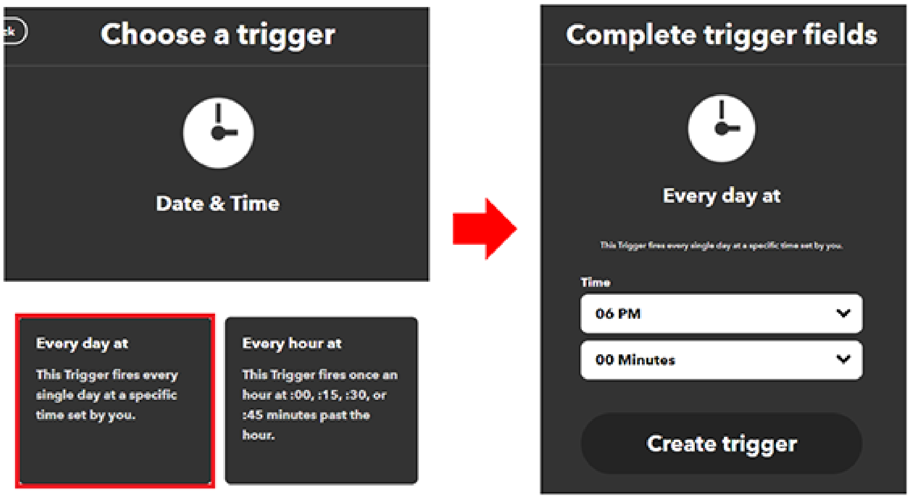
IFTTT Applet: 	In this example: If “it is 6pm today”, then “turn on the light”



A. Trigger Part 1.Search “Date” and select “Date & Time”

2.Finish the time setting.

- Choose trigger “Every day at”
- Select the time (e.g. 06PM, 00Minutes; it menas every day at 6pm) and click “Create trigger”



Every day at

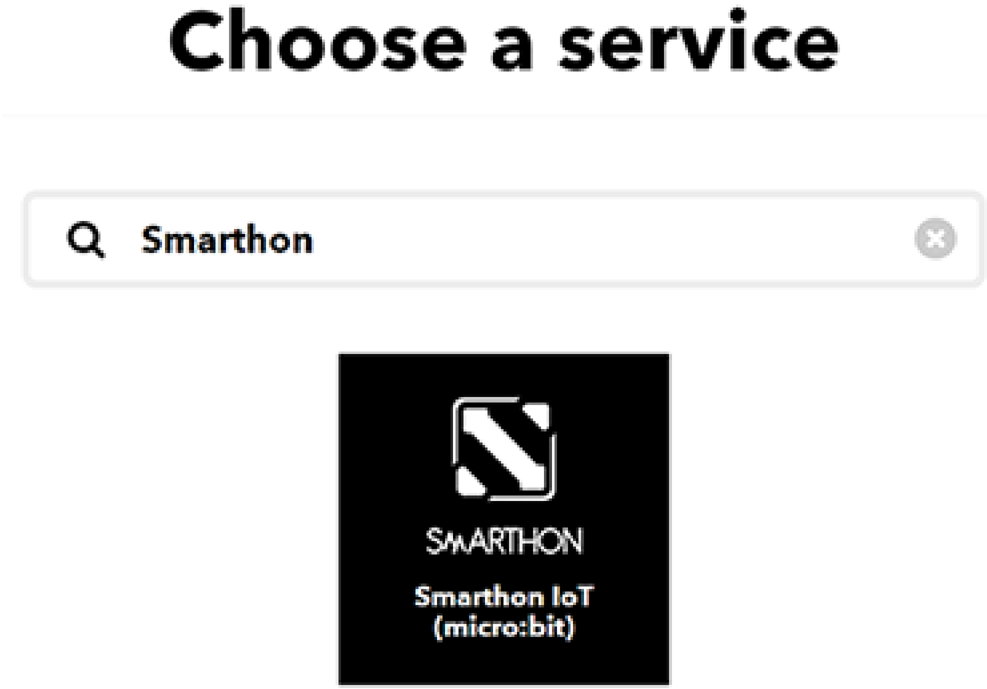
This Trigger fires every single day at a specific time set by you.

Every hour at

This Trigger fires once an hour at :00, :15, :30, or :45 minutes past the hour.

This Trigger fires every single day at a specific time set by you.

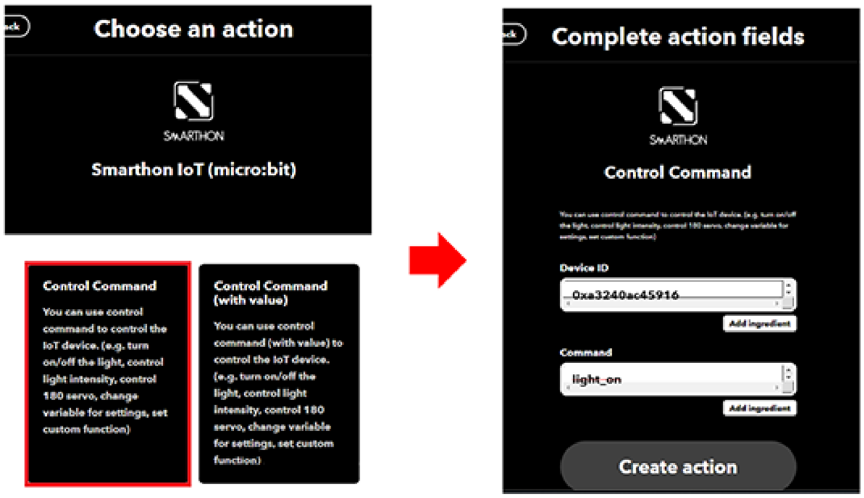
B. Action Part



1.Search “micro:bit” and you will see “Smarthon IoT(micro:bit)”

2.Finish the Micro:bit setting.

- Choose action “Control Command”
- Input your `Device ID` (e.g. 0xa3240ac45916) and `control command` (e.g. light_on). Then click “Create action”



Control Command
You can use control command to control the IoT device. (e.g. turn on/off the light, control light intensity, control 180 servo, change variable for settings, set custom function)

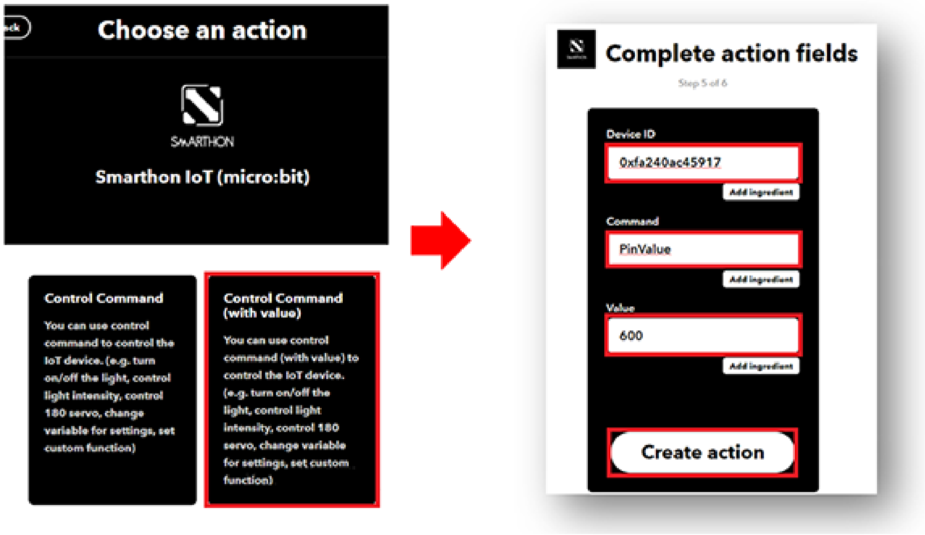
Control Command (with value)
You can use control command (with value) to control the IoT device. (e.g. turn on/off the light, control light intensity, control 180 servo, change variable for settings, set custom function)

Control Command
You can use control command to control the IoT device. (e.g. turn on/off the light, control light intensity, control 180 servo, change variable for settings, set custom function)

Device ID: 0xa3240ac45916

Command: light_on

Optional If you need to send the command with value to the micro:bit



Choose an action

SMARTHON

Smarthon IoT (micro:bit)

Control Command

You can use control command to control the IoT device. (e.g. turn on/off the light, control light intensity, control 180 servo, change variable for settings, set custom function)

Control Command (with value)

You can use control command (with value) to control the IoT device. (e.g. turn on/off the light, control light intensity, control 180 servo, change variable for settings, set custom function)

Complete action fields

Step 5 of 6

Device ID

0xfa240ac45917

Add ingredient

Command

PinValue

Add ingredient

Value

600

Add ingredient

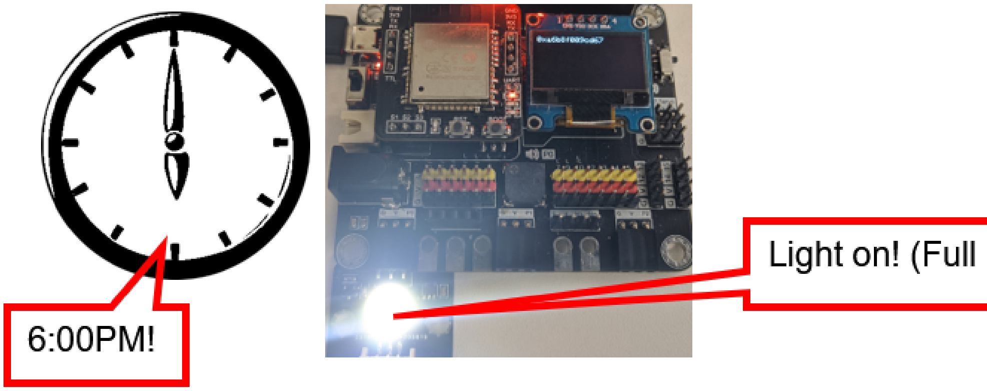
Create action

Result

Normal Case:

After connected to WiFi, micro:bit will connect to control server and keep listening to the WAN command

- Every day at 6pm, clock will trigger IFTTT to send out micro:bit WAN command "light_On"
- Micro:bit will recevie the command

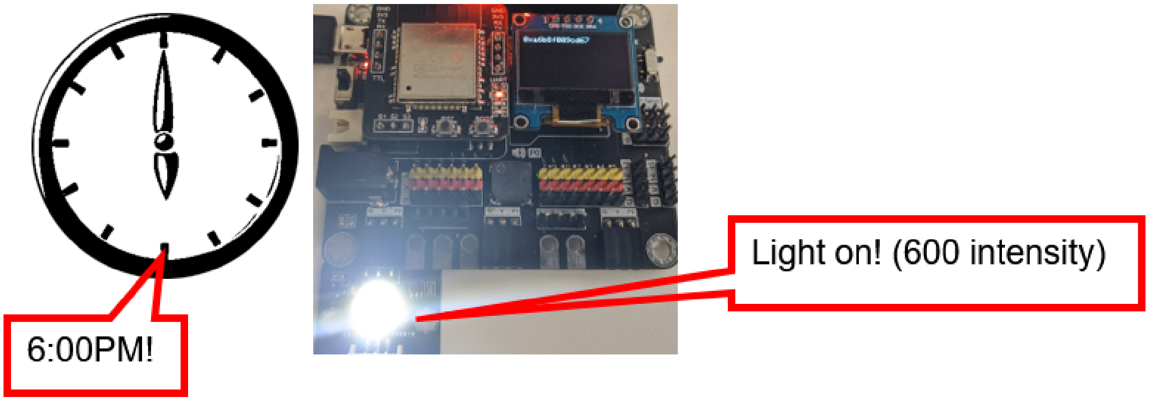


- The LED on P0 will be turned on.

Advanced Usage(with value):

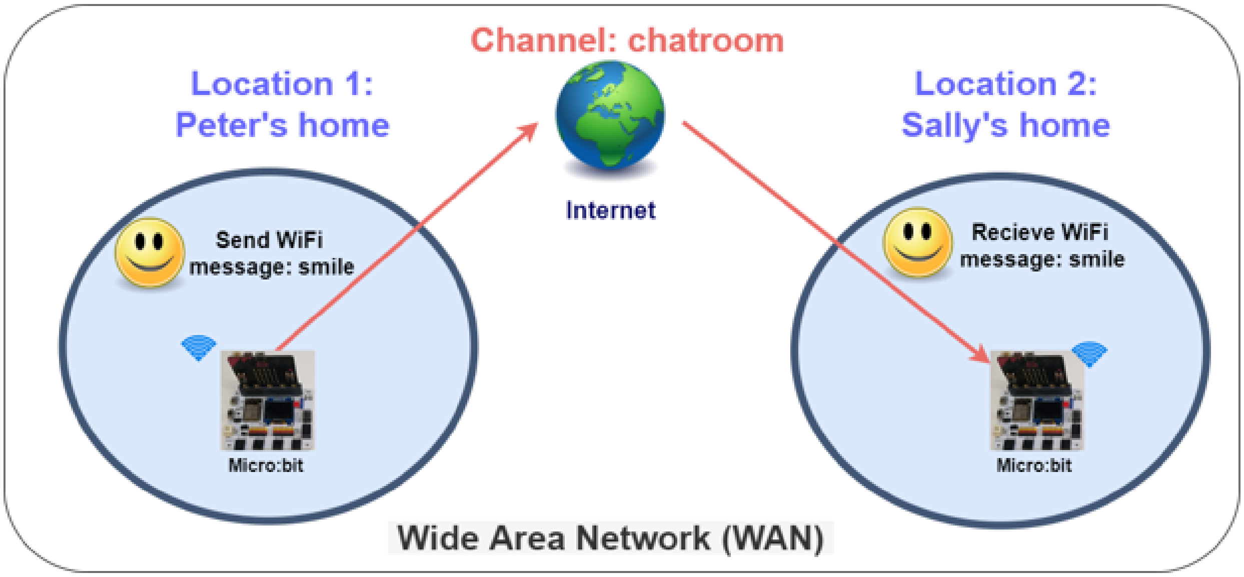
When change the applet to contain value version, it will have an additional variable `value` included

- Every day at 6pm, clock will trigger IFTTT to send out micro:bit WAN command "LightValue" (with value "600")
- Micro:bit will recevie the command
- The LED on P0 will be turned on (with light intensity: 600).



1.1.9 Chapter 5: Object to Object communication

In this example, micro:bit (sender and receiver) are connected to a channel called “chatroom”. Peter want to send a smile message to Sally via internet (WAN). Peter’s micro:bit will send a Wi-Fi message “smile” to Sally’s micro:bit. When Sally receive a WiFi message smile, her micro:bit LED will show a smile icon.

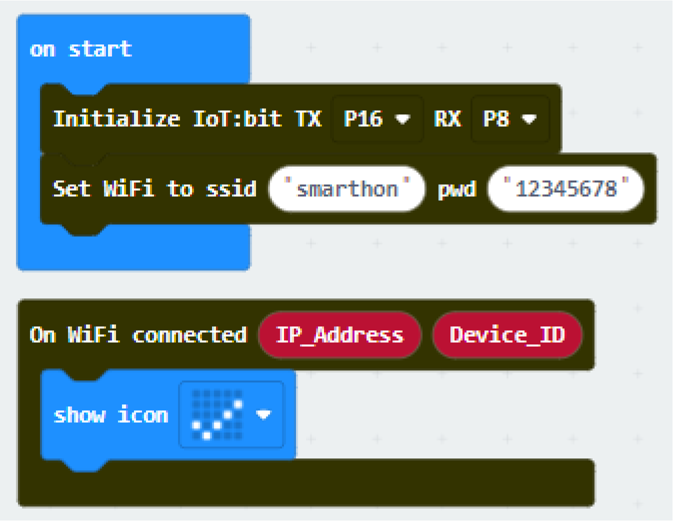


*Before we control the micro:bit, please make sure your micro:bit is connected to the Wi-Fi.

Receiver side

Goal: The receiver joins the channel, set the corresponding action when message received.

Step 1: Connect Wi-Fi Before we join the channel, we need to connect to the network first. We have already known

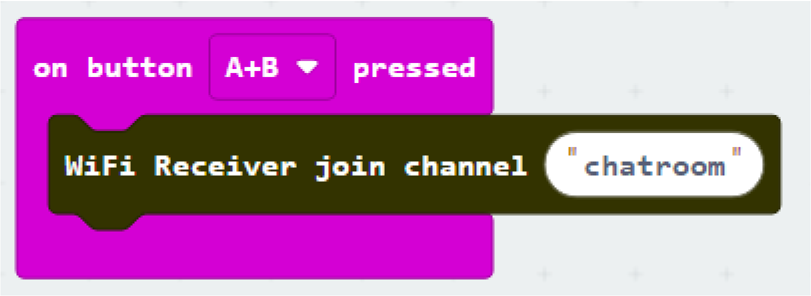


```
on start
    Initialize IoT:bit TX P16 ▾ RX P8 ▾
    Set WiFi to ssid "smarthon" pwd "12345678"

On WiFi connected IP_Address Device_ID
    show icon ▾
```

how to connect to the Wi-Fi on the first chapter.

Step 2: Join the channel We can let the Micro:bit listening on specific channel to receive the command

- Snap the `on button A+B pressed` to stage
- Put the `WiFi Receiver join channel` to the `on button A+B pressed`
- Type the name of the channel "Chatroom"

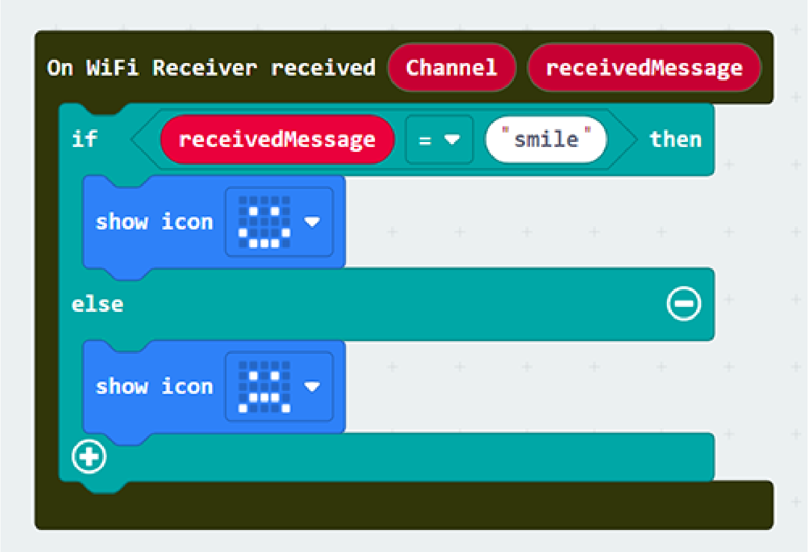


```
on button A+B ▾ pressed
    WiFi Receiver join channel "chatroom"
```

*The user can join more than 1 channel, the maximum is 4.

Step 3: Set the corresponding action After press the A and B button at the same time, Micro:bit will start listening on channel "chatroom" to wait other people send message to it. We need to use the `On WiFi Receiver received` to read the command and take action.

- Snap `On WiFi Receiver received` to stage
- If need to received include value message, can use the include value version `On WiFi Receiver received` at the same time
- Do the `if-condition statement` to the variable `receivedMessage`
- When the `receivedMessage` called "smile", the LED will show a smile icon;
- When the `receivedMessage` not called "smile", the LED will show a sad icon.



```
On WiFi Receiver received Channel receivedMessage
    if receivedMessage = "smile" then
        show icon
    else
        show icon
```

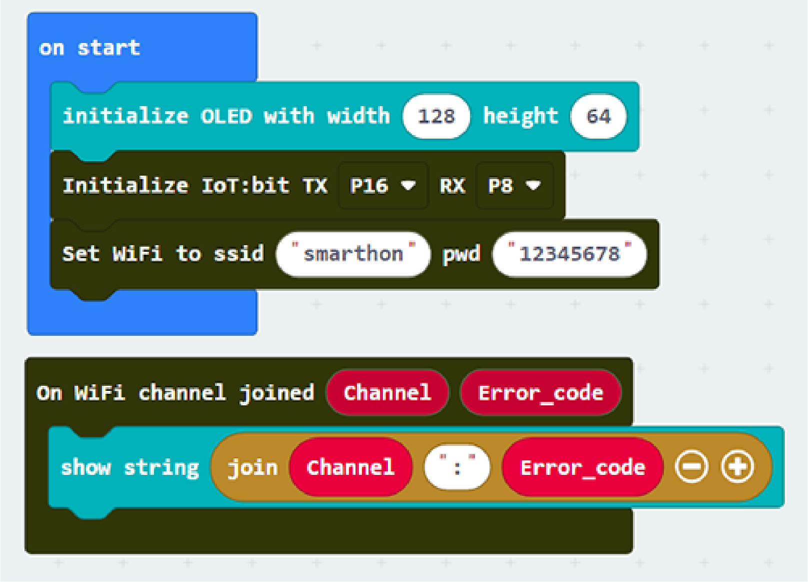
Full Solution MakeCode: https://makecode.microbit.org/_KbxHgu7xfHhP You could also download the program from the following website:

Optional:

When the Receiver was joined the channel, it can check the listening was started successfully or not. The `On WiFi channel joined` will provide the `Error_code` for you to determine the result.

- Go to the OLED Tab
- Snap `initialize OLED ...` to `on Start`
- Snap the `On WiFi channel joined` to stage
- Use `show string` to show the `Error_code` with `channel` name

If `Error_code` is 0, means no error occur If `Error_code` is not 0, means error occur Normally, if no internet connection or the Channel Server is down, the code will be -28674

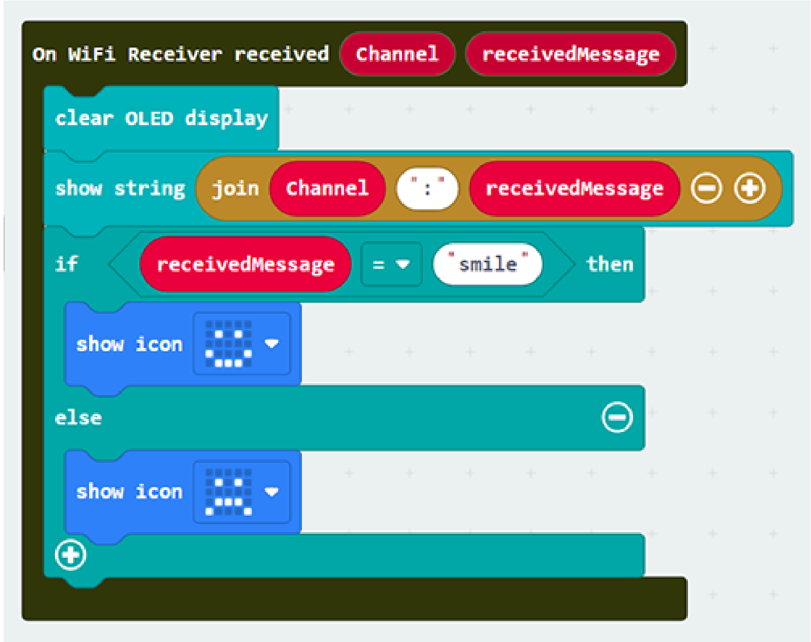


```
on start
    initialize OLED with width 128 height 64
    Initialize IoT:bit TX P16 RX P8
    Set WiFi to ssid "smarthon" pwd "12345678"

On WiFi channel joined Channel Error_code
    show string join Channel ":" Error_code
```

2. Show the source channel name of the message

When you join different channels, you may receive same command from each of them alternately, so you need to identify the message is come from which channel. The `On WiFi Receiver received` provide the variable used to identify.

- Snap `clear OLED display` to make sure the display will not overlap.
- Snap `show string` to the `On WiFi Receiver received`
- Draw the `Channel` and `receivedMessage` variable to the `show string` to display the on OLED

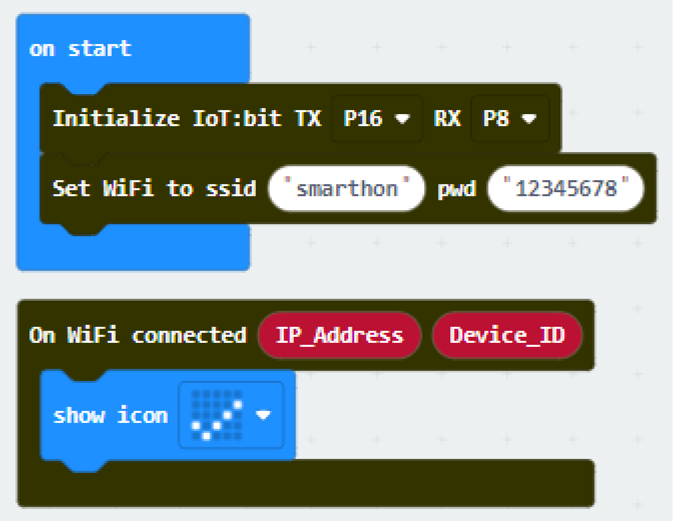


```
On WiFi Receiver received  Channel  receivedMessage
    clear OLED display
    show string  join  Channel  ":"  receivedMessage
    if  receivedMessage = "smile"  then
        show icon
    else
        show icon
```

Full Solution MakeCode: https://makecode.microbit.org/_ApFMuua7d072 You could also download the program from the following website:

Sender side

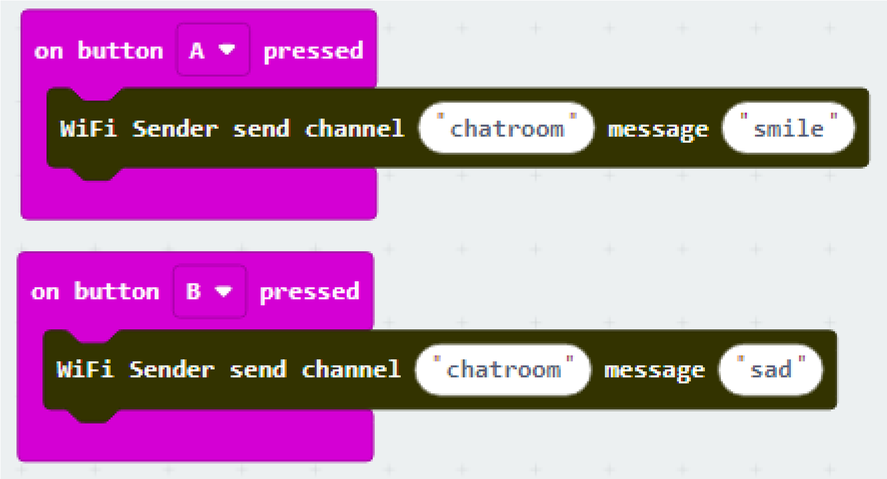
Step 1: Connect WiFi Before we try to use WiFi Control function, we need to connect to the network, we have already know how to connect to the WiFi on the first chapter.



```
on start
    Initialize IoT:bit TX  P16  RX  P8
    Set WiFi to ssid  "smarthon"  pwd  "12345678"

On WiFi connected  IP_Address  Device_ID
    show icon
```

Step 2: Send the message After connected to the internet, we can start send the message to our friend though the channel.

- Snap the `on button ... press` to stage
- Snap the `WiFi Sender send channel...` inside the `on button ... press`
- Input the correct channel name for your target channel, for example chatroom
- Input the message which need to be send
- If need to send the message with value, use another version of `WiFi Sender send channel...` which **include the value input**



```
on button A ▾ pressed
    WiFi Sender send channel "chatroom" message "smile"

on button B ▾ pressed
    WiFi Sender send channel "chatroom" message "sad"
```

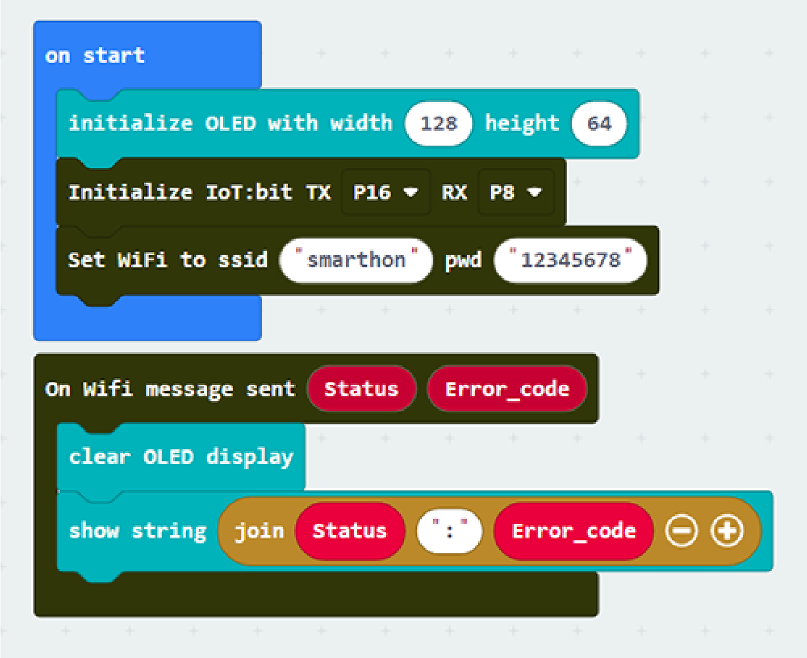
Full Solution MakeCode: https://makecode.microbit.org/_AygdmqgfTCXb You could also download the program from the following website:

Optional:

When the sender sending the message, if use the `On WiFi message sent`, it can know the sending result of message to channel.

- Go to OLED Tab
- Snap the `initialize OLED...` to `on Start`
- Snap the `On Wifi message sent` to stage
- Snap the `show string` inside to `On Wifi message sent`
- Snap the `clear OLED display` to avoid overlap
- Draw the `Status` and `Error_code` variable into the `show string` text placeholder

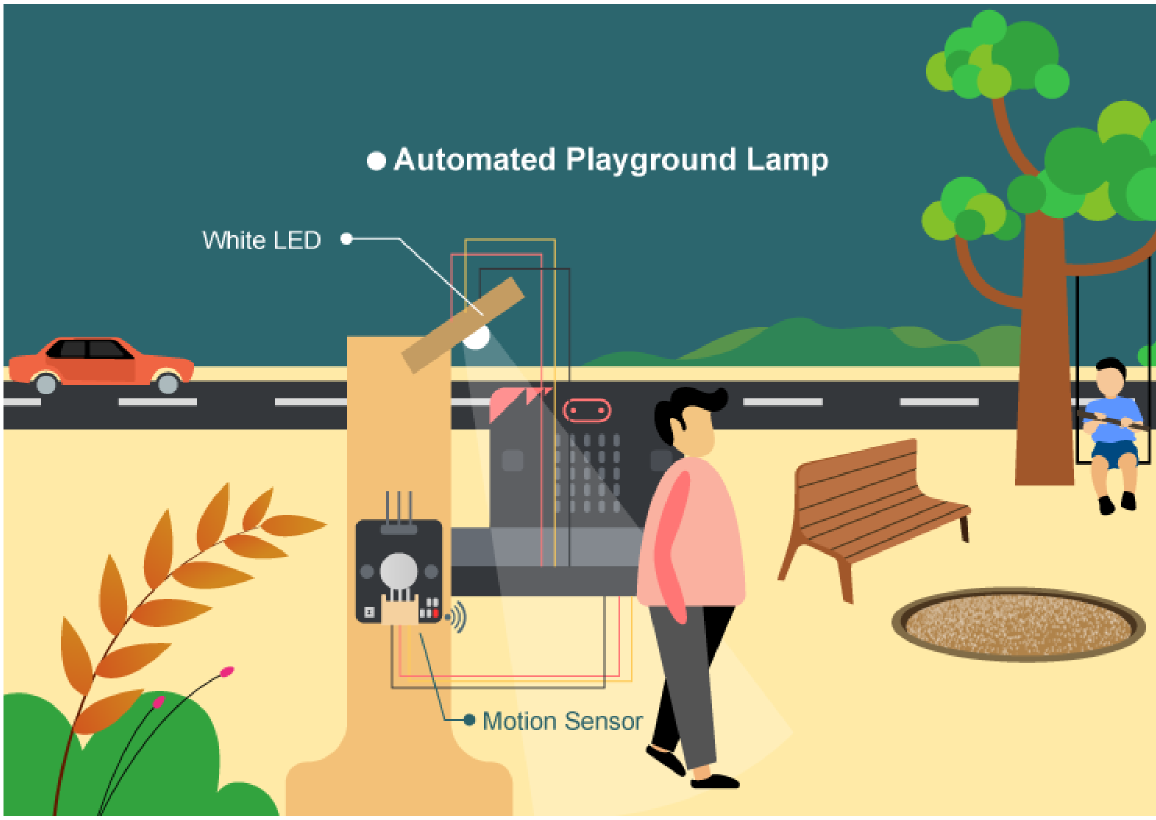
If the sending result is success, the `Status` will be `OK` and `Error_code` is `0` If the sending result is fail, the `Status` will be `FAIL` with `Error_code` Normally, if no internet connection or the channel Server is down, it will be -28674



```
on start
    initialize OLED with width 128 height 64
    Initialize IoT:bit TX P16 ▾ RX P8 ▾
    Set WiFi to ssid "smarthon" pwd "12345678"

On Wifi message sent Status Error_code
    clear OLED display
    show string join Status ":" Error_code ⊖ ⊕
```

Full Solution MakeCode: https://makecode.microbit.org/_2H1fHvgMv6jg You could also download the program from the following website:

1.1.10 Case 01: Automated Smart Playground Lamp



Level: ★ ☆ ☆ ☆ ☆

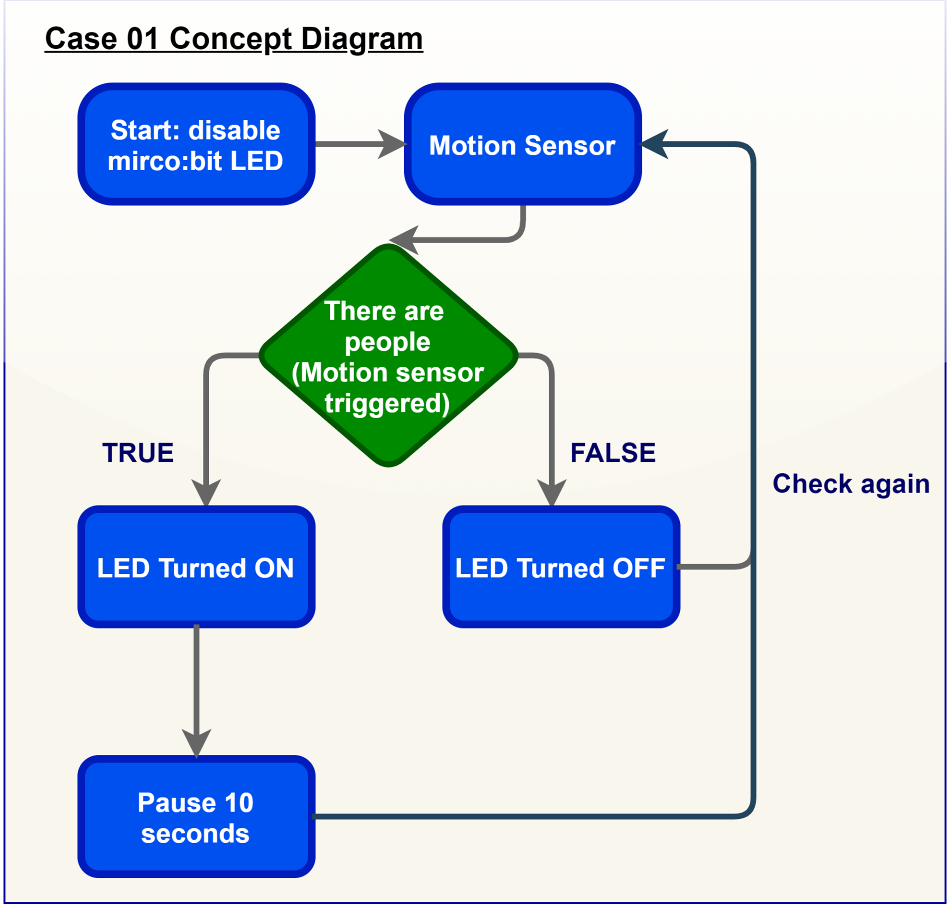
Goal

Background

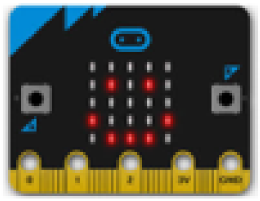
Smart playground lamp is a lamp which can open automatically when someone passes by. Installing an auto-light can help the earth save electricity. When no one passes by, the light will automatically turn off.

Smart playground lamp operation

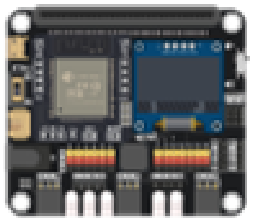
Motion sensor should be able to detect if there are people moving in the playground. If there are people moving in the playground, the LED light should turn on, vice versa.



Part List

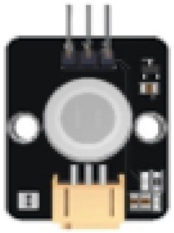


micro:bit (1)



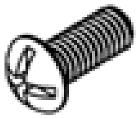
IoT:bit (1)

White LED Light (1)

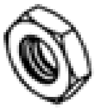


Motion Sensor (1)

3-pin module wire (2)

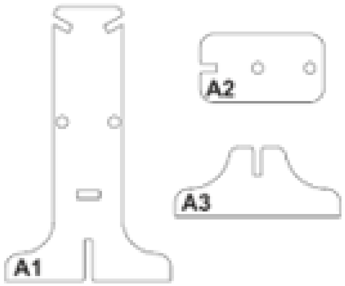


M4*10mm Screw (4)



M4*10mm Nut (4)

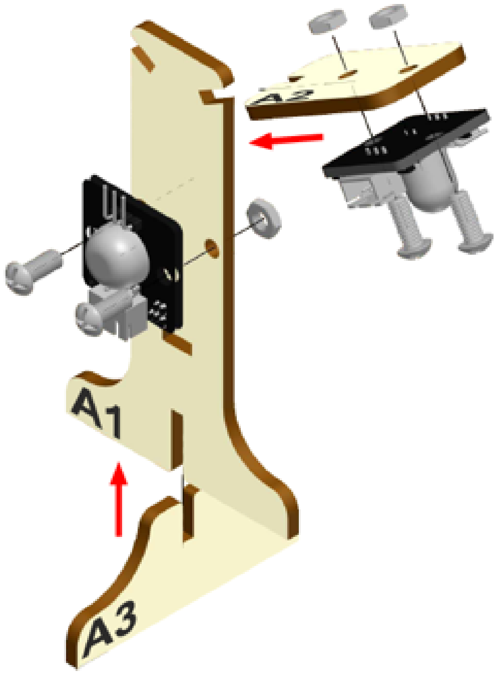
Screwdriver (1)

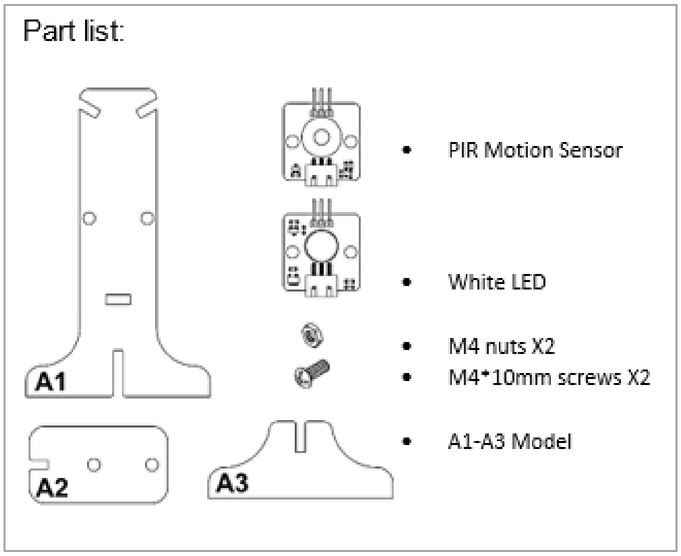


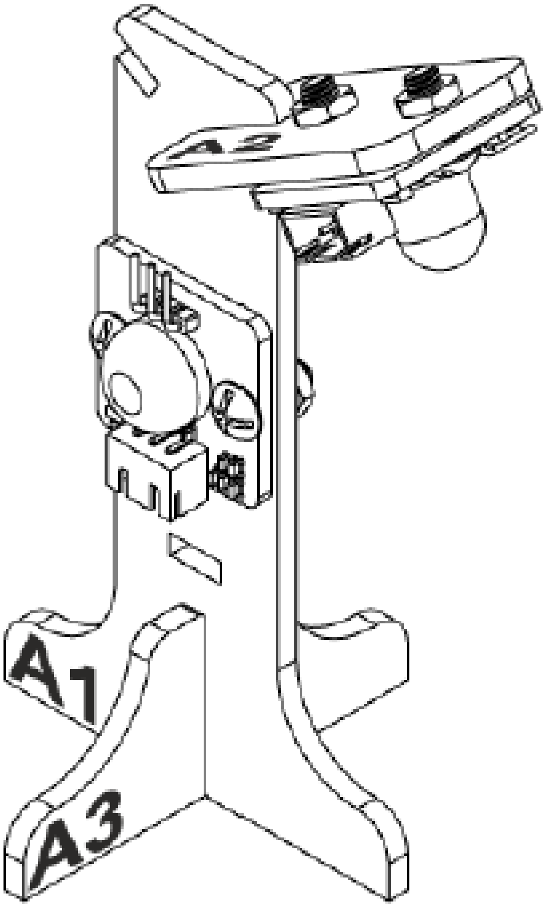
A1-A3 Model (1)

Assembly step

Step 1 Attach motion sensor and white LED to A1 model with M4 * 10mm screws and nuts.



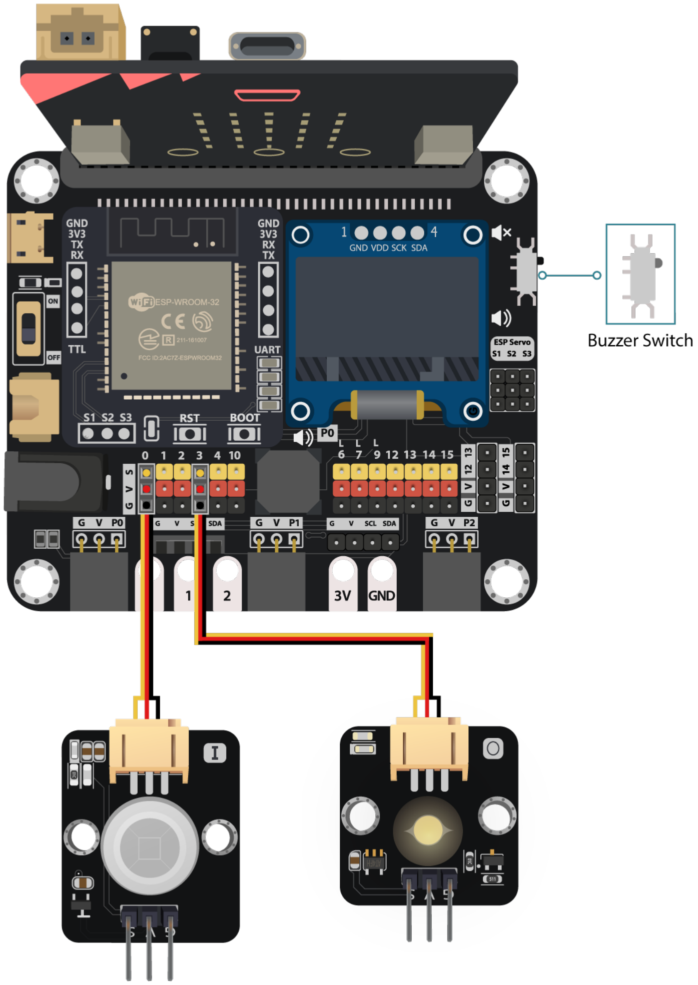




Step 2 Assembly completed!

Hardware connect

Connect Motion sensor to the P0 port of Smarthon IoT:bit Connect LED light to the P3 port of Smarthon IoT:bit

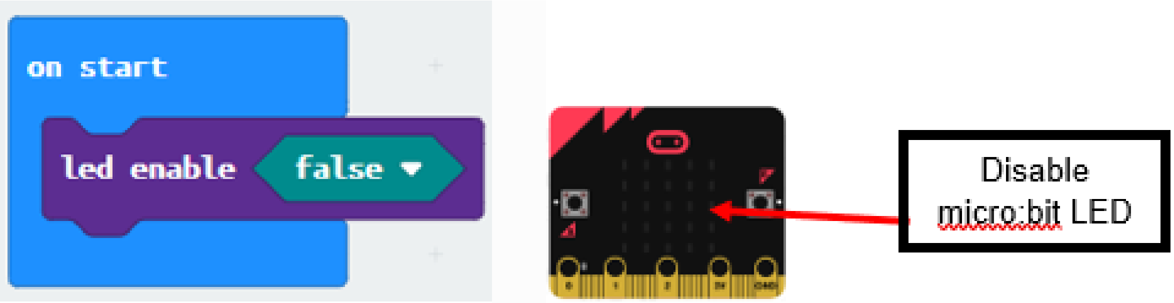


Pull the buzzer switch 'up' to disconnect the buzzer in this execrise

Programming (MakeCode)

Step 1. Disable micro:bit LED.

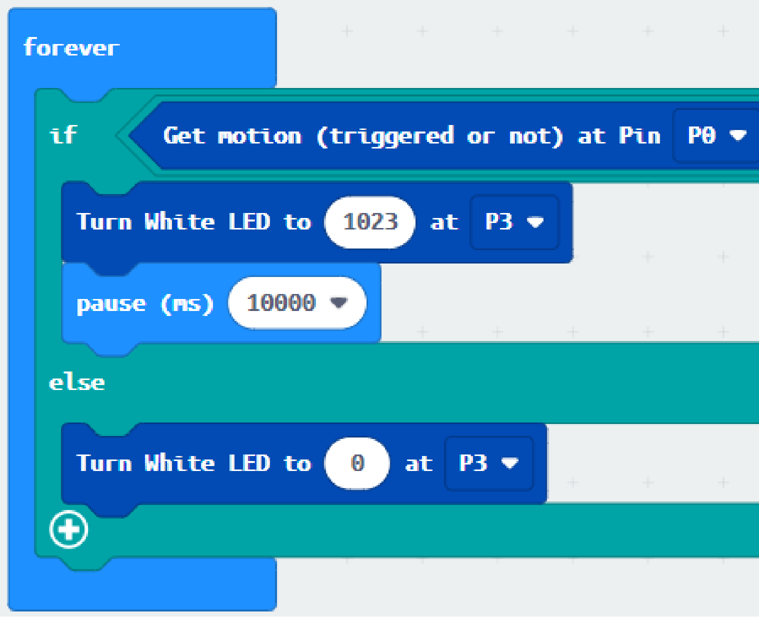
- Snap `led enable false` to `on start`
- Note that P3 is used as LED in default setting, LED need to be disable



```
on start
    led enable  false ▾
```

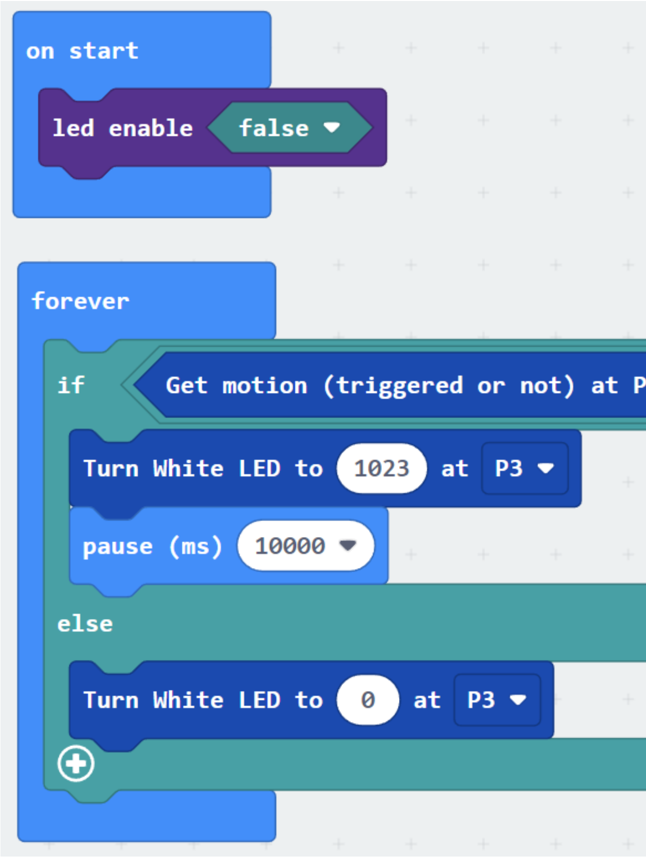
Step 2. Turn on LED by motion sensor

- Drag `forever` from `Basic`
- Snap `if statement` into `forever`
- Set `get motion (triggered or not) at P0 = true`, into `if statement` that's say motion is triggered, someone passes by.
- Then, `turn white LED to 1023 at P3` as turning on white LED and pause 10 seconds.



```
forever
    if  Get motion (triggered or not) at Pin  P0 ▾
        Turn White LED to  1023  at  P3 ▾
        pause (ms)  10000 ▾
    else
        Turn White LED to  0  at  P3 ▾
    ⊕
```

- Else, `turn white LED at P3 to 0` as turning off.



```
on start
    led enable  false ▾

forever
    if  Get motion (triggered or not) at P
        Turn White LED to  1023  at  P3 ▾
        pause (ms)  10000 ▾
    else
        Turn White LED to  0  at  P3 ▾
    ⊕
```

Full Solution MakeCode: https://makecode.microbit.org/_X4xYjzHU6d91

Result

Motion sensor is used to detect if there are people moving in the playground. If there are, the LED light will be turned on; otherwise, it will be turned off.

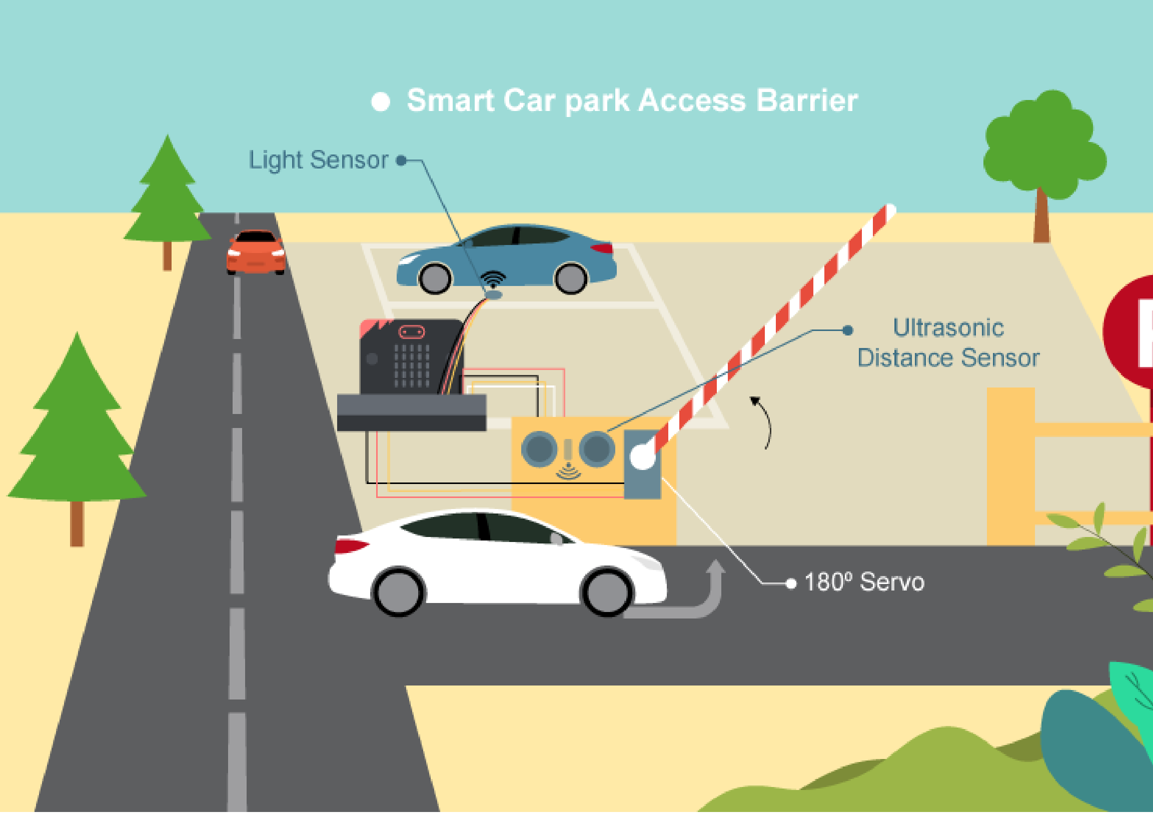
Think

Q1. How can you use motion sensors, other than turning on the light automatically?

Q2. Can you show motion sensor value on OLED?

1.1.11 Case 02: Smart Car park Access Barrier

Level: ★ ★ ☆ ☆ ☆



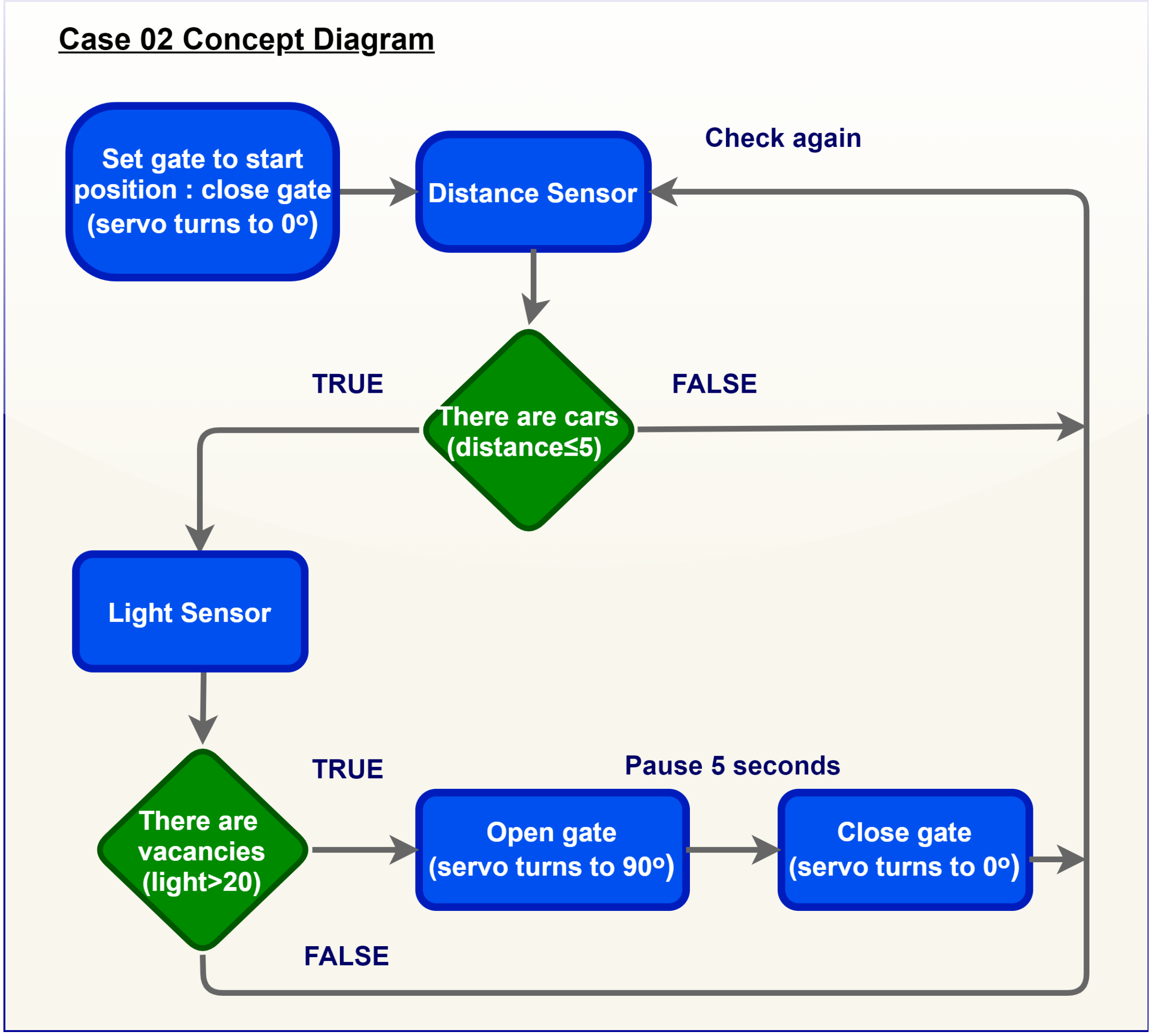
Goal

Make a smart car park gate which opens automatically if there are vacancies in the car park and there are cars coming in.

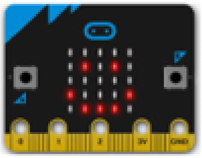
Background

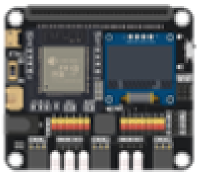
What is smart car park access barrier? Smart Car park Access Barrier is used to allow people to live conveniently. It can reduce manpower and time in controlling the gate and manage information (e.g. car park vacancies). It will be opened automatically if there are vacancies in the car park and there are cars coming in.

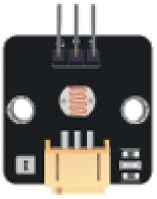
Car park gate operation The car park gate open and close operation is controlled by 180 servo. It will be opened if there are vacancies in the car park (i.e. detected by light sensor) and cars near the car park gate (i.e. detected by distance sensor). Otherwise, the gate will be closed. When there are no vacancies (detected by light sensor), then the gate will keep closed.


Case 02 Concept Diagram
Set gate to start position : close gate (servo turns to 0º)
Distance Sensor
Check again
TRUE
There are cars (distance≤5)
FALSE
Light Sensor
TRUE
Pause 5 seconds
There are vacancies (light>20)
Open gate (servo turns to 90º)
Close gate (servo turns to 0º)
FALSE

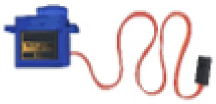
Part List


micro:bit (1)

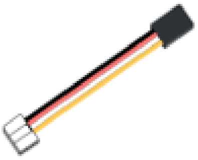

IoT:bit (1)


Light Sensor (1)

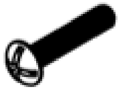
Distance Sensor (1)


180° Servo (1)

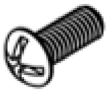
3-pin module wire (1)


4-pin module wire (1)

Servo M2*7.5mm sharp screw (1) (Included in servo pack)

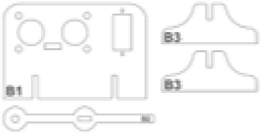

M2*10mm Screw (2)

M2 Nut (2)


M4*10mm Screw (4)

M4 Nut (4)

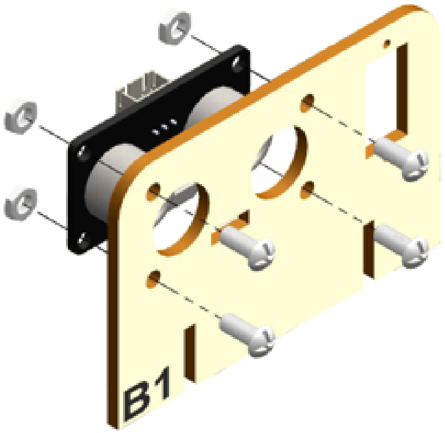
Screwdriver (1)



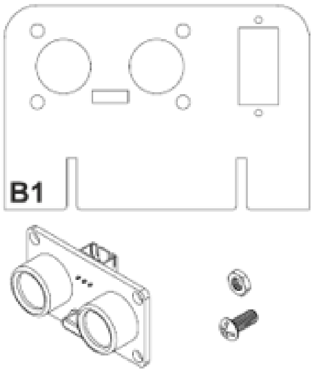
B1-B3 Model (1)

Assembly step

Step 1 Attach the distance sensor to B1 model with M4 * 10mm screws and nuts.

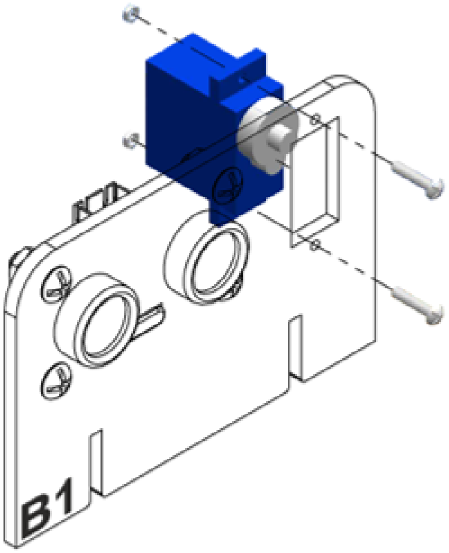


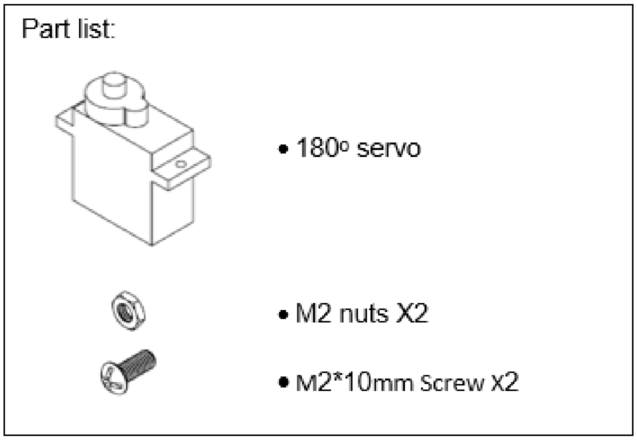
Part list:



- B1 model
- Distance Sensor
- M4 nuts X4
- M4*10mm screw X4

Step 2 Attach the servo to B1 model with M2 * 10mm screws and nuts

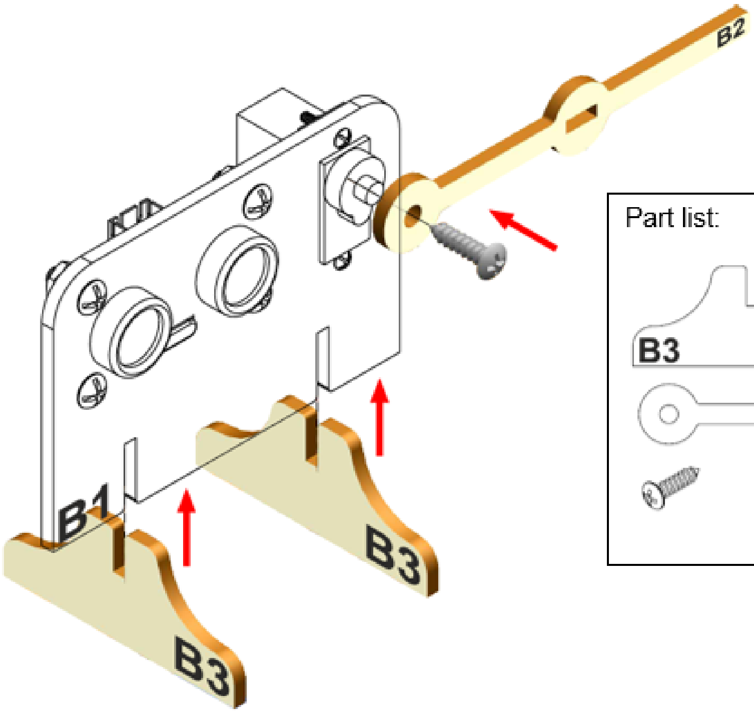


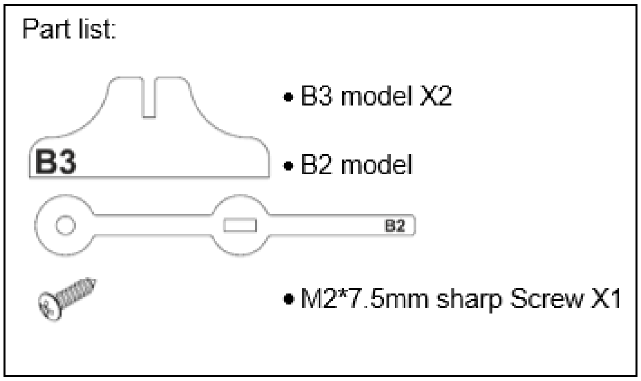


Part list:

- 180o servo
- M2 nuts X2
- M2*10mm Screw X2

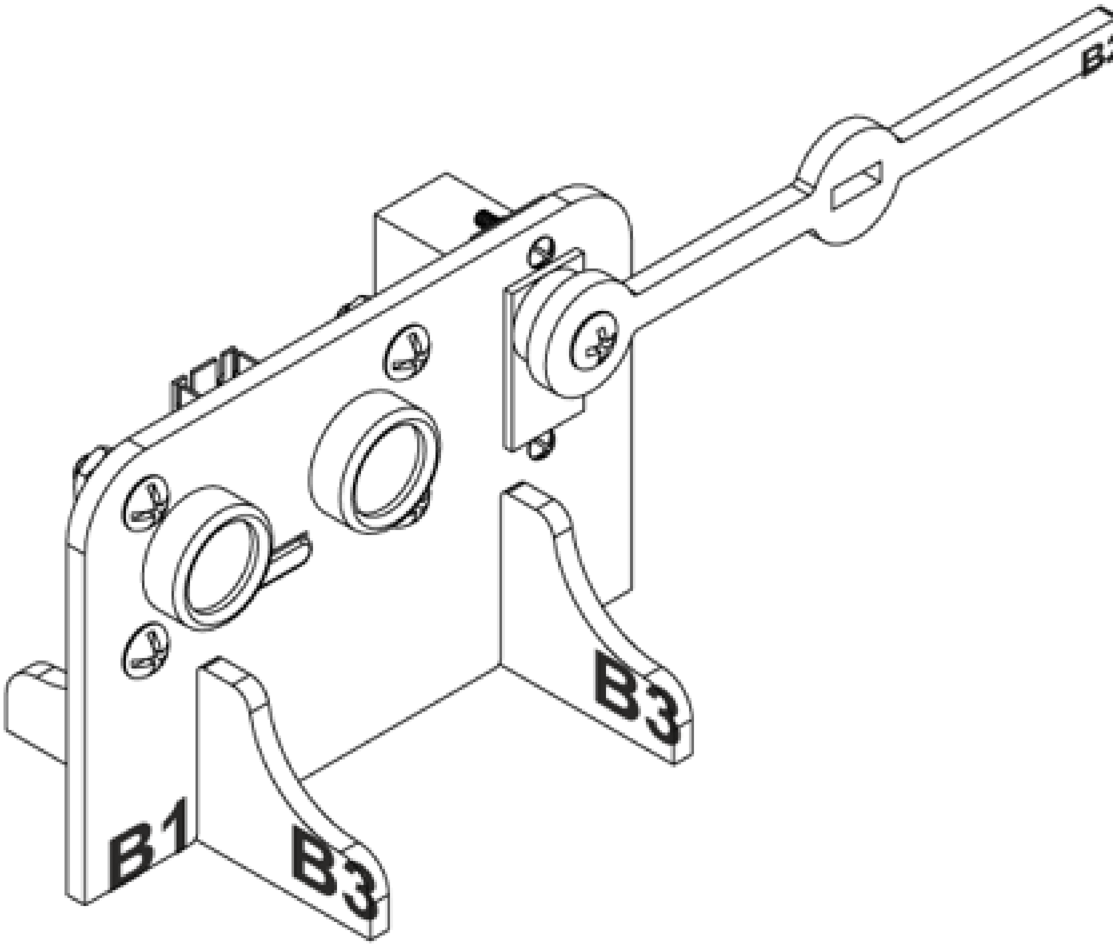
Step 3 Attach the B2 model to the servo with servo M2 * 7.5mm sharp screw. Put B3 model onto B1 model.





Part list:

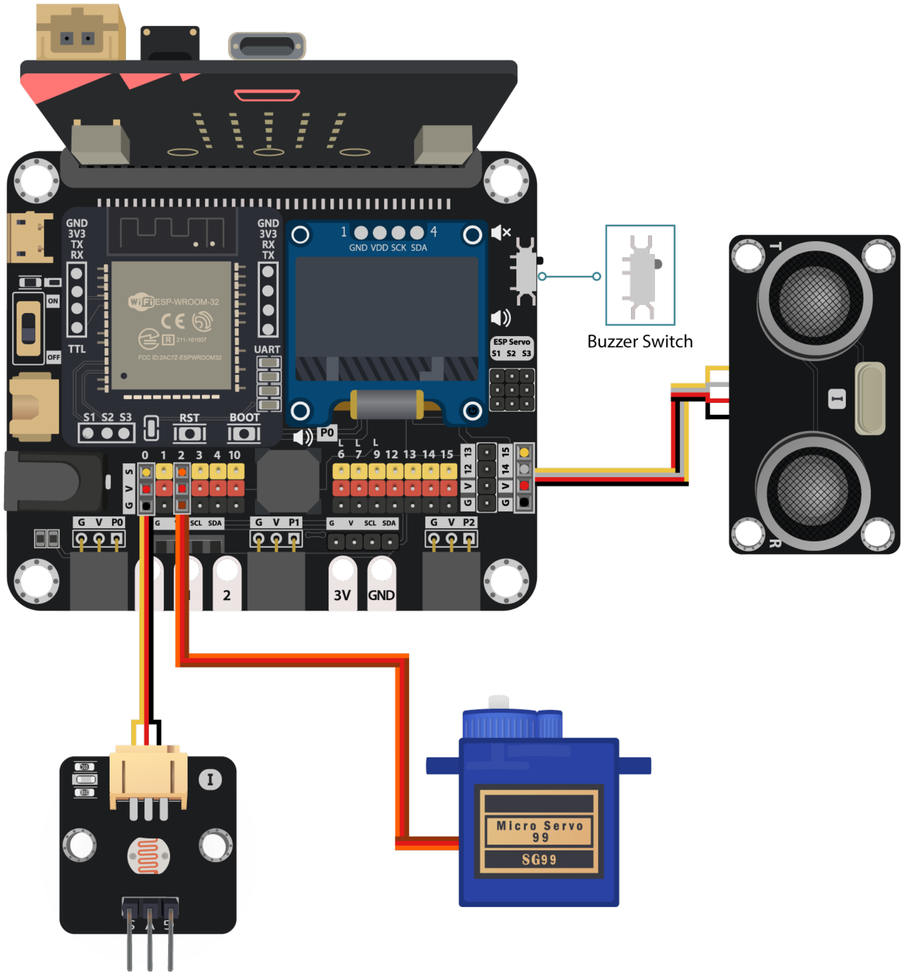
- B3 model X2
- B2 model
- M2*7.5mm sharp Screw X1



Step 4 Assembly completed!

Hardware connect

Connect the Distance Sensor to P14 (trig)/ P15 (echo) port of IoT:bit Connect Light Sensor to P0 port of IoT:bit Connect 180° Servo to P2 port of IoT:bit

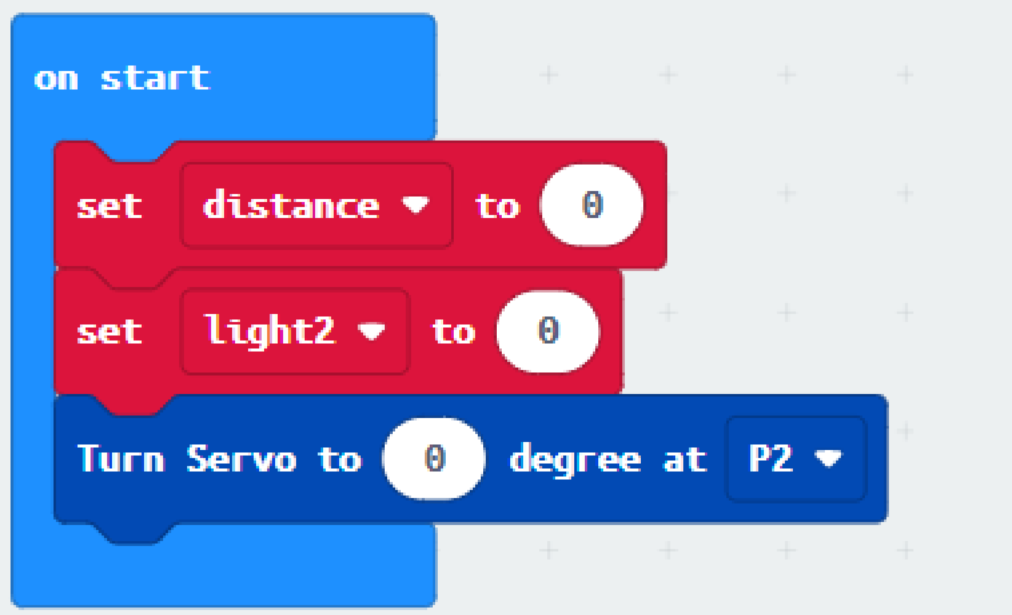


Pull the buzzer switch 'up' to disconnect the buzzer in this execrise

Programming (MakeCode)

Step 1. Set variables and servo at start position

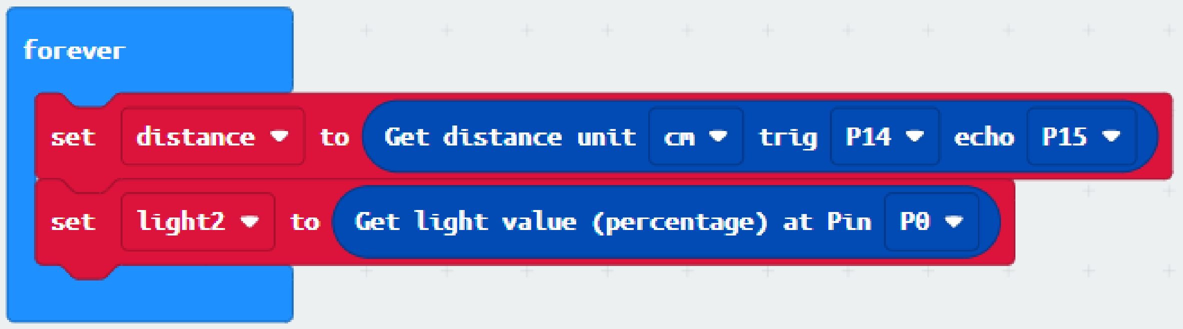
- Inside `on start,` snap `set variable distance to 0` and `set light to 0` from `variables.` Snap `Turn Servo to 0 degree at P2.`



```
on start
  set distance ▾ to 0
  set light2 ▾ to 0
  Turn Servo to 0 degree at P2 ▾
```

Step 2. Get distance and light value

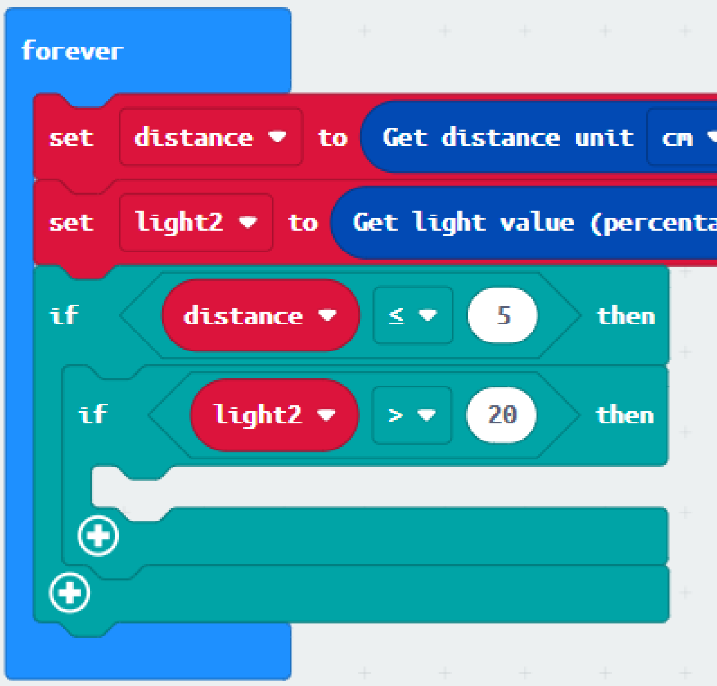
- Drag `get distance` to `get distance unit cm trig P14 echo P15`, store the value to variable `distance`.
- `Get light value (percentage) at Pin P0,` store the value to variable `light`



```
forever
  set distance ▾ to Get distance unit cm ▾ trig P14 ▾ echo P15 ▾
  set light2 ▾ to Get light value (percentage) at Pin P0 ▾
```

Step 3. Open/close gate with light value

- Snap `if statement` into forever, set if variable `distance 5`



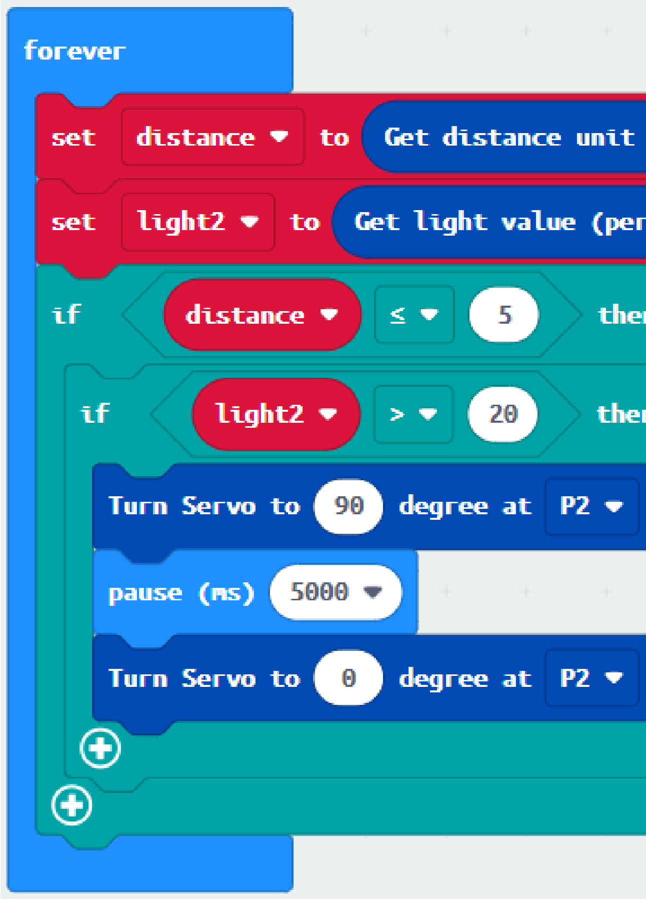
```
forever
  set distance ▾ to Get distance unit cm ▾
  set light2 ▾ to Get light value (percenta
  if distance ▾ ≤ ▾ 5 then
    if light2 ▾ > ▾ 20 then
```

- Snap another `if statement` set variable `light` value >20

Step 4. Set servo position

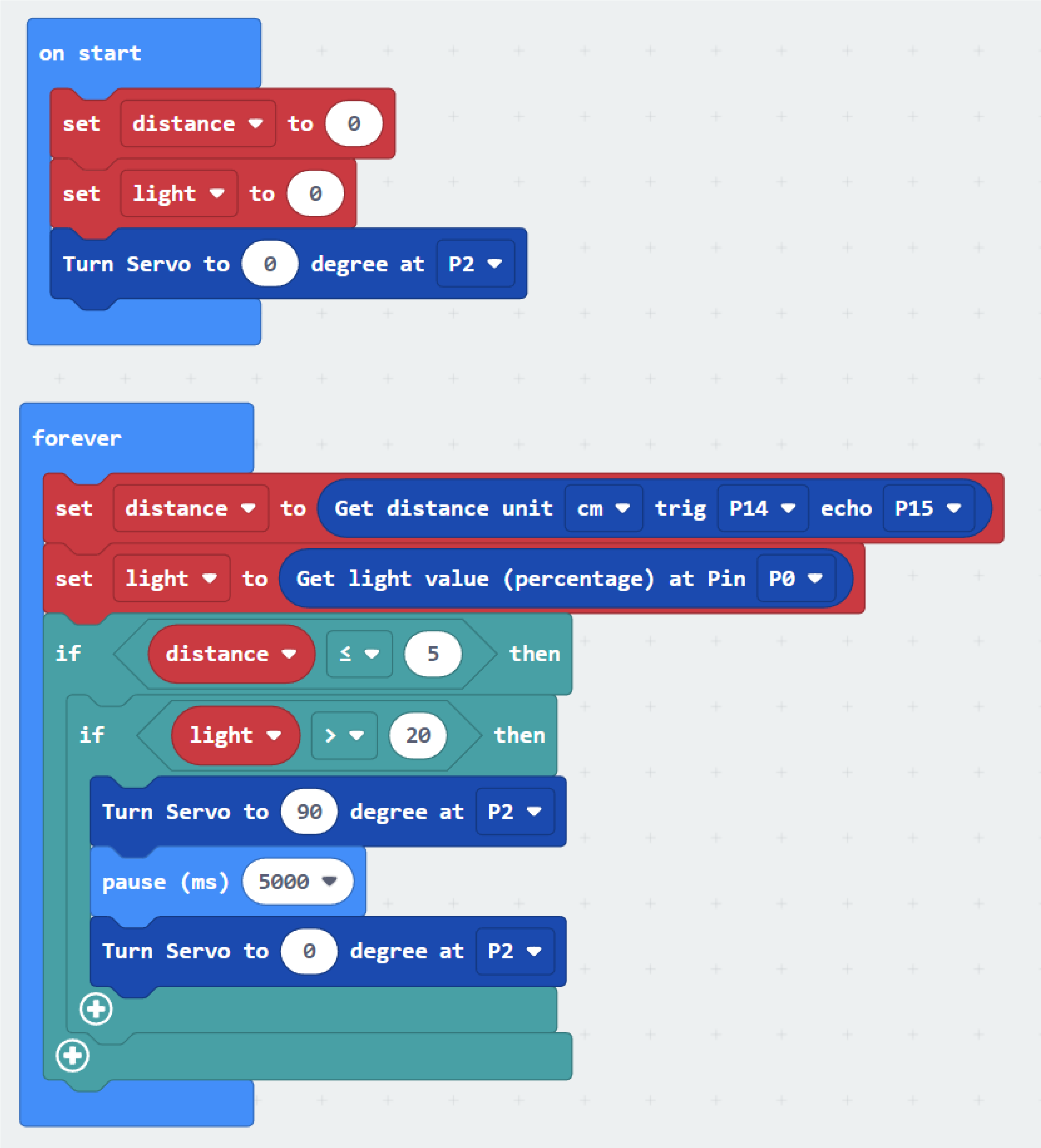
- Snap `Turn Servo to 90 degree at P2` as the gate is opened.
- Snap `pause` to the loop to wait 5 seconds

- Snap `Turn Servo to 0 degree at P2` as the gate is closed.



```
forever
  set distance to Get distance unit
  set light2 to Get light value (per
  if distance ≤ 5 then
    if light2 > 20 then
      Turn Servo to 90 degree at P2
      pause (ms) 5000
      Turn Servo to 0 degree at P2
```

Full Solution MakeCode: https://makecode.microbit.org/_UgRKmX2XzH2u



```
on start
    set distance ▾ to 0
    set light ▾ to 0
    Turn Servo to 0 degree at P2 ▾

forever
    set distance ▾ to Get distance unit cm ▾ trig P14 ▾ echo P15 ▾
    set light ▾ to Get light value (percentage) at Pin P0 ▾
    if distance ▾ ≤ ▾ 5 then
        if light ▾ > ▾ 20 then
            Turn Servo to 90 degree at P2 ▾
            pause (ms) 5000 ▾
            Turn Servo to 0 degree at P2 ▾
        ⊕
    ⊕
```

Result

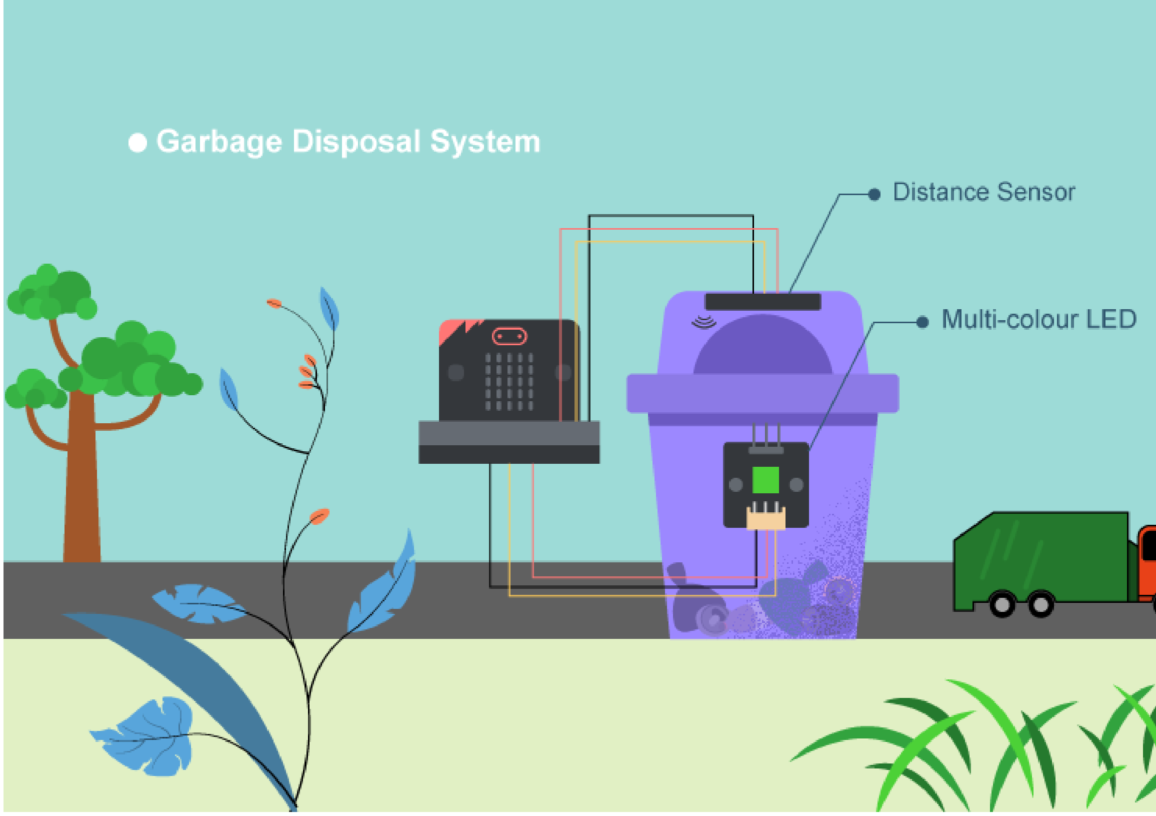
The light sensor is used to check the vacancies in the car park while the distance sensor is used to detect if there are any cars coming near the car park gate The car park gate is controlled by 180 servo. When there are vacancies in the car park and there are cars near the car park gate, the car park is available. The gate will be opened for 5 seconds and then closed to let the car entering the car park.

Think

Q1. How can you show the light and distance value on OLED? Q2. Can you show signals on micro:bit LED to let drivers know there are vacancies in the car park?

1.1.12 Case 03: Garbage disposal system

Level: ★ ★ ☆ ☆ ☆



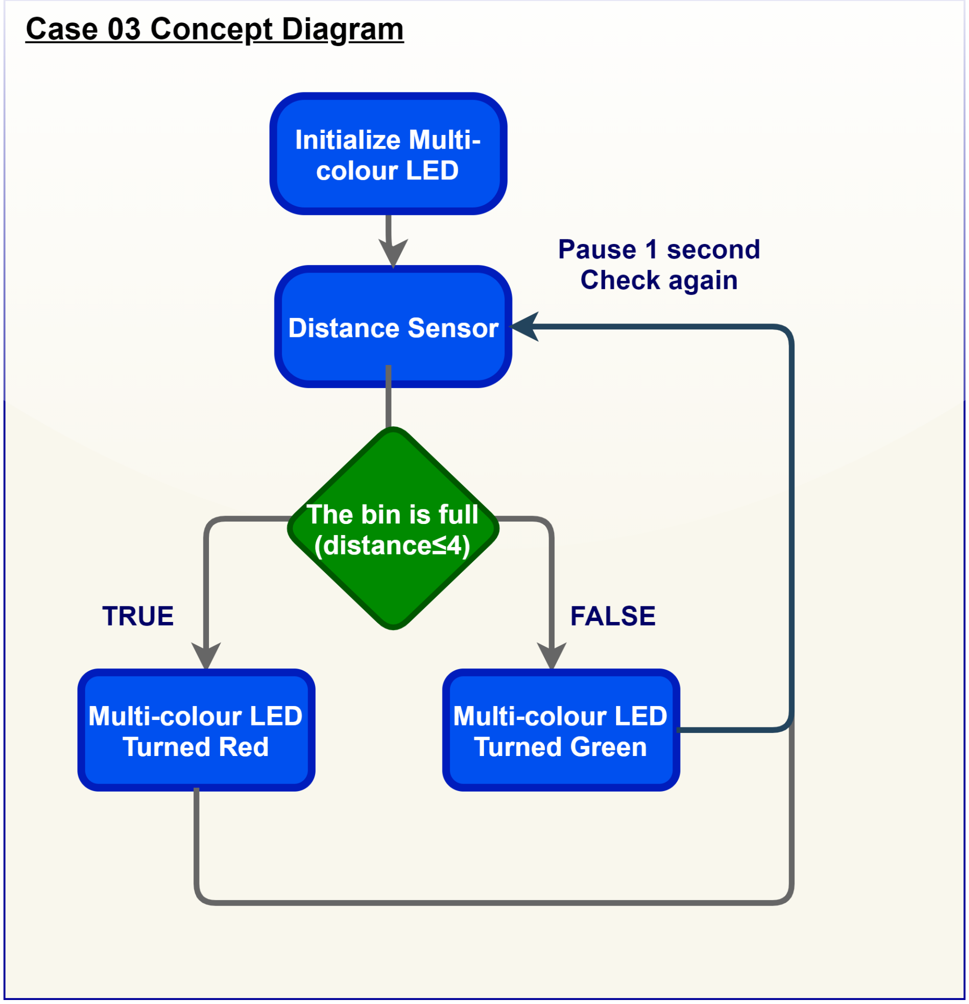
Goal

Make a smart garbage bin by showing different colors of light according to the volume of garbage inside.

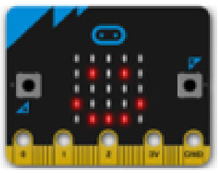
Background

What is garbage disposal system? LED light on the garbage bin can show people the amount of garbage inside the garbage bin so that garbage truck can easily determine if the garbage is full or not. This can minimize the wastage of the garbage bags and become a more environmental-friendly city.

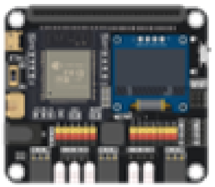
Garbage disposal system operation The distance sensor should be able to sense the amount of garbage is inside the garbage bin. Multi-colour LED can be used to emit different colour of light (green and red light), which represent the emptiness of the garbage bin.


Case 03 Concept Diagram
Initialize Multi-colour LED
Pause 1 second
Check again
Distance Sensor
The bin is full (distance≤4)
TRUE
FALSE
Multi-colour LED Turned Red
Multi-colour LED Turned Green

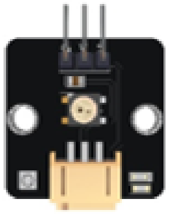
Part List



micro:bit (1)



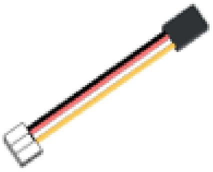
IoT:bit (1)



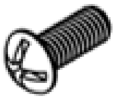
Multi-colour LED (WS2812) (1)

Distance Sensor (1)

3-pin module wire (1)



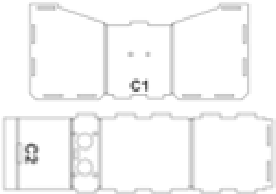
4-pin module wire (1)



M4*10mm Screw (6)

M4 Nut (6)

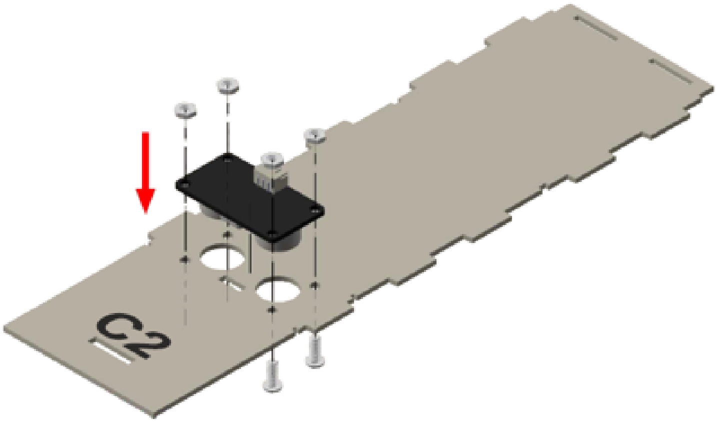
Screwdriver (1)

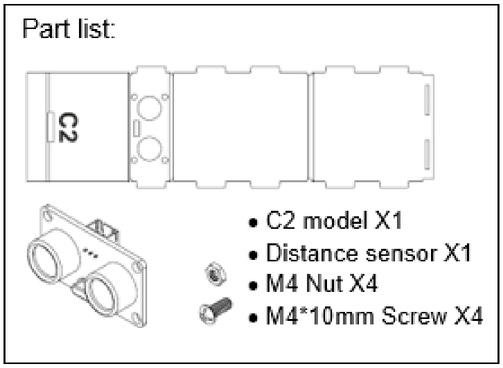


C1-C2 Model (1)

Assembly step

Step1 Attach the Multi-colour LED to C1 model with M4 * 10mm screws and nuts.

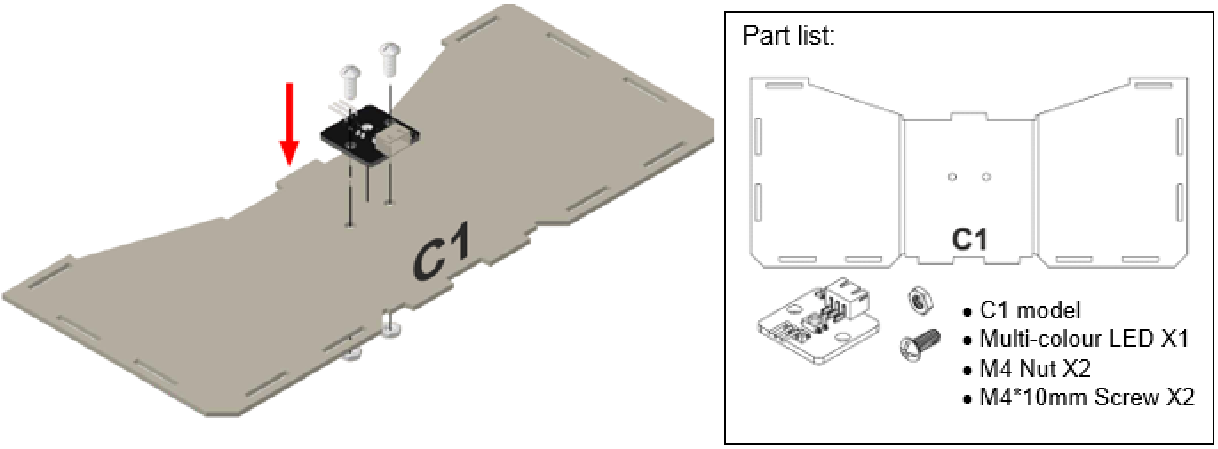


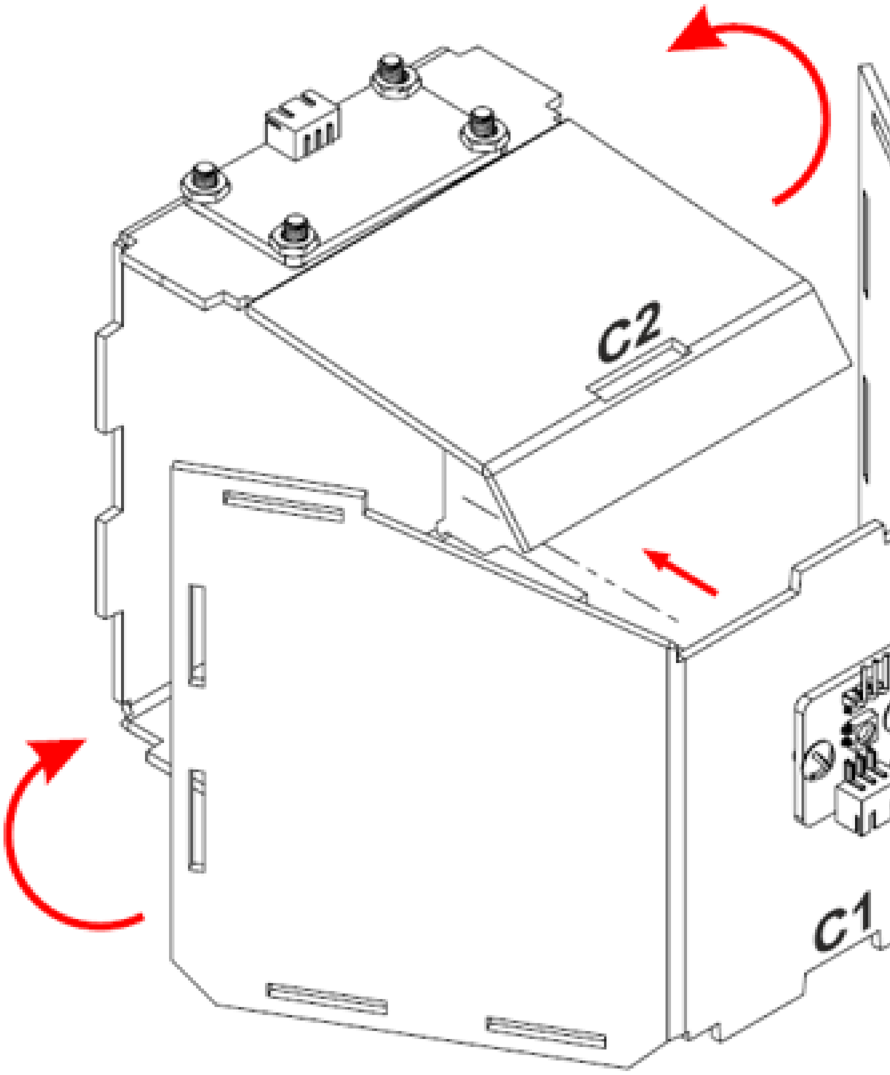


Part list:

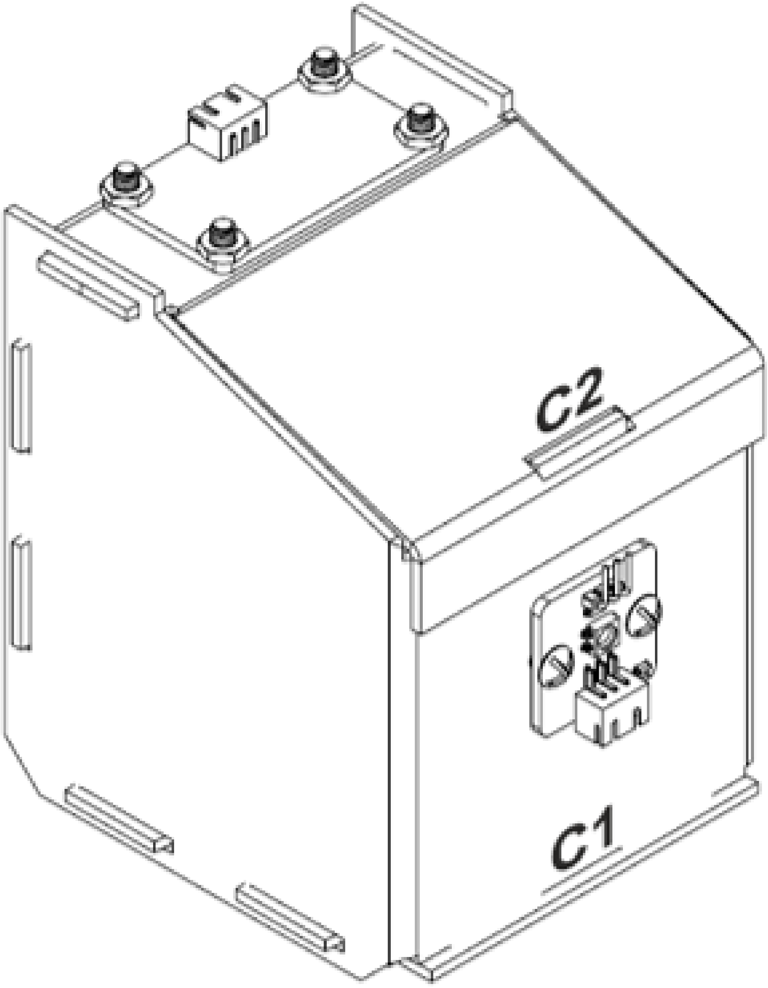
- C2 model X1
- Distance sensor X1
- M4 Nut X4
- M4*10mm Screw X4

Step 2 Attach the distance sensor to C2 model with M4 * 10mm screw and nuts.





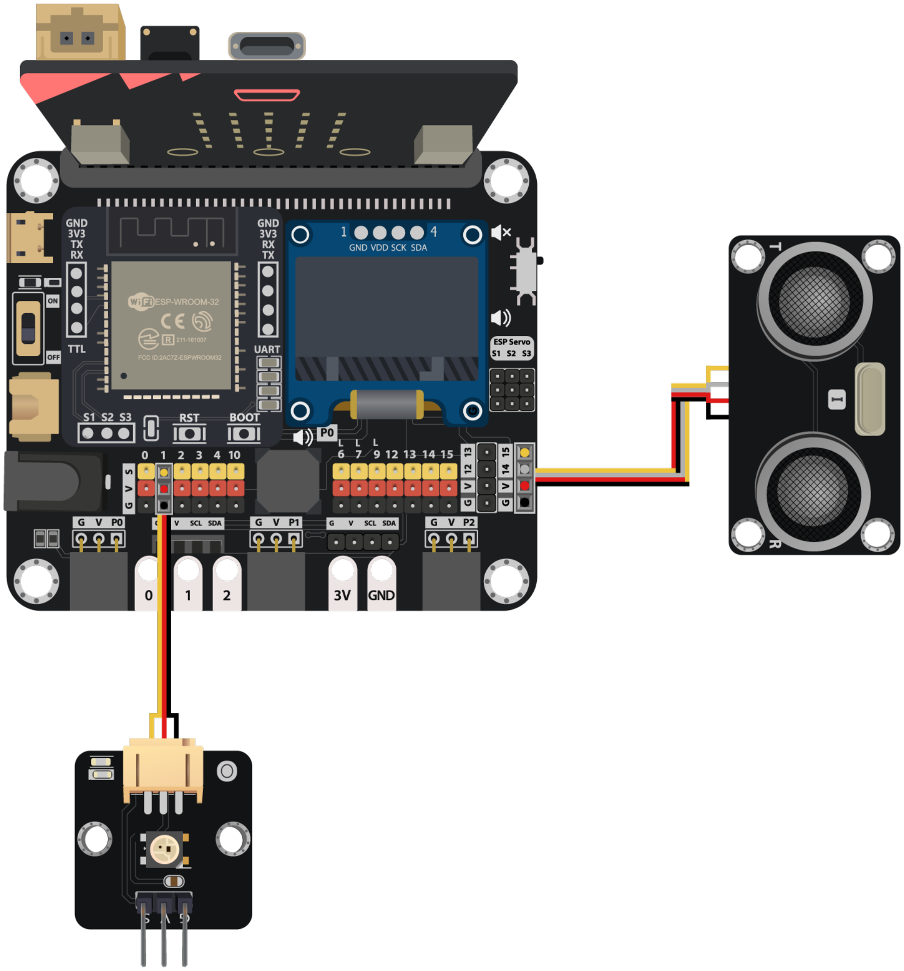
Step 3 Put together all the cardboard parts (C1-C2).



Step 4 Assembly completed!

Hardware connect

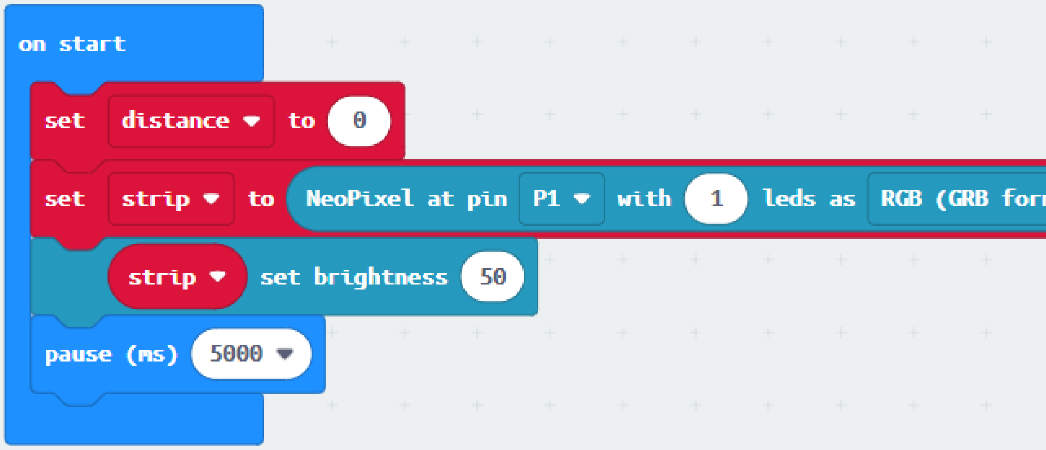
Connect the Distance Sensor to P14 (trig)/ P15 (echo) port of IoT:bit Connect Multi-color LED to P1 port of IoT:bit



Programming (MakeCode)

Step 1. Set variable and initialize multi-colour LED

- Inside `on start`, snap `set variable distance to 0` from `variables`
- Snap `set strip to NeoPixel at pin...` from `Neopixel`. Set pin P1 with 1 led of the block.
- Snap `strip set brightness` from `Neopixel` > `more` and set brightness 50

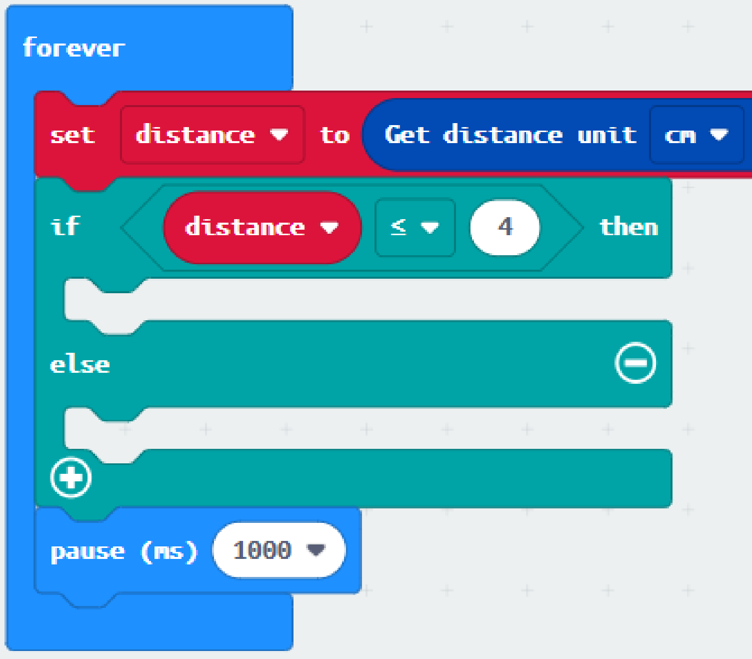


```
on start
  set distance ▾ to 0
  set strip ▾ to NeoPixel at pin P1 ▾ with 1 leds as RGB (GRB for
  strip ▾ set brightness 50
  pause (ms) 5000 ▾
```

- Snap `pause` to wait 5 seconds

Step 2. Get distance value

- Inside block `forever`. Set `distance` to `get distance unit cm trig P14 echo P15`, that's say get the distance value by connecting the distance sensor to P14 and P15
- Snap `if statement` into `forever`, set `distance 4` into `if statement`

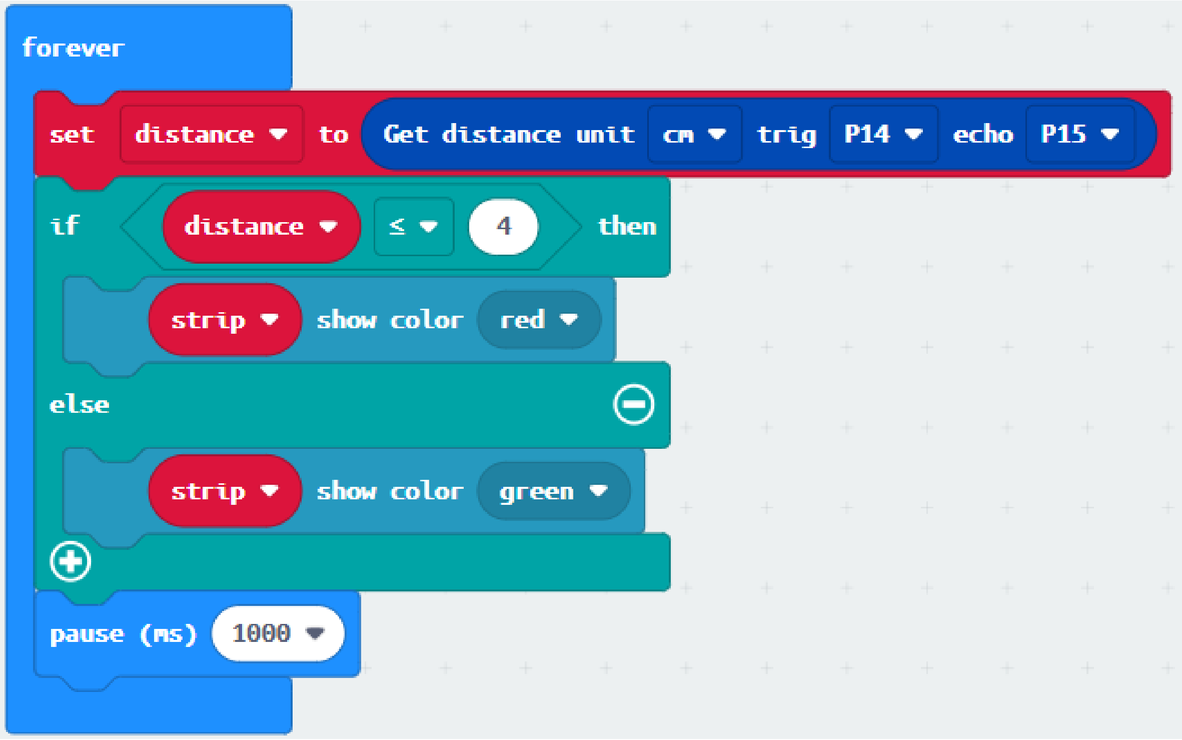


```
forever
  set distance ▾ to Get distance unit cm ▾
  if distance ▾ ≤ ▾ 4 then

  else ⊖

  ⊕
  pause (ms) 1000 ▾
```

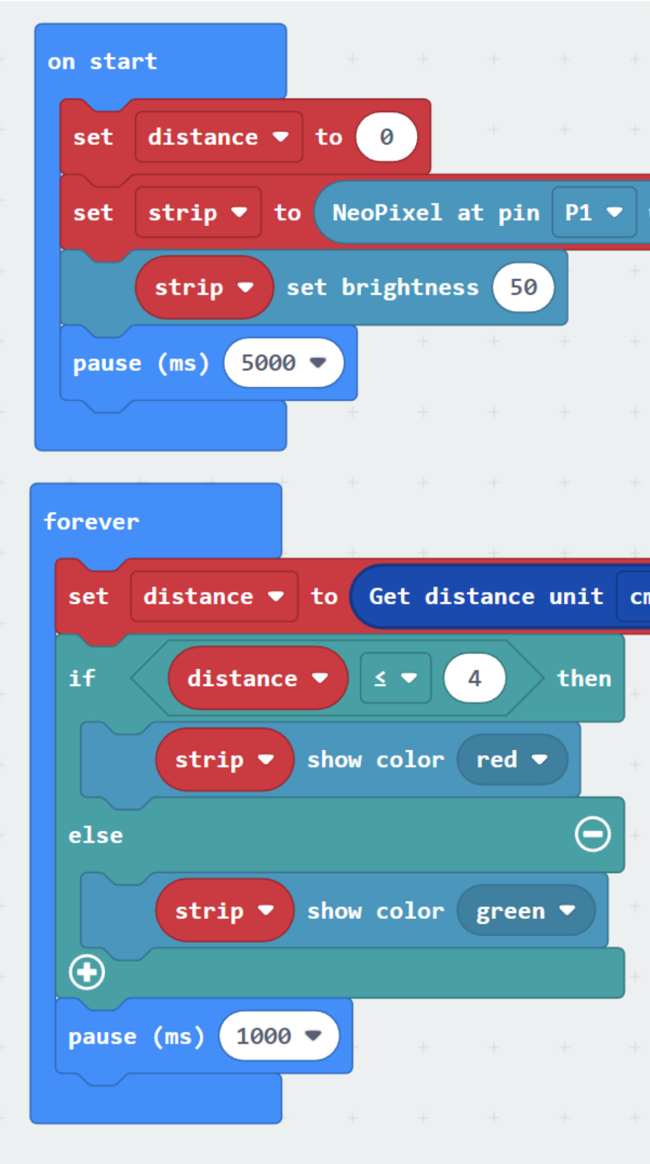
- Snap `Pause` to the loop to wait 1 second for next checking

Step 3. Show indicating colours with distance value

- If distance 4, then `strip show color red`, else `strip show color green`



```
forever
    set distance ▾ to Get distance unit cm ▾ trig P14 ▾ echo P15 ▾
    if distance ▾ ≤ ▾ 4 then
        strip ▾ show color red ▾
    else ⊖
        strip ▾ show color green ▾
    ⊕
    pause (ms) 1000 ▾
```



```
on start
    set distance to 0
    set strip to NeoPixel at pin P1
    strip set brightness 50
    pause (ms) 5000

forever
    set distance to Get distance unit cm
    if distance ≤ 4 then
        strip show color red
    else
        strip show color green
    pause (ms) 1000
```

Full Solution MakeCode: https://makecode.microbit.org/_F2x6j9UaM6wf

Result

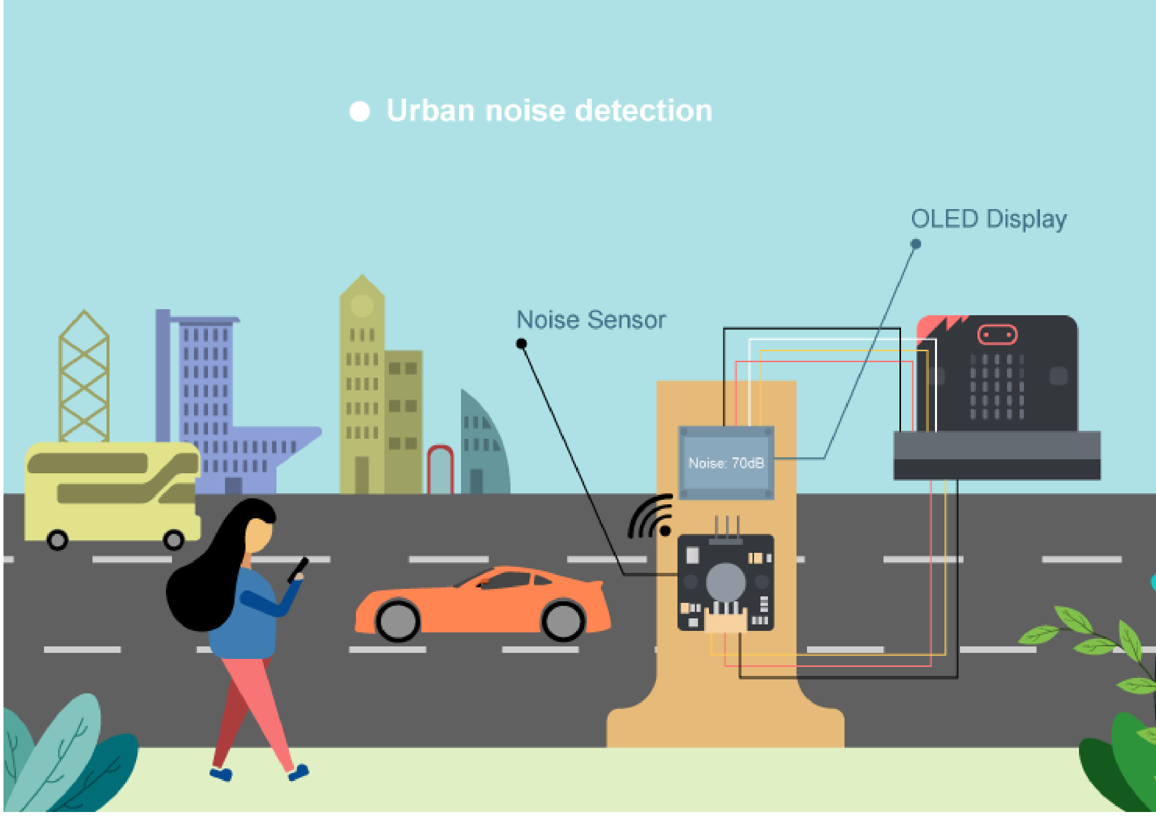
The distance sensor can return the distance value between the top of the garbage bin and the level of garbage inside the garbage bin. The LED light is used to indicate if the bin is full or not. If it is empty or there are some garbage, the LED turns green, if it is full, the LED turns red.

Think

Q1. How to make sound notification if there are full of garbage (i.e. using buzzer)?

1.1.13 Case 04: Urban noise detection

Level: ★ ★ ☆ ☆ ☆



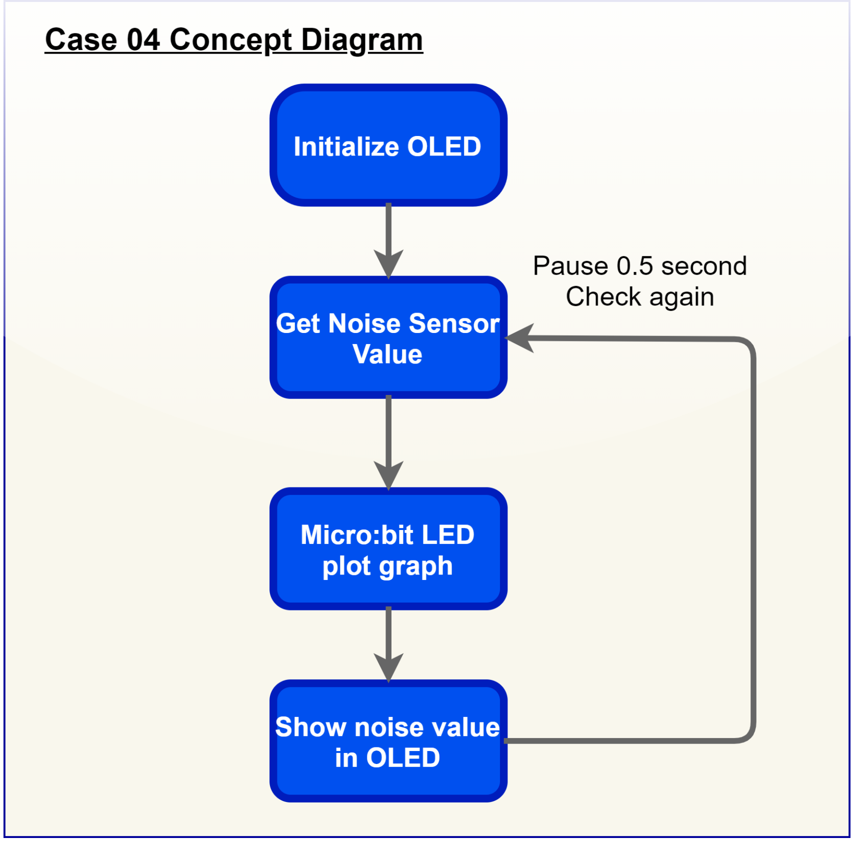
Goal

Make a noise detection point to detect the noise volume near the roadside using noise sensor.

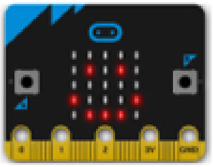
Background

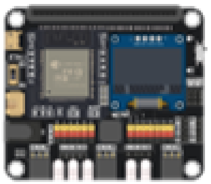
What is urban noise detection? It is a system to detect noise near the road as noise pollution caused by cars on the road seriously affect the living standard of people. By installing a monitor to detect the noise volume near the roadside can help engineer to gather noise information and find solution to solve the problem in the future.

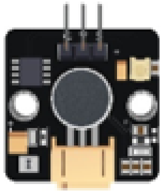
Noise detection operation The noise sensor can detect the volume in dB near the roadside and show the volume on the OLED. A bar graph of sound intensity will be shown on the micro:bit.


Case 04 Concept Diagram
Initialize OLED
Get Noise Sensor Value
Micro:bit LED plot graph
Show noise value in OLED
Pause 0.5 second
Check again

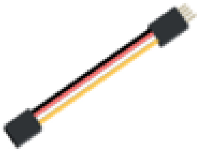
Part List

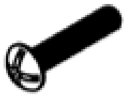

micro:bit (1)


IoT:bit with OLED (1)

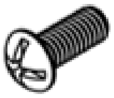

Noise Sensor (1)

3-pin module wire (1)


4-pin extension wire (1)

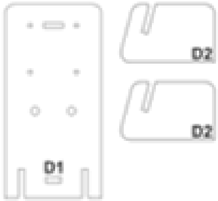

M2*10mm Screw (2)

M2 Nut (2)


M4*10mm Screw (4)

M4 Nut (4)

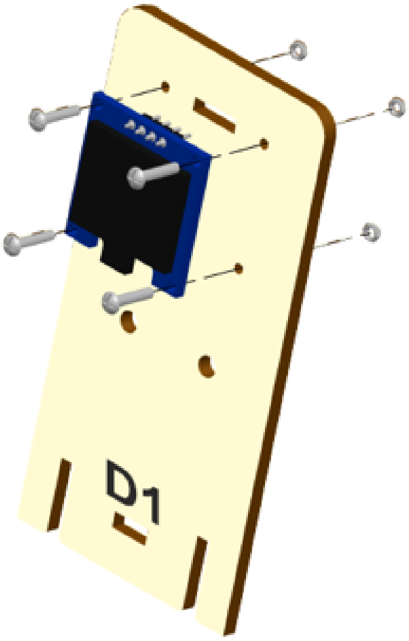
Screwdriver (1)

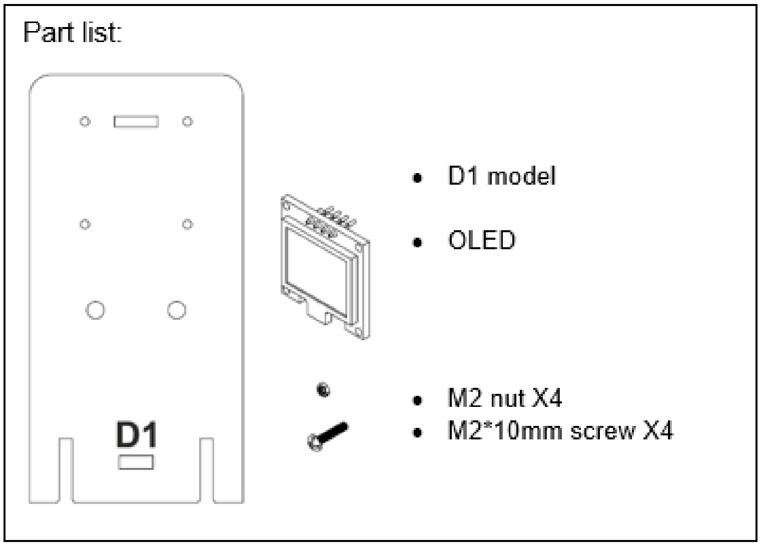


D1-D2 Model (1)

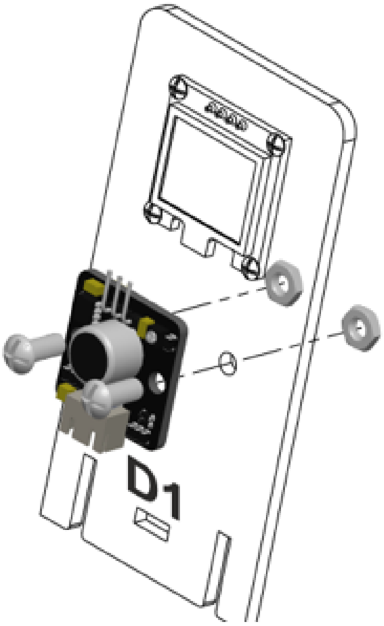
Assembly step

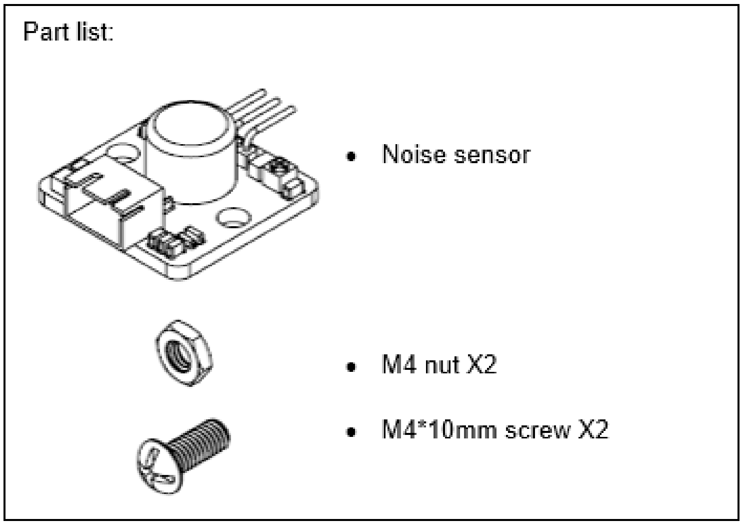
Step 1 Attach the OLED to D1 model with M2 * 10mm screws and nuts.

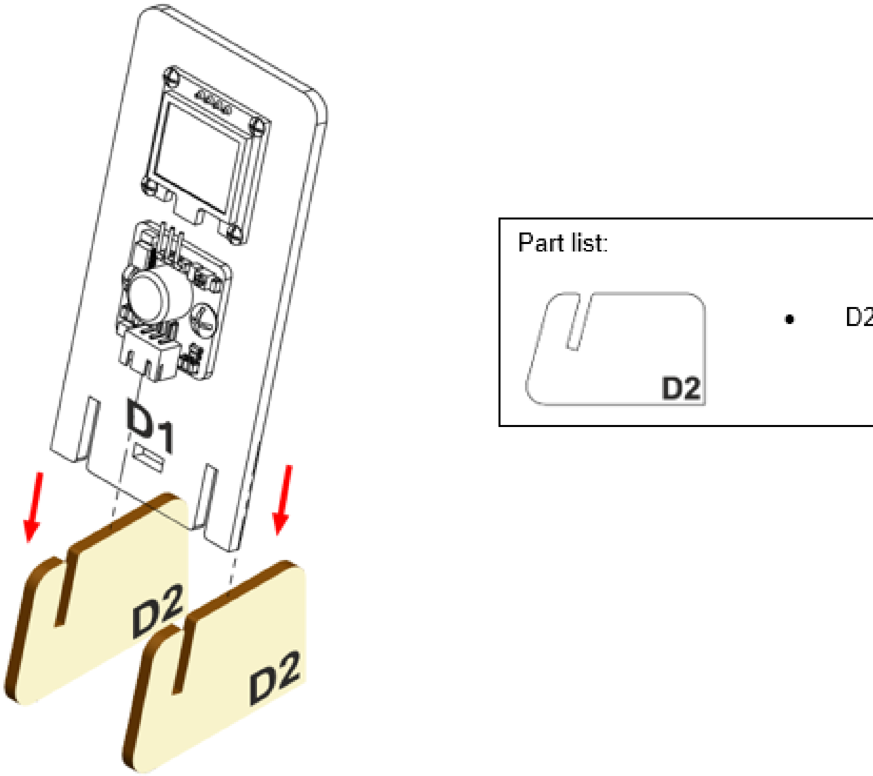




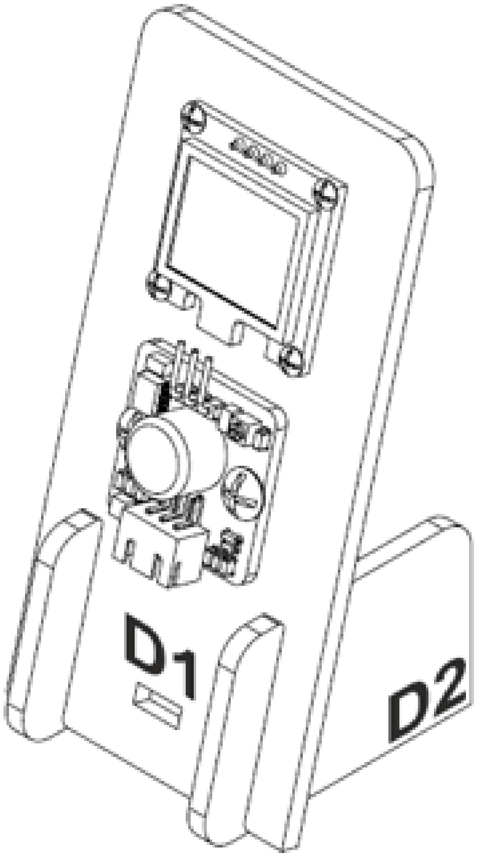
Step 2 Attach noise sensor to D1 model with M4 * 10mm screws and nuts.







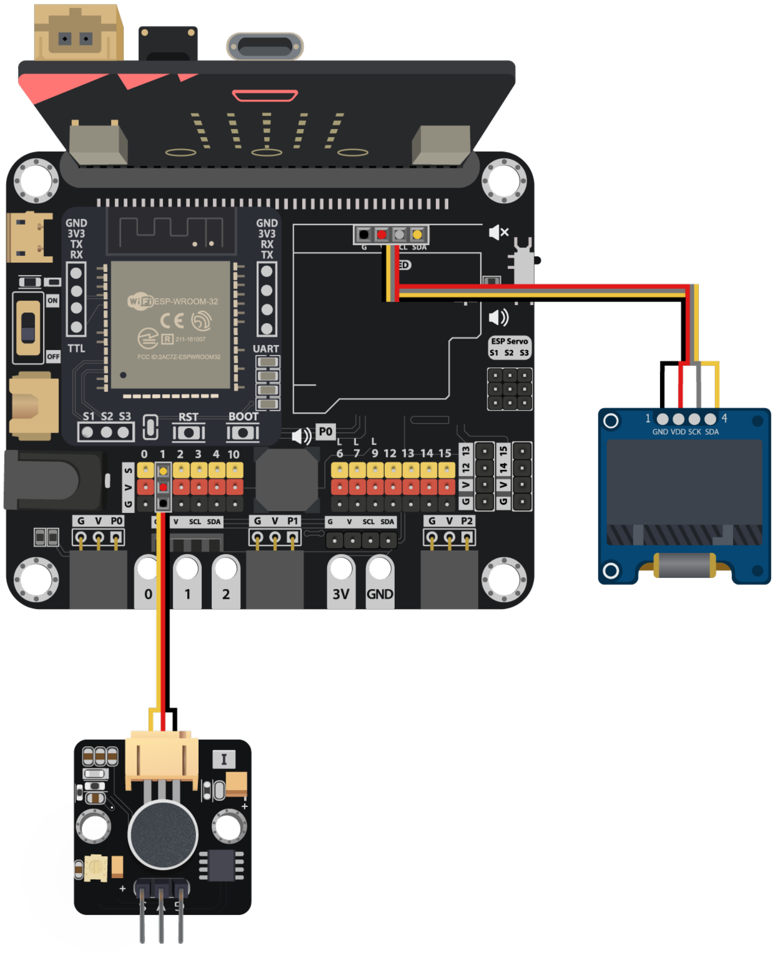
Step 3 Put together all the cardboard parts (D1-D2).



Step 4 Assembly completed!

Hardware connect

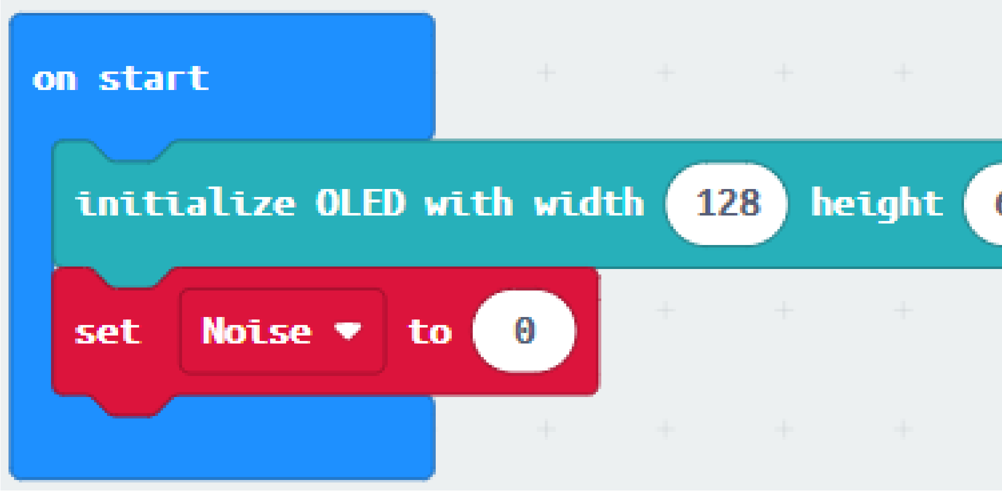
Connect Noise Sensor to P1 port of IoT:bit Extend the connection of OLED to I2C connection port of IoT:bit



Programming (MakeCode)

Step 1 Step 1. Initialize OLED screen and variable

- Drag `Initialize OLED with width:128, height: 64` to `on start`

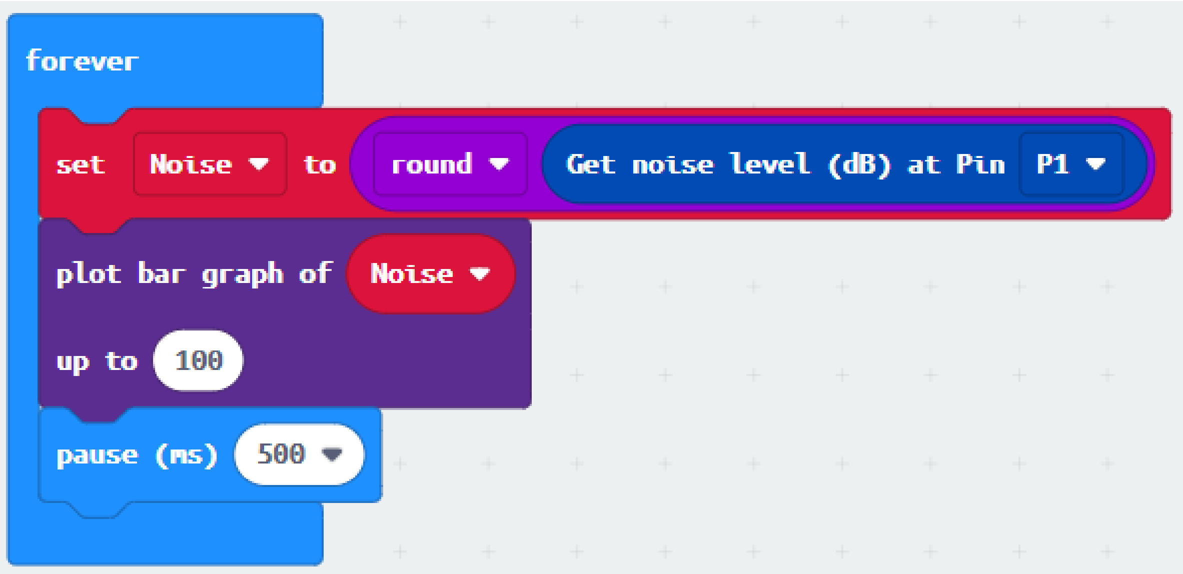


```
on start
    initialize OLED with width 128 height
    set Noise ▾ to 0
```

- Set `Noise` to 0 from `variables`

Step 2. Show noise value on micro:bit LED

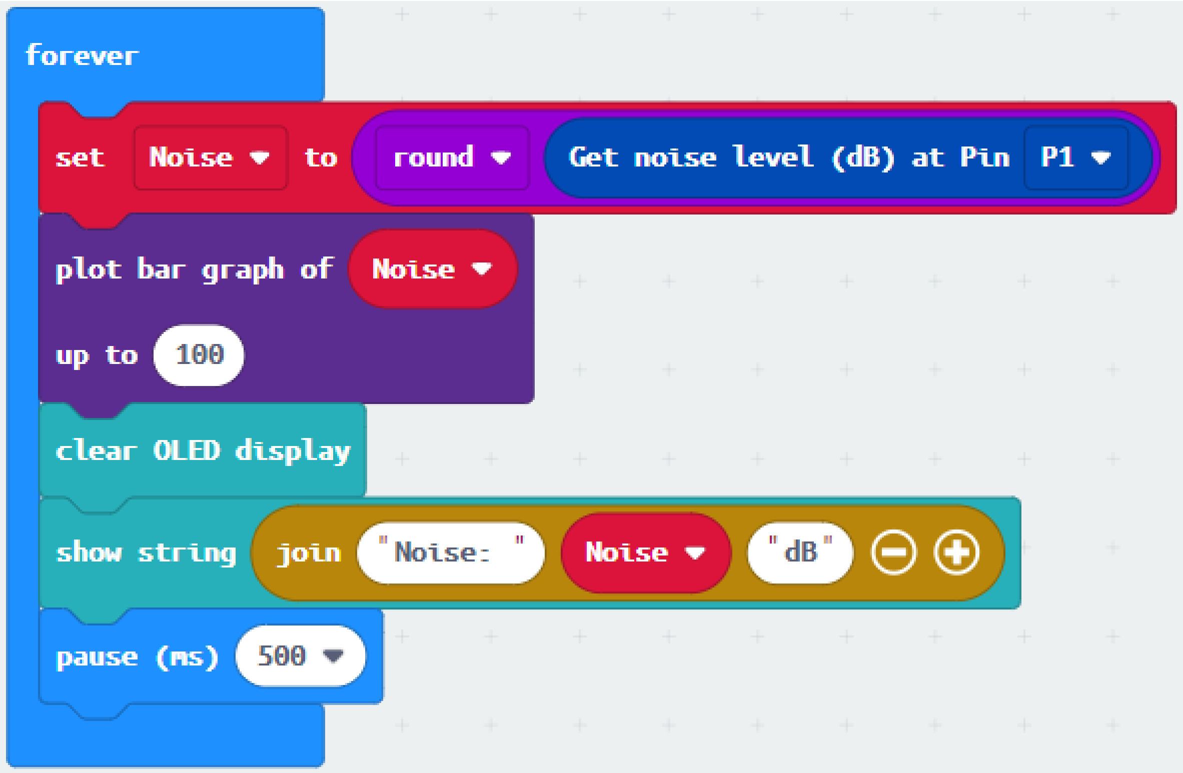
- In block `forever`. Set `noise` to round `get noise level (dB) at pin P1`
- Drag `plot bar graph of ...` from `Led` and draw variable `Noise` into the plotted value. Set value up to 100
- Drag `Pause` to the loop for wait for 0.5 second for next checking



```
forever
    set Noise ▾ to round ▾ Get noise level (dB) at Pin P1 ▾
    plot bar graph of Noise ▾
    up to 100
    pause (ms) 500 ▾
```

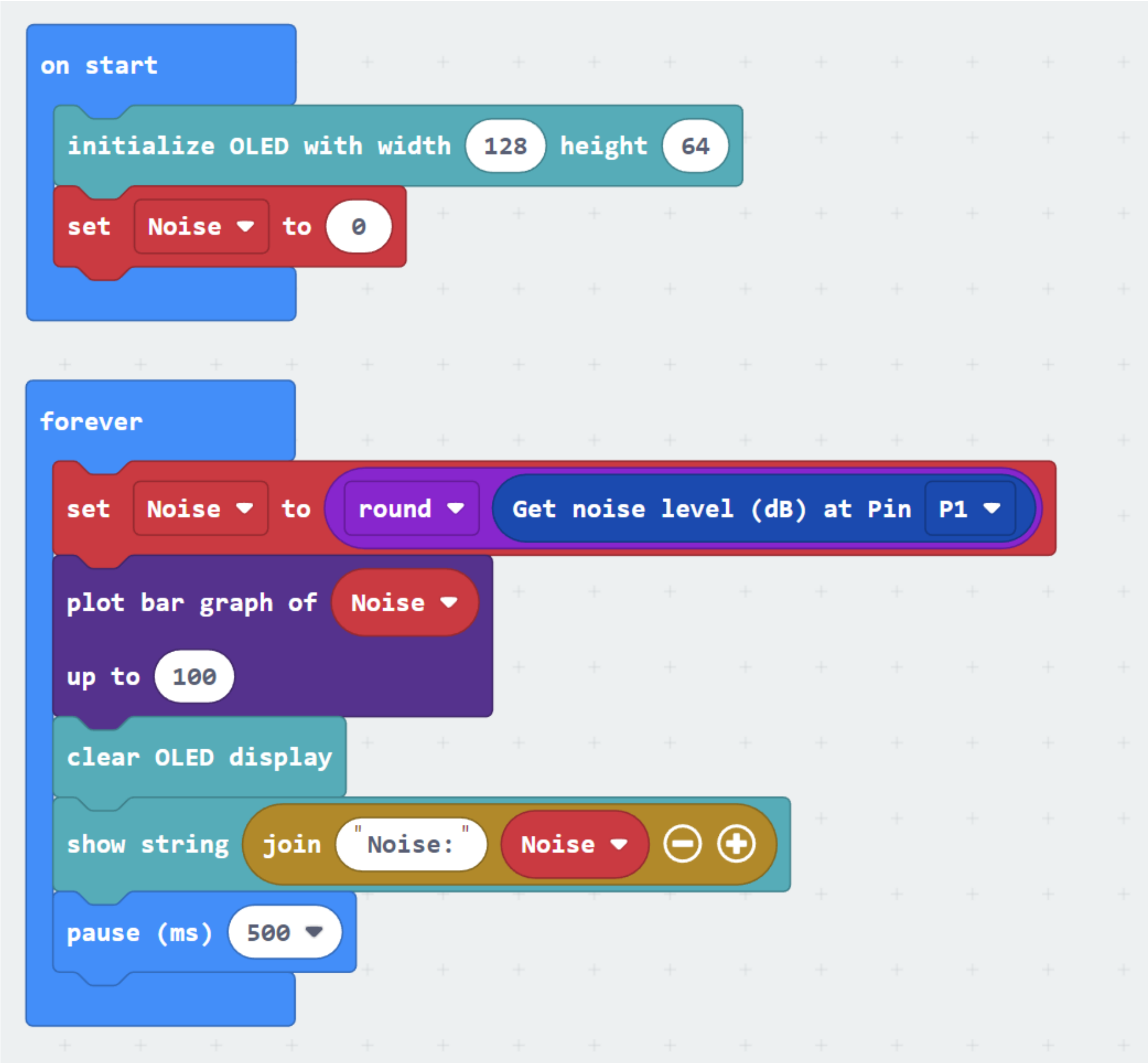
Step 3. Show the noise value on OLED screen

- Snap `clear OLED display` from `OLED` to avoid overlap
- Snap `show string` to the loop and show value of the variable `Noise`



```
forever
    set Noise ▾ to round ▾ Get noise level (dB) at Pin P1 ▾
    plot bar graph of Noise ▾
    up to 100
    clear OLED display
    show string join "Noise: " Noise ▾ "dB" ⊖ ⊕
    pause (ms) 500 ▾
```

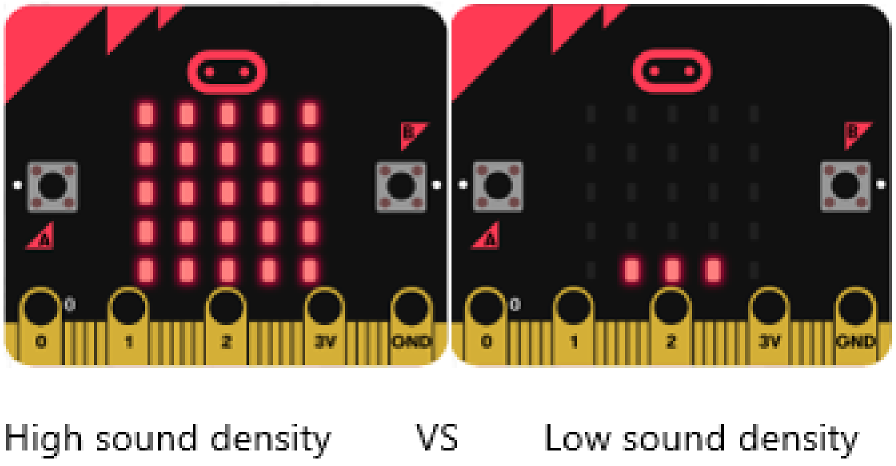
Full Solution MakeCode: https://makecode.microbit.org/_TiW9H7CuU5M5



```
on start
    initialize OLED with width 128 height 64
    set Noise to 0

forever
    set Noise to round ( Get noise level (dB) at Pin P1 )
    plot bar graph of Noise
    up to 100
    clear OLED display
    show string join "Noise:" Noise
    pause (ms) 500
```

Results

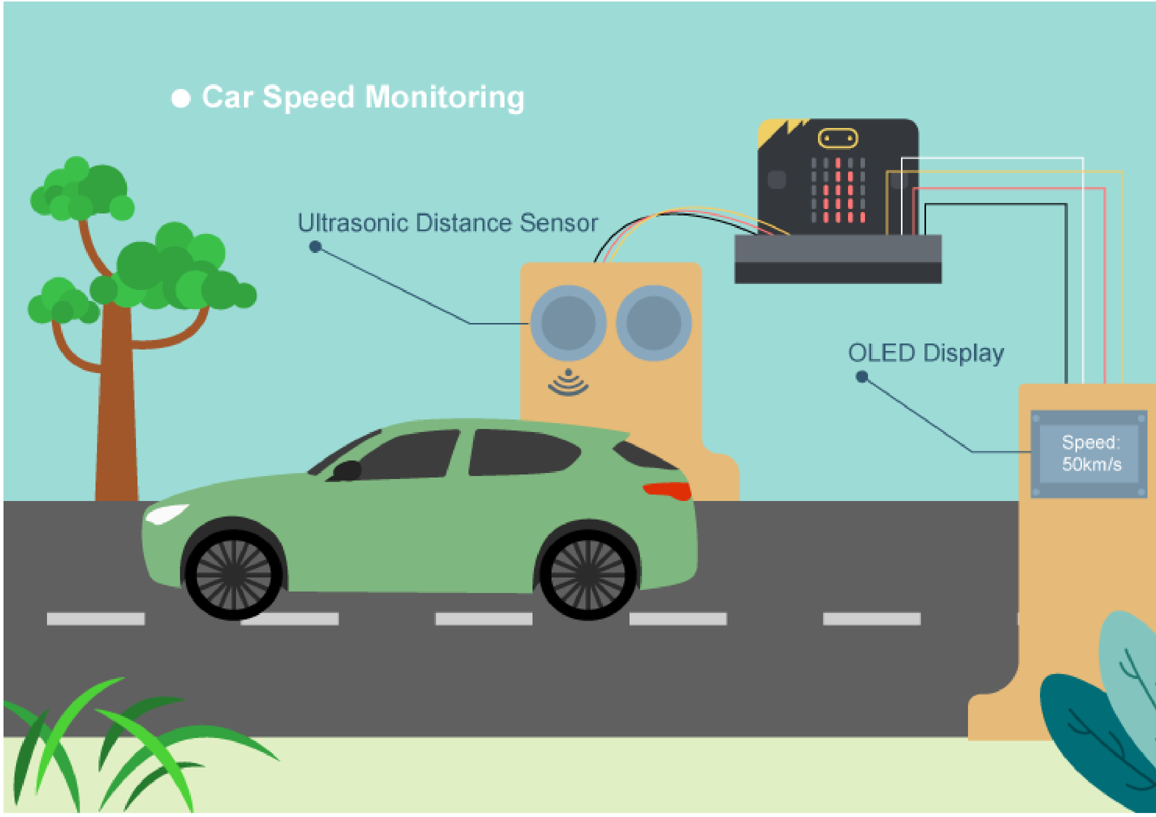
After initializing WiFi extension board and OLED, micro:bit will show a bar graph for the sound intensity in a city.



Think

Q1. How to make a notification if noise pollution problem is serious? i.e. showing red LED

1.1.14 Case 05: Car speed monitoring



Level: ★ ★ ★ ☆ ☆

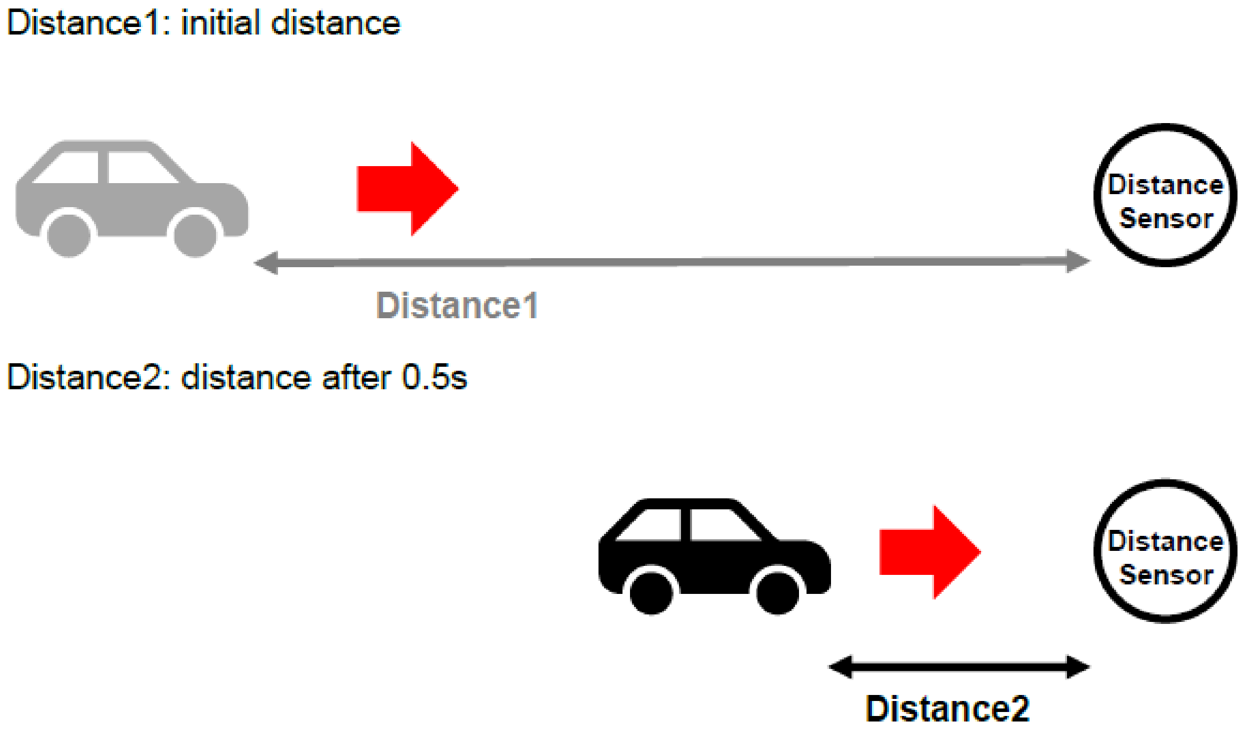
Goal

Make a car speed monitor to detect car speed on the road.

Background

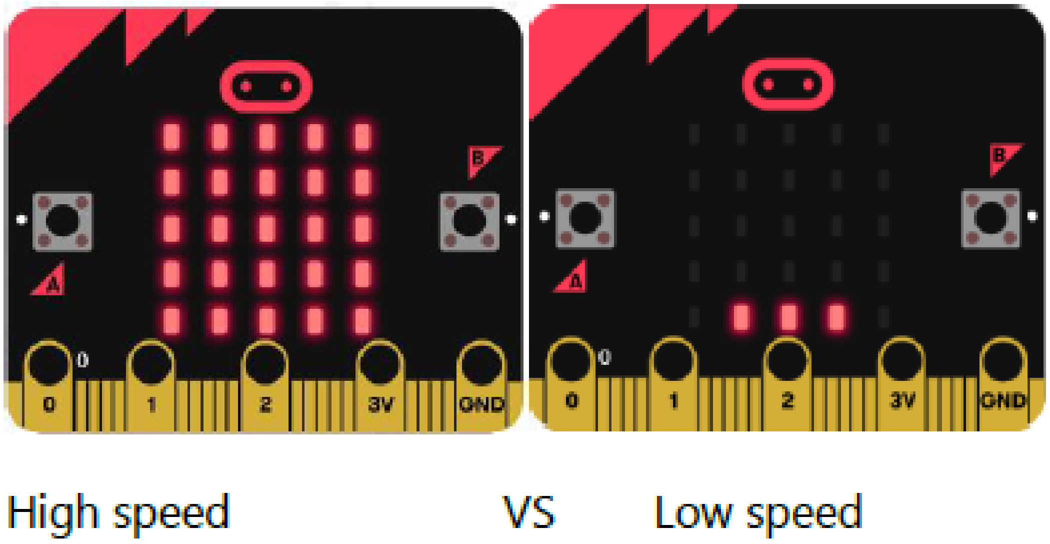
What is car speed monitoring? It is an automatic system to check car speed on the road at certain time interval. There are cars often over-speed causing traffic accidents, therefore installing a car speed monitoring is a must to minimize the chances of traffic accidents.

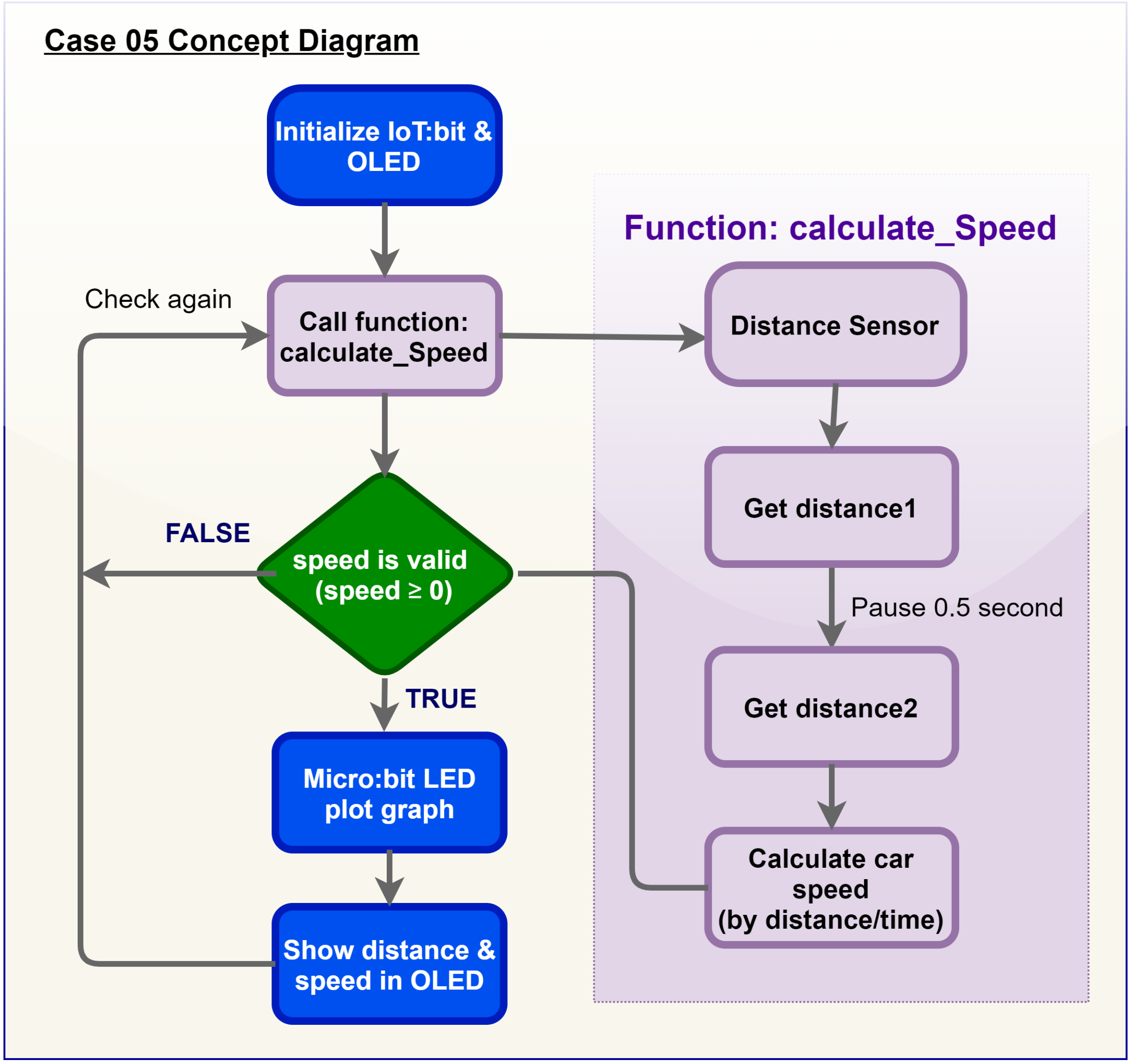
Car speed monitor operation The distance sensor measures two different distances at certain time interval, and therefore car speed can be calculated and shows it on the OLED. On every 500ms (0.5 second), the distance sensor will keep updating distance between the sensor and the car.



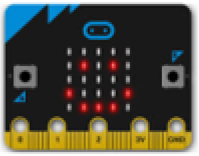
By equation, $\text{Speed} = \frac{\text{distance}}{\text{time}}$,we found that $\text{Speed} = \frac{\text{distance1} - \text{distance2}}{\text{time}}$

If distance 1 distance 2, that's say the car is moving towards. The moving distance is distance1 -distance2. The speed is (distance1-distance2)/0.5 (unit: cm/s) If distance 1 = distance 2, that's say the car has stop moving or there are no cars. The moving distance and speed are 0 For car speed < 0, it is the exceptional case (the car turns left/right and leave the road) and the speed will not be shown. A bar graph on micro:bit LED is shown to indicate the instant speed of the cars.

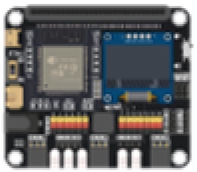

0
1
2
3V
GND
A
B
High speed
VS
Low speed


Case 05 Concept Diagram
Initialize IoT:bit & OLED
Call function: calculate_Speed
Check again
speed is valid (speed ≥ 0)
FALSE
TRUE
Micro:bit LED plot graph
Show distance & speed in OLED
Function: calculate_Speed
Distance Sensor
Get distance1
Pause 0.5 second
Get distance2
Calculate car speed (by distance/time)

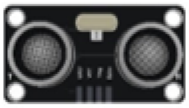
Part List



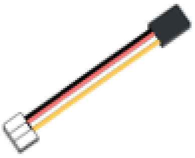
micro:bit (1)



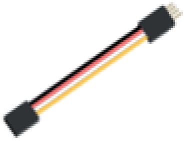
IoT:bit with OLED (1)



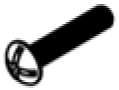
Distance Sensor (1)



4-pin module wire (1)



4-pin extension wire (1)



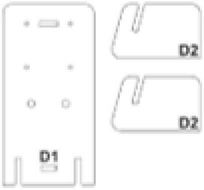
M2*10mm Screw (4)

M2 Nut (4)

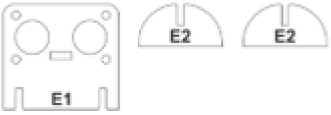
M4*10mm Screw (4)

M4 Nut (4)

Screwdriver (1)



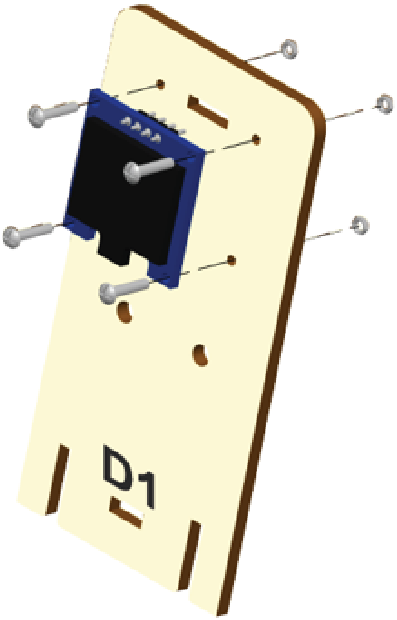
D1-D2 Model (1)

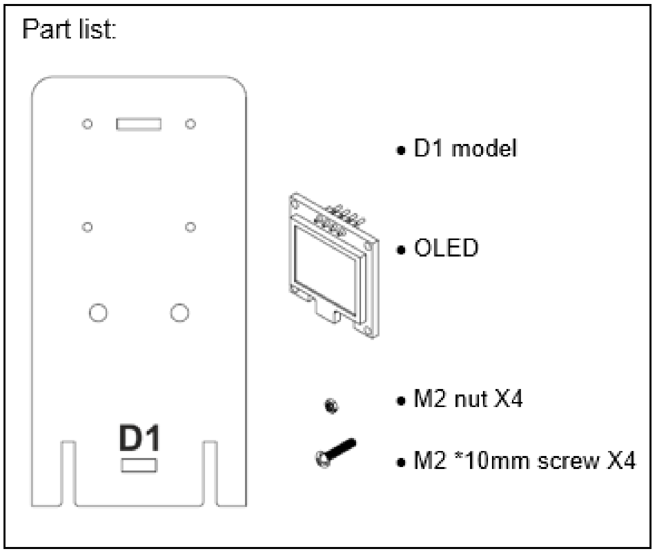


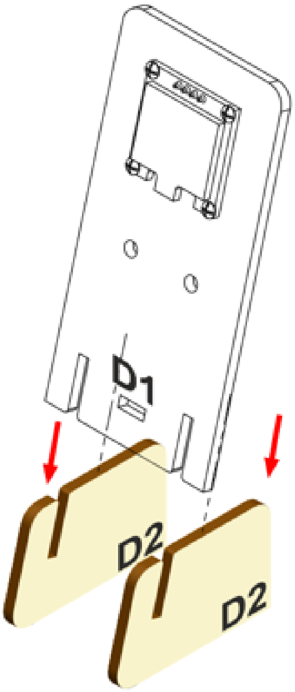
E1-E2 Model (1)

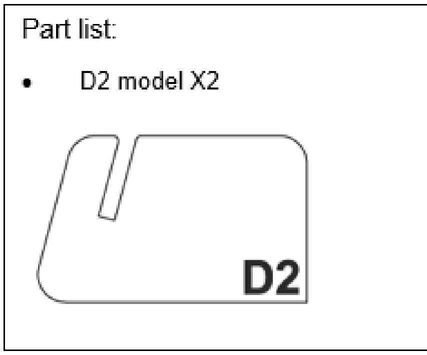
Assembly step

Step 1 Attach the OLED to D1 model using M2 * 10mm screw and nut.



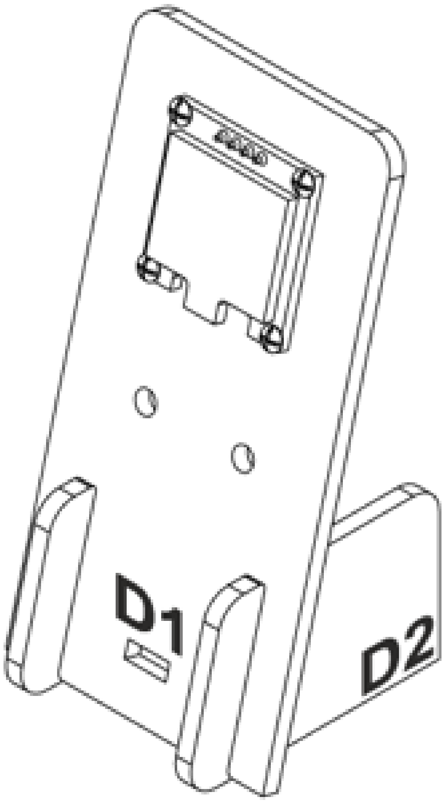




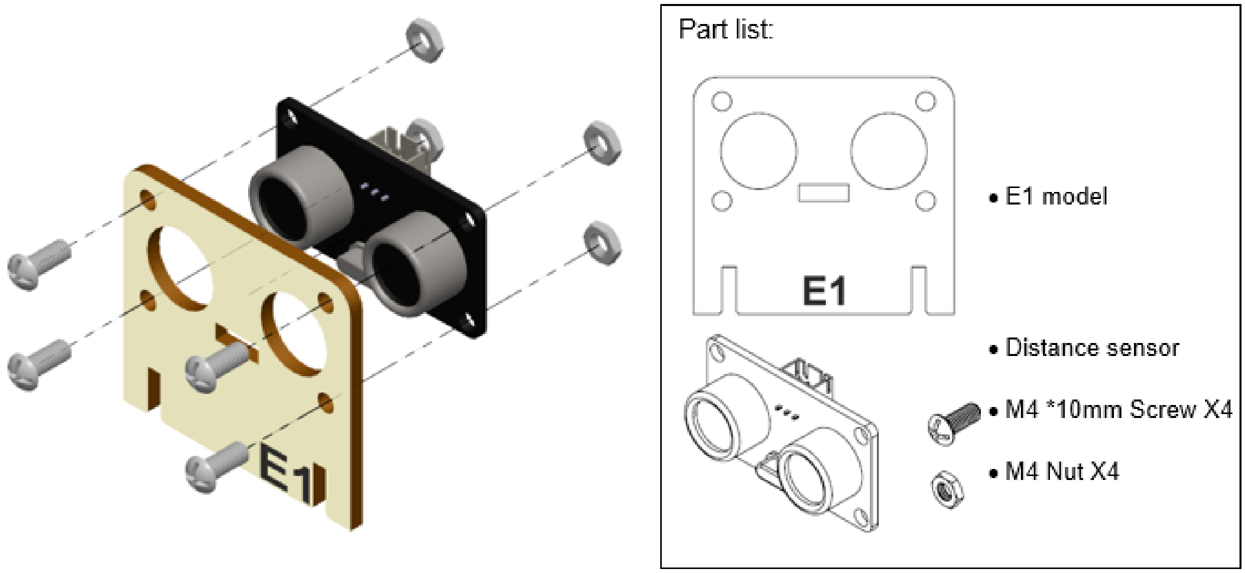

Part list:

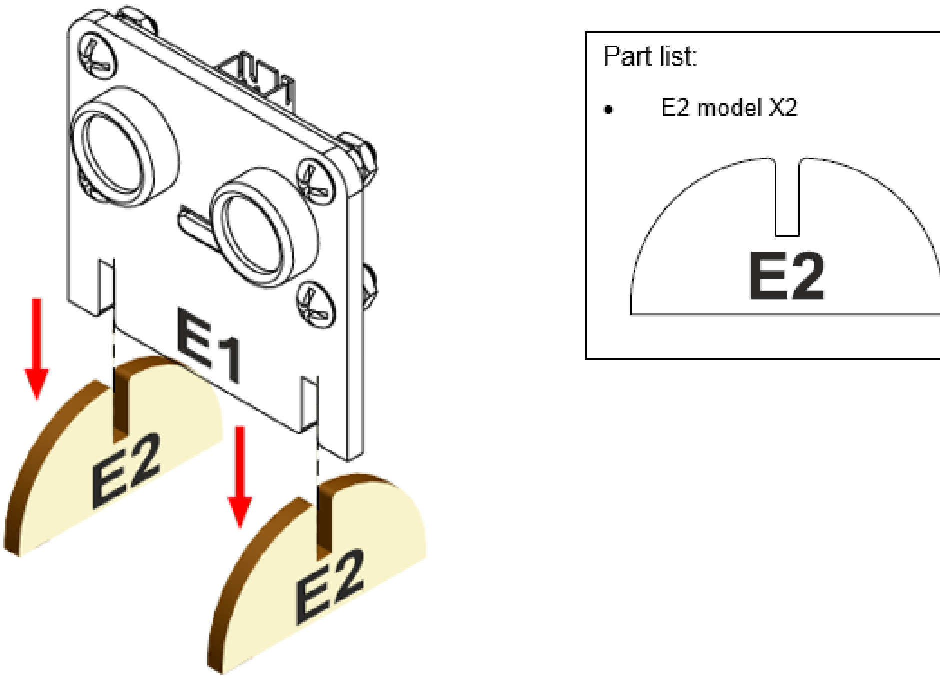
- D2 model X2

Step 2 Put D2 model onto the D1 model.

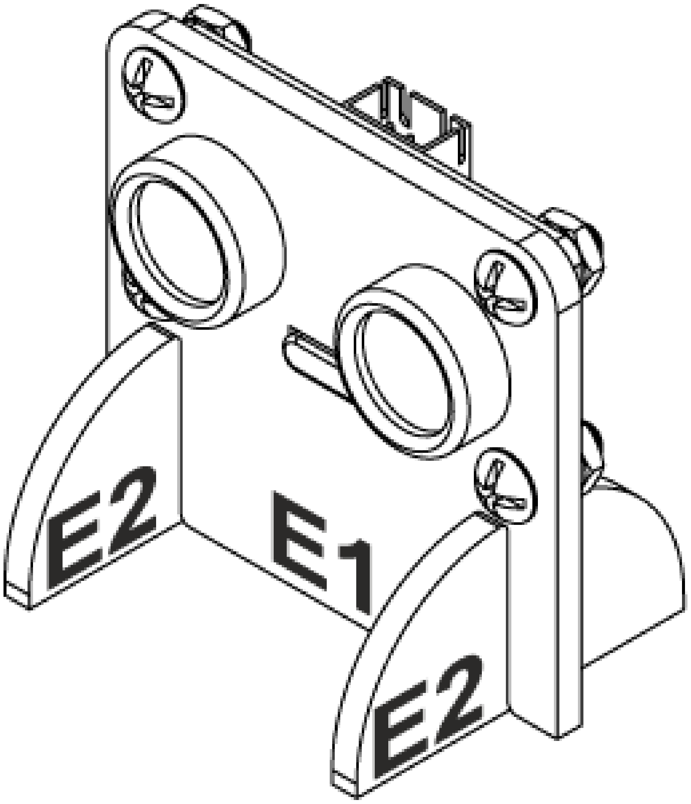


Step 3 Assembly completed! Step 4 Attach the distance sensor to E1 model using M4 screw and nut.





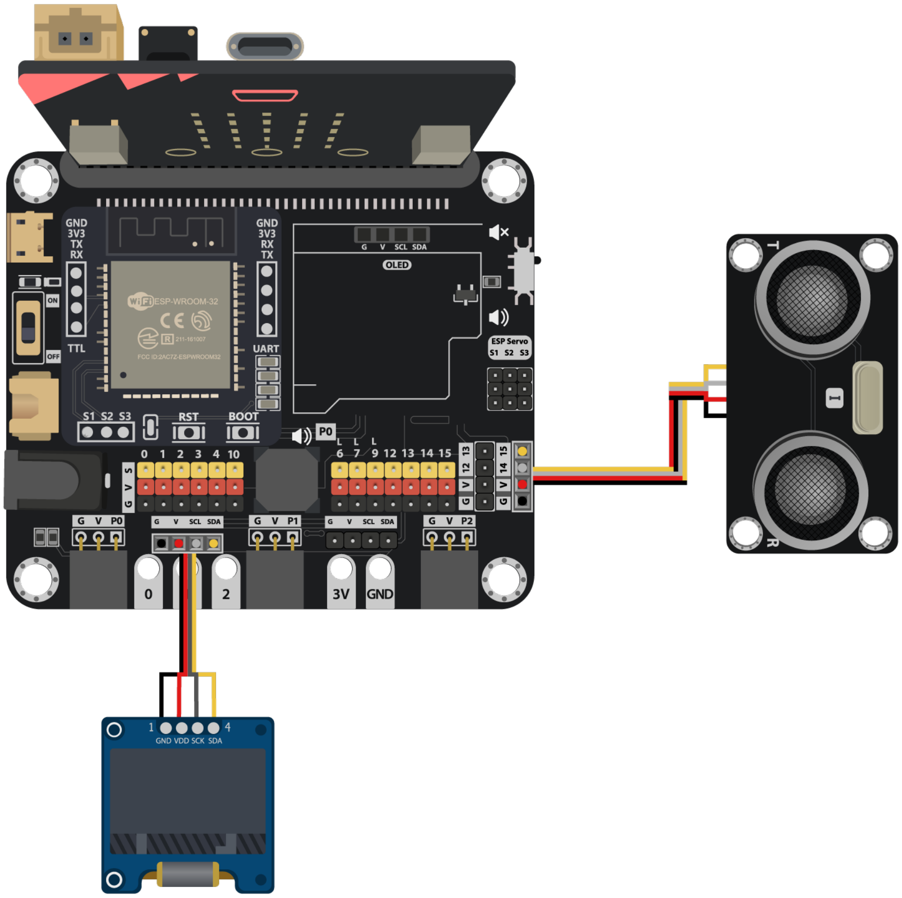
Step 5 Put the E2 model onto E1 model.



Step 6 Assembly completed!

Hardware connect

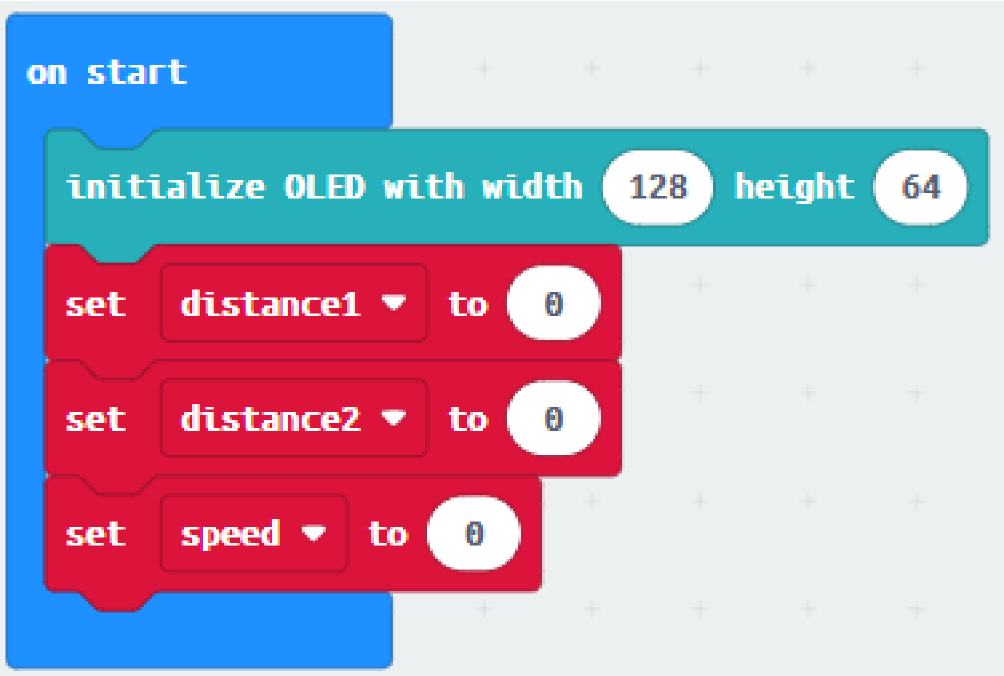
Connect the Distance Sensor to P14 (trig)/ P15 (echo) port of IoT:bit
Extend the connection of OLED to the I2C connection port of IoT:bit



Programming (MakeCode)

Step 1. Initialize OLED screen

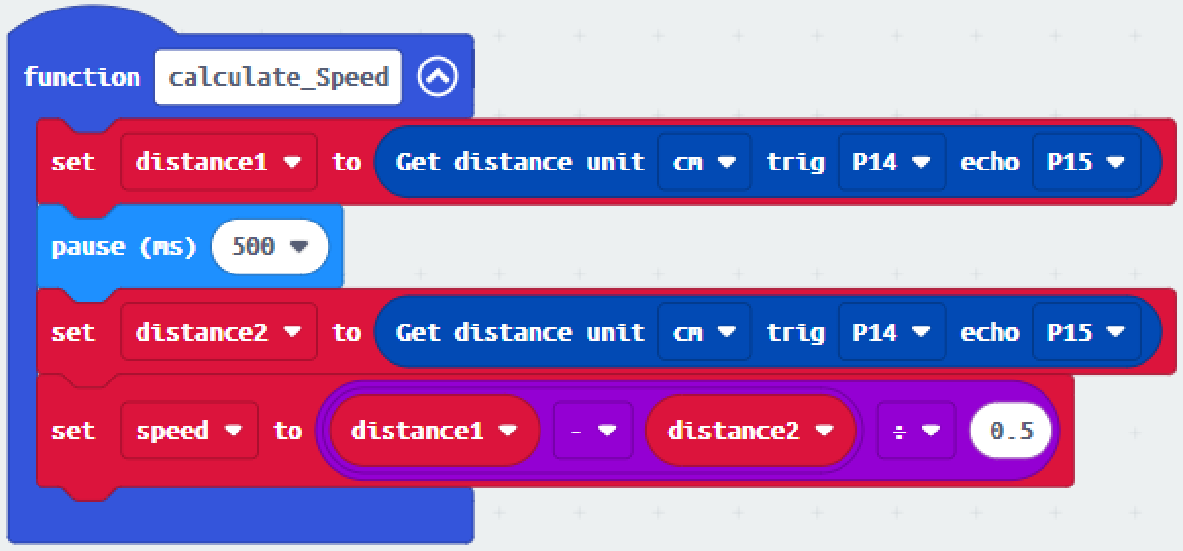
- Drag `Initialize OLED with width:128, height: 64` to `on start`
- Set `distance1`, `distance2` and `speed` to 0 from `variables`



```
on start
    initialize OLED with width 128 height 64
    set distance1 to 0
    set distance2 to 0
    set speed to 0
```

Step 2. Set up function (calculate_Speed)

- Set up a new function `calculate_Speed` from `Advanced` > `Functions`.
- Set `distance1` to `get distance unit cm trig P14 echo P15` (distance from the car to the distance sensor before 0.5 second) Drag `Pause` to wait 500ms and set `distance2` to `get distance unit cm trig P14 echo P15` (distance from the car to the distance sensor after 0.5 second)
- By the equation of speed = distance / time. We get the `speed` of the moving car to `(distance1-distance2)`/0.5 (unit: cm/s)

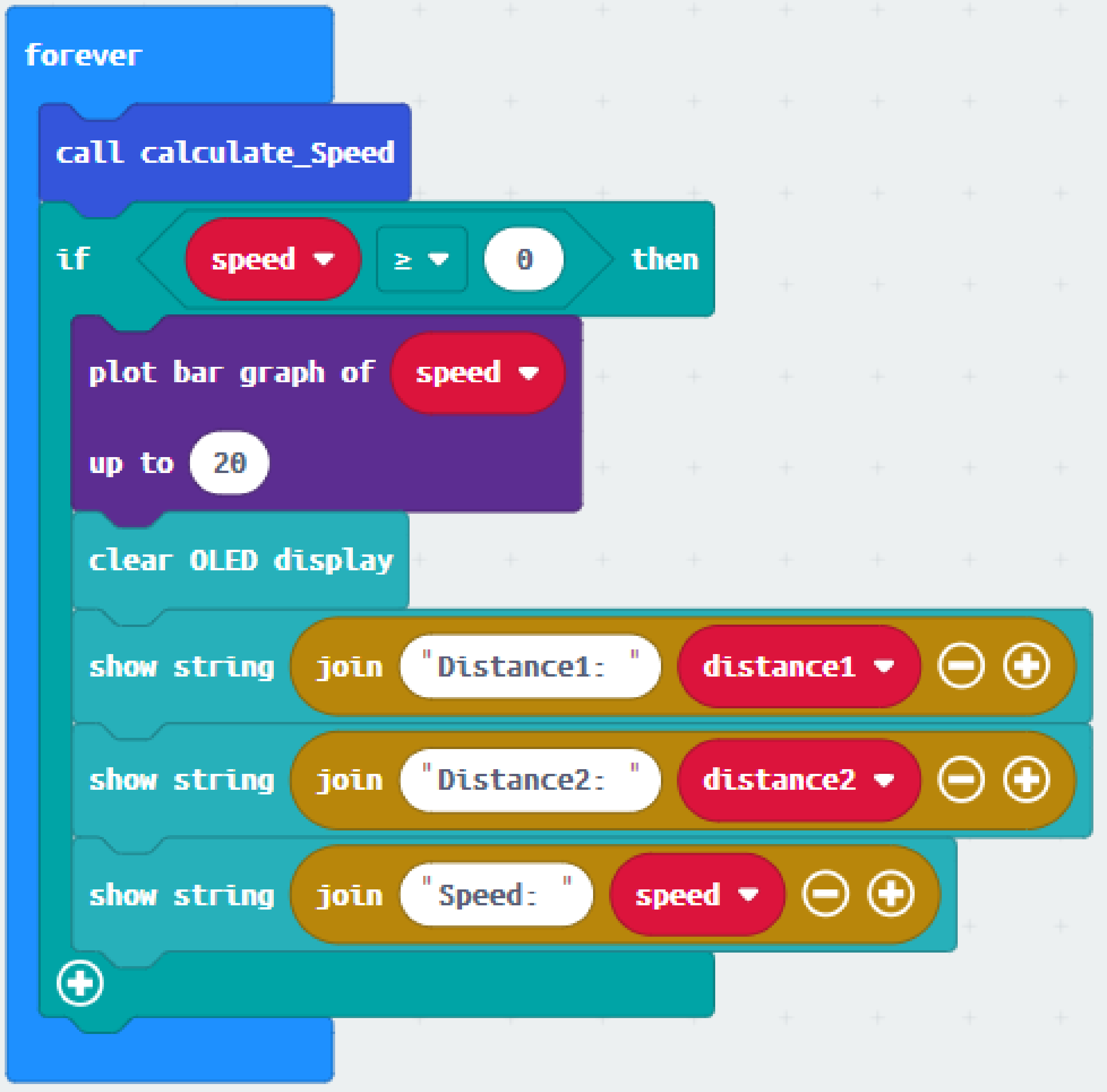


```
function calculate_Speed
    set distance1 to Get distance unit cm trig P14 echo P15
    pause (ms) 500
    set distance2 to Get distance unit cm trig P14 echo P15
    set speed to distance1 - distance2 ÷ 0.5
```

Step 3. Calculate car speed

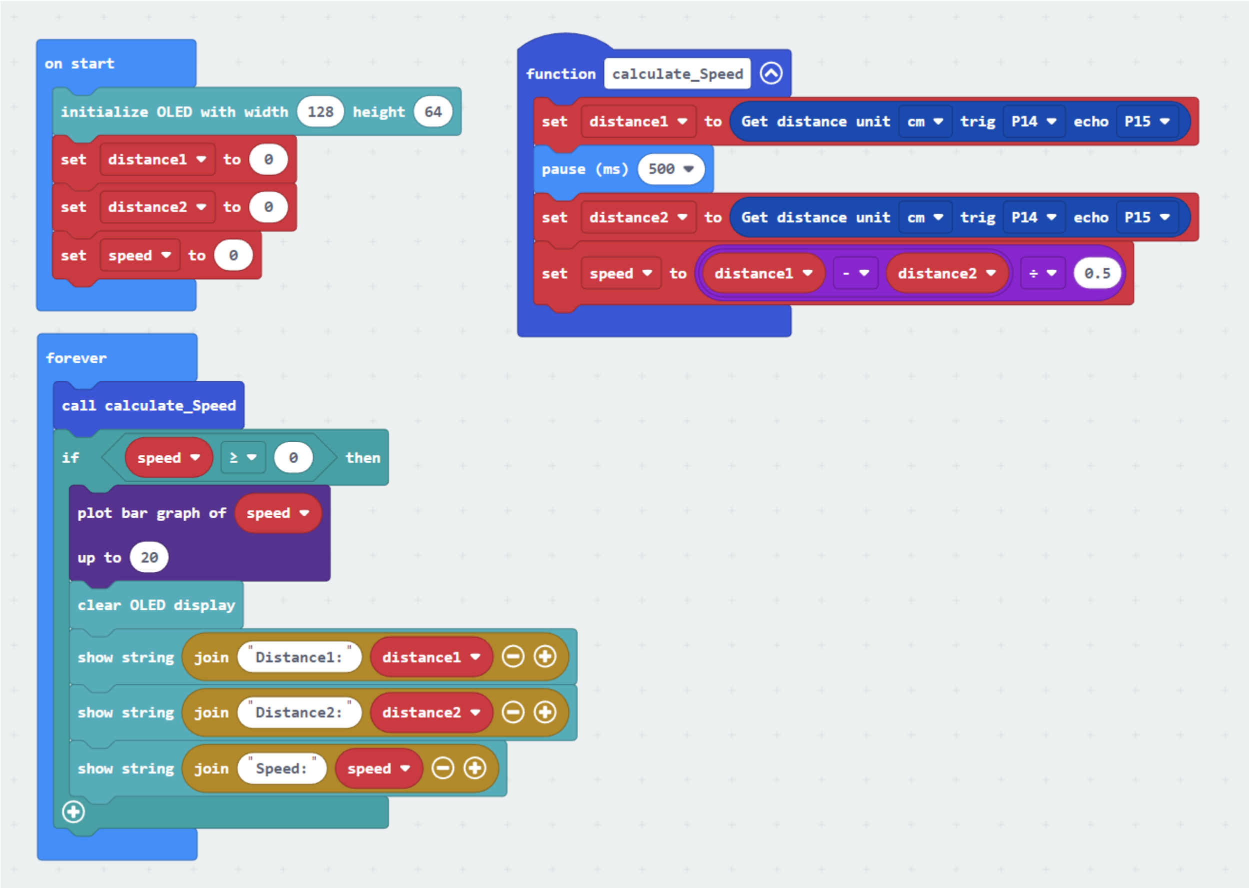
- In block `forever`, call function `calculate_Speed` from `Advanced` > `Functions` to get the speed of the moving car
- Snap `If statement` into the loop

- If speed 0, then it will plot bar graph of ... from Led and draw variable speed into the plotted value. Set value up to 20
- Snap clear OLED display from OLED to avoid overlap
- Snap show string and show value of variables distance1, distance2 and speed



```
forever
    call calculate_Speed
    if speed ▾ ≥ ▾ 0 then
        plot bar graph of speed ▾
        up to 20
        clear OLED display
        show string join "Distance1: " distance1 ▾ ⊖ ⊕
        show string join "Distance2: " distance2 ▾ ⊖ ⊕
        show string join "Speed: " speed ▾ ⊖ ⊕
    ⊕
```

Full Solution MakeCode: https://makecode.microbit.org/_0Y0h5MWPde4A



```
on start
    initialize OLED with width 128 height 64
    set distance1 to 0
    set distance2 to 0
    set speed to 0

function calculate_Speed
    set distance1 to Get distance unit cm trig P14 echo P15
    pause (ms) 500
    set distance2 to Get distance unit cm trig P14 echo P15
    set speed to distance1 - distance2 ÷ 0.5

forever
    call calculate_Speed
    if speed ≥ 0 then
        plot bar graph of speed
        up to 20
        clear OLED display
        show string join "Distance1:" distance1
        show string join "Distance2:" distance2
        show string join "Speed:" speed
```

Results

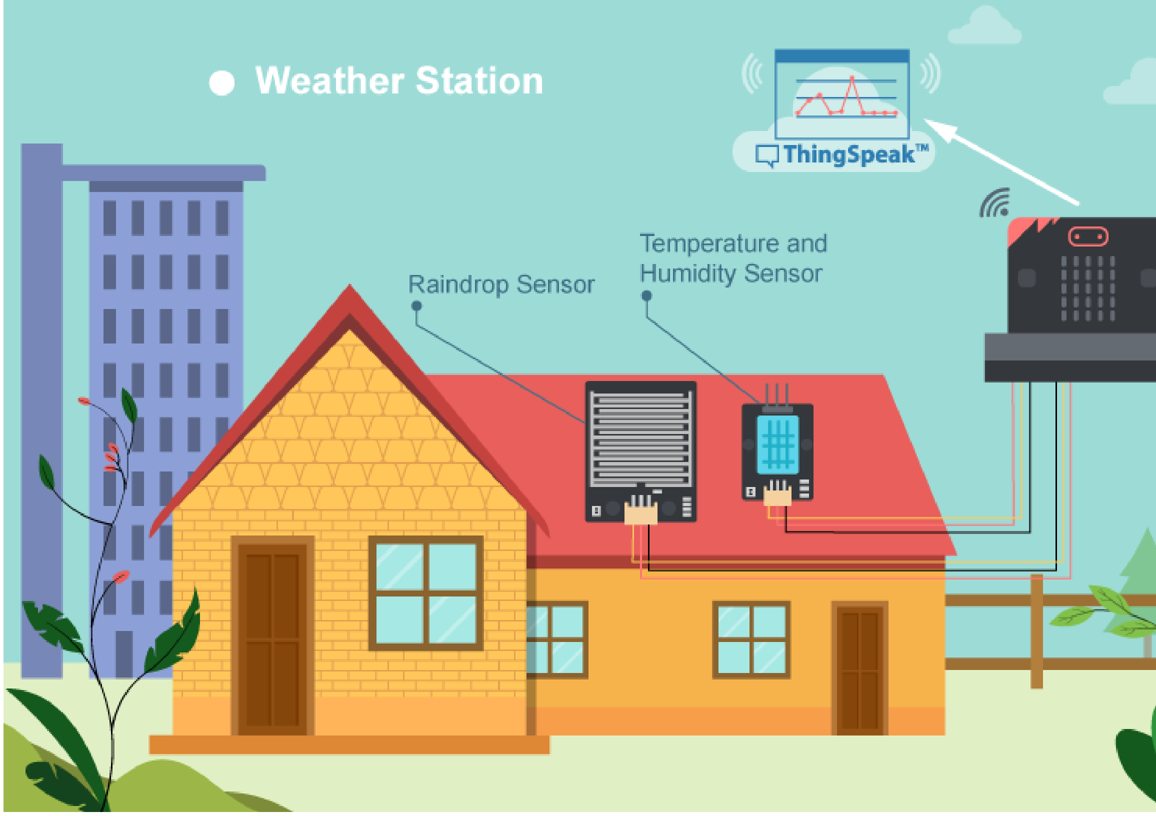
It will keep checking the distance of cars from distance sensor by distance sensor in every 500ms. The speed of the cars will be shown on OLED. A bar chart of speed will be shown on micro:bit LED also.

Think

How can we set a sound alert to notify that there is car over-speeding?

1.1.15 IoT Case 06: Weather Station

Level: ★ ★ ☆ ☆ ☆



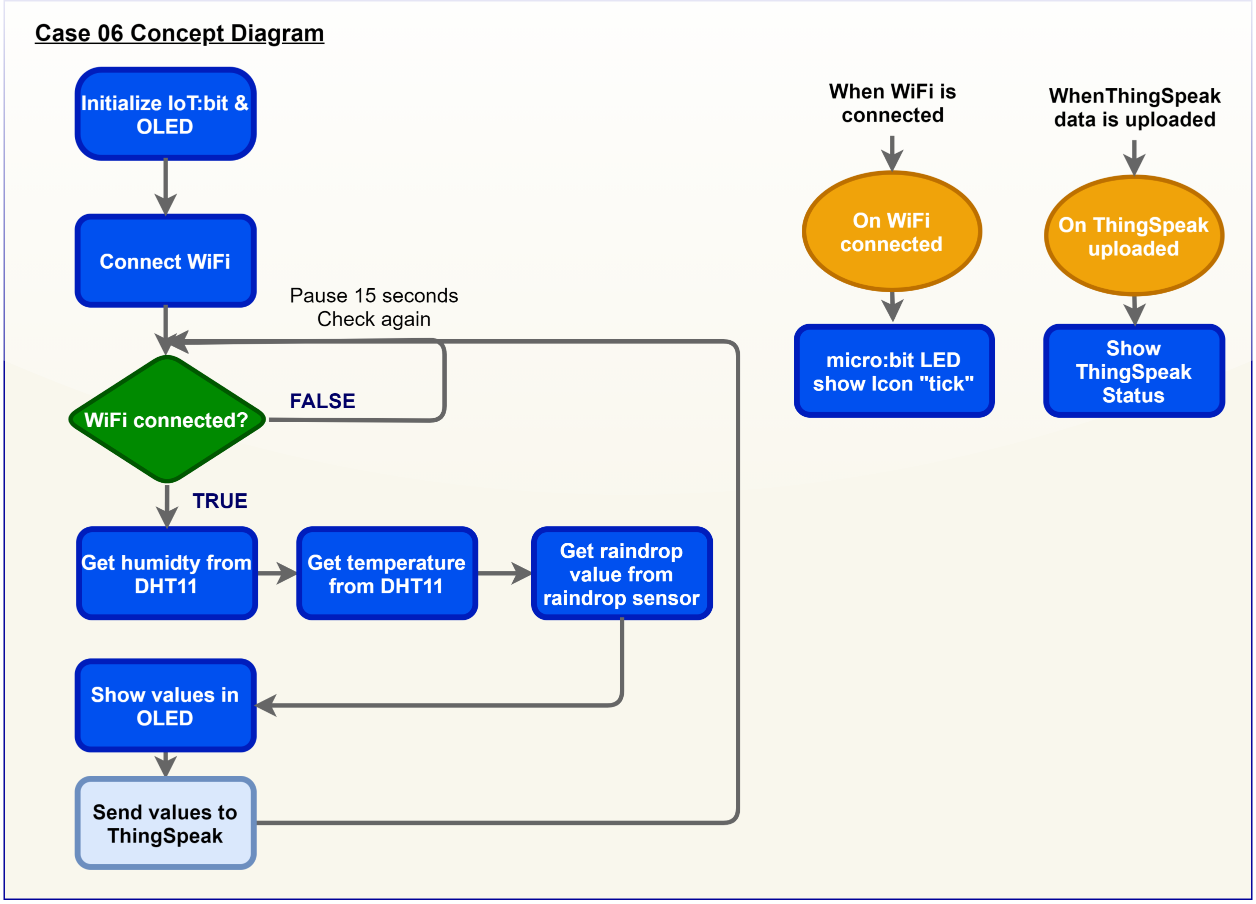
Goal

Make a weather station which gets the values from the raindrop sensor, temperature and humidity sensor. The data will be sent to IoT platform - ThingSpeak.

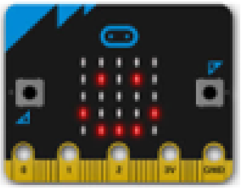
Background

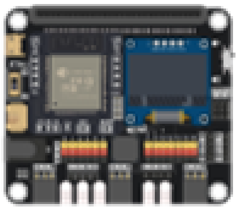
What is Thingspeak? ThingSpeak is an IoT analytics platform service which provides instant visualizations of data posted by your devices to ThingSpeak. In this case, we will use this platform to update our data to plot instant graphs.

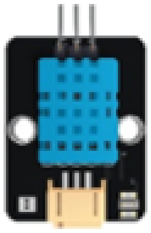
Weather station operation Collecting temperature, humidity and raindrop consistently and uploading the data by using Thingspeak. This can help us to do analytical work more conveniently as we can refer to the automatically plotted graphs.


Case 06 Concept Diagram
Initialize IoT:bit & OLED
Connect WiFi
Pause 15 seconds
Check again
WiFi connected?
FALSE
TRUE
Get humidty from DHT11
Get temperature from DHT11
Get raindrop value from raindrop sensor
Show values in OLED
Send values to ThingSpeak
When WiFi is connected
On WiFi connected
micro:bit LED show Icon "tick"
WhenThingSpeak data is uploaded
On ThingSpeak uploaded
Show ThingSpeak Status

Part List

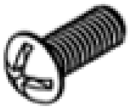

micro:bit (1)

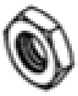

IoT:bit with OLED (1)


Temperature and humidity Sensor (1)

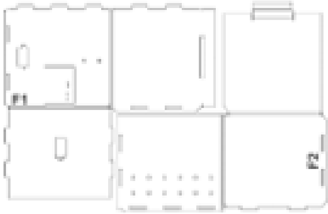
Raindrop Sensor (1)

3-pin module wire (2)

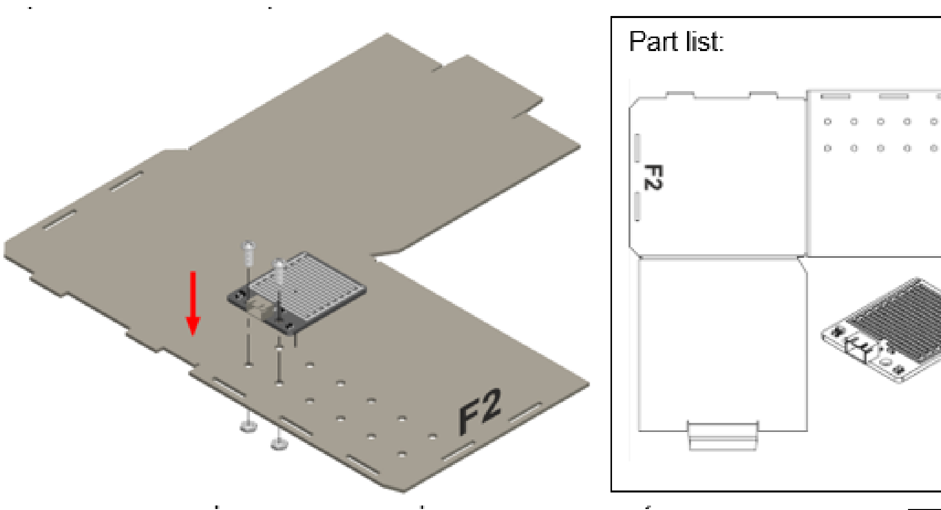

M4*10mm Screw (4)


M4 Nut (4)

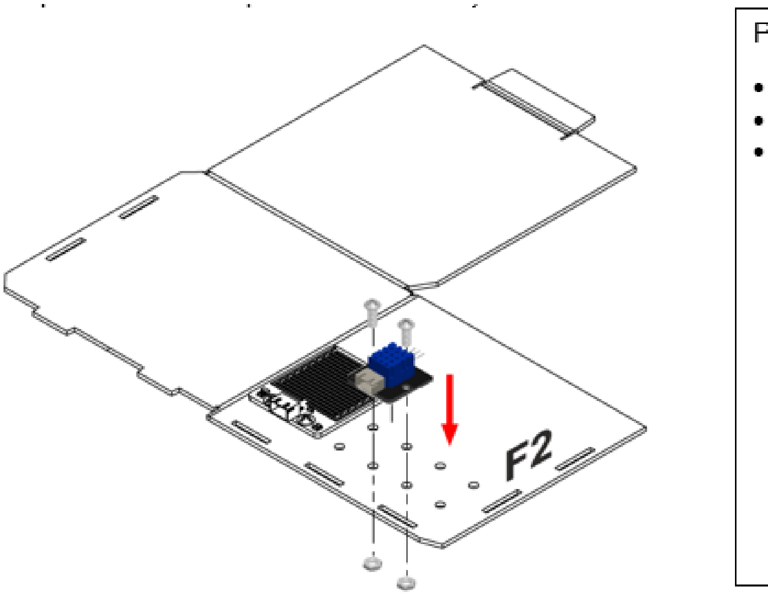
Screwdriver (1)


F1-F2 Model (1)

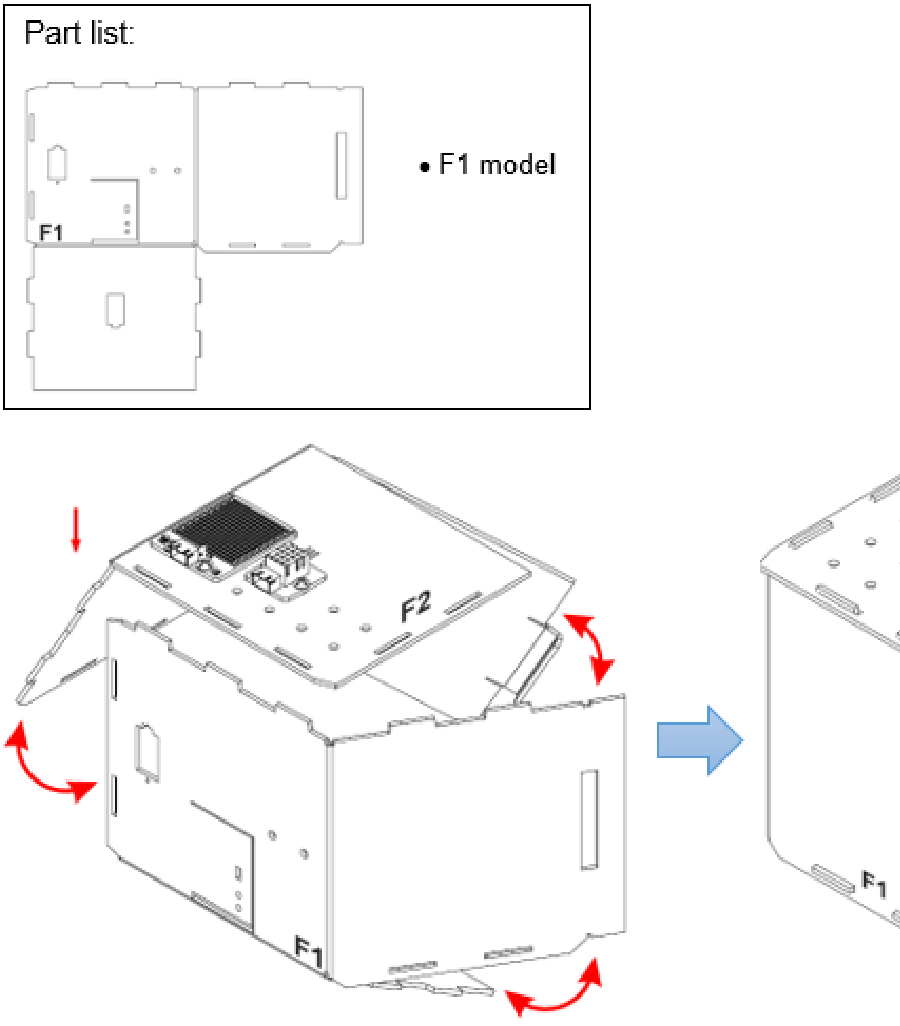
Assembly step



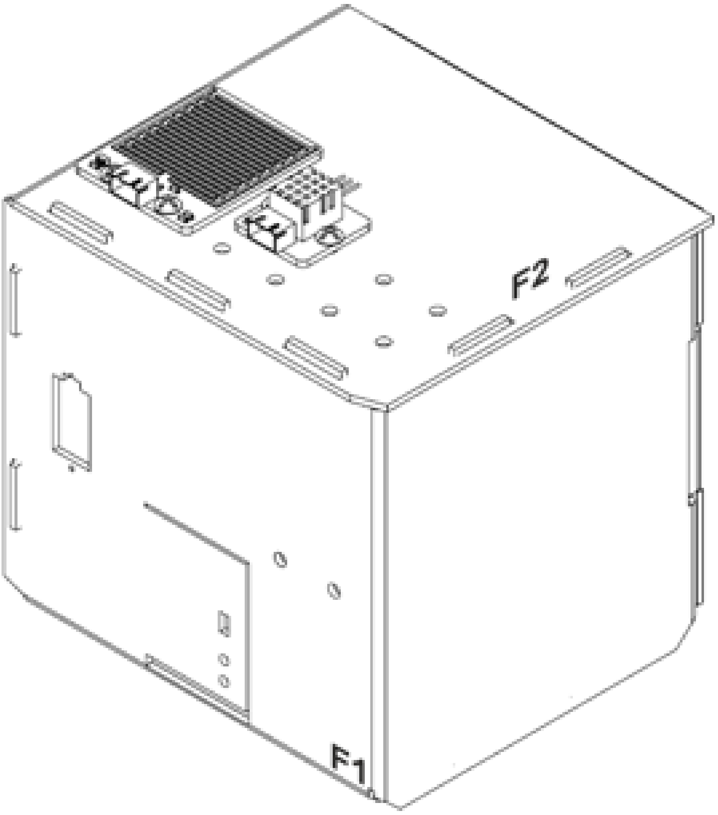
Step 1 Attach the raindrop sensor to F2 model.



Step 2 Attach the temperature and humidity sensor to F2 model.



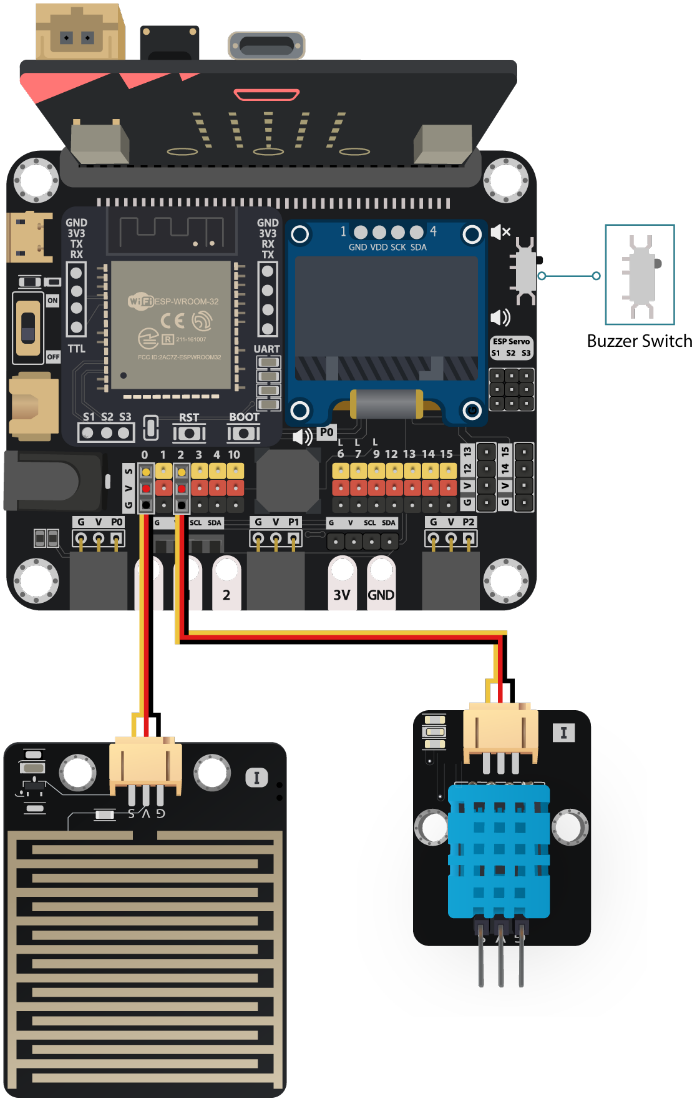
Step 3 Put together all the cardboard parts (F1-F2).



Step 4 Assembly completed!

Hardware connect

Connect the Raindrop Sensor to P0 port of IoT:bit Connect the Temperature and humidity Sensor to P2 port of IoT:bit



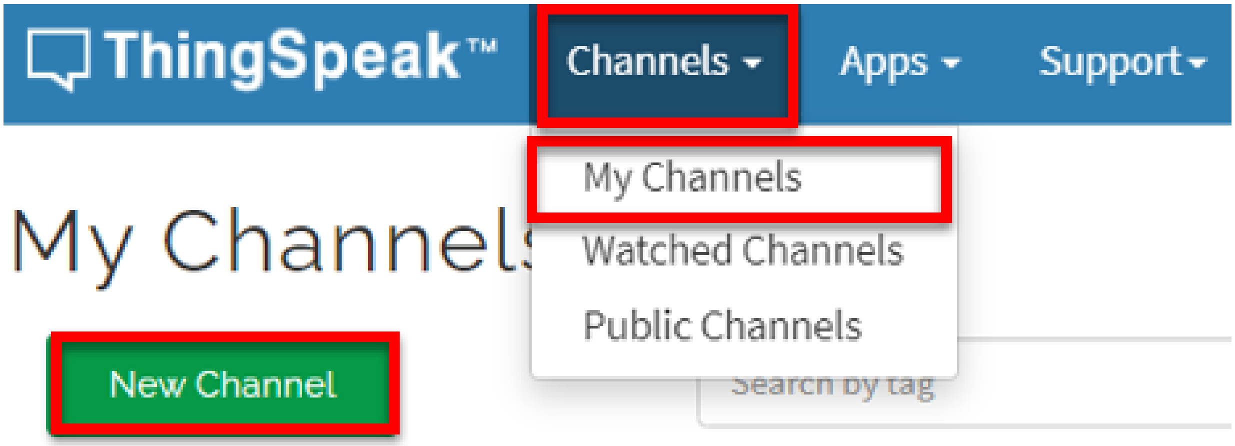
Buzzer Switch

Pull the buzzer switch 'up' to disconnect the buzzer in this execrise

IoT (ThingSpeak)

* For more details, please refer to Chapter 1: Upload Data to ThingSpeak

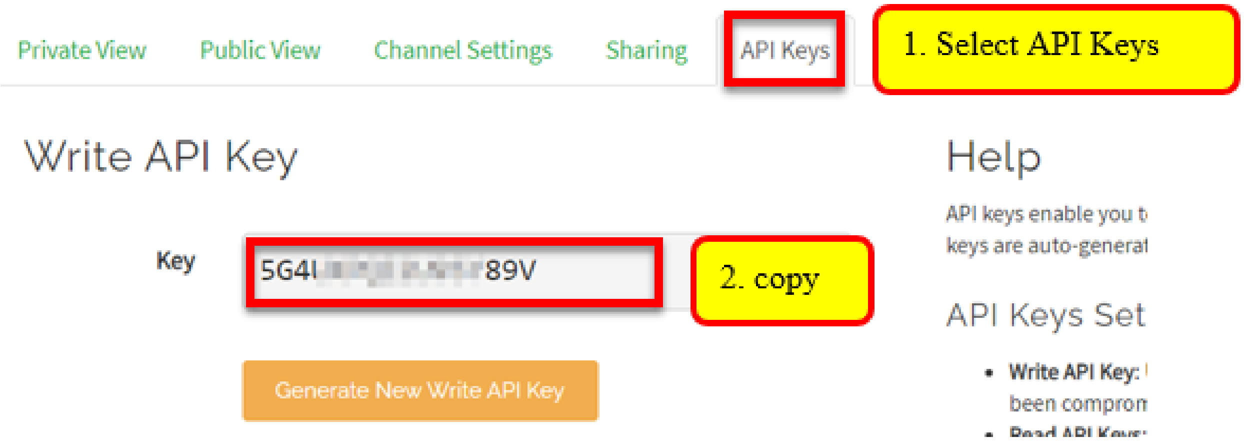
Step 1 Go to https://thingspeak.com, Choose Channels -> My Channels -> New Channel



Step 2 Input Channel name, Field1 and Field2 , then click "Save Channel"

- Channel name: Smart Weather Station
- Field 1: temperature
- Field 2: humidity
- Freld 3: raindrop

Step 3 Select your channel > "API Keys" copy the API key as follows:

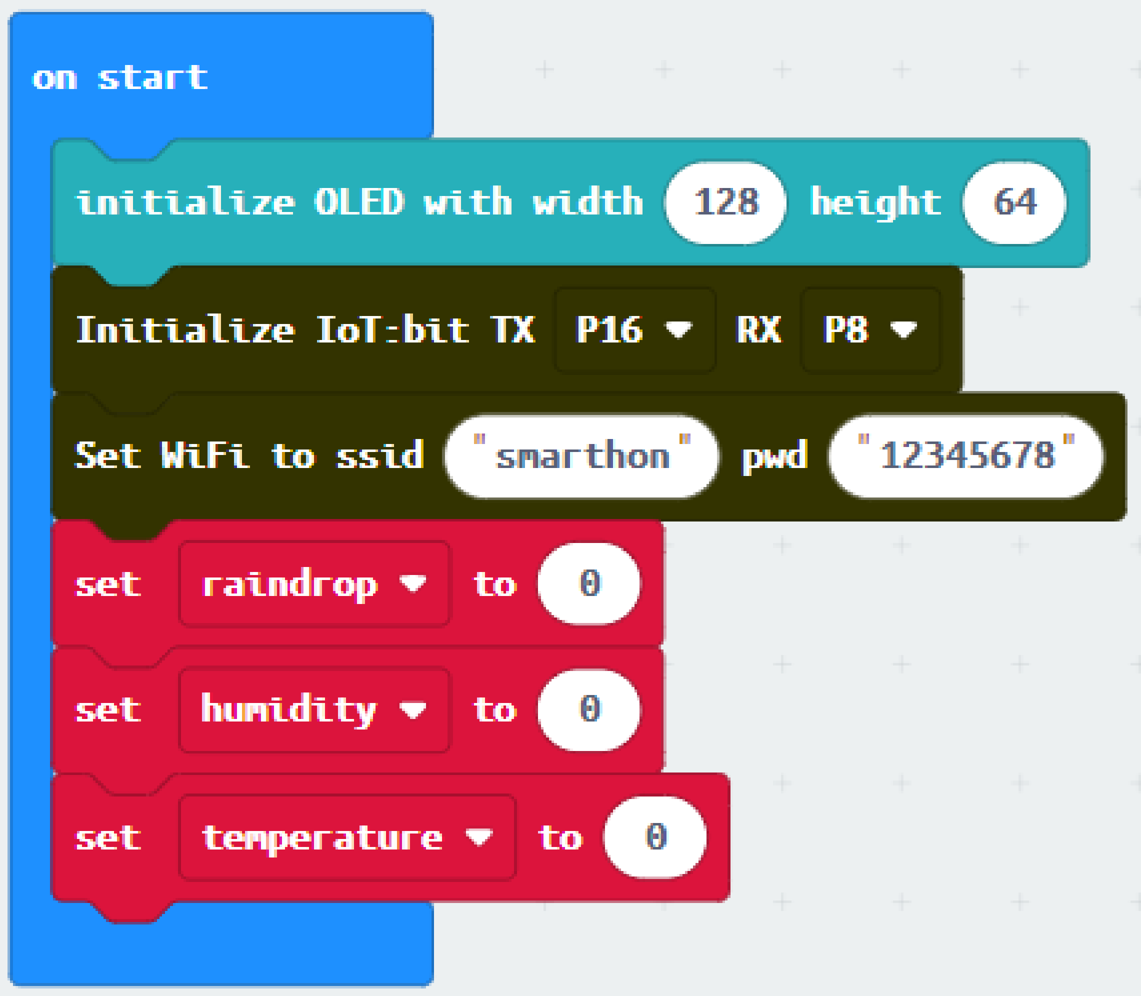


Programming (MakeCode)

Step 1. Initialize OLED, IoT:bit and connect to WiFi

- Snap `Initialize OLED with width:128, height:  64` to `on start`
- Snap `Initialize IoT:bit TX P16 RX P8` from `IoT:bit` to `on start`
- Snap `Set Wi-Fi to ssid pwd` from `IoT:bit`
- Enter your Wi-Fi name and password. Here we set `smarthon` as `SSID` and `12345678` as `password`

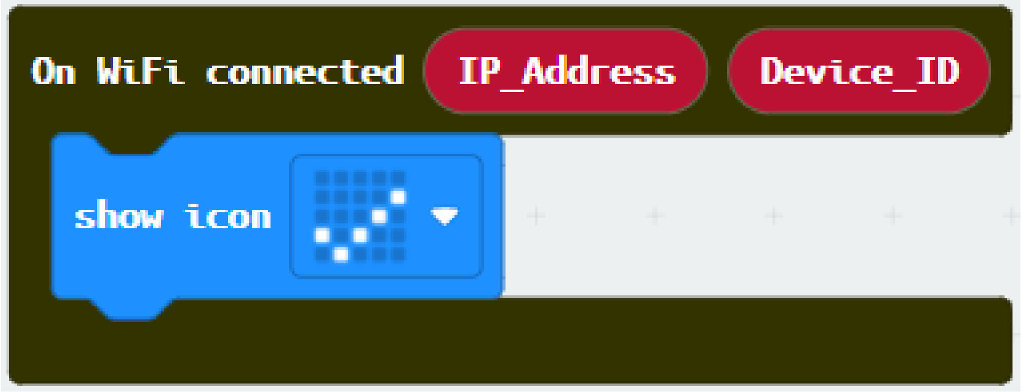
- Set variable `raindrop`, `humidity` and `temperature` to 0 from `variables*`



```
on start
    initialize OLED with width 128 height 64
    Initialize IoT:bit TX P16 ▾ RX P8 ▾
    Set WiFi to ssid "smarthon" pwd "12345678"
    set raindrop ▾ to 0
    set humidity ▾ to 0
    set temperature ▾ to 0
```

Step 2. Show icon "tick" after WiFi connection

- Snap `show icon` from `basic` to `On WiFi connected` and select icon `tick`

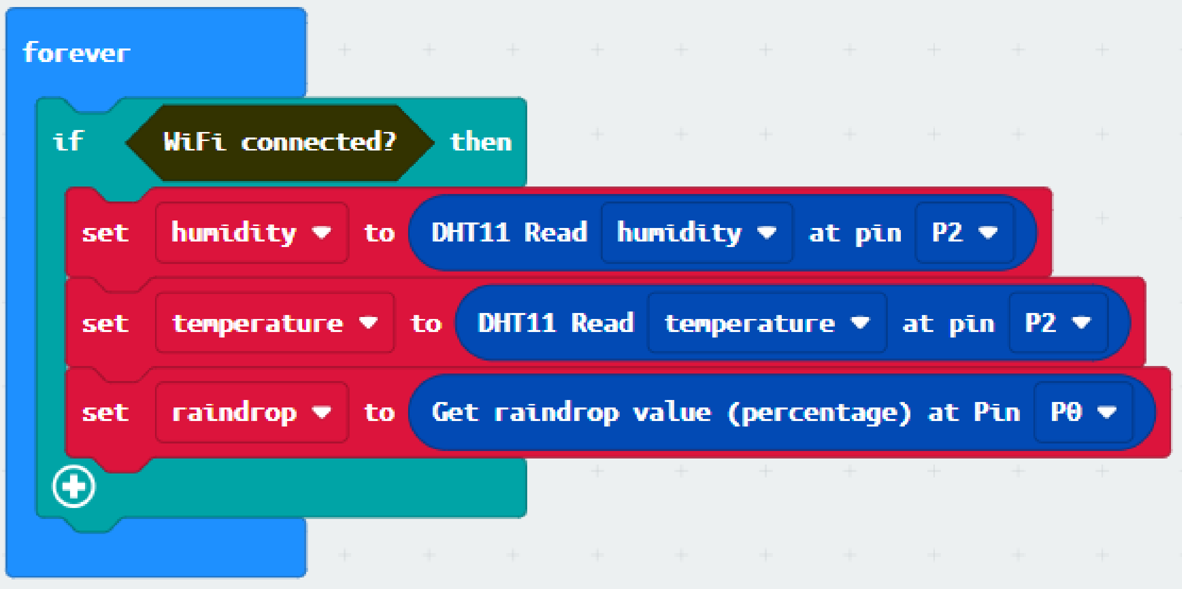


```
On WiFi connected IP_Address Device_ID
    show icon ▾
```

Step 3. Get temperature, humidity and raindrop values

- Snap `if statement` to block `forever`
- If `WiFi is connected`
- Then, set `humidity` to `read humidity from DHT11 at Pin P2`

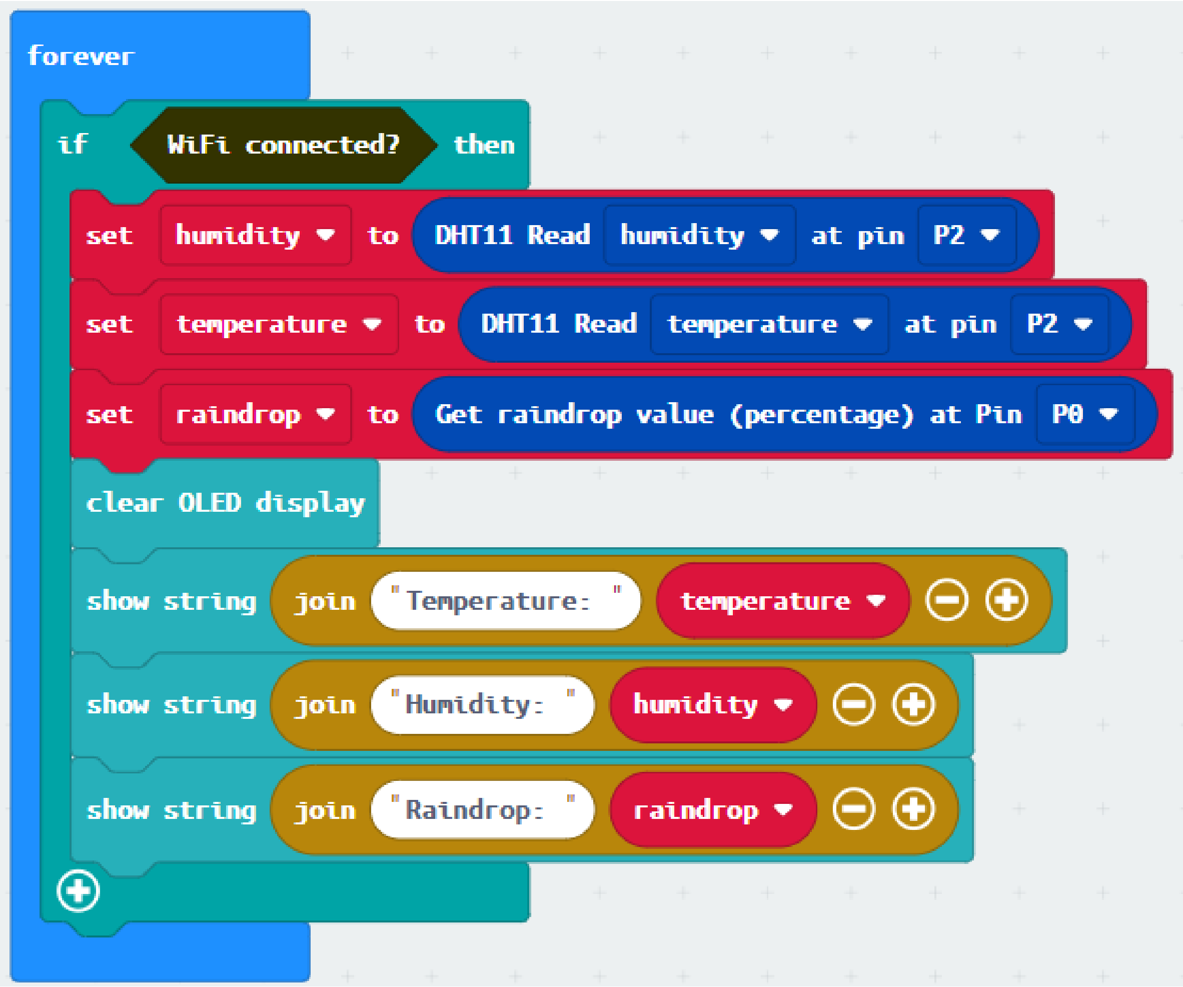
- Set `temperature` to `read temperature from DHT11`
- Set `raindrop` to `get raindrop value (percentage) at Pin P0`



```
forever
  if WiFi connected? then
    set humidity ▾ to DHT11 Read humidity ▾ at pin P2 ▾
    set temperature ▾ to DHT11 Read temperature ▾ at pin P2 ▾
    set raindrop ▾ to Get raindrop value (percentage) at Pin P0 ▾
  ⊕
```

Step 4. Show values on OLED

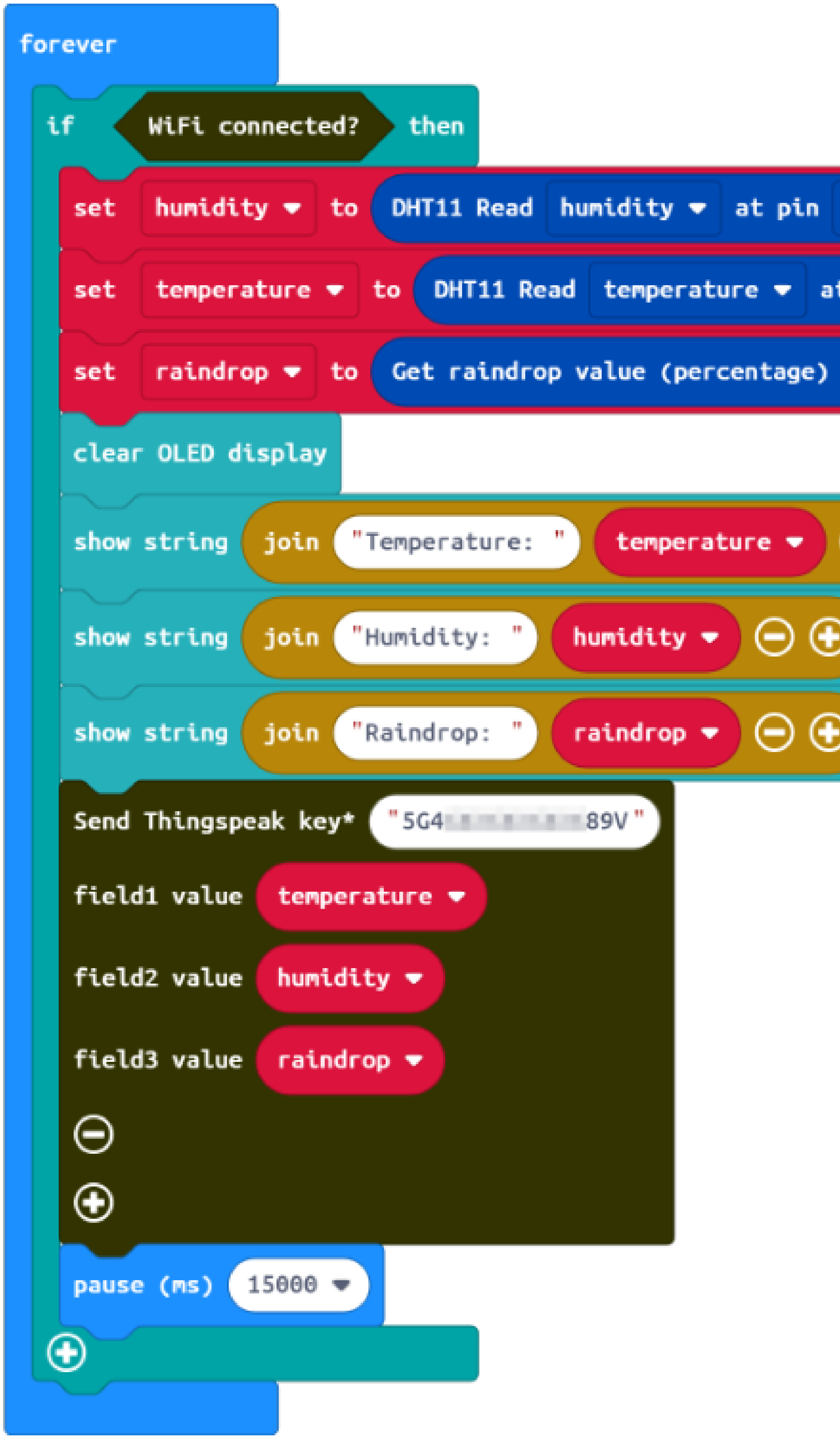
- Snap `clear OLED display` from `OLED` to avoid overlap
- Snap `show string` and `show value` of variables `Temperature`, `Humidity` and `Raindrop`



```
forever
  if WiFi connected? then
    set humidity ▾ to DHT11 Read humidity ▾ at pin P2 ▾
    set temperature ▾ to DHT11 Read temperature ▾ at pin P2 ▾
    set raindrop ▾ to Get raindrop value (percentage) at Pin P0 ▾
    clear OLED display
    show string join "Temperature: " temperature ▾ ⊖ ⊕
    show string join "Humidity: " humidity ▾ ⊖ ⊕
    show string join "Raindrop: " raindrop ▾ ⊖ ⊕
  ⊕
```

Step 5. Upload data to ThingSpeak

- Snap `Send Thingspeak key...` inside the `if-statement`
- Fill in the `api key` from Thingspeak with temperature, humidity and raindrop value

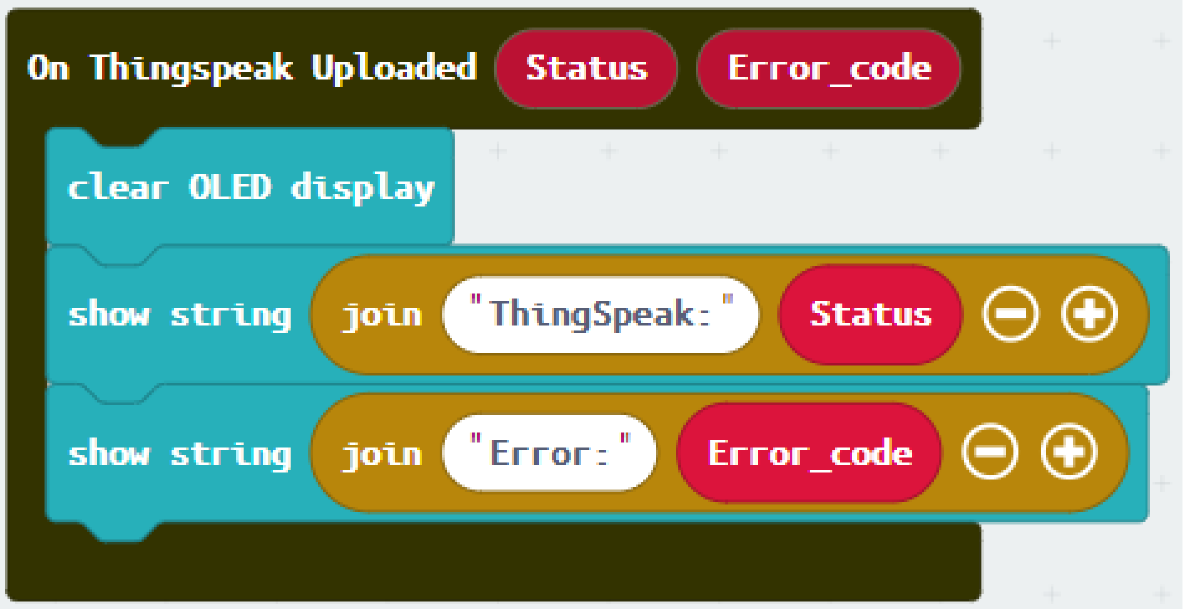


```
forever
  if WiFi connected? then
    set humidity ▾ to DHT11 Read humidity ▾ at pin
    set temperature ▾ to DHT11 Read temperature ▾ at
    set raindrop ▾ to Get raindrop value (percentage)
    clear OLED display
    show string join "Temperature: " temperature ▾
    show string join "Humidity: " humidity ▾ ⊖ ⊕
    show string join "Raindrop: " raindrop ▾ ⊖ ⊕
    Send Thingspeak key* "5G4[redacted]89V"
    field1 value temperature ▾
    field2 value humidity ▾
    field3 value raindrop ▾
    ⊖
    ⊕
    pause (ms) 15000 ▾
  ⊕
```

- Snap pause to the loop to wait 15 second

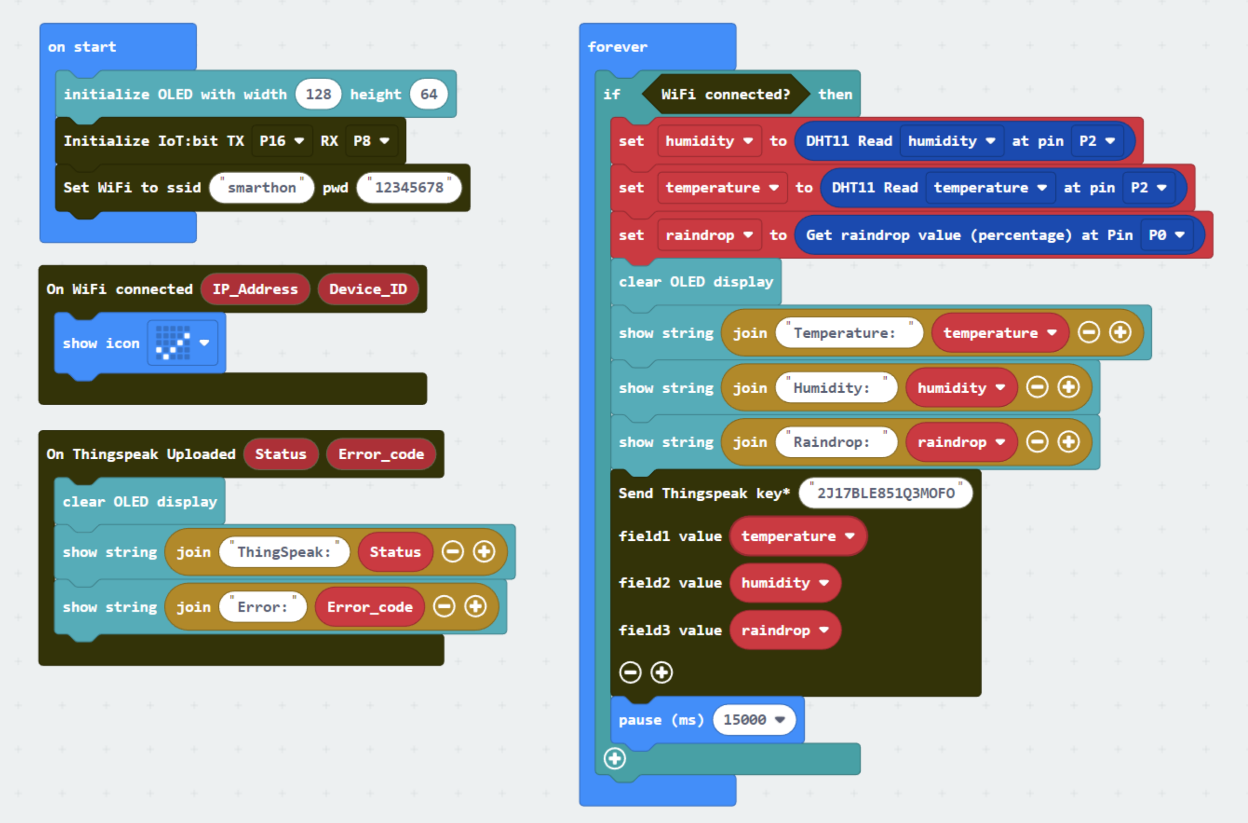
Step 6. Show ThingSpeak upload status

- Snap show string inside On Thingspeak Uploaded
- Draw the variable Status and Error_code to block show string



```
On Thingspeak Uploaded  Status  Error_code
    clear OLED display
    show string  join  "ThingSpeak:"  Status  ⊖ ⊕
    show string  join  "Error:"  Error_code  ⊖ ⊕
```

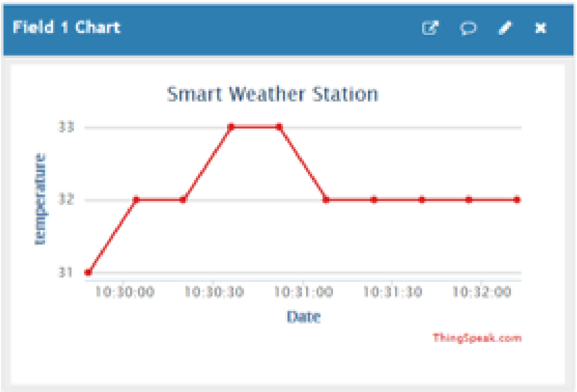
Full Solution MakeCode: https://makecode.microbit.org/_TMyYxL8Re1du

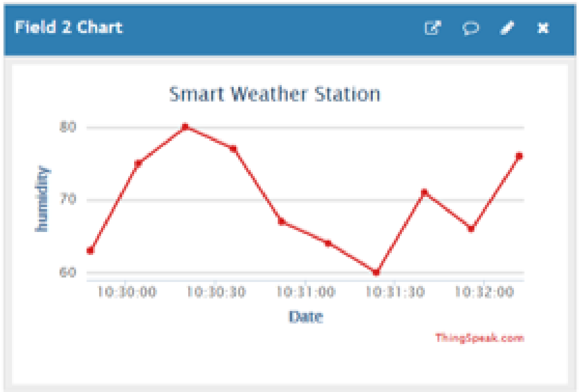


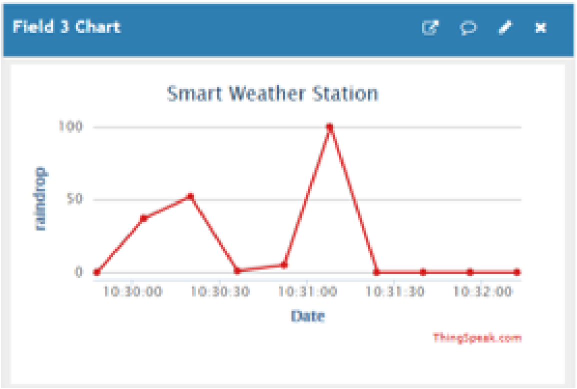
```
on start
    initialize OLED with width 128 height 64
    Initialize IoT:bit TX P16 ▾ RX P8 ▾
    Set WiFi to ssid "smarthon" pwd "12345678"

On WiFi connected  IP_Address  Device_ID
    show icon ▾

On Thingspeak Uploaded  Status  Error_code
    clear OLED display
    show string  join  "ThingSpeak:"  Status  ⊖ ⊕
    show string  join  "Error:"  Error_code  ⊖ ⊕

forever
    if  WiFi connected?  then
        set humidity ▾ to DHT11 Read humidity ▾ at pin P2 ▾
        set temperature ▾ to DHT11 Read temperature ▾ at pin P2 ▾
        set raindrop ▾ to Get raindrop value (percentage) at Pin P0 ▾
        clear OLED display
        show string  join  "Temperature:"  temperature ▾  ⊖ ⊕
        show string  join  "Humidity:"  humidity ▾  ⊖ ⊕
        show string  join  "Raindrop:"  raindrop ▾  ⊖ ⊕
        Send Thingspeak key* "2J17BLE851Q3MOFO"
        field1 value  temperature ▾
        field2 value  humidity ▾
        field3 value  raindrop ▾
        ⊖ ⊕
        pause (ms) 15000 ▾
    ⊕
```

Result

When micro:bit is connected to WiFi, it will check weather information (temperature, humidity from Temperature and humidity Sensor and raindrop value from raindrop sensor). Then, those data will be sent to ThingSpeak and pause for 15 seconds for another update. We can find three graphs on Thingspeak including temperature, humidity and raindrop (field1, field2, field3) respectively by collecting data using different sensors.





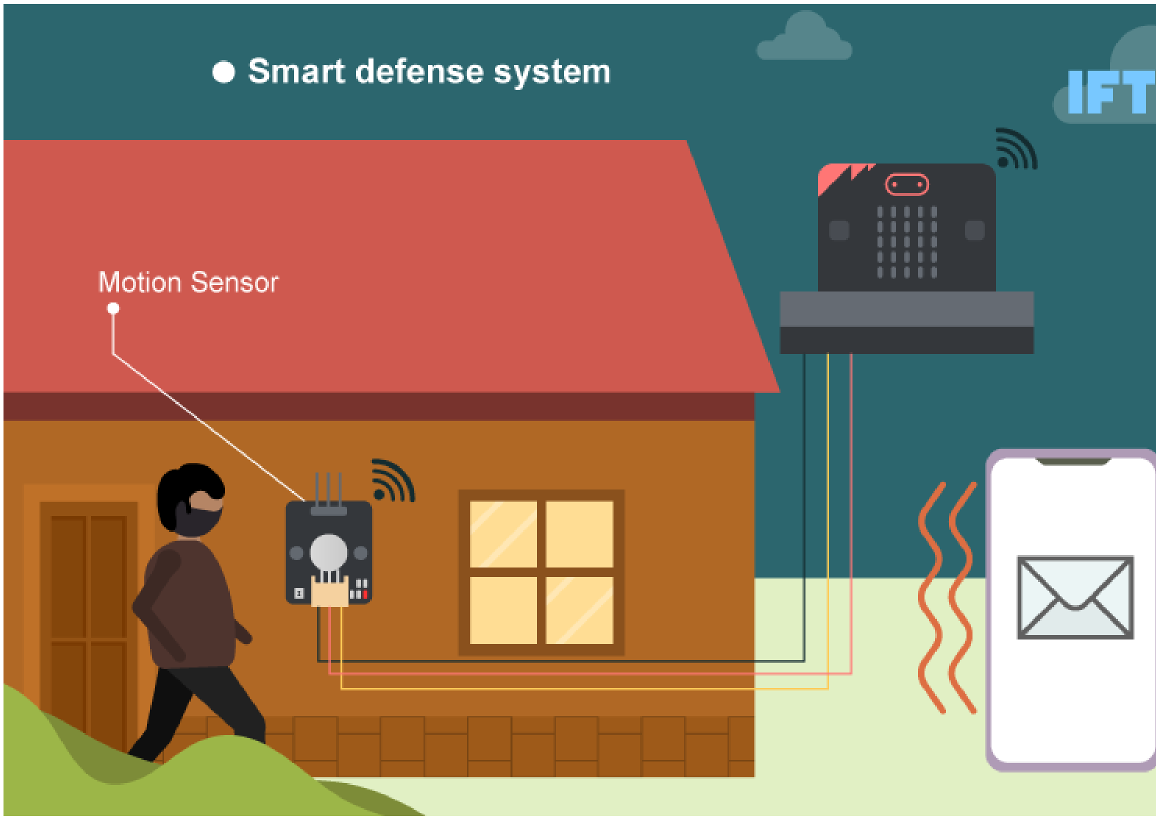


Think

Q1. How can we upload other module values (e.g. noise) to ThingSpeak?

1.1.16 IoT Case 07: Smart defense system

Level: ★ ★ ★ ☆ ☆

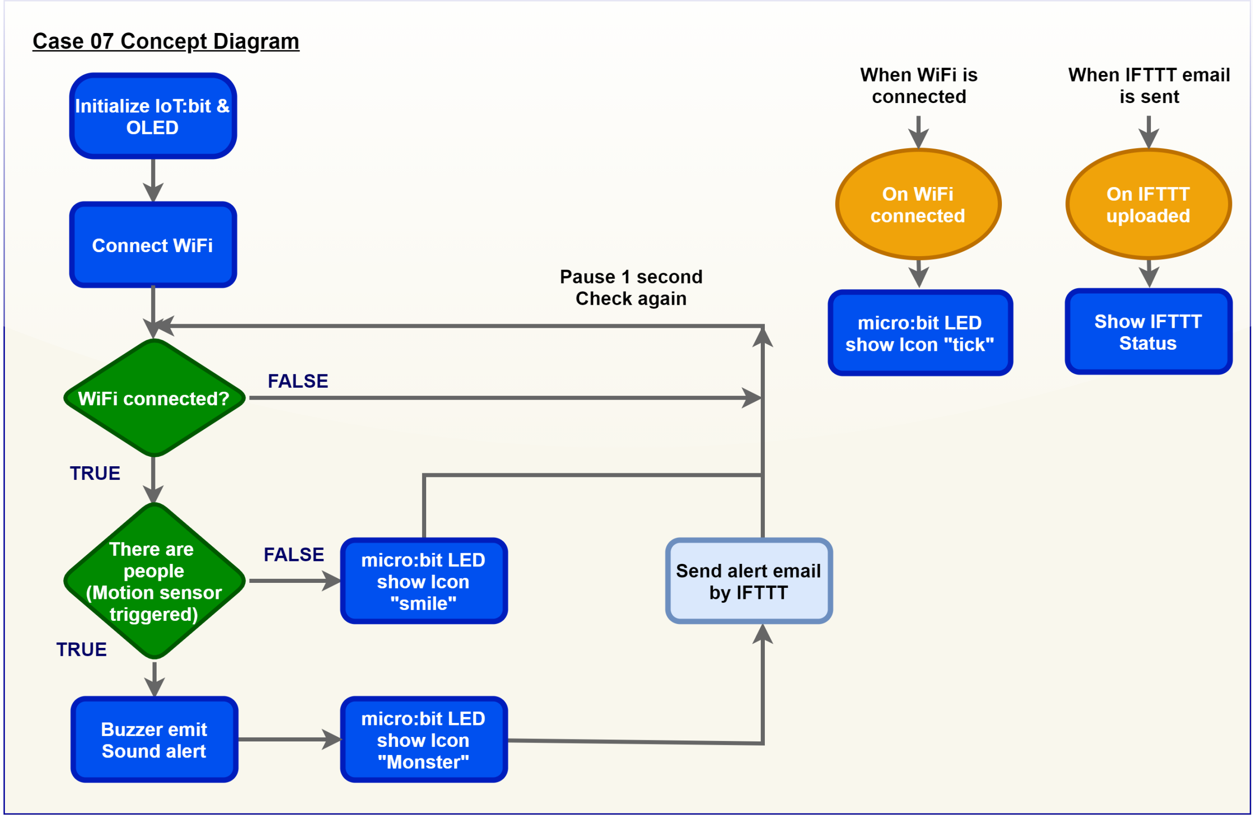


Goal

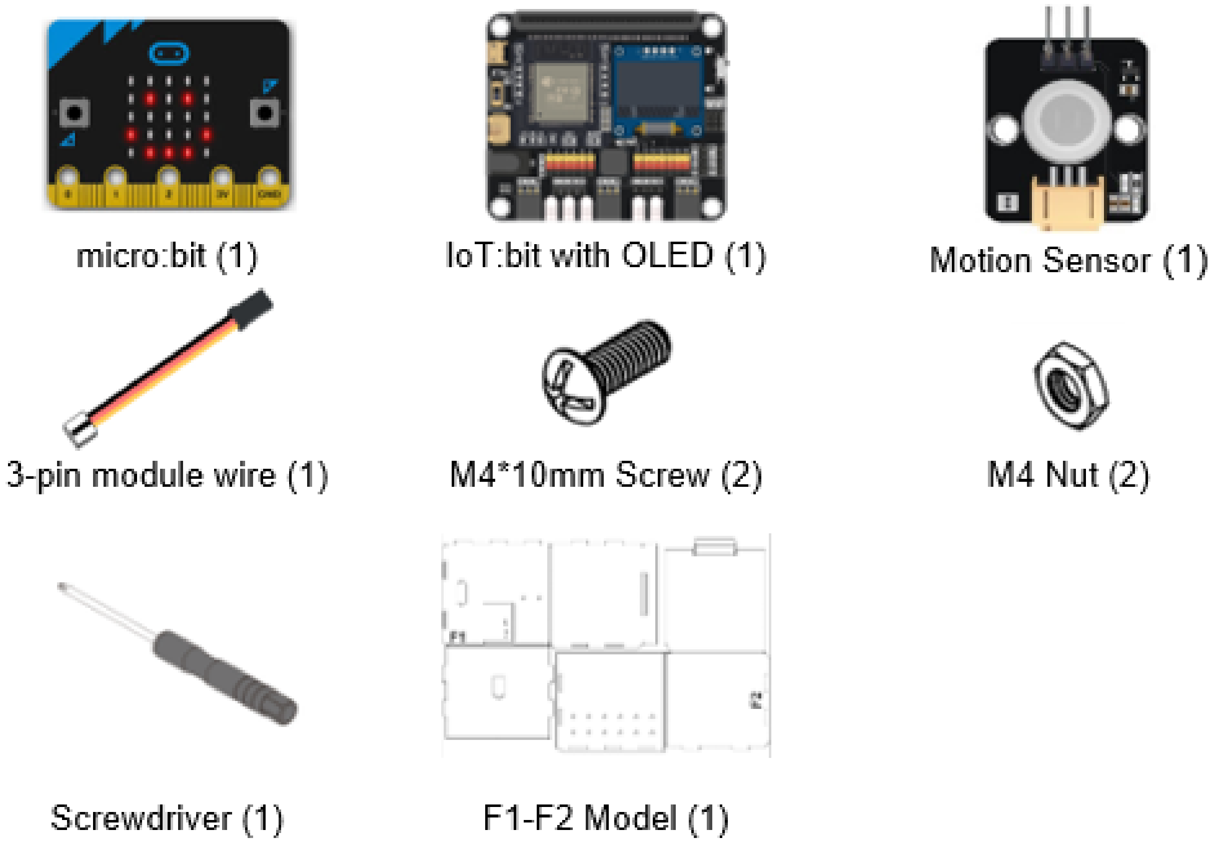
Make a smart defense system which emits sound and sends an email to the house owner if there are any suspicious movement near the door.

Background

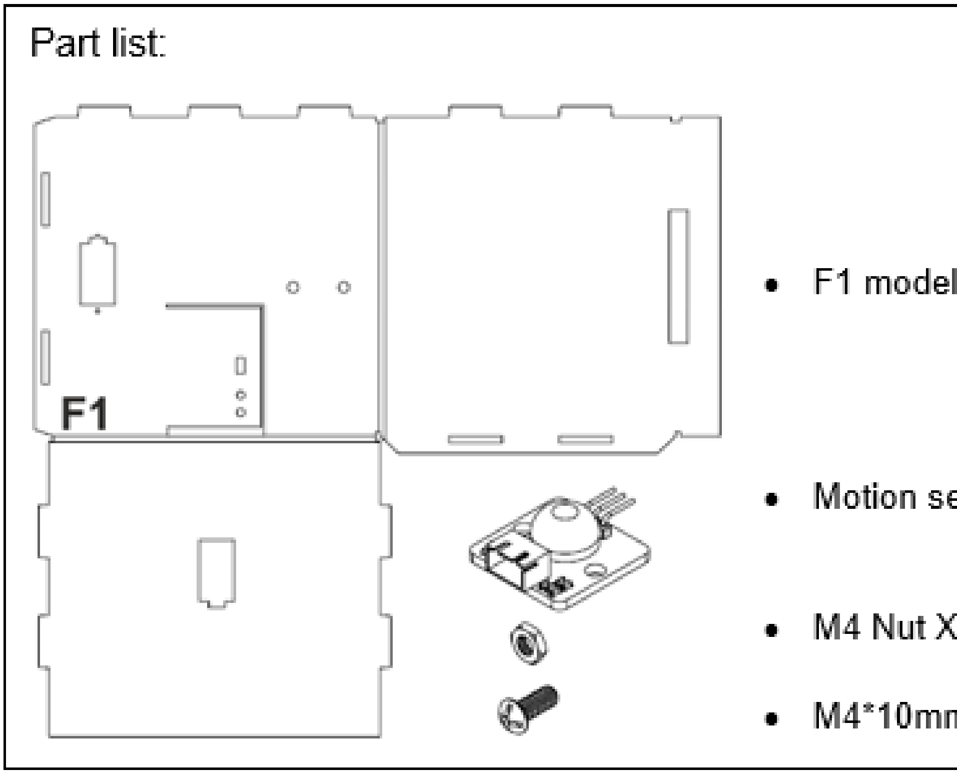
What is IFTTT? IFTTT can help connect all your different apps and devices which enables them to work together to do specific tasks. Smart defense system operation The motion sensor can deliver a motion signal to the micro:bit. When the micro:bit detects the signal, the buzzer will emit sound and send an email to specific email account through IFTTT. Also, a monstor icon will be shown on the micro:bit if there are suspicious people passes by.


Case 07 Concept Diagram
Initialize IoT:bit & OLED
Connect WiFi
WiFi connected?
FALSE
TRUE
Pause 1 second
Check again
There are people (Motion sensor triggered)
FALSE
TRUE
micro:bit LED show Icon "smile"
Send alert email by IFTTT
Buzzer emit Sound alert
micro:bit LED show Icon "Monster"
When WiFi is connected
On WiFi connected
micro:bit LED show Icon "tick"
When IFTTT email is sent
On IFTTT uploaded
Show IFTTT Status

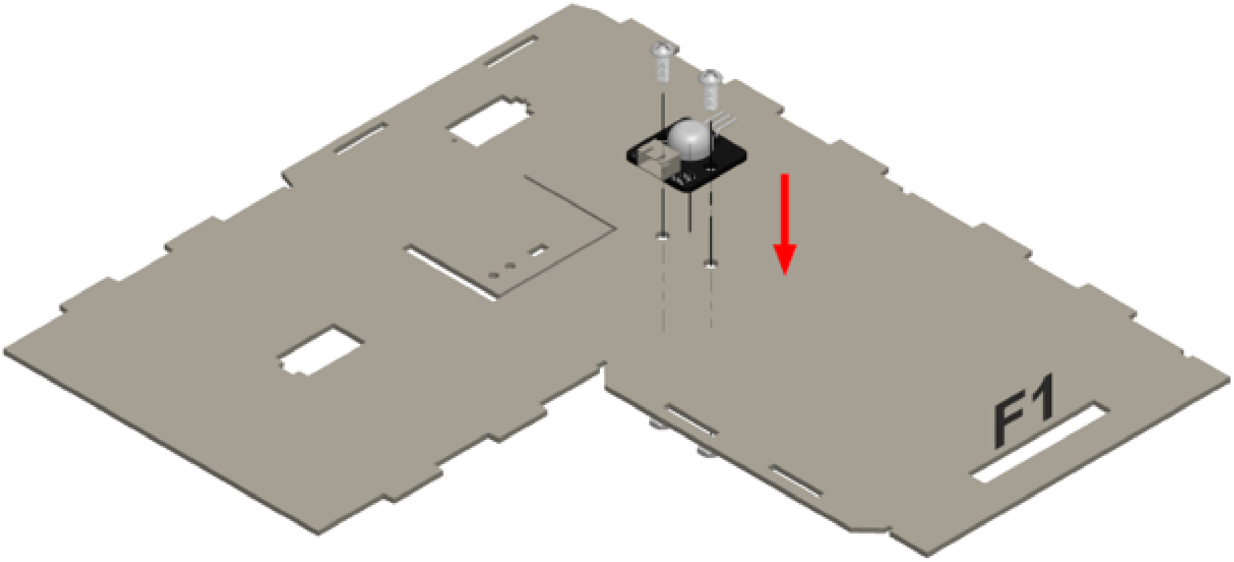
Part List

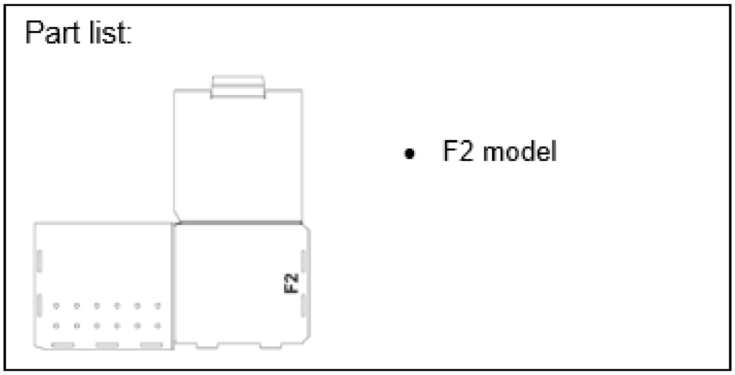

micro:bit (1)
IoT:bit with OLED (1)
Motion Sensor (1)
3-pin module wire (1)
M4*10mm Screw (2)
M4 Nut (2)
Screwdriver (1)
F1-F2 Model (1)

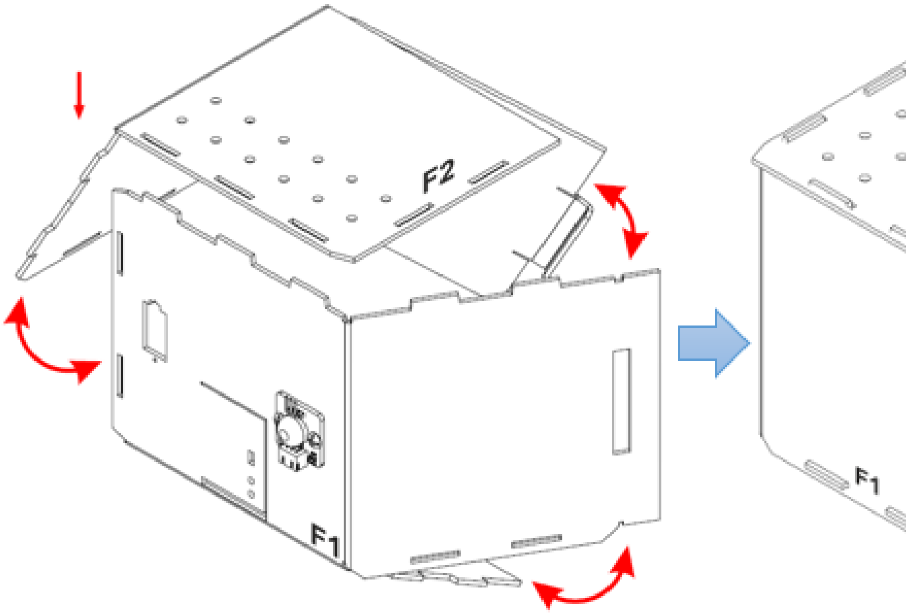
Assembly step



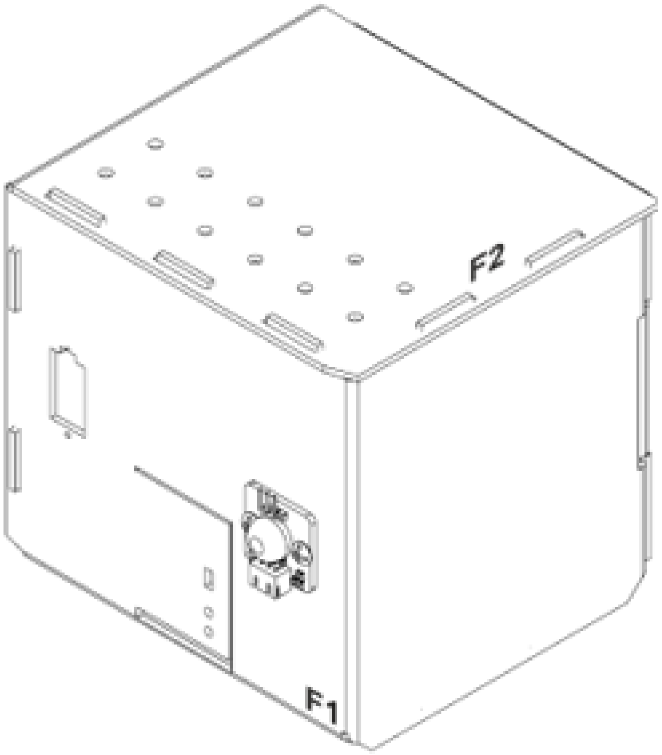
Step 1 Attach the motion sensor to F1 model.







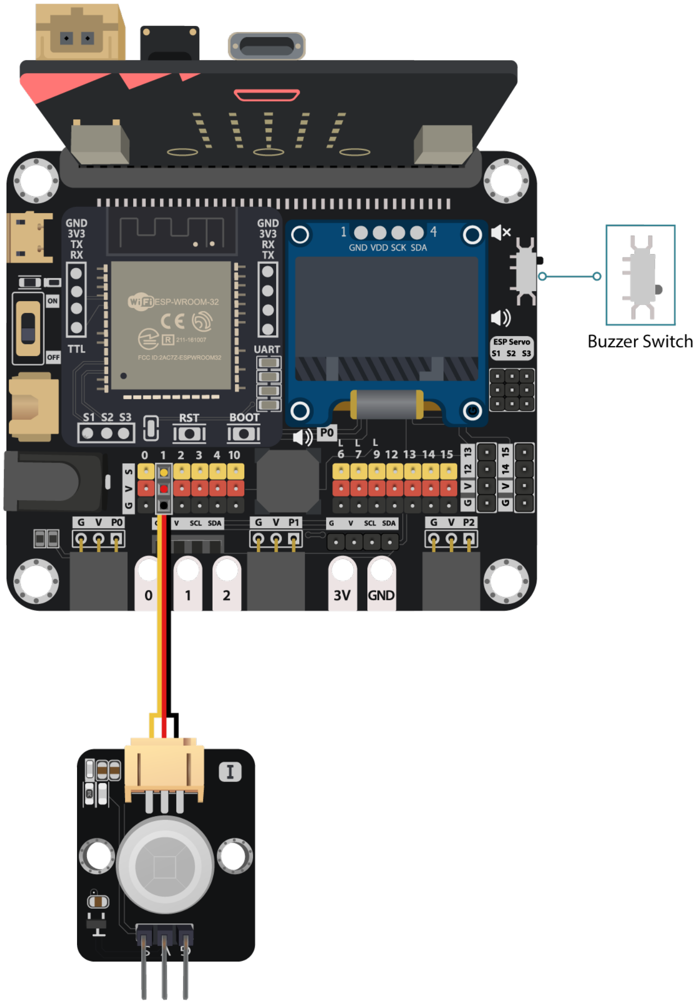
Step 2 Put together all the cardboard parts (F1-F2)



Step 3 Assembly completed!

Hardware connect

Connect the Motion Sensor to P1 port of IoT:bit Turn on the Buzzer Switch on P0 port of IoT:bit



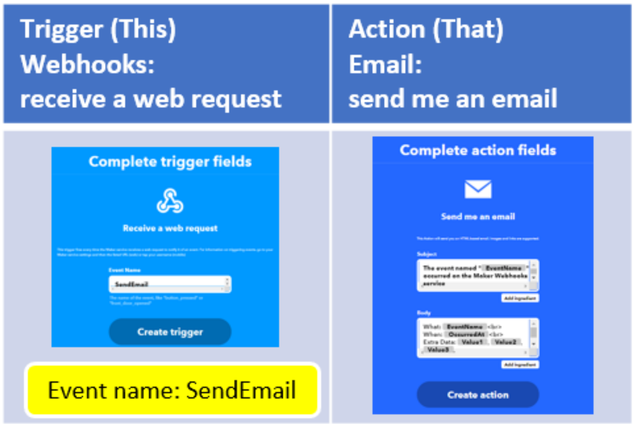
Pull the buzzer switch 'down' to connect the buzzer in this execrise

IoT (IFTTT)

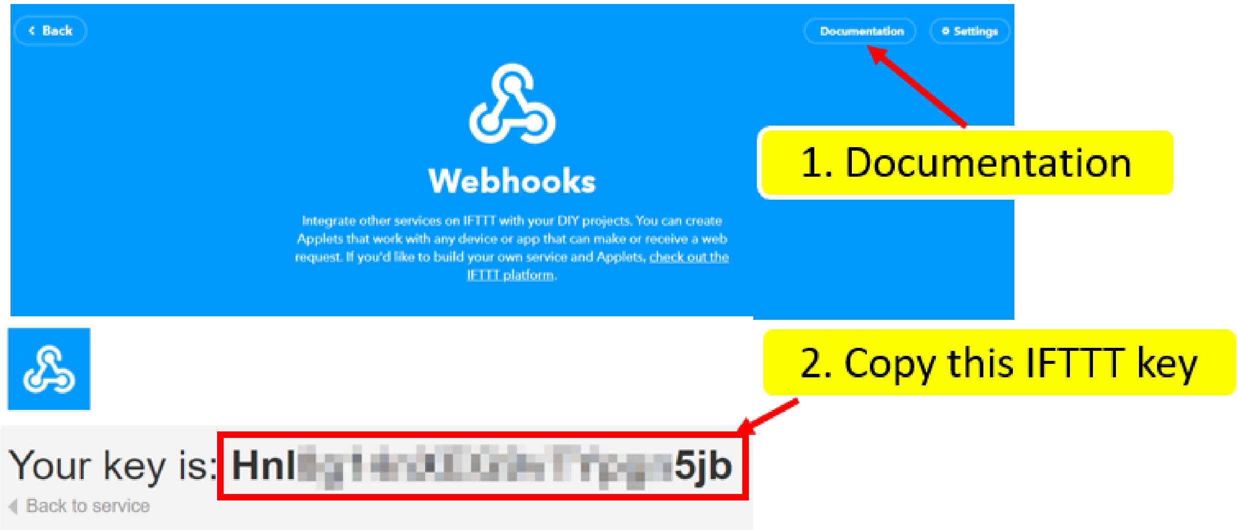
* For more details, please refer to Chapter 2: Send Email by IFTTT

Step 1 Go to https://ifttt.com , create applet (if webhooks then Email)

This: webhooks
If This
Add
That: Email
Then That



Step 2 Go to "My services" > "Webhooks", select "Documentation" . Copy your Webhooks Key as follows:



Webhooks

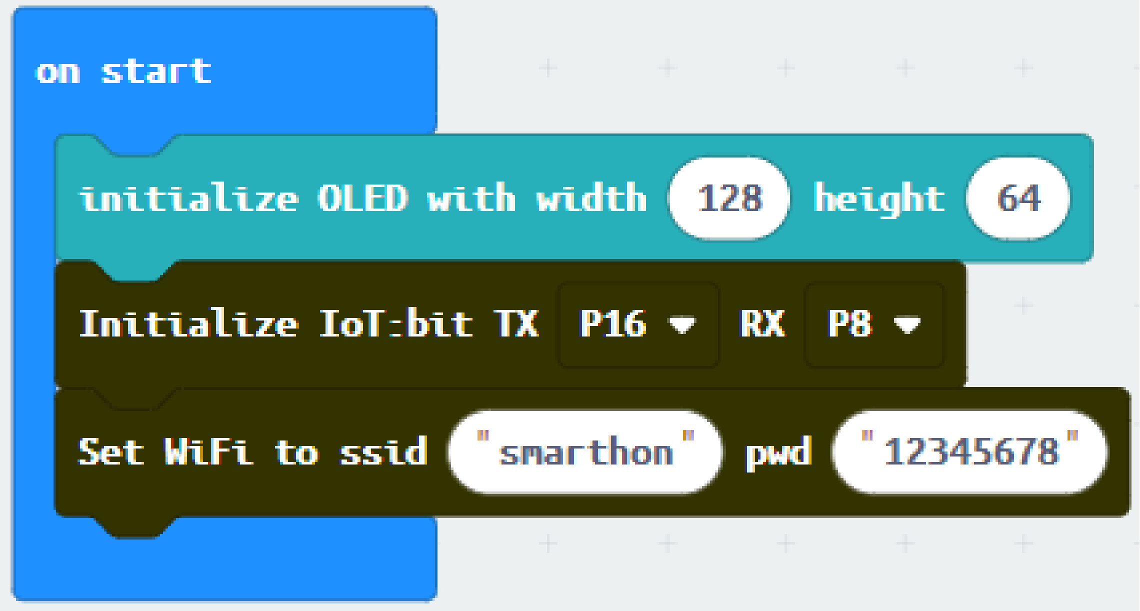
Integrate other services on IFTTT with your DIY projects. You can create Applets that work with any device or app that can make or receive a web request. If you'd like to build your own service and Applets, check out the IFTTT platform.

Your key is: Hnl[redacted]5jb

Programming (MakeCode)

Step 1. Initialize OLED, IoT:bit and connect to WiFi

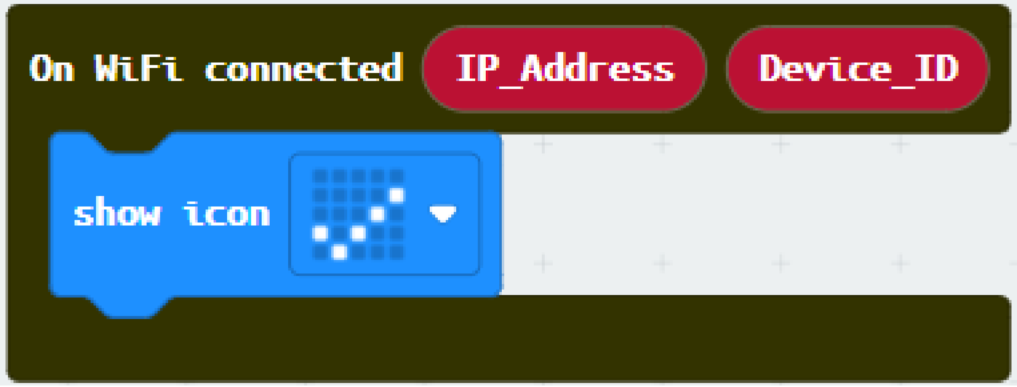
- Snap `Initialize OLED with width:128, height: 64` to `on start`
- Snap `Initialize IoT:bit TX P16 RX P8` from `IoT:bit` to `on start`
- Snap `Set Wi-Fi to ssid pwd` from `IoT:bit`
- Enter your Wi-Fi name and password. Here we set `smarthon` as `SSID` and `12345678` as `password`



```
on start
  initialize OLED with width 128 height 64
  Initialize IoT:bit TX P16 RX P8
  Set WiFi to ssid "smarthon" pwd "12345678"
```

Step 2: Show icon "tick" after WiFi connection

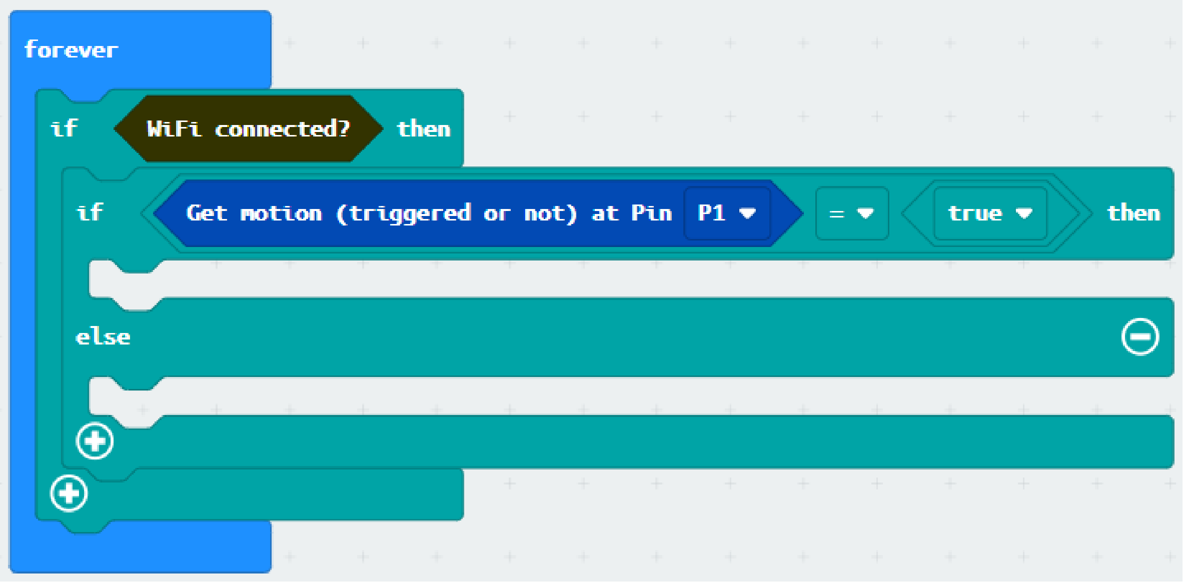
- Snap `show icon` from `basic` to `On WiFi connected` and select icon `tick`



```
On WiFi connected IP_Address Device_ID
  show icon ✓
```

Step 3. Check motion sensor value

- Snap `if statement` to block `forever`
- If `WiFi is connected`,
- If `get motion (triggered or not) at Pin P1` = `true` then, that's say someone is near the door.

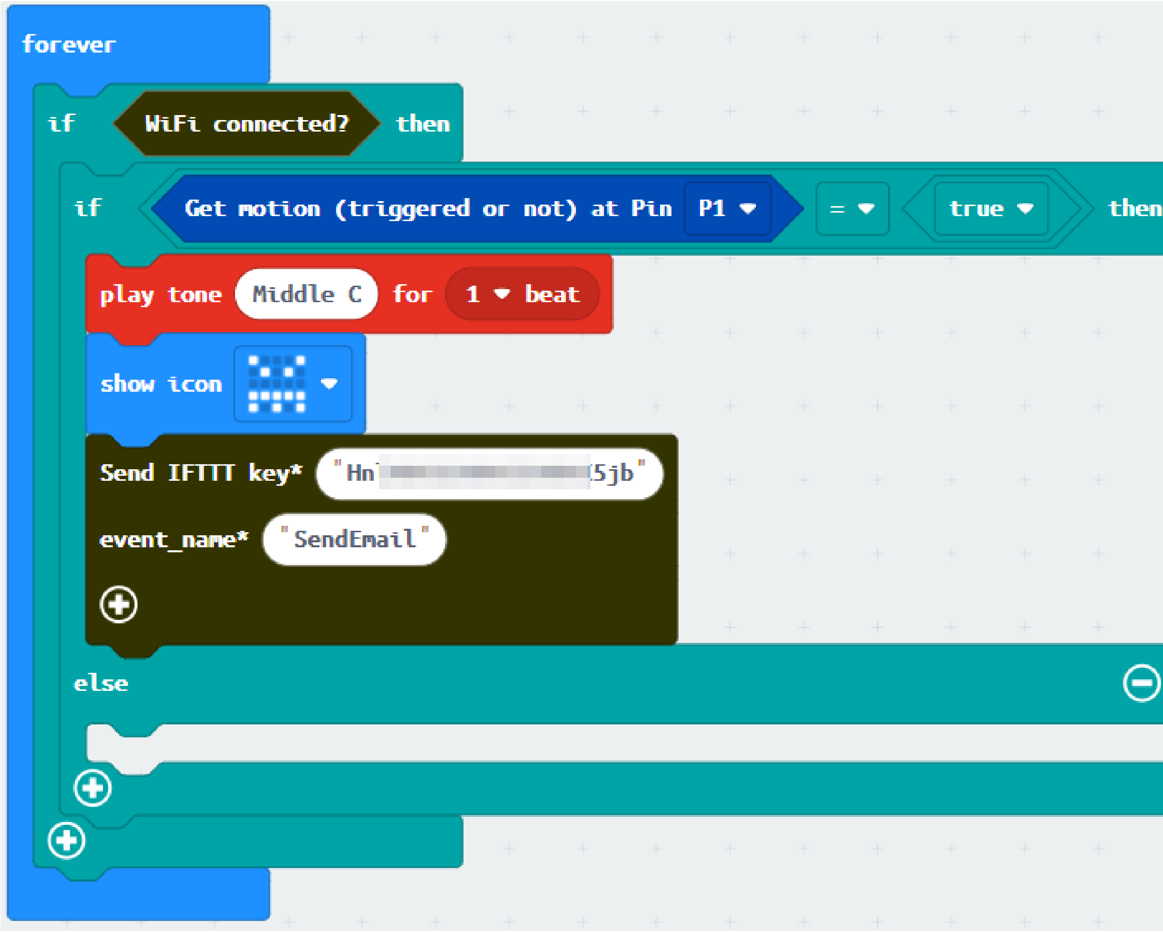


```
forever
  if WiFi connected? then
    if Get motion (triggered or not) at Pin P1 = true then
    else
```

Step 4. Send notification when someone pass by

- Snap `if statement` to block `forever`
- If `WiFi is connected`, then snap `play tone Middle C for 1 beat` from `music`
- Snap `show icon` from `basic` and select a `monster icon`
- Snap `Send IFTTT key...` from `IoT:bit` > `IoT Services`, input your `IFTTT key` and input event

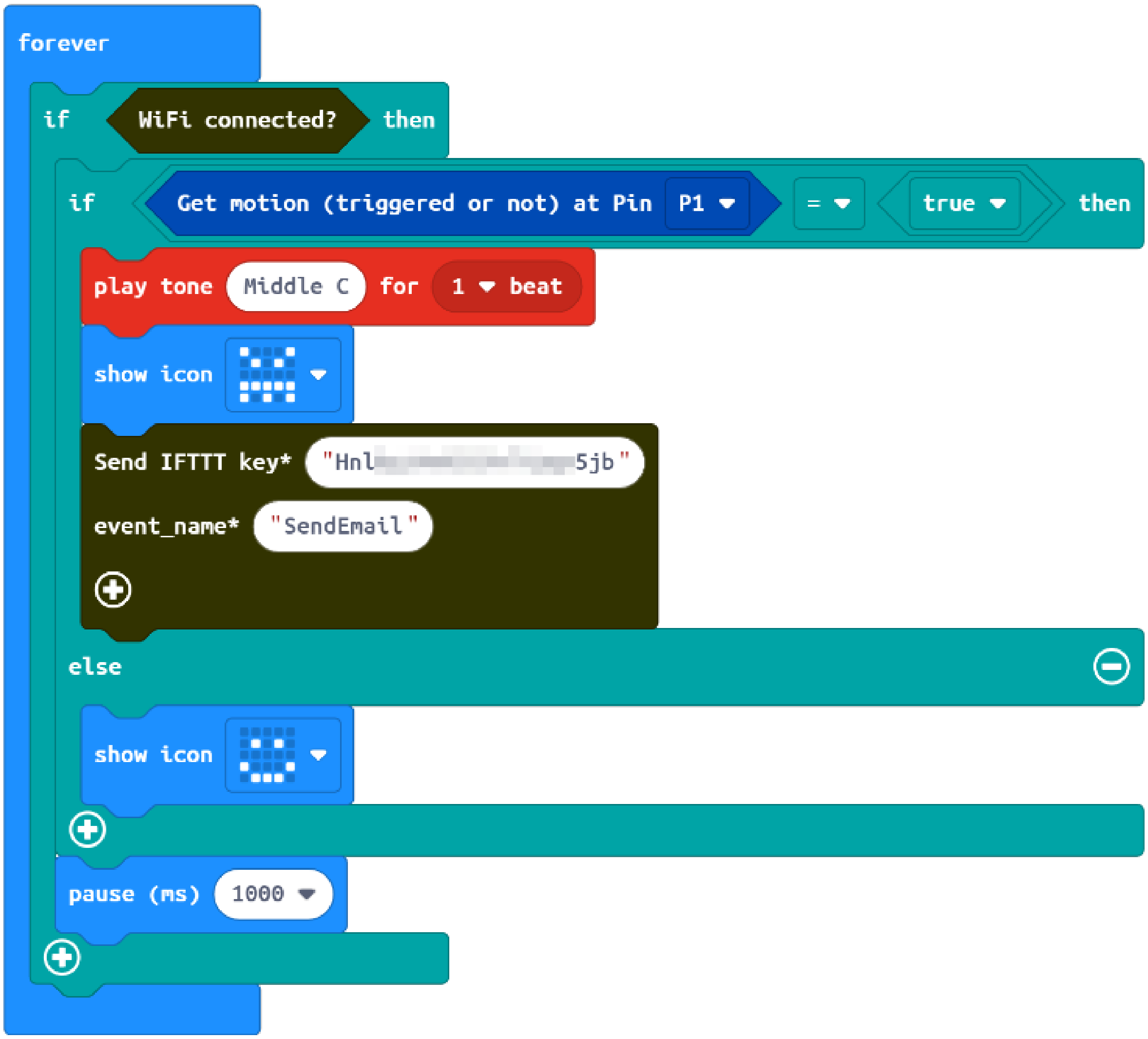
name SendEmail



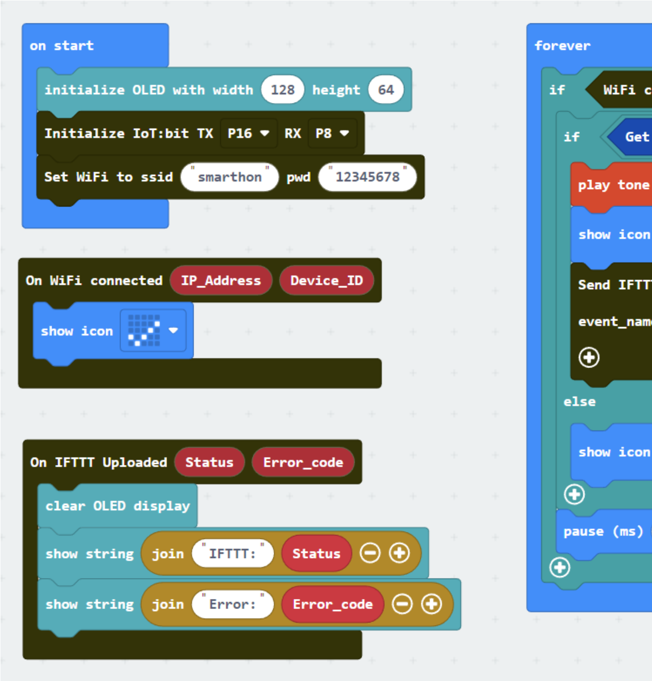
```
forever
  if WiFi connected? then
    if Get motion (triggered or not) at Pin P1 = true then
      play tone Middle C for 1 beat
      show icon
      Send IFTTT key* "Hn[redacted]5jb"
      event_name* "SendEmail"
    else

```

Step 5. Show smile icon when no one pass by

- Snap `show icon` into `Else`, and select icon `smile`
- Snap `Pause` after `if statement` to the loop for 1 second delay for checking



```
forever
  if WiFi connected? then
    if Get motion (triggered or not) at Pin P1 = true then
      play tone Middle C for 1 beat
      show icon
      Send IFTTT key* "Hnl[redacted]5jb"
      event_name* "SendEmail"
    else
      show icon
  pause (ms) 1000
```

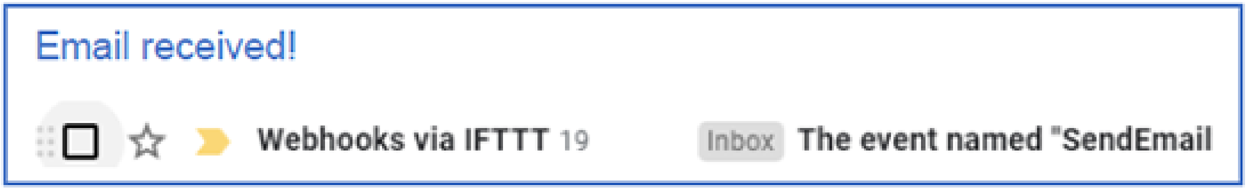


```
on start
  initialize OLED with width 128 height 64
  Initialize IoT:bit TX P16 ▾ RX P8 ▾
  Set WiFi to ssid "smarthon" pwd "12345678"

On WiFi connected IP_Address Device_ID
  show icon

On IFTTT Uploaded Status Error_code
  clear OLED display
  show string join "IFTTT:" Status ⊖ ⊕
  show string join "Error:" Error_code ⊖ ⊕

forever
  if WiFi c
    if Get
      play tone
      show icon
      Send IFTT
      event_nam
      ⊕
    else
      show icon
    ⊕
    pause (ms)
  ⊕
```

Full Solution MakeCode: https://makecode.microbit.org/_H4uU8R3716xj

Result

When WiFi is connected, if there are any suspicious movement near the door, the buzzer will emit sound and an email will be sent to the house owner. Also, a monster icon will be shown on the micro:bit.



Email received!

□ ☆ Webhooks via IFTTT 19 Inbox The event named "SendEmail

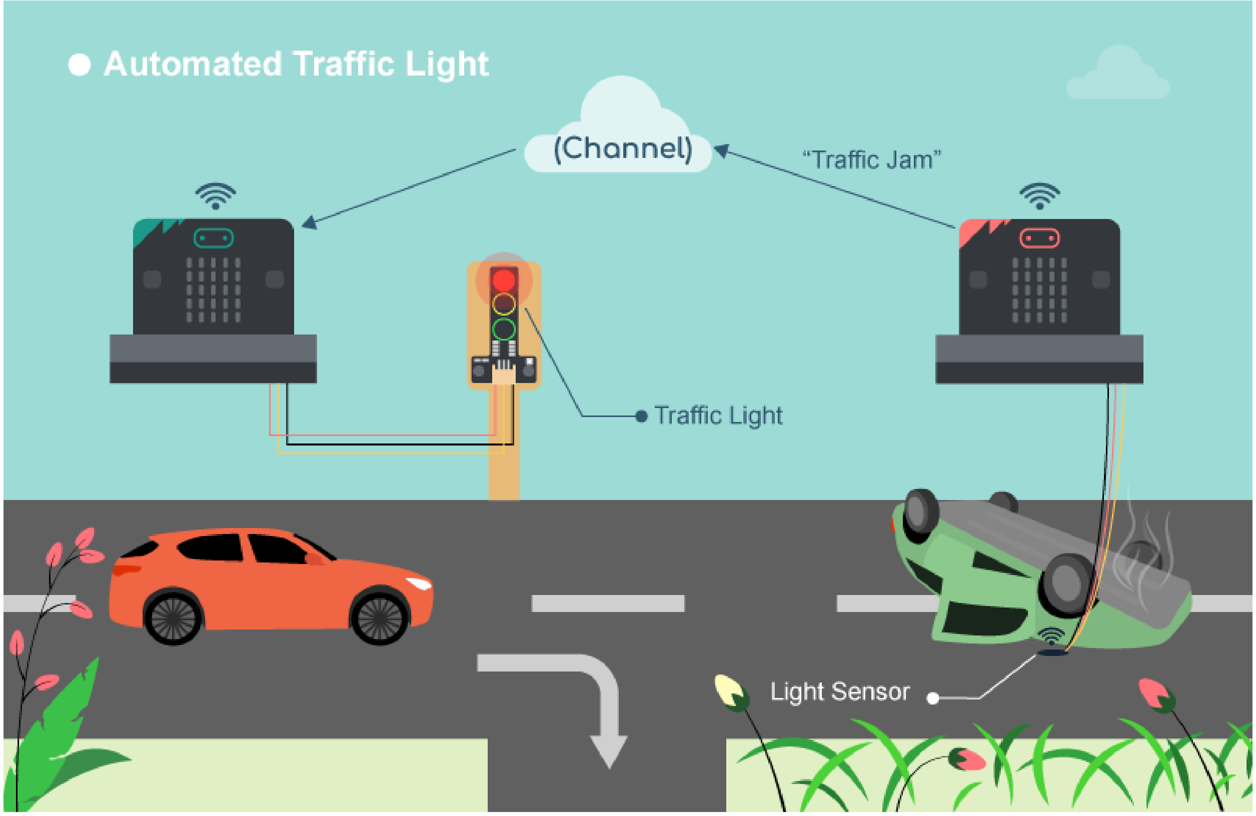
Think

Q1. How to avoid sending duplicate emails? (tips: using variable) Q2. How to distinguish suspicious people (e.g. password authentication)?

1.1.17 IoT Case 08: Automated Traffic Light

Level: ★ ★ ★ ★ ★

* For more details, please refer to “Chapter 5: Object to Object communication”



Sender

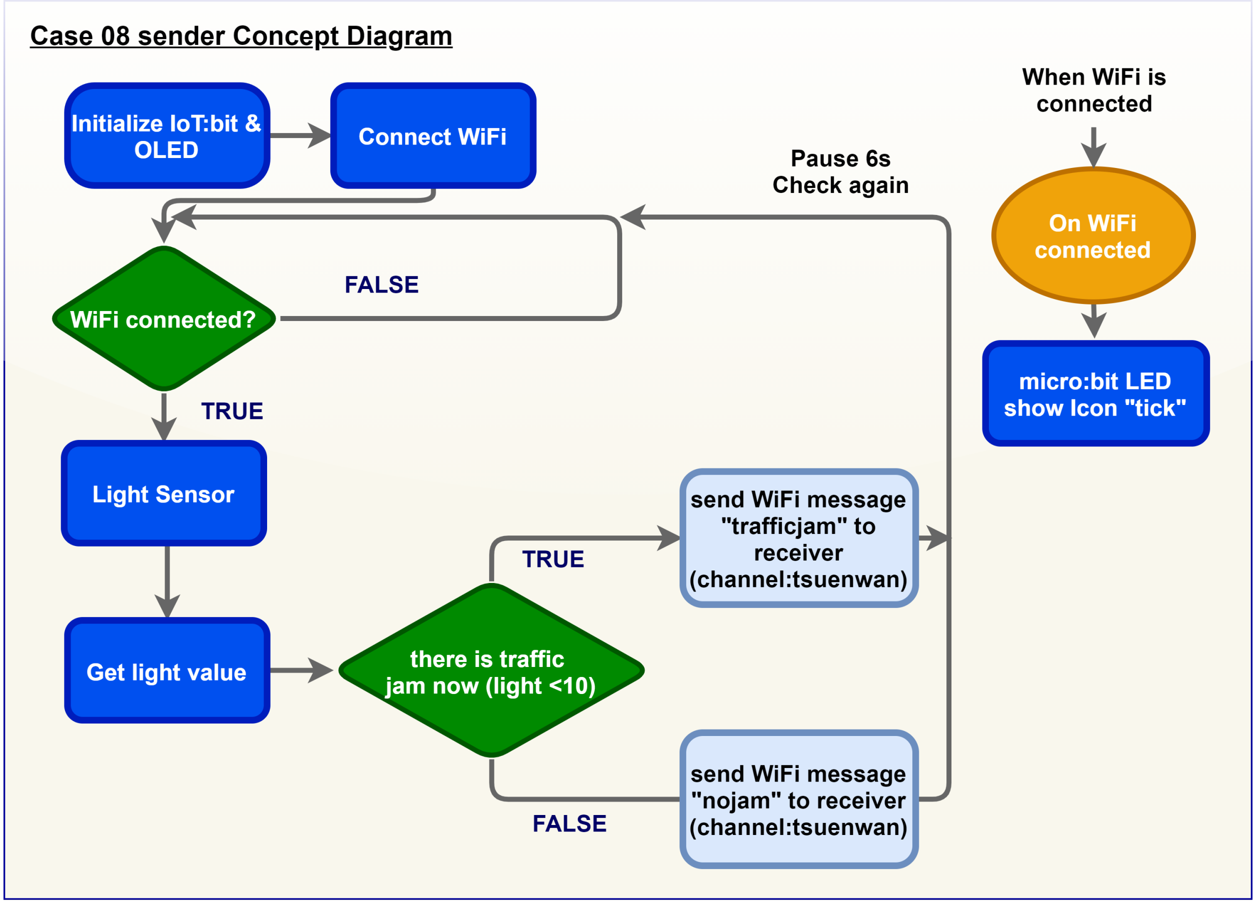
Goal

Make a sender to send signal to another micro:bit to show if there is traffic jam or not.

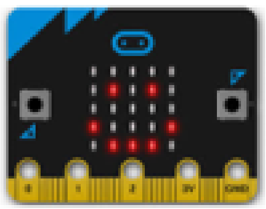
Background

How to send signal to another micro:bit? Micro:bit (sender and receiver) are connected to the same channel so wifi message can be sent between the micro:bit. Such that when the sender micro:bit sends a Wi-Fi message "trafficjam" to receiver micro:bit. When receiver micro:bit receives a WiFi message "trafficjam", the connected traffic light will turn red.

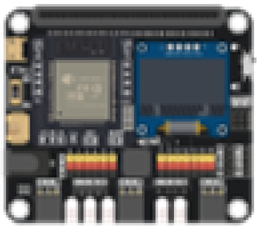
Sender micro:bit operation When the light value detected is too low, this would represent there is a traffic jam and and a 'trafficjam' wifi message to another microbit. When the light value detected is high, this would represent there is no traffic jam and send a "nojam" wifi message to another micro:bit.


Case 08 sender Concept Diagram
Initialize IoT:bit & OLED
Connect WiFi
WiFi connected?
FALSE
TRUE
Light Sensor
Get light value
there is traffic jam now (light <10)
TRUE
FALSE
send WiFi message "trafficjam" to receiver (channel:tsuenwan)
send WiFi message "nojam" to receiver (channel:tsuenwan)
Pause 6s Check again
When WiFi is connected
On WiFi connected
micro:bit LED show Icon "tick"

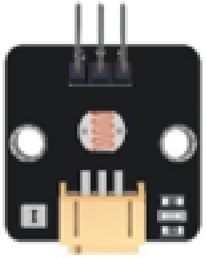
Part List



micro:bit (1)



IoT:bit with OLED (1)



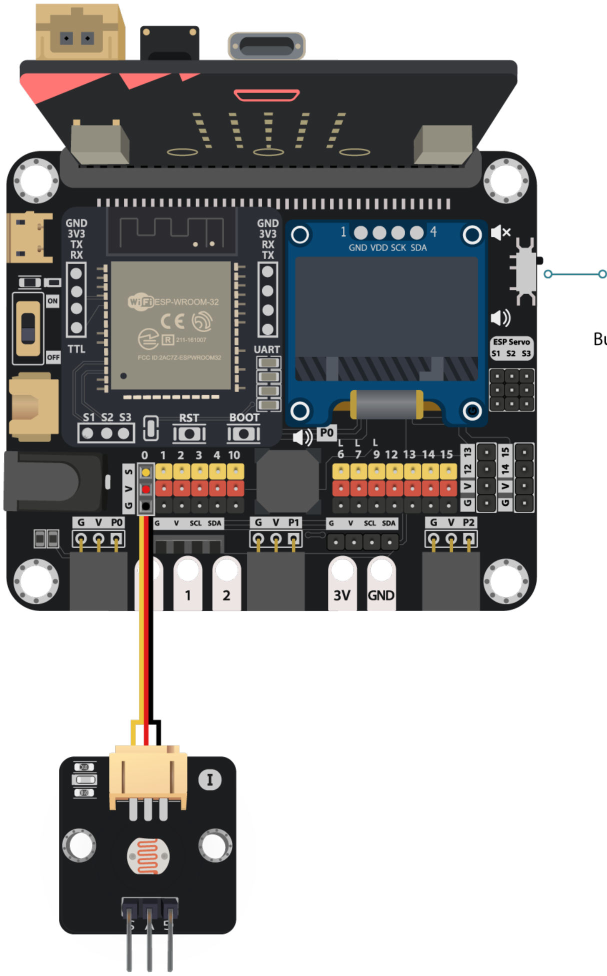
Light Sensor (1)

3-pin module wire (1)

Assembly step

N/A

Hardware connect



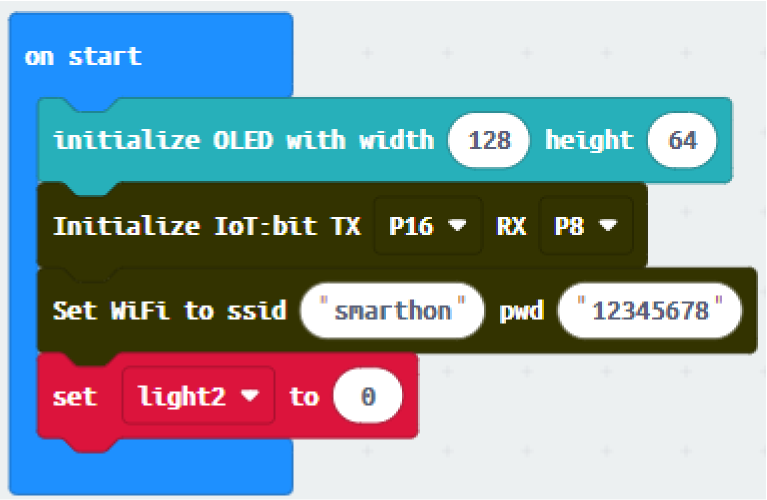
Connect the Light Sensor to P0 port of IoT:bit

Pull the buzzer switch 'up' to disconnect the buzzer in this execrise

Programming (MakeCode)

Step 1. Initialize OLED and IoT:bit and connect to WiFi

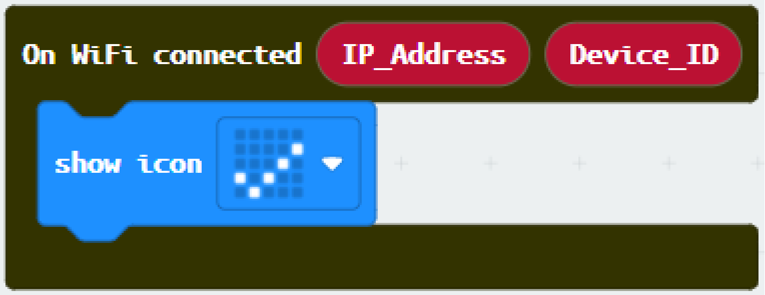
- Snap `Initialize OLED with width:128, height: 64` to `on start`
- Snap `Initialize IoT:bit TX P16 RX P8` from `IoT:bit` to `on start`
- Snap `Set Wi-Fi to ssid pwd` from `IoT:bit`
- Enter your Wi-Fi name and password. Here we set `smarthon` as `SSID` and `12345678` as `password`
- Set variable `light2` to 0 from `variables`



```
on start
    initialize OLED with width 128 height 64
    Initialize IoT:bit TX P16 ▾ RX P8 ▾
    Set WiFi to ssid "smarthon" pwd "12345678"
    set light2 ▾ to 0
```

Step 2. Show icon "tick" after WiFi connection

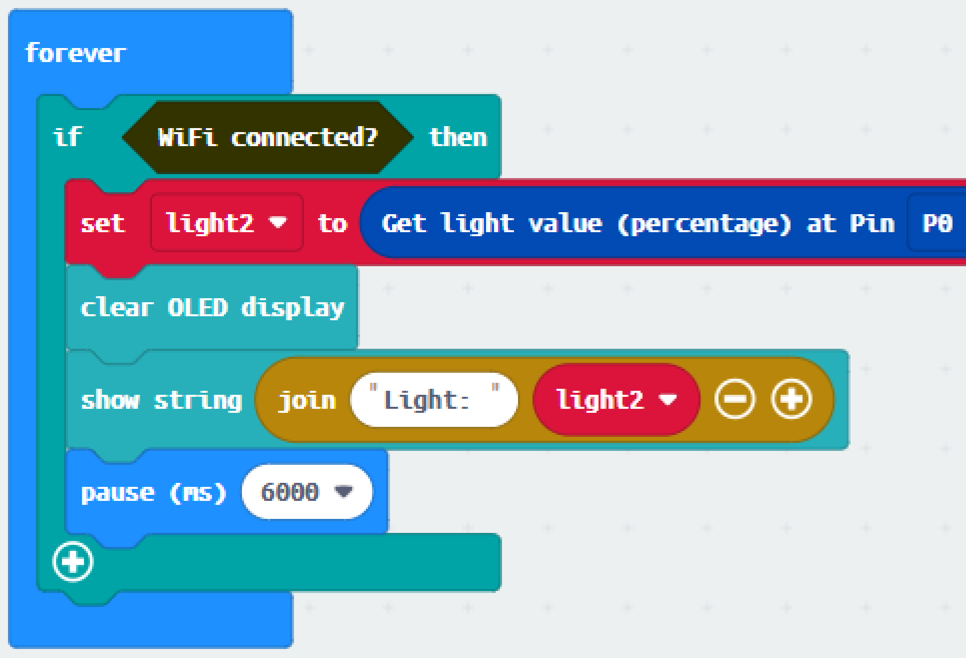
- Snap `show icon` from `basic` to `On WiFi connected` and select icon `tick`



```
On WiFi connected IP_Address Device_ID
    show icon ▾
```

Step 3. Check traffic status

- Snap `if statement` to block `forever`
- If `WiFi is connected`,
- Set `variable2` to `light value from light sensor at P0`
- Snap `clear OLED display` from `OLED` to avoid overlap
- Snap `show string` and show value of variables `light2`

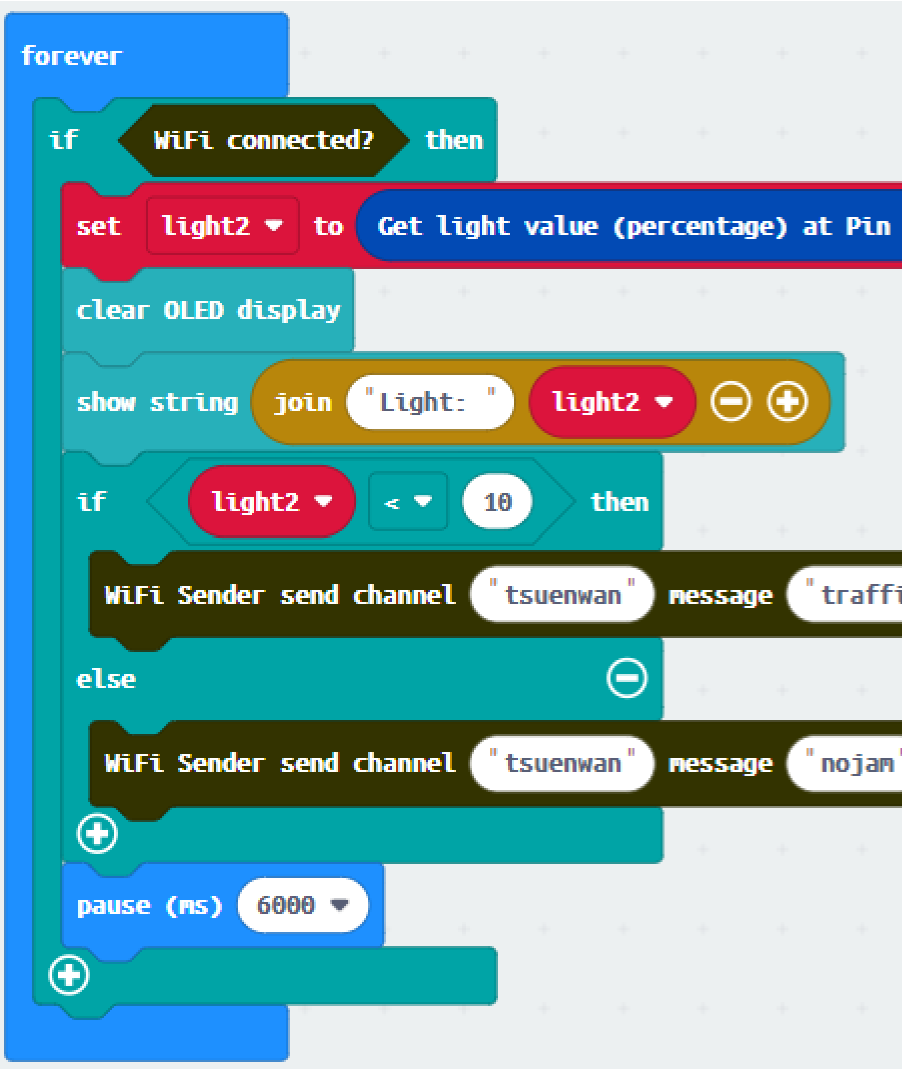


```
forever
  if WiFi connected? then
    set light2 to Get light value (percentage) at Pin P0
    clear OLED display
    show string join "Light: " light2
    pause (ms) 6000
```

- `Pause` 6000 (ms) to another checking

Step 4. Send notification when someone pass by

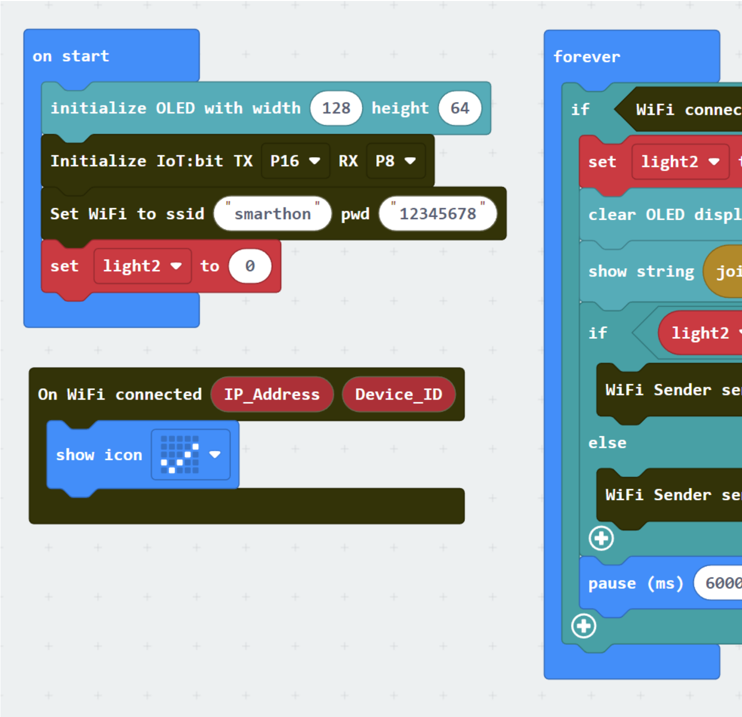
- Snap `if statement`
- Set `light` < `10` in to if-condition (traffic jam is detected)
- Snap `WiFi Sender send channel... message...` from `IoT:bit` > `channel`
- Set channel to `tsuenwan`, message `trafficjam`
- Set `else-condition` (traffic jam is not detected)
- Snap `WiFi Sender send channel... message...` from `IoT:bit` > `channel`



```
forever
  if WiFi connected? then
    set light2 to Get light value (percentage) at Pin
    clear OLED display
    show string join "Light: " light2
    if light2 < 10 then
      WiFi Sender send channel "tsuenwan" message "traffi
    else
      WiFi Sender send channel "tsuenwan" message "nojam"
    pause (ms) 6000
```

- Set channel to `tsuenwan`, message `nojam`

Full Solution MakeCode: https://makecode.microbit.org/_P5fAujhip5Lh

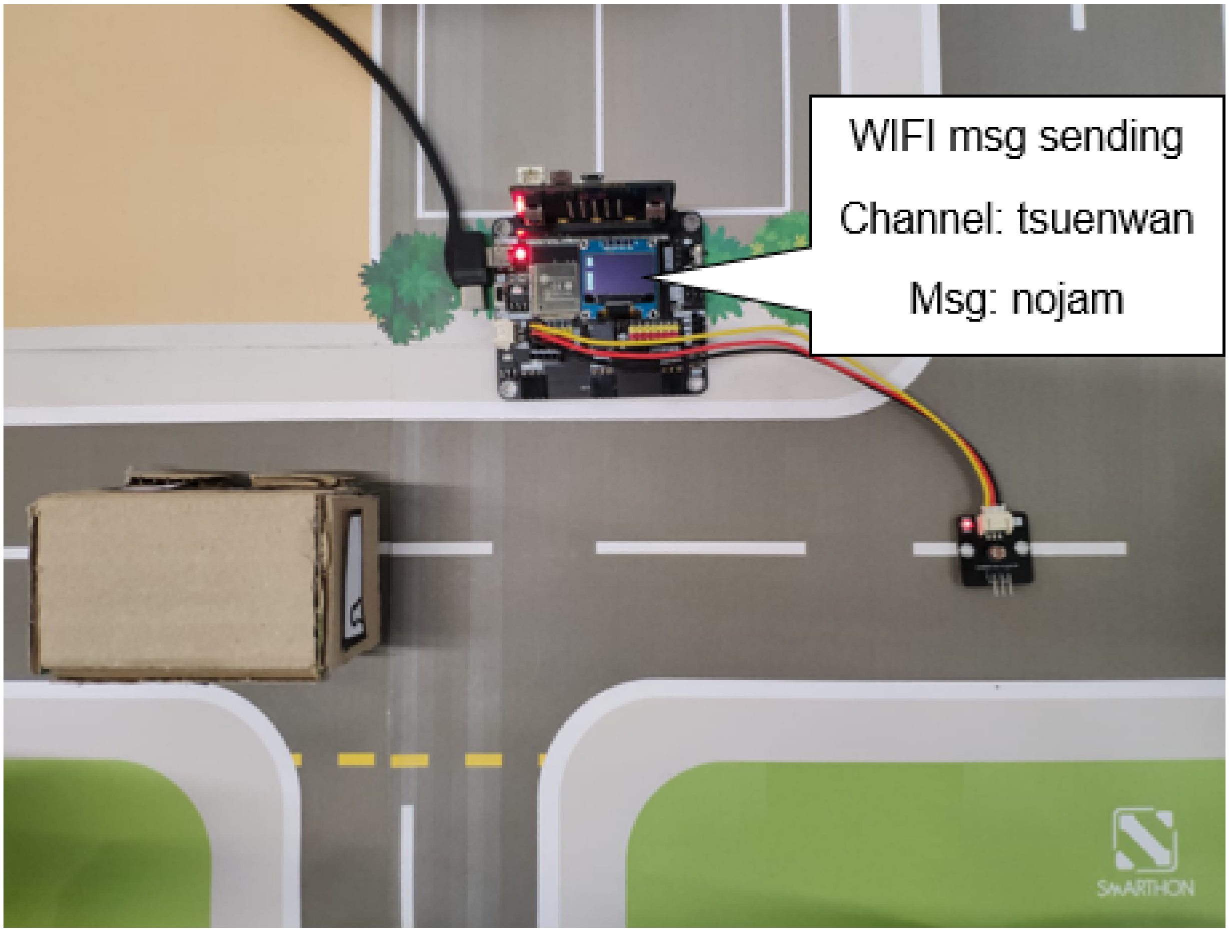


```
on start
    initialize OLED with width 128 height 64
    Initialize IoT:bit TX P16 RX P8
    Set WiFi to ssid "smarthon" pwd "12345678"
    set light2 to 0

On WiFi connected IP_Address Device_ID
    show icon

forever
    if WiFi connec
        set light2 t
        clear OLED displ
        show string joi
        if light2
            WiFi Sender se
        else
            WiFi Sender se
        pause (ms) 6000
```

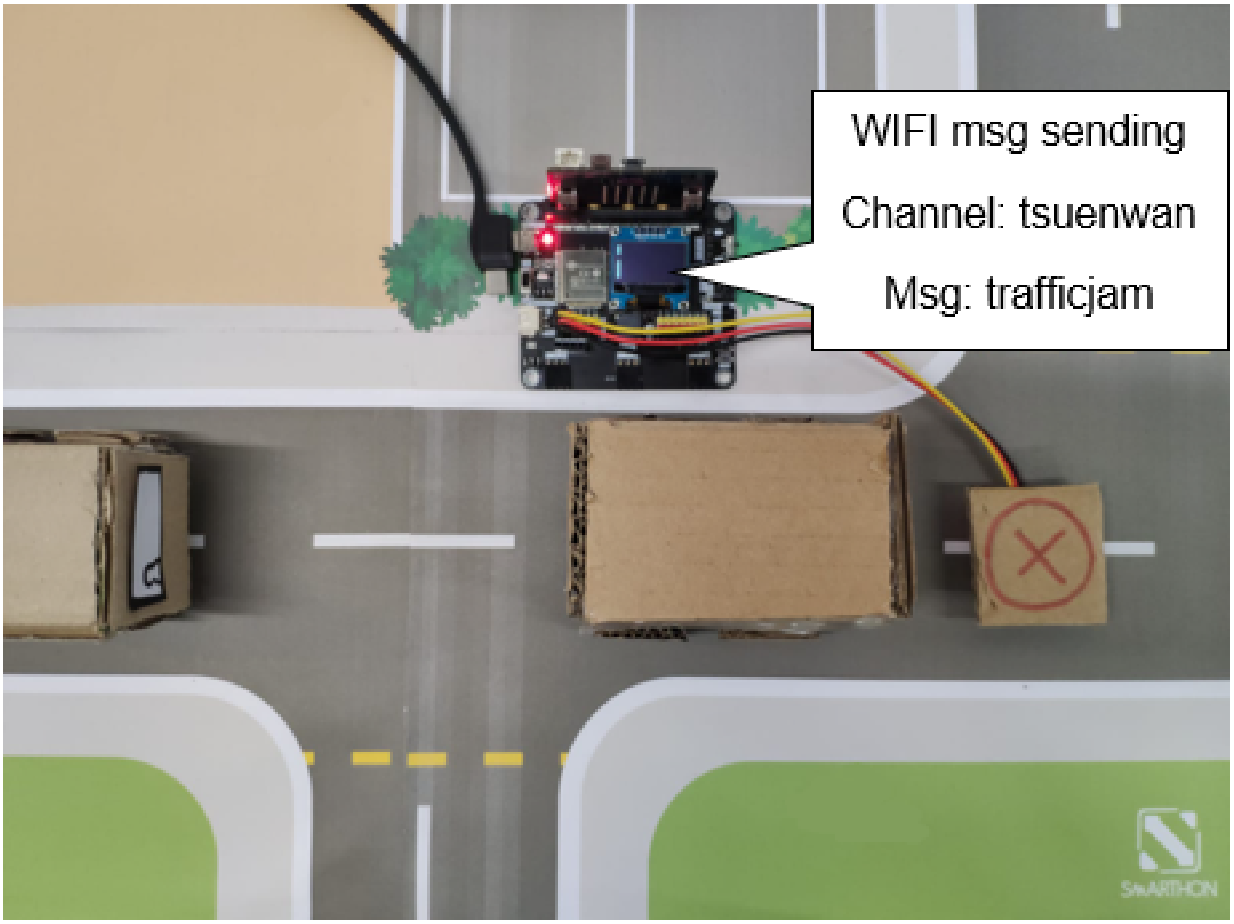
You could also download the program from the following website:

Result

Light sensor is used to detect if there are traffic jam. Once the light intensity is not low, it indicates that there is a traffic jam on the road. Wi-Fi message "nojam" will be sent to another micro:bit (receiver).



Once the light intensity is too low, it indicates that there is a traffic jam on the road. Wi-Fi message "trafficjam" will be sent to another micro:bit (receiver).



Think

Q1. How can we use distance sensor to detect traffic status?

Receiver

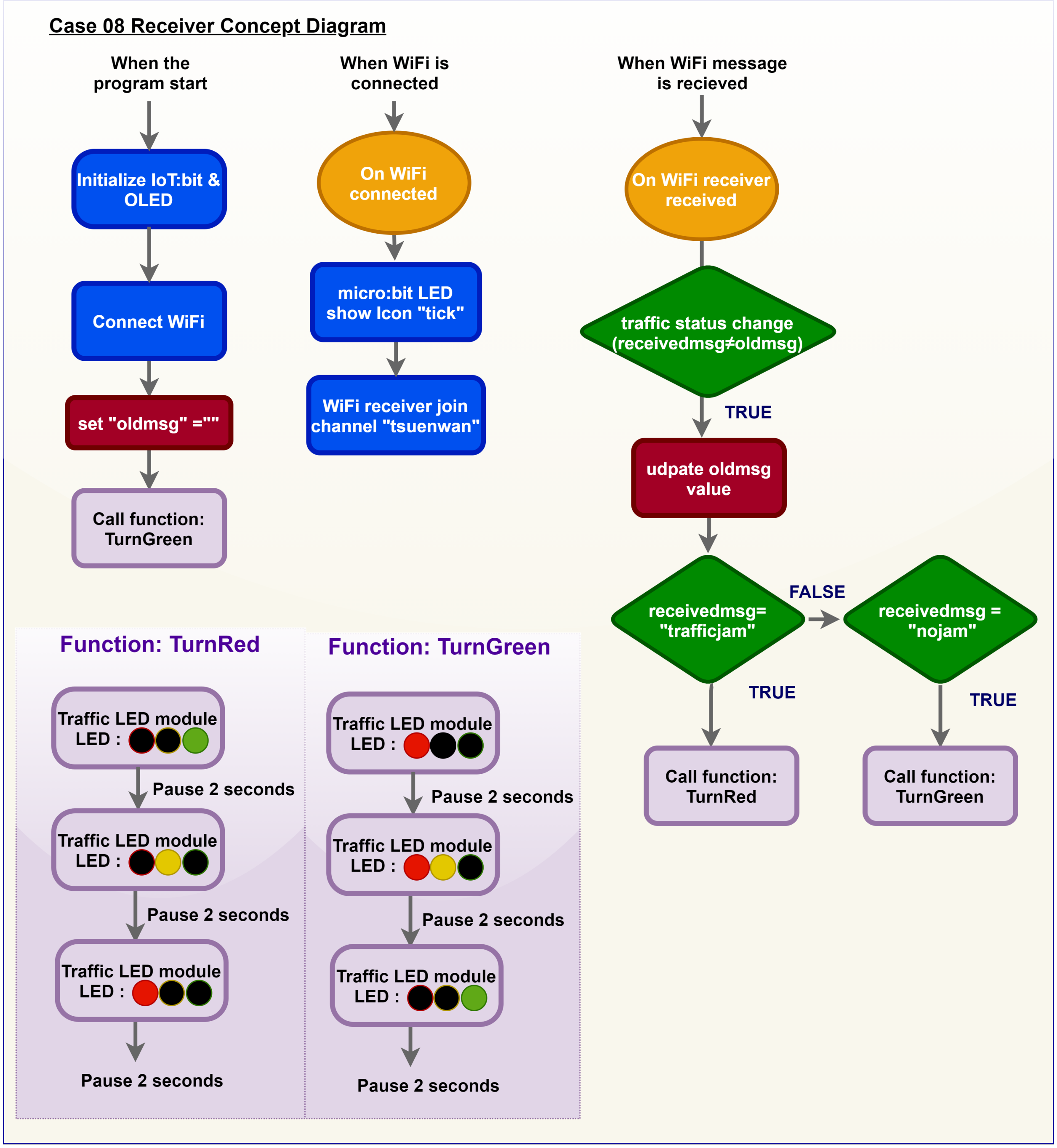
Goal

Make an auto-traffic LED Module by receiving signal from sender micro:bit.

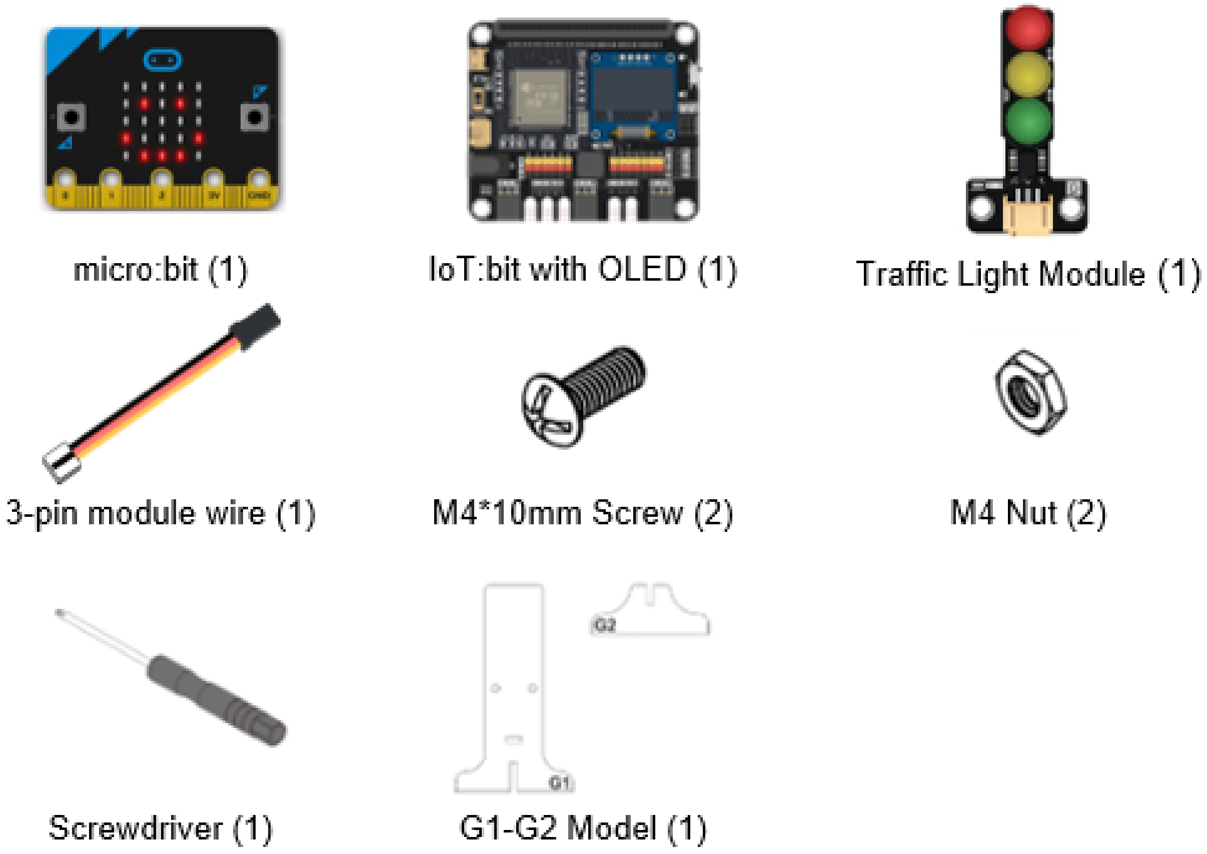
Background

How to receive signal from another micro:bit? Micro:bit (sender and receiver) are connected to the same channel so wifi message can be sent between the micro:bit. If receiver micro:bit receives a WiFi message “trafficjam” from sender, the connected traffic light will turn red.

Receiver micro:bit operation When a wifi message “trafficjam” is received, it means there is traffic jam forward. The traffic LED Module will turn red. When a wifi message “nojam” is received, it means there is no traffic jam forward. The traffic LED Module will turn green. By using smart traffic light, the problem of traffic jam can be reduced as automatic traffic control is used.

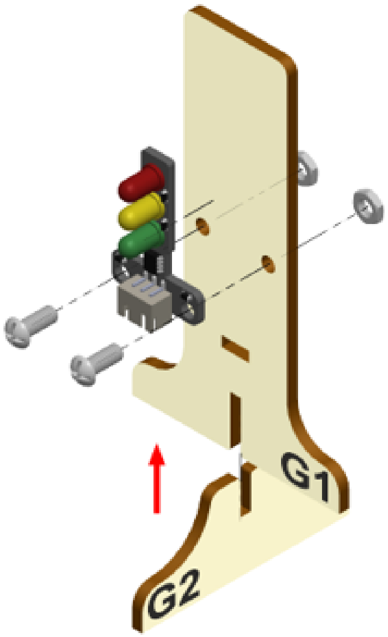

Case 08 Receiver Concept Diagram
When the program start
Initialize IoT:bit & OLED
Connect WiFi
set "oldmsg" =""
Call function: TurnGreen
When WiFi is connected
On WiFi connected
micro:bit LED show Icon "tick"
WiFi receiver join channel "tsuenwan"
When WiFi message is recieved
On WiFi receiver received
traffic status change (receivedmsg≠oldmsg)
TRUE
udpate oldmsg value
receivedmsg= "trafficjam"
FALSE
receivedmsg = "nojam"
TRUE
TRUE
Call function: TurnRed
Call function: TurnGreen
Function: TurnRed
Traffic LED module LED :
Pause 2 seconds
Traffic LED module LED :
Pause 2 seconds
Traffic LED module LED :
Pause 2 seconds
Function: TurnGreen
Traffic LED module LED :
Pause 2 seconds
Traffic LED module LED :
Pause 2 seconds
Traffic LED module LED :
Pause 2 seconds

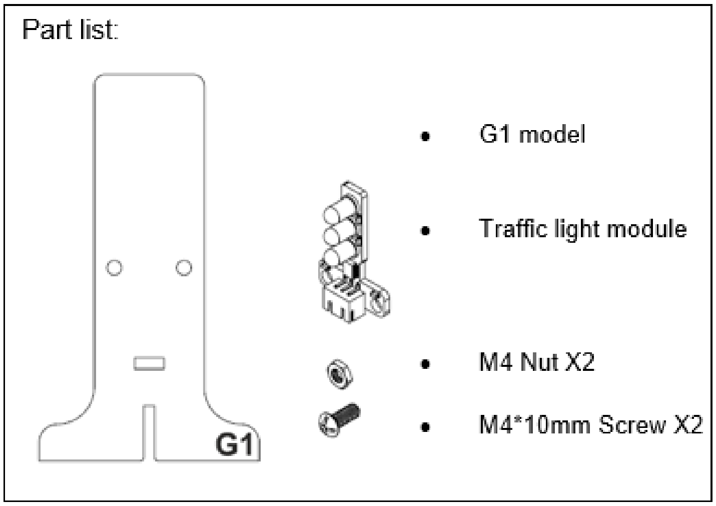
Part List

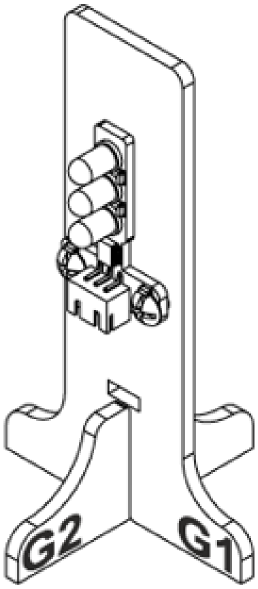

micro:bit (1)
IoT:bit with OLED (1)
Traffic Light Module (1)
3-pin module wire (1)
M4*10mm Screw (2)
M4 Nut (2)
Screwdriver (1)
G2
G1
G1-G2 Model (1)

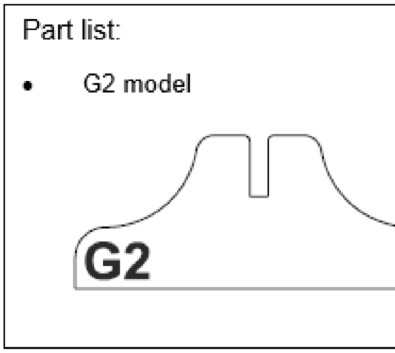
Assembly step

Step 1 Attach the Traffic light Module to G1 model using M4 screw and nut. Put together all the cardboard parts (G1-G2)



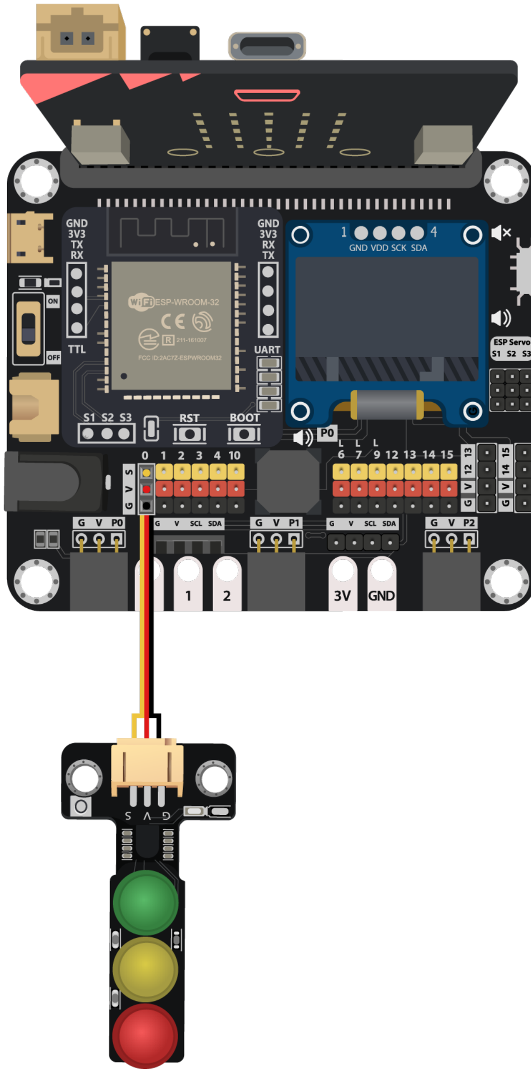






Step 2 Assembly completed!

Hardware connect

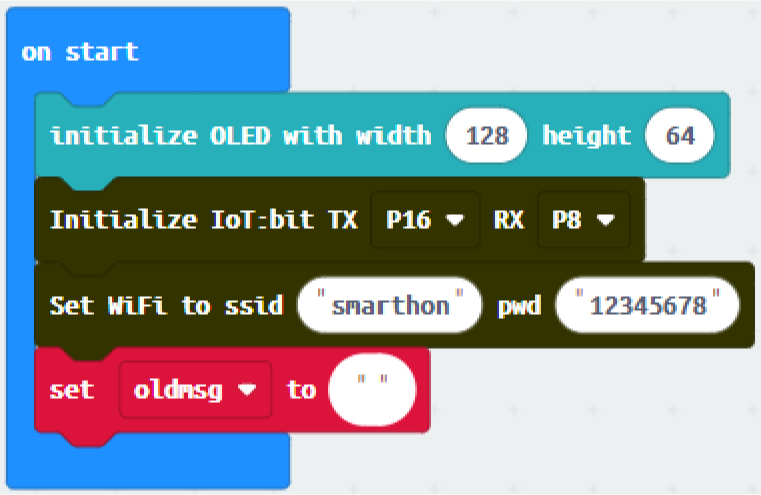


Connect the Traffic LED Module to P0 port of IoT:bit

Programming (MakeCode)

Step 1. Initialize OLED and IoT:bit and connect to WiFi

- Snap `Initialize OLED with width:128, height:  64` to `on start`
- Snap `Initialize IoT:bit TX P16 RX P8` from `IoT:bit` to `on start`
- Snap `Set Wi-Fi to ssid pwd` from `IoT:bit`
- Enter your Wi-Fi name and password. Here we set `smarthon` as `SSID` and `12345678` as `password`

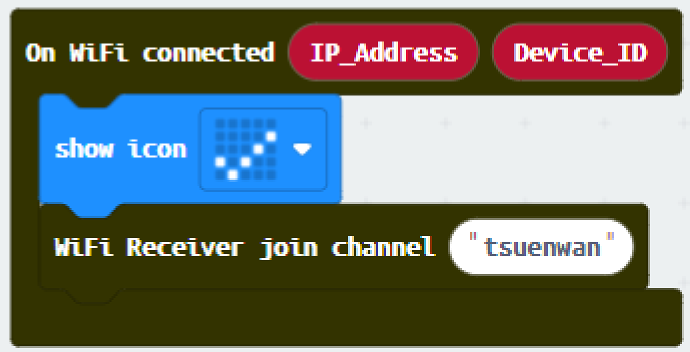


```
on start
    initialize OLED with width 128 height 64
    Initialize IoT:bit TX P16 RX P8
    Set WiFi to ssid "smarthon" pwd "12345678"
    set oldmsg to " "
```

- Set variable `oldmsg` to “” from `variables`

Step 2. Join Channel “tsuenwan”

- Snap `show icon` from `basic` to `On WiFi connected` and select icon `tick`
- Snap `WiFi Receiver join channel` from `IoT` > `Chanel`

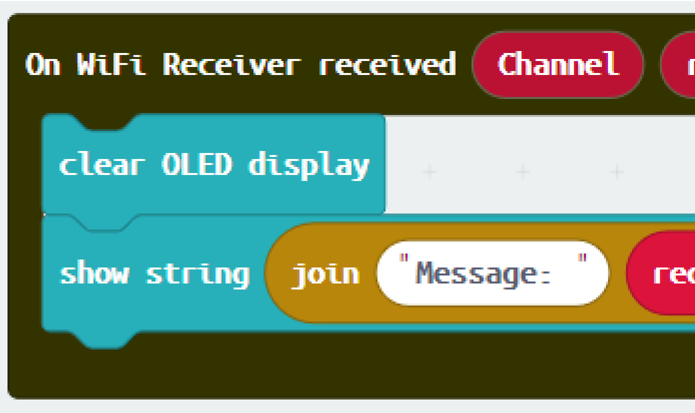


```
On WiFi connected IP_Address Device_ID
    show icon ✓
    WiFi Receiver join channel "tsuenwan"
```

- Set channel name `tsuenwan`

Step 3. Receive WiFi message

- Snap `On WiFi Receiver received`
- Go to OLED
- Snap the `clear OLED display` to `On WiFi received` to avoid overlap
- Snap the `show string`

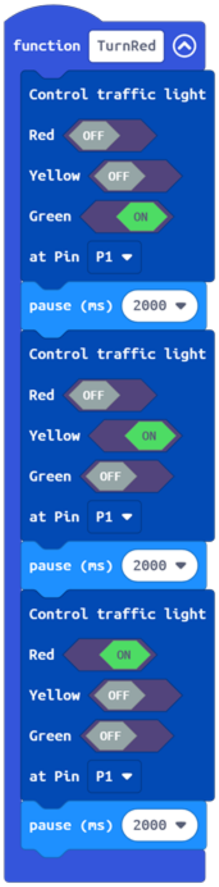


```
On WiFi Receiver received Channel r
    clear OLED display
    show string join "Message: " rec
```

- Draw variable `receivedMessage` to show string placeholder

Step 4. Set up a new function (TurnRed)

- `Control traffic light at P1` green on from `SmartCity` > `Output`
- `Pause` for 2000ms from basic
- `Control traffic light at P1` yellow on, `pause` for 2000ms

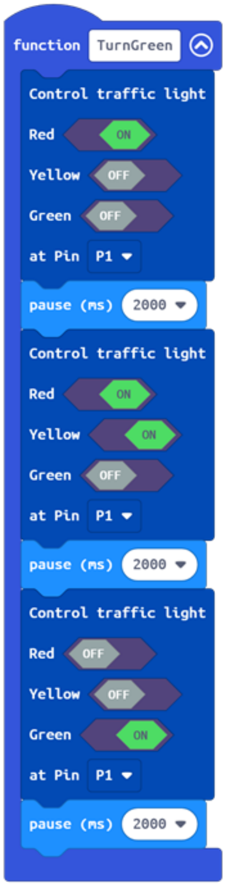


```
function TurnRed
  Control traffic light
  Red OFF
  Yellow OFF
  Green ON
  at Pin P1
  pause (ms) 2000
  Control traffic light
  Red OFF
  Yellow ON
  Green OFF
  at Pin P1
  pause (ms) 2000
  Control traffic light
  Red ON
  Yellow OFF
  Green OFF
  at Pin P1
  pause (ms) 2000
```

- `Control traffic light at P1` red on and `pause` for 2000ms.

Step 5. Set up a new function (TurnGreen)

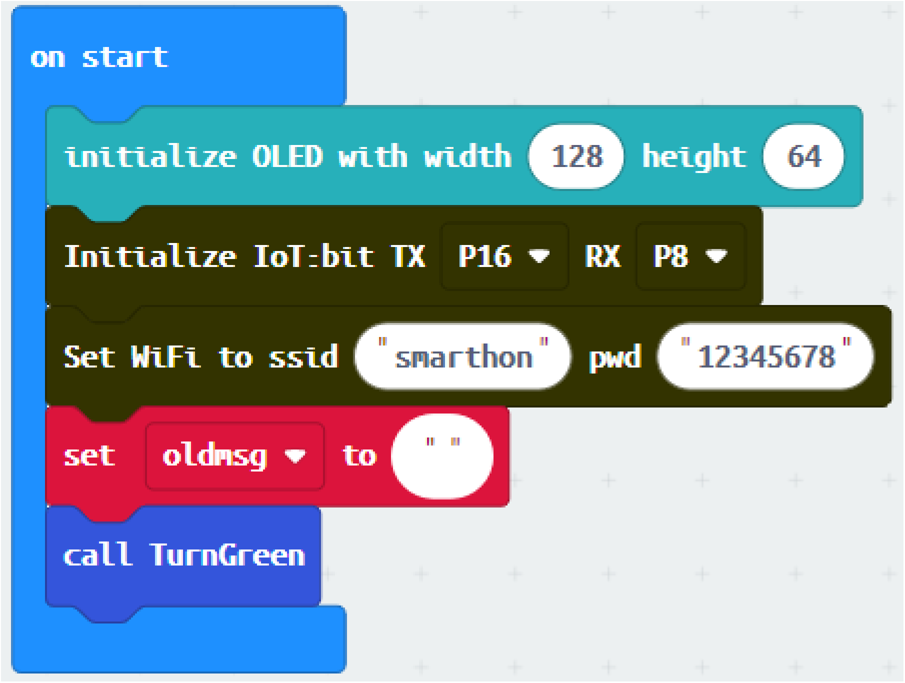
- `Control traffic light at P1` red on from `SmartCity` > `Output`
- `Pause` for 2000ms from basic
- `Control traffic light at P1` red and yellow on, `pause` for 2000ms



```
function TurnGreen
    Control traffic light
    Red ON
    Yellow OFF
    Green OFF
    at Pin P1
    pause (ms) 2000
    Control traffic light
    Red ON
    Yellow ON
    Green OFF
    at Pin P1
    pause (ms) 2000
    Control traffic light
    Red OFF
    Yellow OFF
    Green ON
    at Pin P1
    pause (ms) 2000
```

- `Control traffic light at P1` green on and `pause` for 2000ms.

Step 6. Set traffic light default status

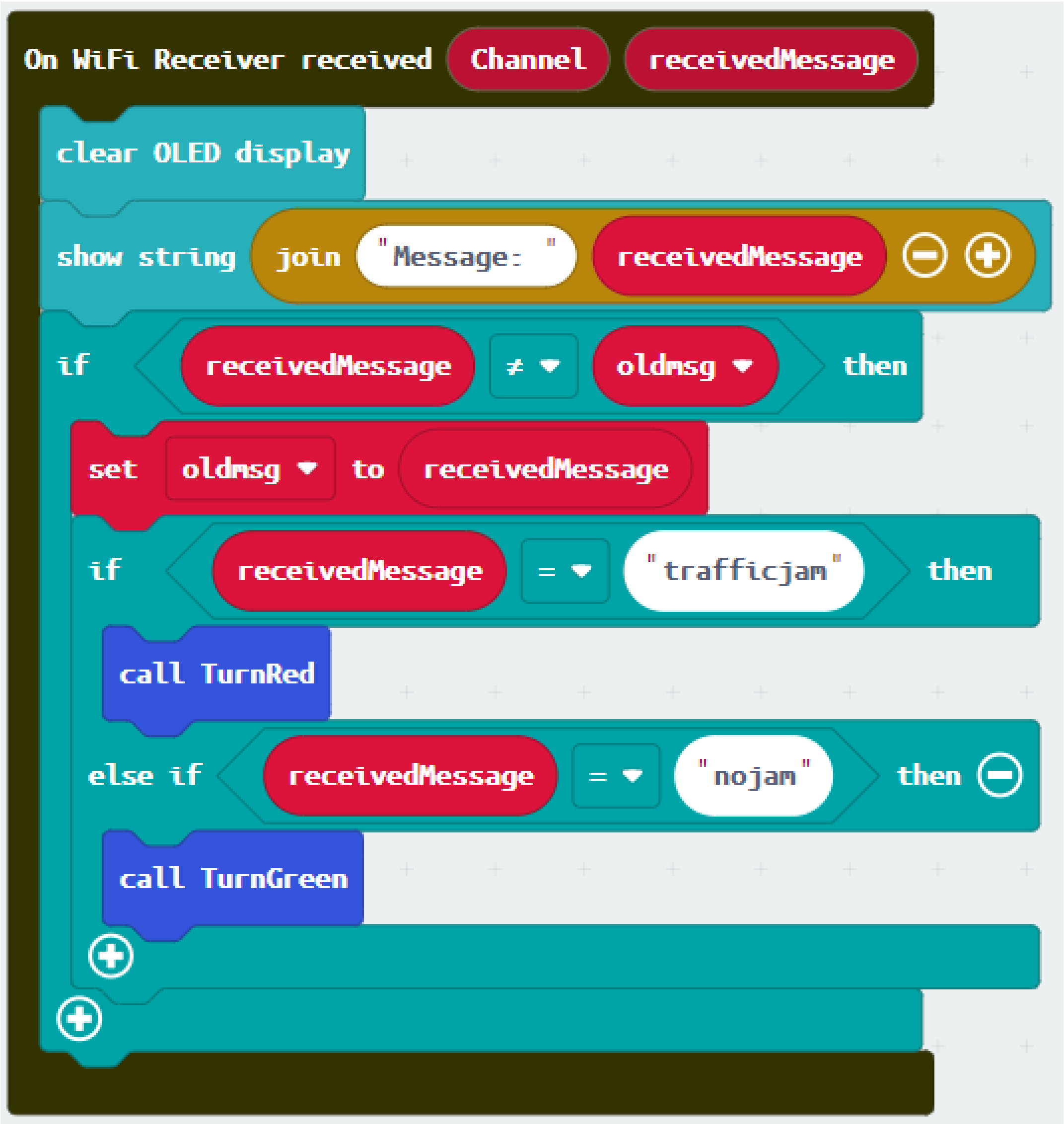


```
on start
    initialize OLED with width 128 height 64
    Initialize IoT:bit TX P16 RX P8
    Set WiFi to ssid "smarthon" pwd "12345678"
    set oldmsg to ""
    call TurnGreen
```

- Insert function `TurnGreen` to `on start`

Step 7. Change traffic light status

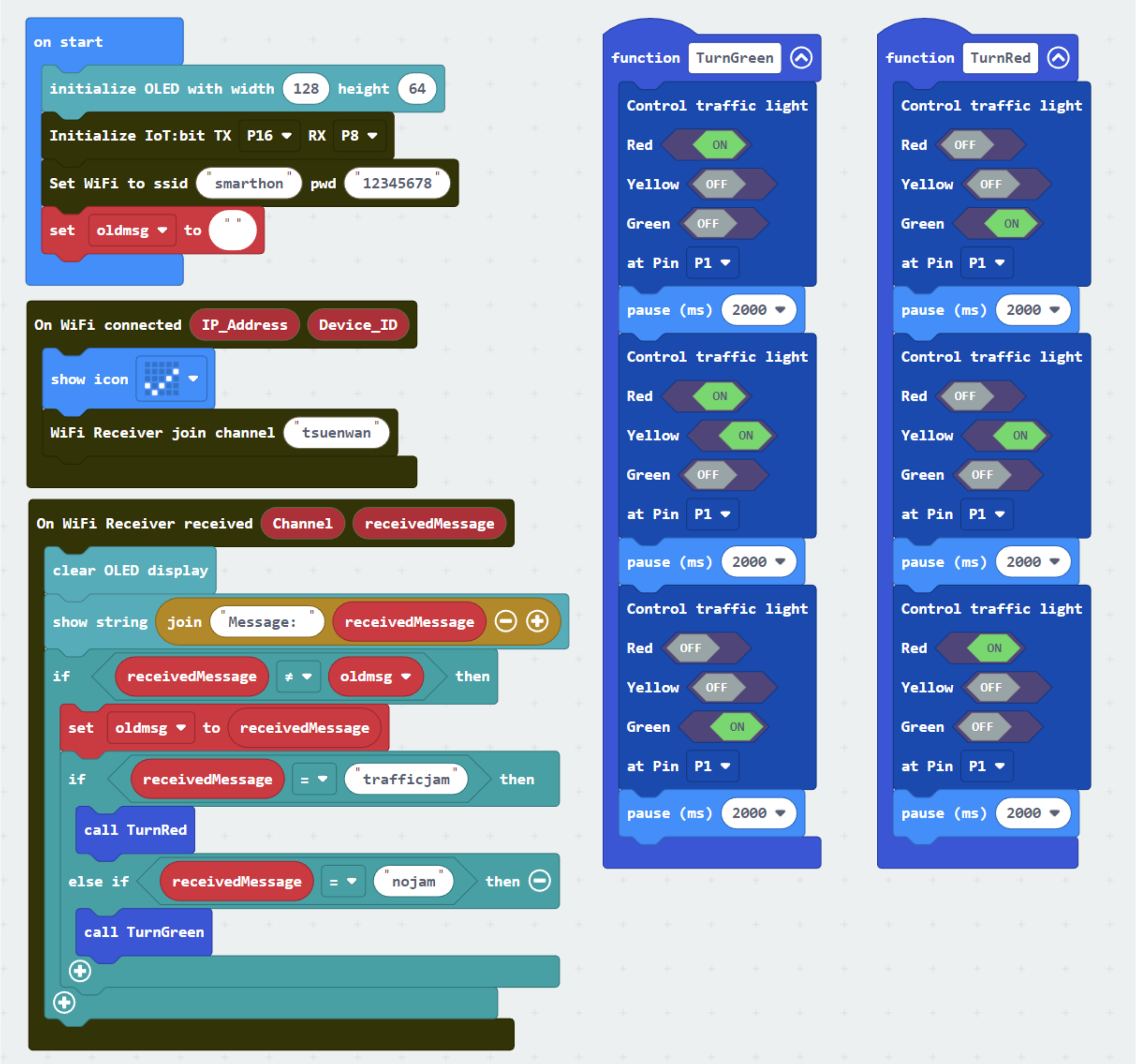
- Snap `if-statement`, set `received Message oldmsg` to `if-condition`
- Set `oldmsg` = `receivedMessage` (renew the latest traffic status)
- Snap another `if-statement`
- Set `receivedMessage` = `trafficjam` to `if-condition`
- Call function `TurnRed`
- Set `receivedMessage` = `nojam` to `else-if-condition`



```
On WiFi Receiver received Channel receivedMessage
    clear OLED display
    show string join "Message: " receivedMessage
    if receivedMessage ≠ oldmsg then
        set oldmsg to receivedMessage
        if receivedMessage = "trafficjam" then
            call TurnRed
        else if receivedMessage = "nojam" then
            call TurnGreen
```

- Call function `TurnGreen`

Full Solution MakeCode: https://makecode.microbit.org/_V5jcy5Pvm7hF



```
on start
    initialize OLED with width 128 height 64
    Initialize IoT:bit TX P16 RX P8
    Set WiFi to ssid "smarthon" pwd "12345678"
    set oldmsg to " "

On WiFi connected IP_Address Device_ID
    show icon
    WiFi Receiver join channel "tsuenwan"

On WiFi Receiver received Channel receivedMessage
    clear OLED display
    show string join "Message:" receivedMessage
    if receivedMessage ≠ oldmsg then
        set oldmsg to receivedMessage
        if receivedMessage = "trafficjam" then
            call TurnRed
        else if receivedMessage = "nojam" then
            call TurnGreen

function TurnGreen
    Control traffic light
    Red ON
    Yellow OFF
    Green OFF
    at Pin P1
    pause (ms) 2000
    Control traffic light
    Red ON
    Yellow ON
    Green OFF
    at Pin P1
    pause (ms) 2000
    Control traffic light
    Red OFF
    Yellow OFF
    Green ON
    at Pin P1
    pause (ms) 2000

function TurnRed
    Control traffic light
    Red OFF
    Yellow OFF
    Green ON
    at Pin P1
    pause (ms) 2000
    Control traffic light
    Red OFF
    Yellow ON
    Green OFF
    at Pin P1
    pause (ms) 2000
    Control traffic light
    Red ON
    Yellow OFF
    Green OFF
    at Pin P1
    pause (ms) 2000
```

Result

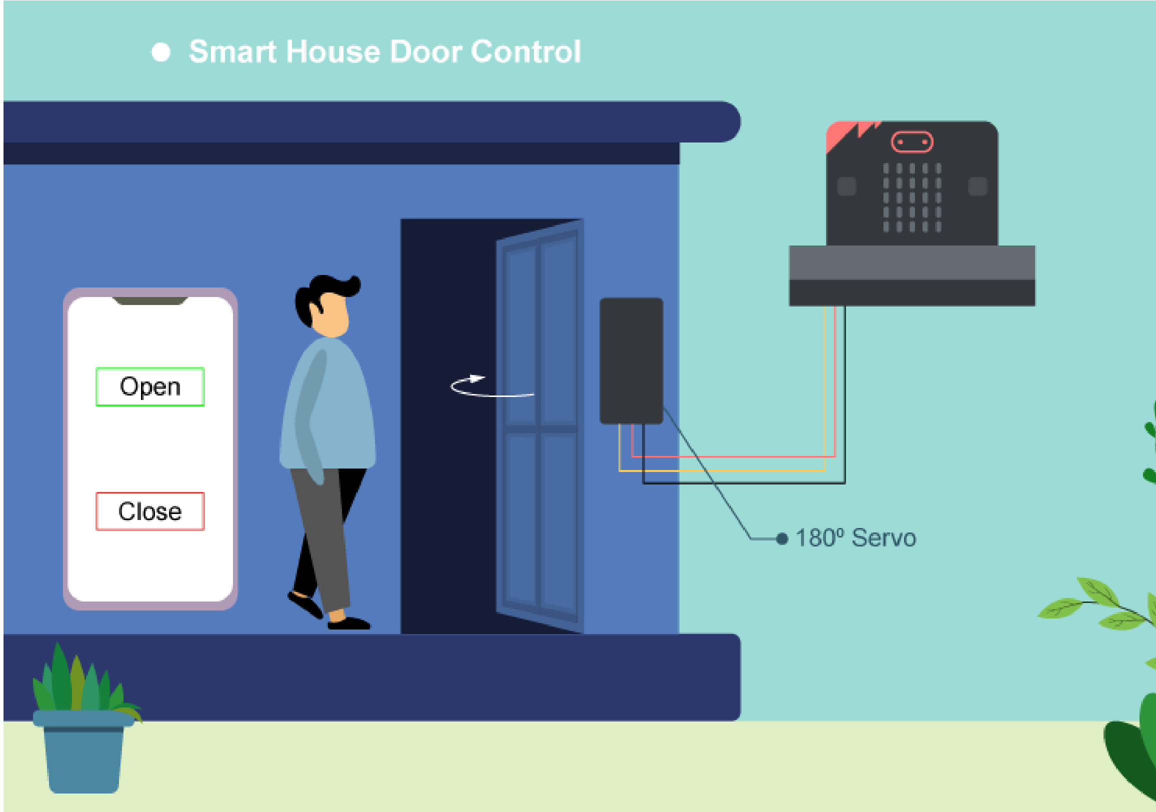
By receiving wifi message, traffic LED Module will turn to the corresponding colour automatically. If there are no traffic jam ahead (detected by light sensor). Micro:bit (sender) will send wifi message "nojam" to the micro:bit (receiver). The traffic light will turn green. If there are traffic jam ahead (detected by light sensor). Micro:bit (sender) will send wifi message "nojam" to the micro:bit (receiver). The traffic light will turn red.

Think

Q1. How can we add sound effect to the traffic LED Module according to the corresponding color?

1.1.18 IoT Case 09: Smart House door control

Level: ★ ★ ★ ★ ☆

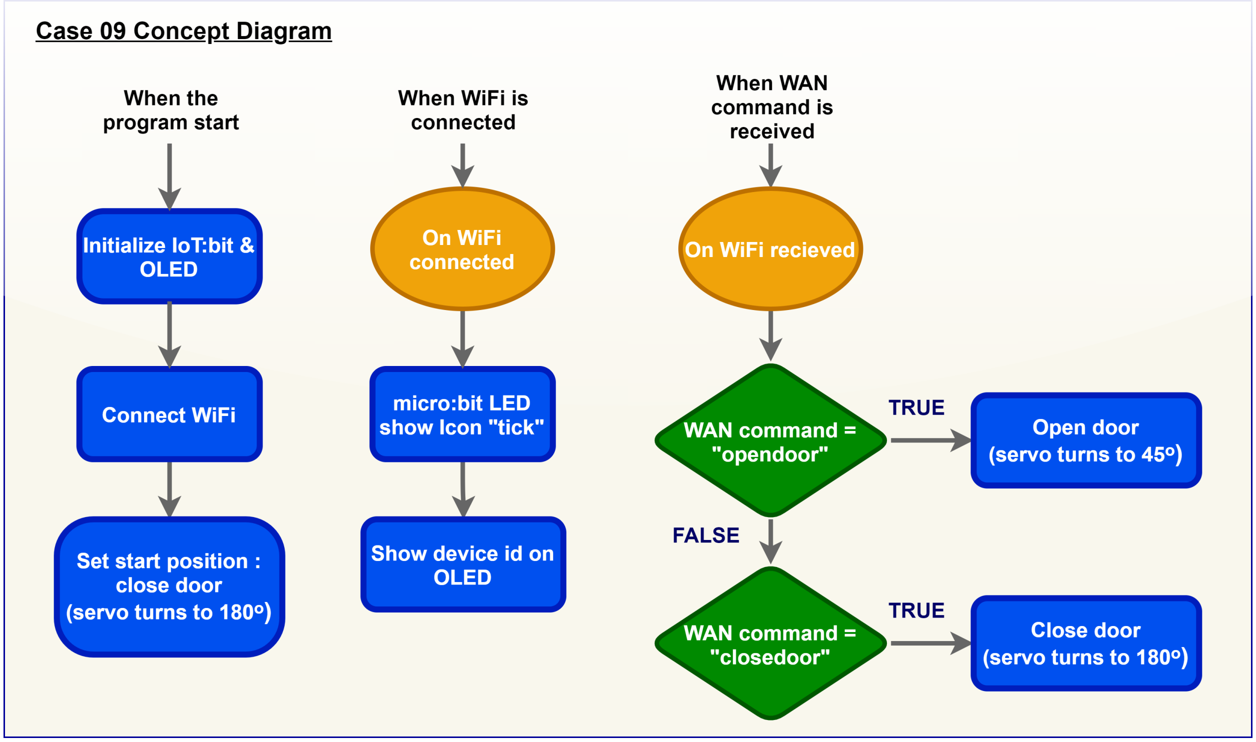


Goal

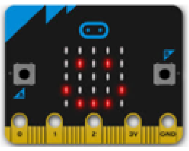
Use app inventor to create an APP to control the door of the house.

Background

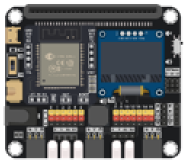
What is app inventor? App Inventor allows user to develop their own app for Android phones or IOS device(iphone, ipad) by simple programming and design work. Smart House door operation When the micro:bit receives the signal "opendoor" from the app, the 180o servo will turn for 180o to open the door. When the micro:bit receives the signal "closedoor" from the app, the 180o servo will turn back 180o to close the door.


Case 09 Concept Diagram
When the program start
Initialize IoT:bit & OLED
Connect WiFi
Set start position : close door (servo turns to 180º)
When WiFi is connected
On WiFi connected
micro:bit LED show Icon "tick"
Show device id on OLED
When WAN command is received
On WiFi recieved
WAN command = "opendoor"
TRUE
Open door (servo turns to 45º)
FALSE
WAN command = "closedoor"
TRUE
Close door (servo turns to 180º)

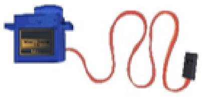
Part List



micro:bit (1)



IoT:bit with OLED (1)



180° Servo (1)

3-Pin module wire (1)

Servo M2*7.5mm sharp screw (1) (Included in servo pack)

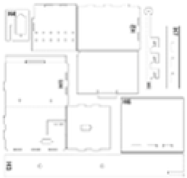
M2*10mm Screw (2)

M2 Nut (2)

M3*10mm Screw (2)

M3 Nuts (1)

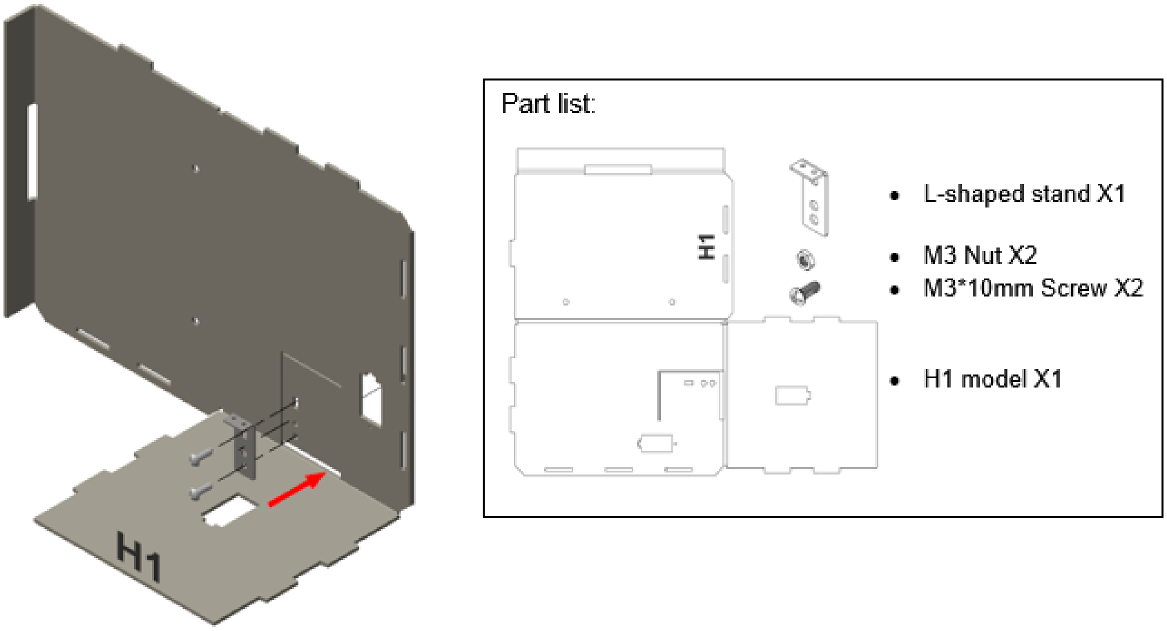
Screwdriver (1)

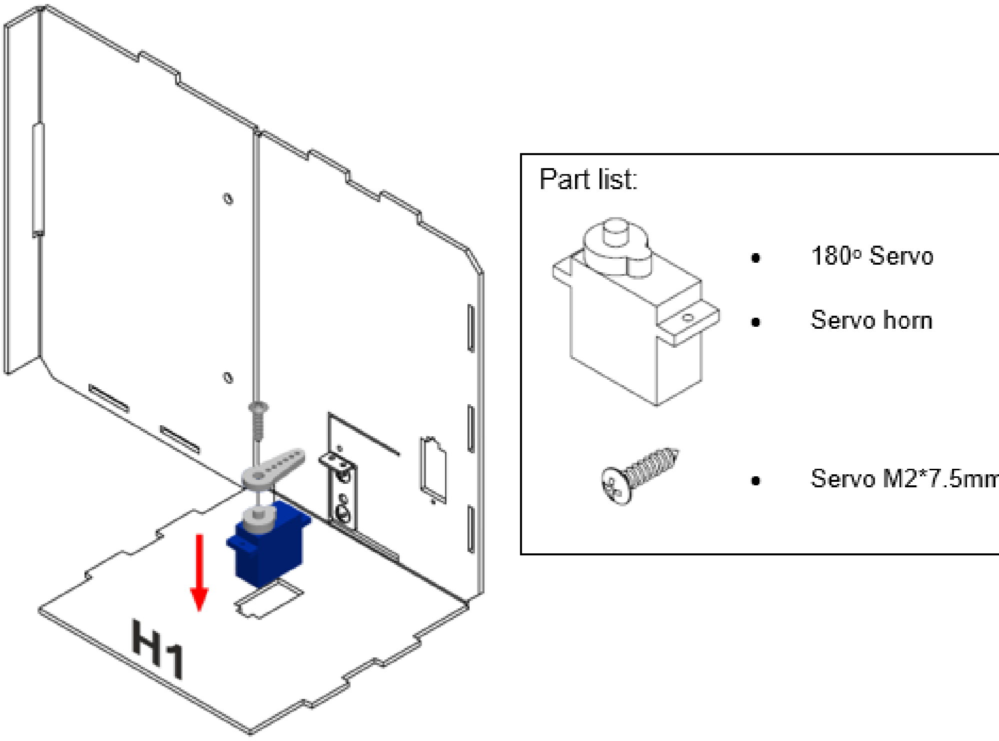


H1-H7 Model (1)

Assembly step

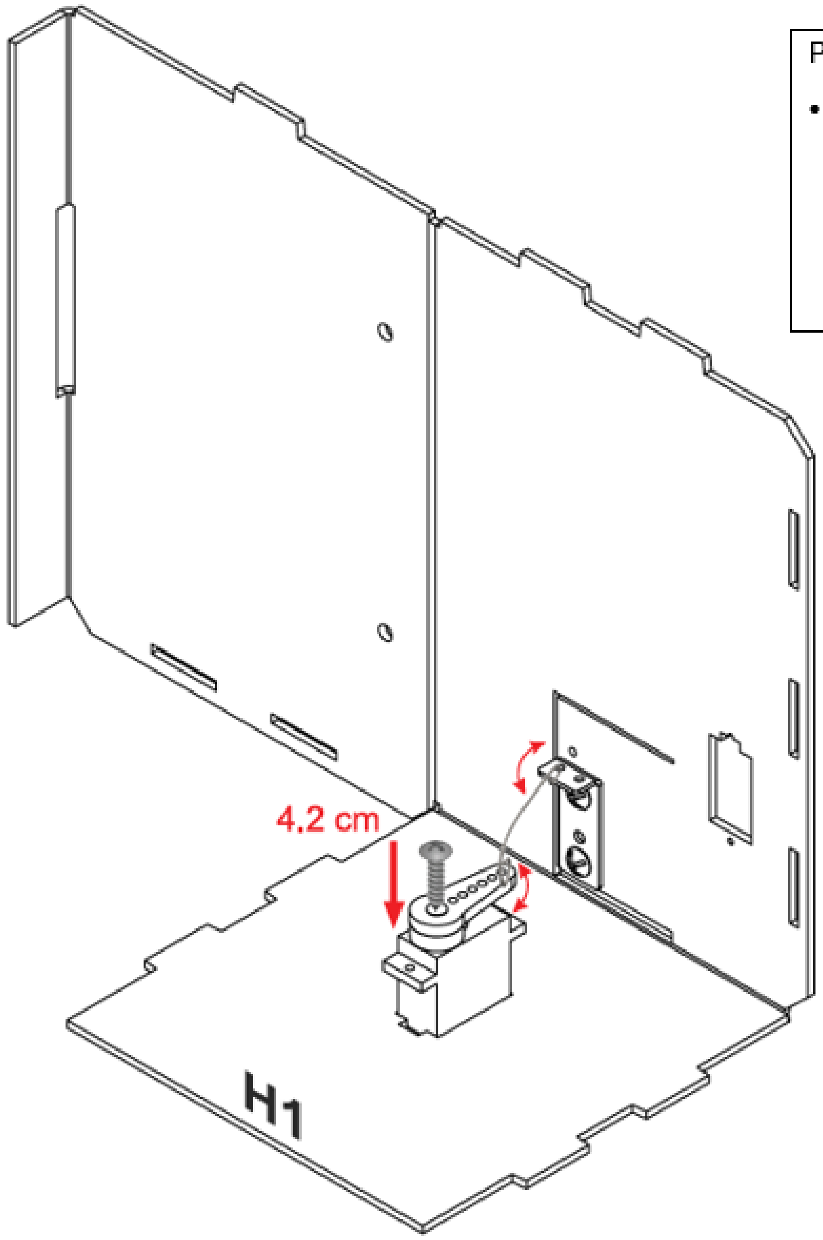
Step 1 Attach the L-shaped stand with M3 screws and nuts to the model H1. Make a small hole to fix into the stand securely.





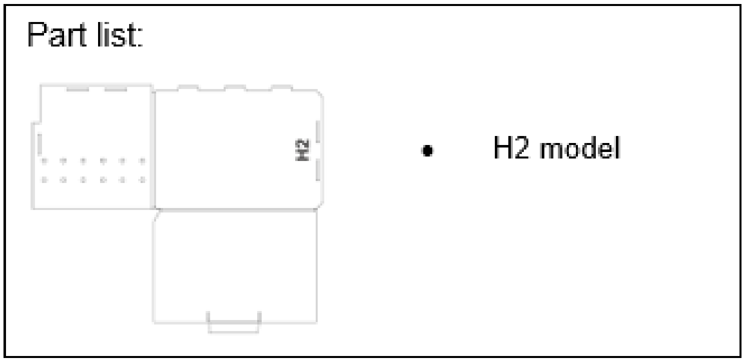
Step 2 Attach the servo to the model H1.

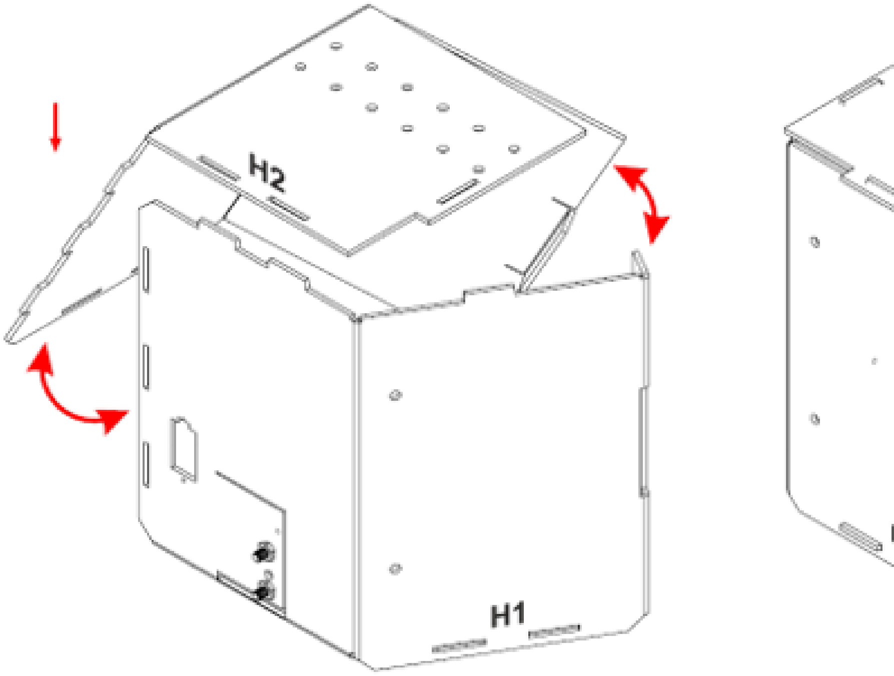
Step 3 Straighten the paper clip and cut it to 6.5 cm. Use pilers to fix the paper clip into the L-shaped stand.



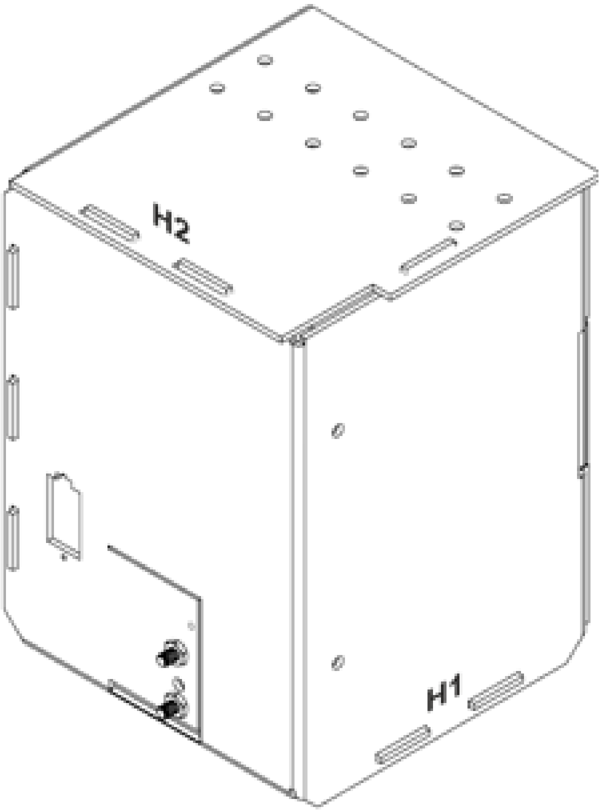
Part list:

- Paper Clip X1





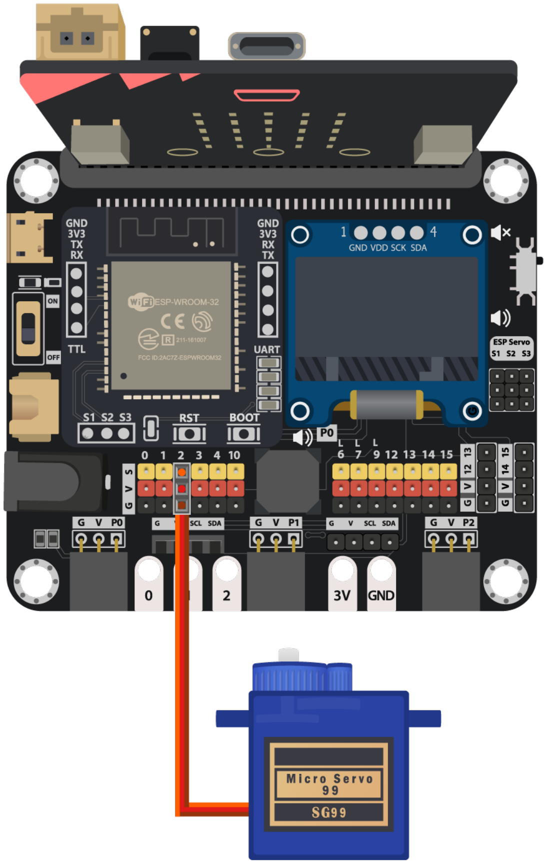
Step 4 Put together all the cardboard parts (H1-H2)



Step 5 Assembly completed!

Hardware connect

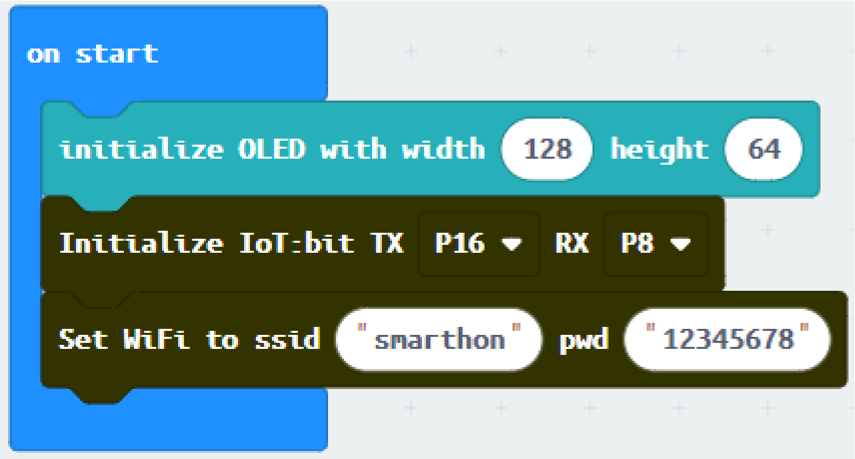
Connect the 180 servo to P2 port of IoT:bit



Programming (MakeCode)

Step 1. Initialize OLED, IoT:bit and connect to WiFi

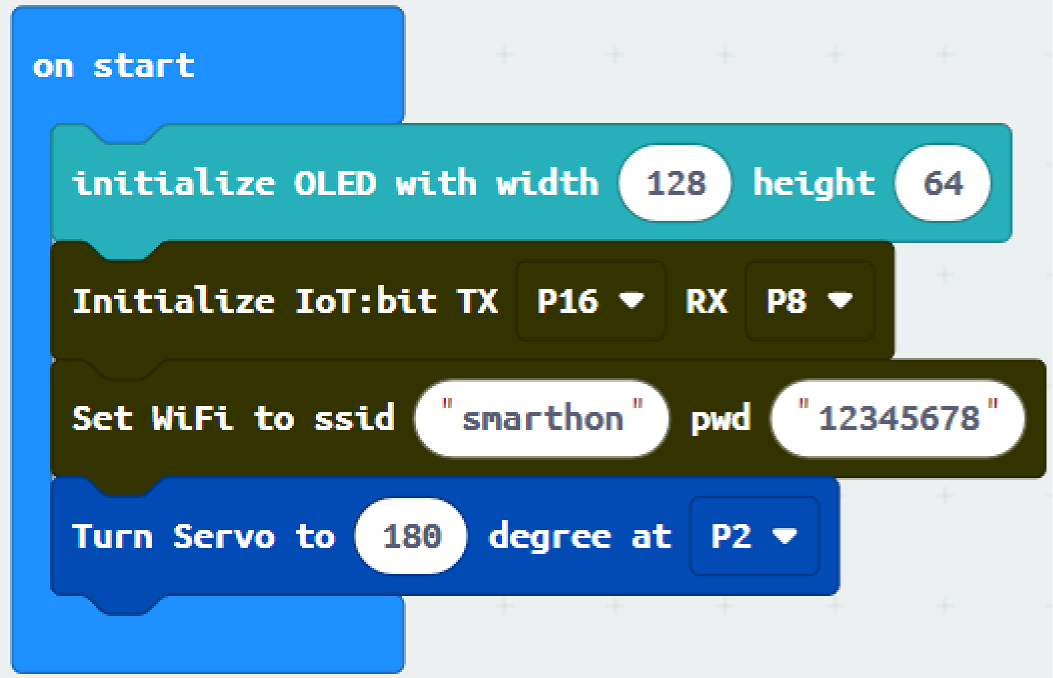
- Snap `Initialize OLED with width:128, height: 64` to `on start`
- Snap `Initialize IoT:bit TX P16 RX P8` from `IoT:bit` to `on start`
- Snap `Set Wi-Fi to ssid pwd` from `IoT:bit`
- Enter your Wi-Fi name and password. Here we set `smarthon` as SSID and `12345678` as `password`



```
on start
    initialize OLED with width 128 height 64
    Initialize IoT:bit TX P16 ▾ RX P8 ▾
    Set WiFi to ssid "smarthon" pwd "12345678"
```

Step 2. Set servo initial position

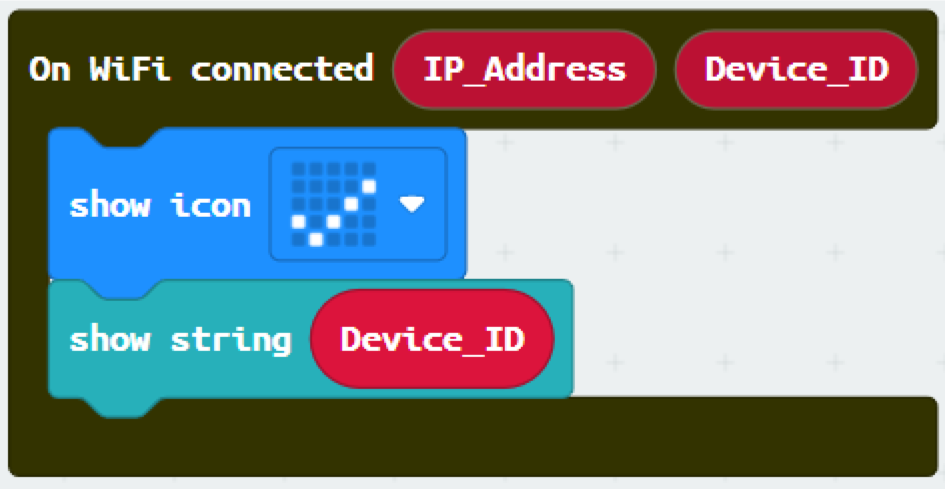
- Snap `Turn Servo to ... degree` from `SmartCity` > `Output`
- Turn servo to `180` degree at pin `P2` (the servo will control the door to be closed initially)



```
on start
    initialize OLED with width 128 height 64
    Initialize IoT:bit TX P16 ▾ RX P8 ▾
    Set WiFi to ssid "smarthon" pwd "12345678"
    Turn Servo to 180 degree at P2 ▾
```

Step 3. Show icon "tick" after WiFi connection

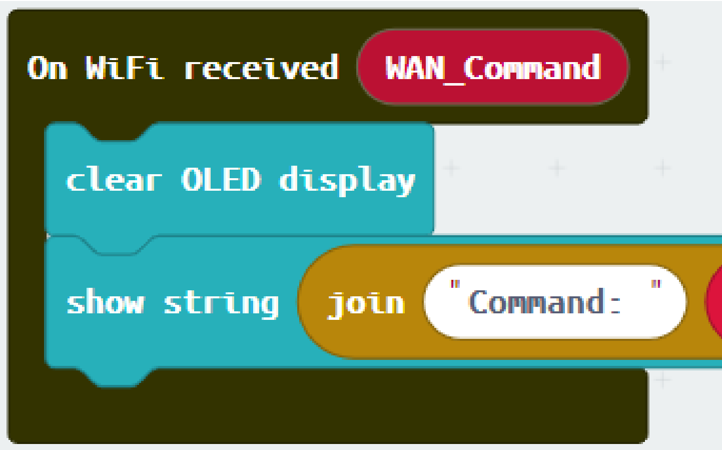
- Snap `show icon` from `basic` to `On WiFi connected` and select icon `tick`
- Draw the `Device ID` variable from `On WiFi connected` to the `show string` block placeholder



```
On WiFi connected IP_Address Device_ID
    show icon
    show string Device_ID
```

Step 4. Receive WAN command

- Go to OLED
- Snap the `clear OLED display` to `On WiFi received` to avoid overlap
- Snap the `show string` to `On WiFi received`

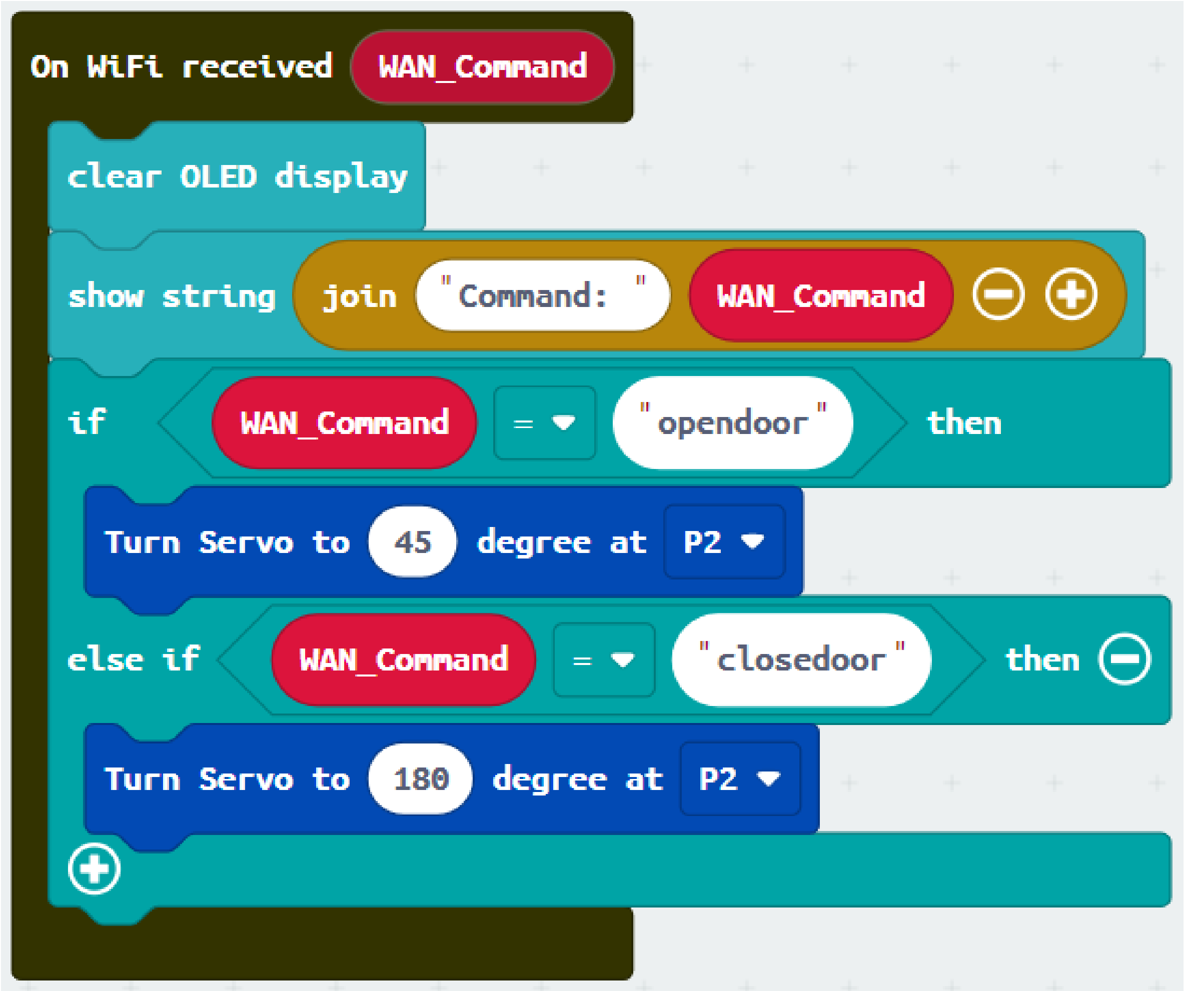


```
On WiFi received WAN_Command
    clear OLED display
    show string join " Command: "
```

- Draw the `WAN_Command` variable to show string placeholder

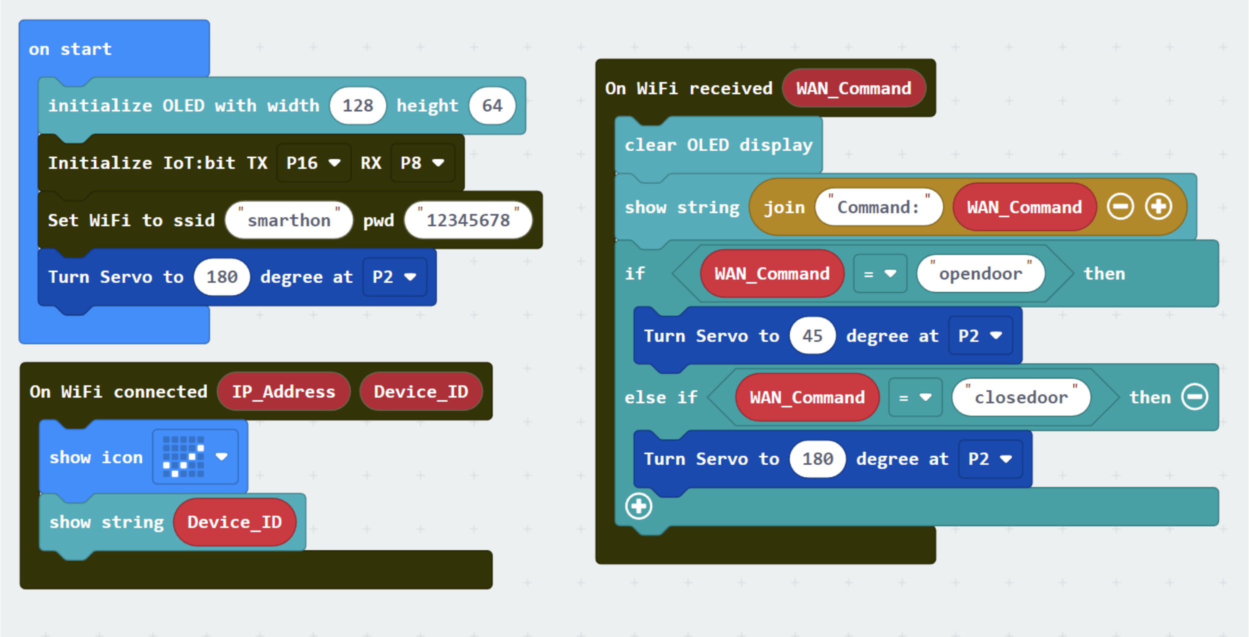
Step 5. Open/close door by WAN command

- If Wan command `opendoor` is received, `turn servo to 45 degree at P2` (open door angle)
- If Wan command `closedoor` is received, `turn servo to 180 degree at P2` (close door angle)



```
On WiFi received WAN_Command
    clear OLED display
    show string join "Command: " WAN_Command
    if WAN_Command = "opendoor" then
        Turn Servo to 45 degree at P2
    else if WAN_Command = "closedoor" then
        Turn Servo to 180 degree at P2
```

Full Solution MakeCode:https://makecode.microbit.org/_YhvAsP3HCgY6



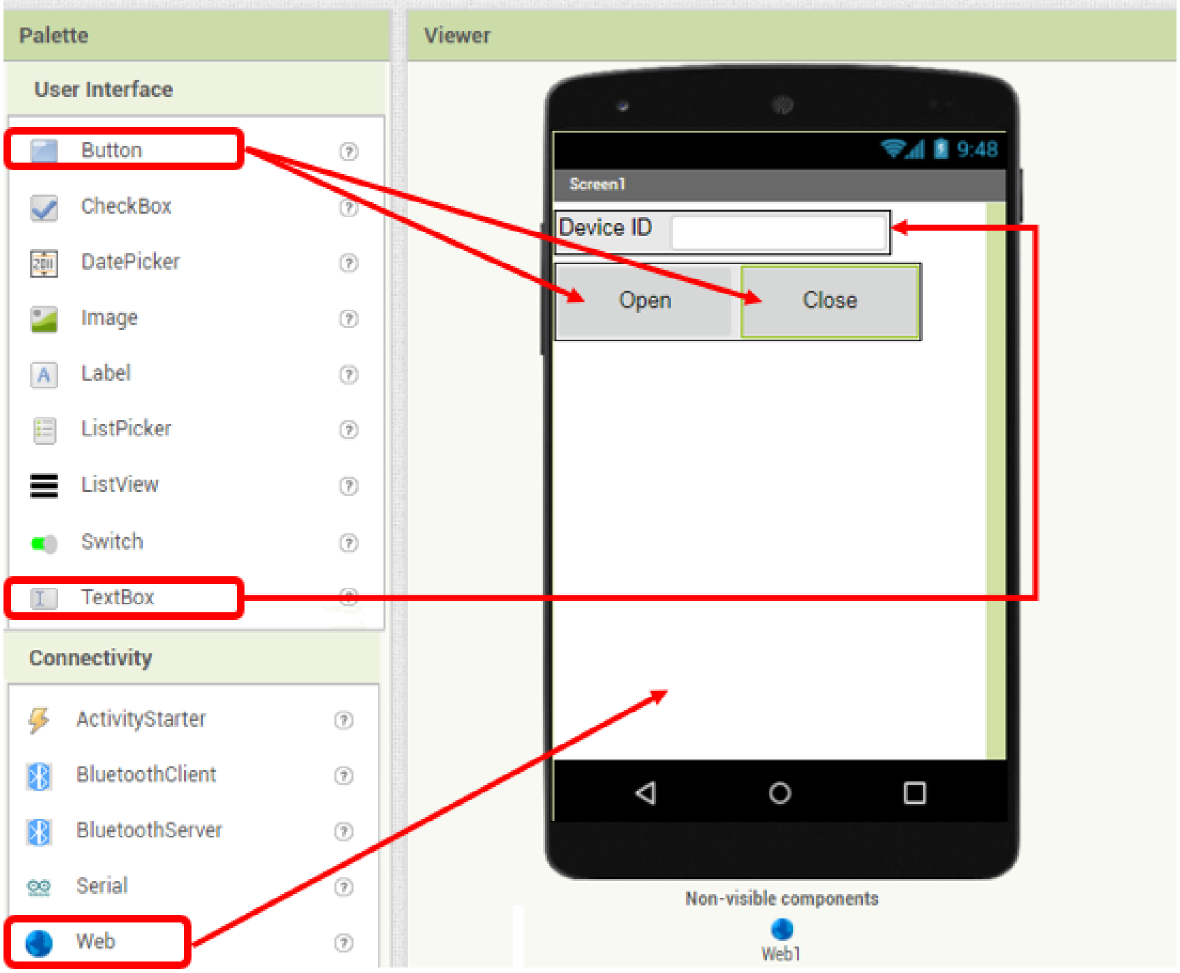
```
on start
    initialize OLED with width 128 height 64
    Initialize IoT:bit TX P16 RX P8
    Set WiFi to ssid "smarthon" pwd "12345678"
    Turn Servo to 180 degree at P2

On WiFi connected IP_Address Device_ID
    show icon
    show string Device_ID

On WiFi received WAN_Command
    clear OLED display
    show string join "Command: " WAN_Command
    if WAN_Command = "opendoor" then
        Turn Servo to 45 degree at P2
    else if WAN_Command = "closedoor" then
        Turn Servo to 180 degree at P2
```

IoT (App Inventor)

* For more details, please refer to "Chapter 3: Direct control micro:bit by App Inventor 2"

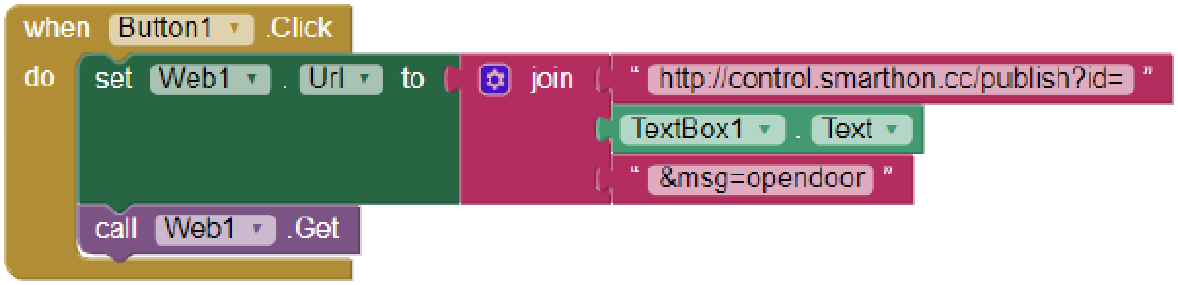
Step 1 Create the page with the components

- On Designer:
- Drag the components from the left menu – we will need buttons to open or close the door and a textbox for input the Device ID number.
- Also, add the invisible component "Web" under connectivity, we will need it for WAN connection.



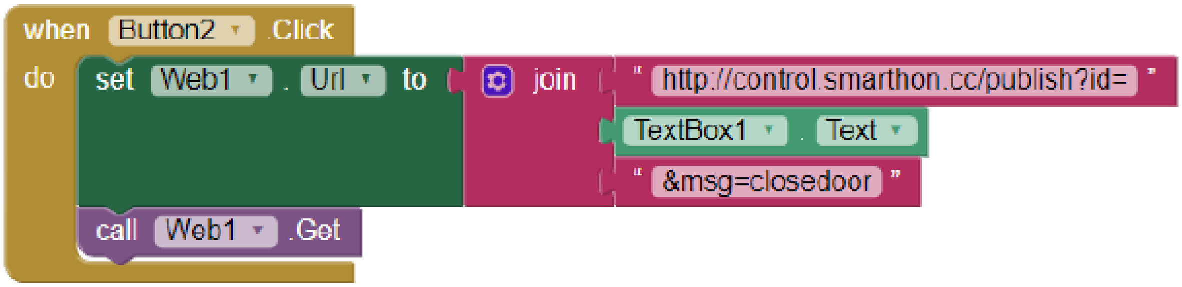
Step 2 Make the programming part

- On Blocks:
- The WAN control command URL is: http://control.smarthon.cc/publish?id=DeviceID&msg=ControlCommand
- When Button1 is clicked, it will direct to the URL with given device ID and command "opendoor".



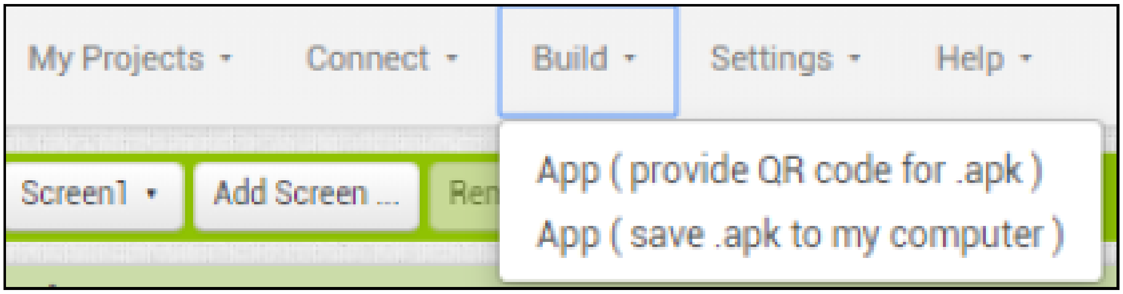
```
when Button1 .Click
do  set Web1 . Url to  join  " http://control.smarthon.cc/publish?id= "
                              TextBox1 . Text
                              " &msg=opendoor "
    call Web1 .Get
```

- When Button2 is clicked, it will direct to the URL with given device ID and command “closedoor”.



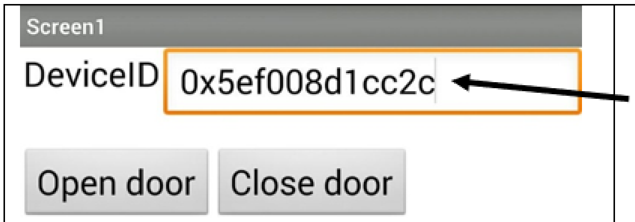
```
when Button2 .Click
do  set Web1 . Url to  join  " http://control.smarthon.cc/publish?id= "
                              TextBox1 . Text
                              " &msg=closedoor "
    call Web1 .Get
```

- Download the App in your phone to by .apk or scan QR code.


My Projects
Connect
Build
Settings
Help
Screen1
Add Screen ...
Ren
App (provide QR code for .apk)
App (save .apk to my computer)

Result

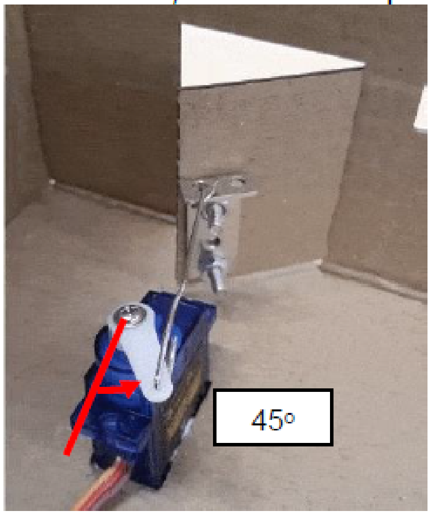
Input your device ID number in the textbox, then click button “opendoor” to send command “opendoor”. Click the button “Close Door” to send command “closedoor”.



* for the micro:bit device ID number, please refer to Chapter 3: Control your micro:bit by App Inventor 2

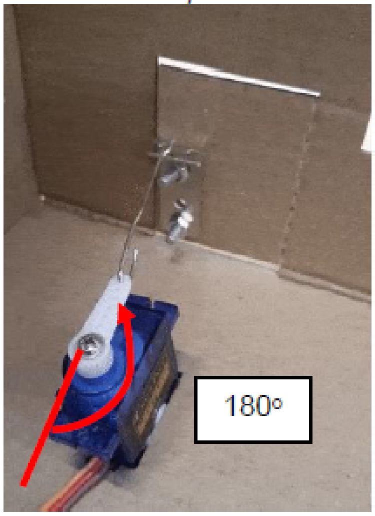
Open door

when command "opendoor" is received, the servo will turn to 45o, the door will be opened.



Close door

When command "closedoor" is received, the servo will turn to 180o, the door will be closed.



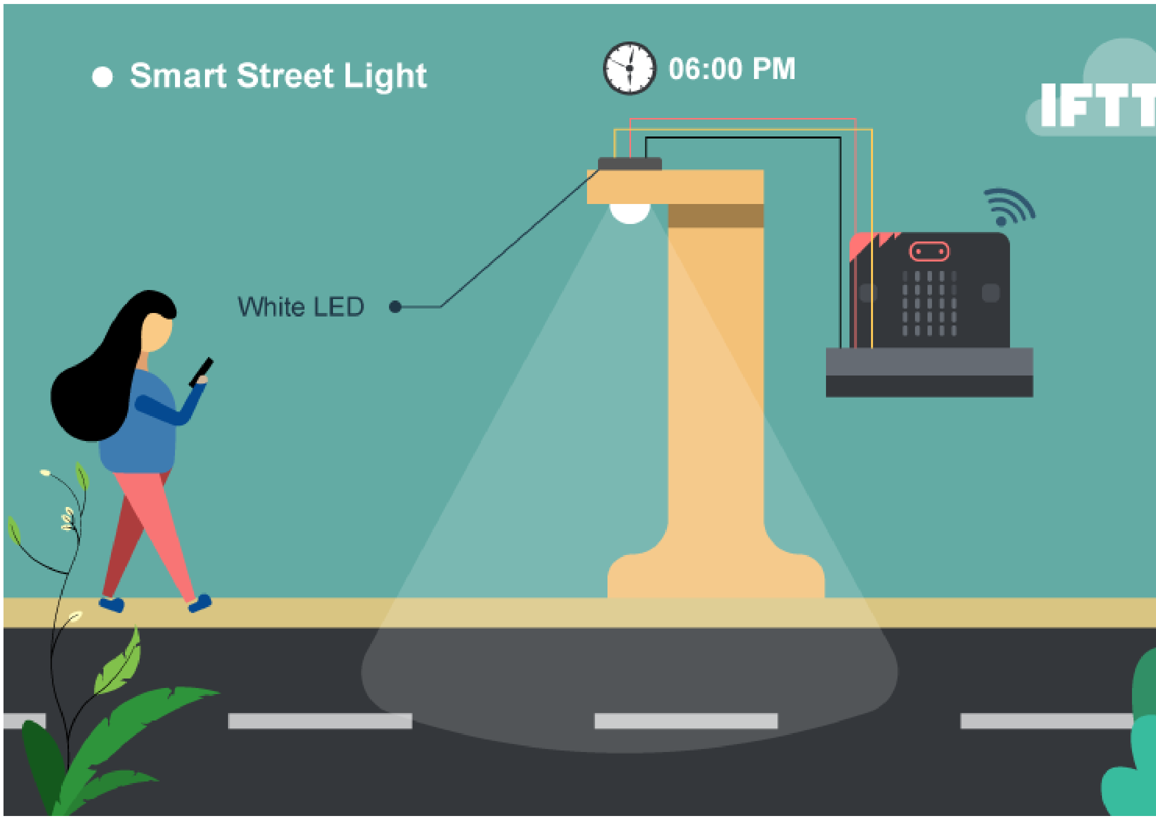
With app inventor and micro:bit, you can control the door open/close by mobile app!

Think

Q1. How can we add password authentication to open the door? Q2. We can install the servo to different position in the house (tips: modify the turning angle of the servo)

1.1.19 IoT Case 10: Smart Street Light

Level: ★ ★ ★ ★ ☆

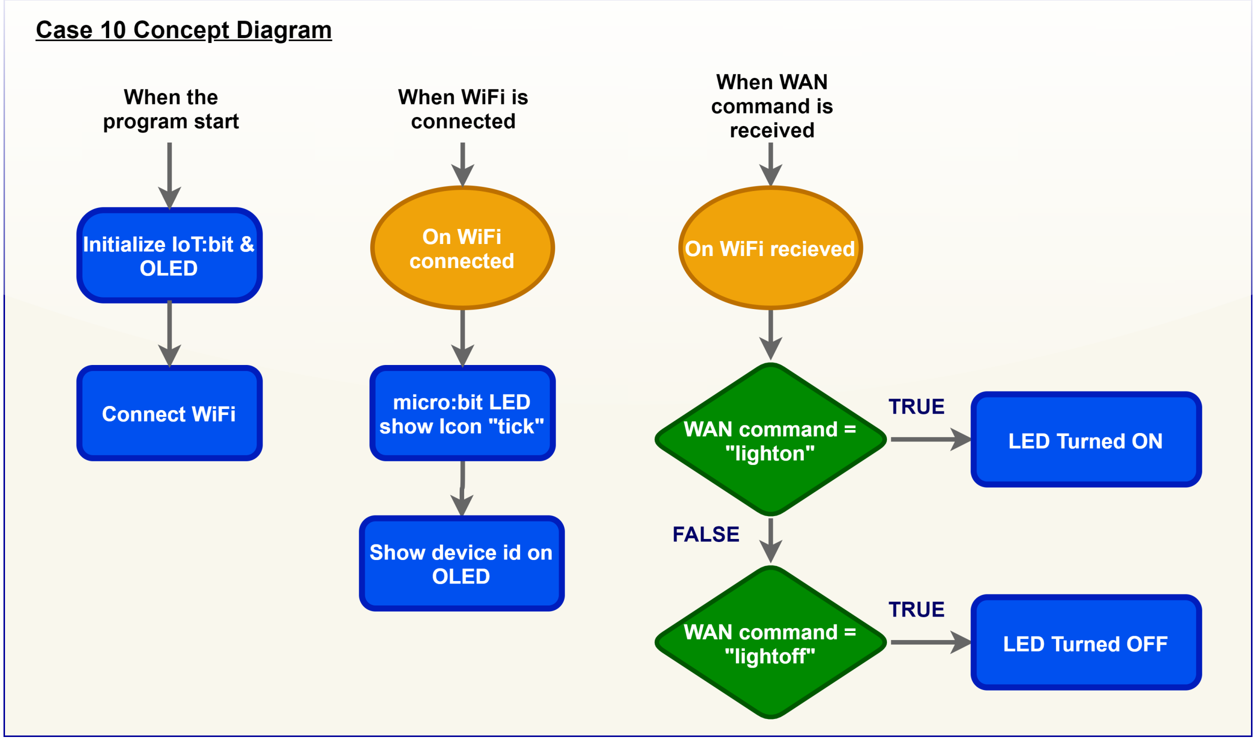


Goal

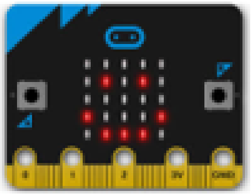
Make a smart streetlight which can have a button to control on off.

Background

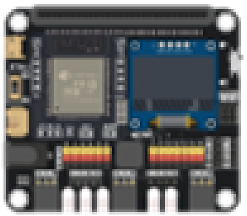
What is Smart Street Light? To improve the living stand of citizen and to save electricity, smart street light can be automatically turned on at night (eg.6pm) and turned off in the morning (eg.6am) Smart Streetlight operation When WAN commands are received from WAN (eg. IFTTT), micro:bit will get the command name. If the command name is "light_on", the LED light will be turned on. If the command name is "light_off", the LED light will be turned off.


Case 10 Concept Diagram
When the program start
Initialize IoT:bit & OLED
Connect WiFi
When WiFi is connected
On WiFi connected
micro:bit LED show Icon "tick"
Show device id on OLED
When WAN command is received
On WiFi recieved
WAN command = "lighton"
TRUE
LED Turned ON
FALSE
WAN command = "lightoff"
TRUE
LED Turned OFF

Part List



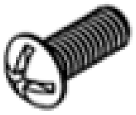
micro:bit (1)



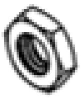
IoT:bit with OLED (1)

White LED Light (1)

3-pin module wire (1)

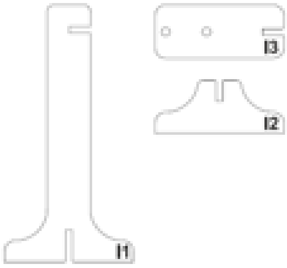


M4*10mm Screw (2)



M4 Nut (2)

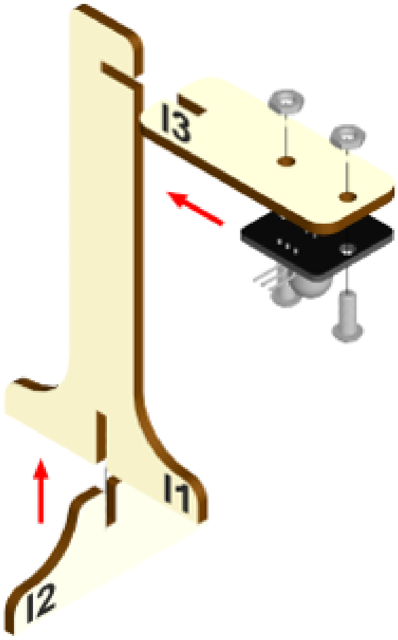
Screwdriver (1)

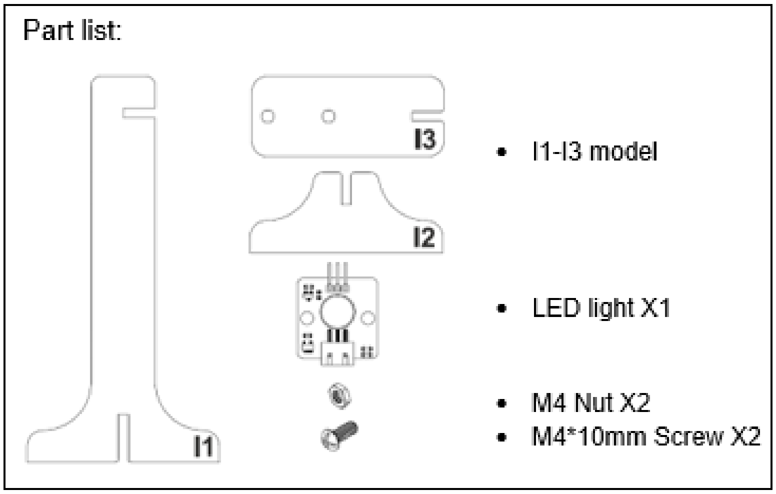


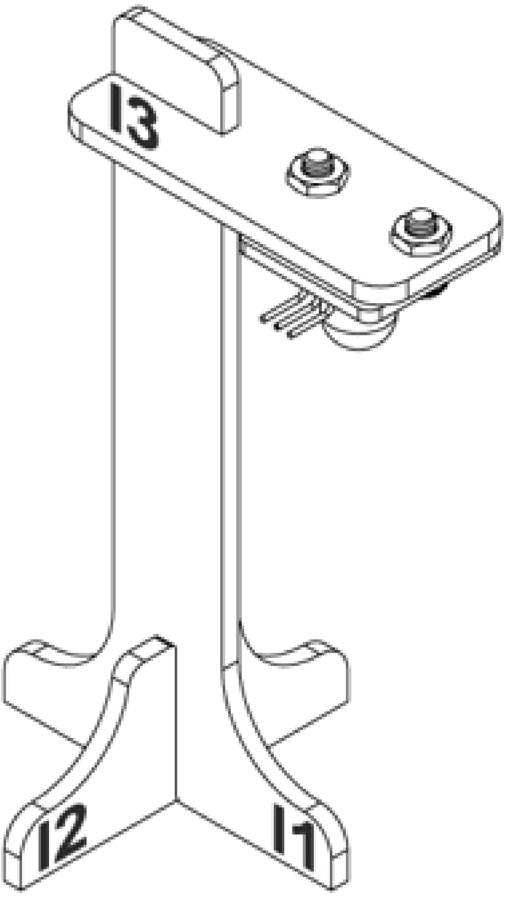
I1-I2 Model (1)

Assembly step

Step 1 Attach the LED light to the model I2 with M4 screw and nuts. Put together all the cardboard parts (I1-I3)

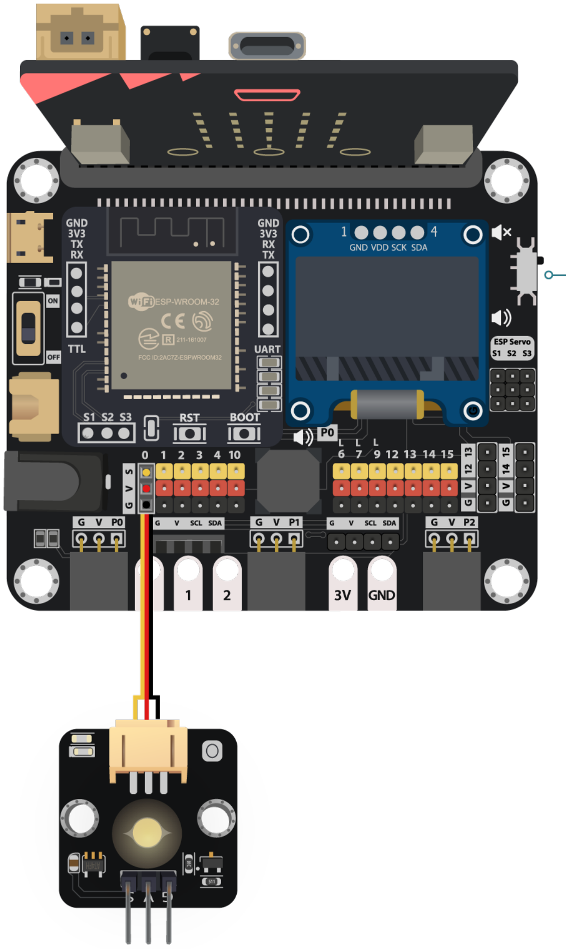






Step 2 Assembly completed!

Hardware connect



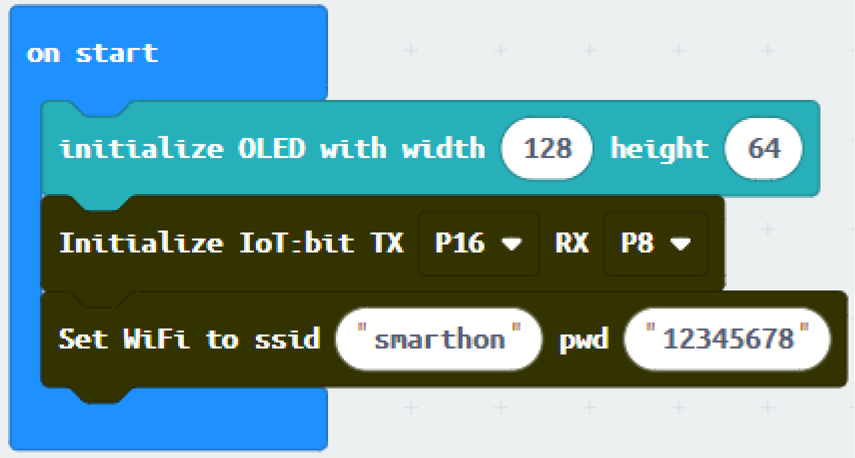
Connect the white LED Light to P0 port of IoT:bit

Pull the buzzer switch 'up' to disconnect the buzzer in this execrise

Programming (MakeCode)

Step 1. Initialize OLED, IoT:bit and connect to WiFi

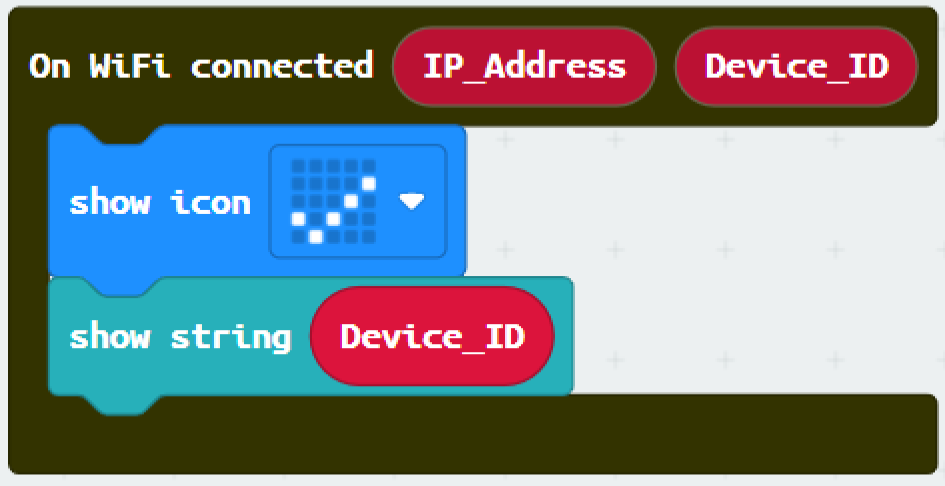
- Snap `Initialize OLED with width:128, height: 64` to `on start`
- Snap `Initialize IoT:bit TX P16 RX P8` from `IoT:bit` to `on start`
- Snap `Set Wi-Fi to ssid pwd` from `IoT:bit`
- Enter your Wi-Fi name and password. Here we set `smarthon` as `SSID` and `12345678` as `password`



```
on start
    initialize OLED with width 128 height 64
    Initialize IoT:bit TX P16 RX P8
    Set WiFi to ssid "smarthon" pwd "12345678"
```

Step 2. Show icon "tick" after WiFi connection

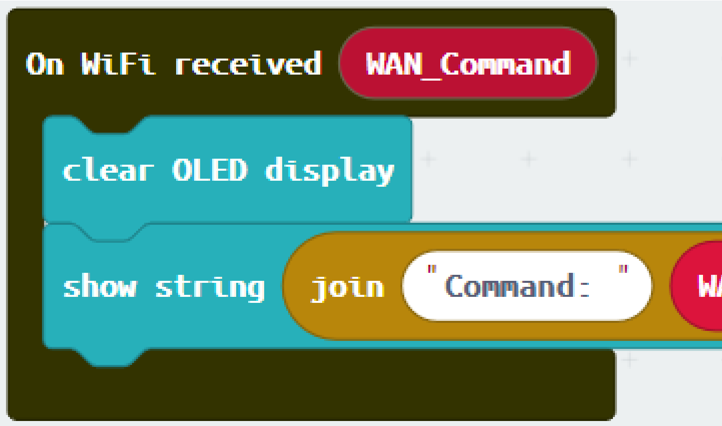
- Snap `show icon` from `basic` to `On WiFi connected` and select icon `tick`
- Draw the `Device ID` variable from `On WiFi connected` to the `show string` block placeholder



```
On WiFi connected IP_Address Device_ID
    show icon
    show string Device_ID
```

Step 3. Receive WAN command

- Go to OLED
- Snap the `clear OLED display` to `On WiFi received` to avoid overlap
- Snap the `show string` to `On WiFi received`

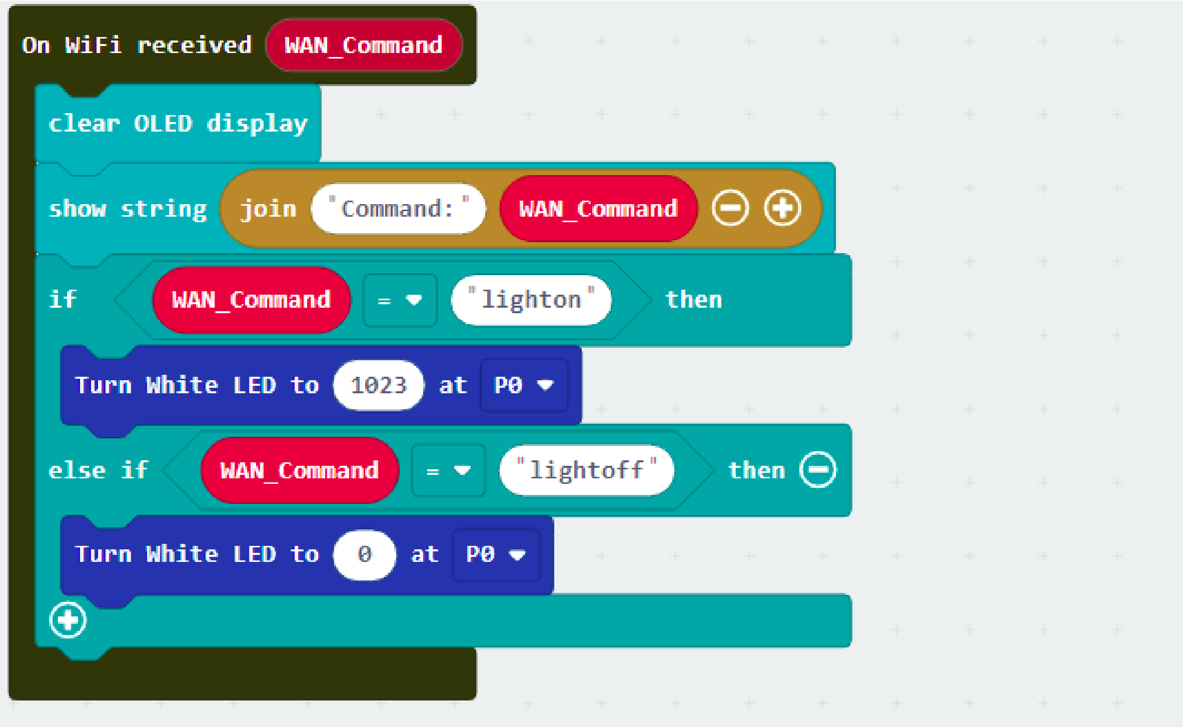


```
On WiFi received WAN_Command
  clear OLED display
  show string join "Command: " WA
```

- Draw the WAN_Command variable to show string placeholder

Step 4. Turn on/off LED by WAN command

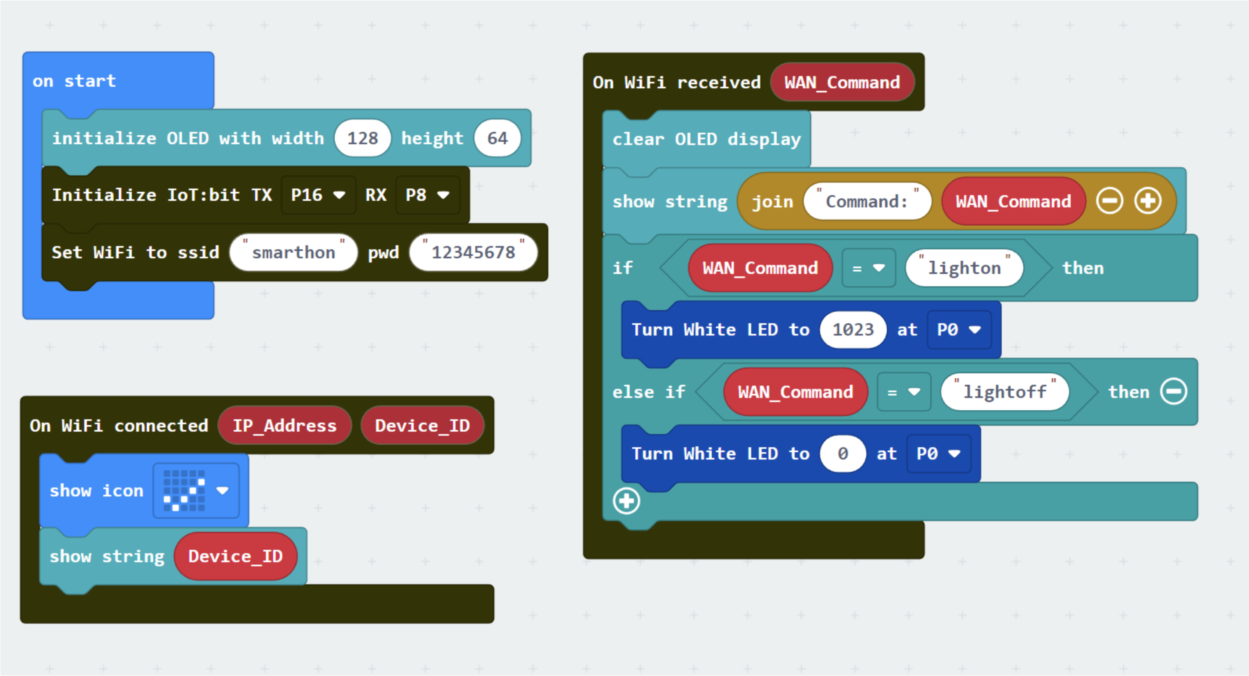
- Snap if-condition
- Set variable WAN_Command = lighton into if-condition
- turn White LED ... from SmartCity > Output, turn white LED with intensity: 1023 at P0
- Set variable WAN_Command = lightoff into else-if-condition
- Snap turn White LED ... from SmartCity > Output, turn white LED with intensity: 0 at P0



```
On WiFi received WAN_Command
  clear OLED display
  show string join "Command: " WAN_Command
  if WAN_Command = "lighton" then
    Turn White LED to 1023 at P0
  else if WAN_Command = "lightoff" then
    Turn White LED to 0 at P0
```

Full Solution

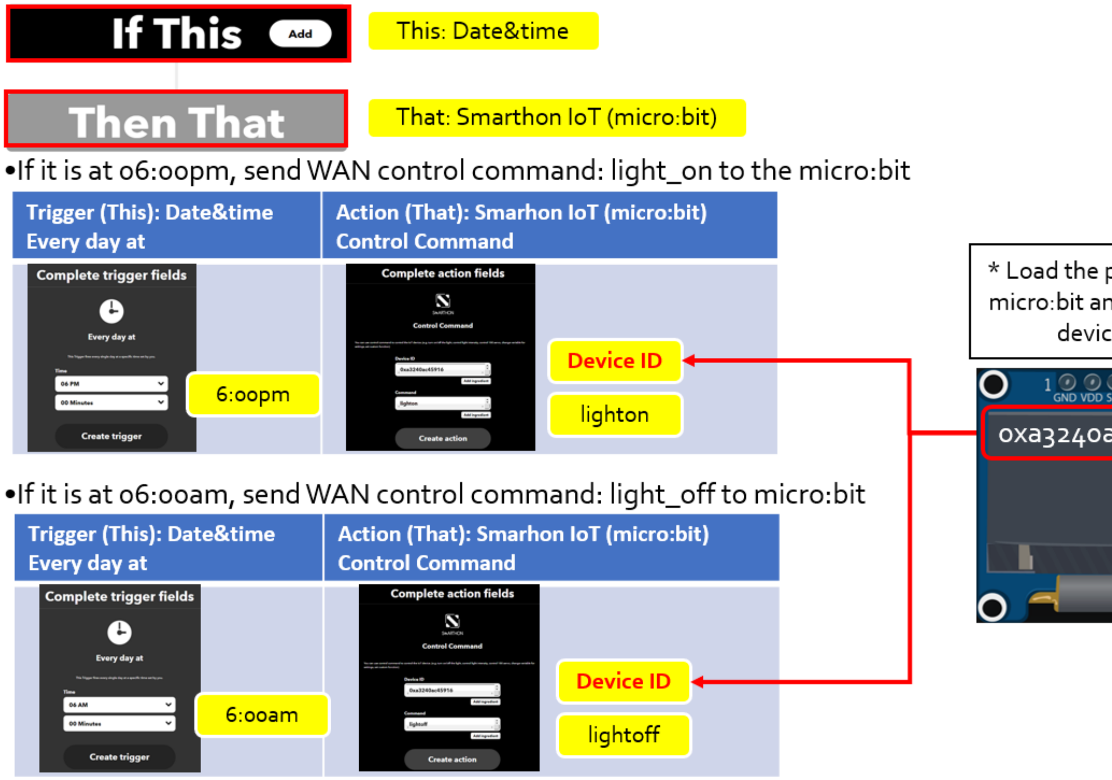
MakeCode: https://makecode.microbit.org/_6DuRvwD5PYpU



```
on start
    initialize OLED with width 128 height 64
    Initialize IoT:bit TX P16 RX P8
    Set WiFi to ssid "smarthon" pwd "12345678"

On WiFi connected IP_Address Device_ID
    show icon
    show string Device_ID

On WiFi received WAN_Command
    clear OLED display
    show string join "Command:" WAN_Command
    if WAN_Command = "lighton" then
        Turn White LED to 1023 at P0
    else if WAN_Command = "lightoff" then
        Turn White LED to 0 at P0
```

IoT (IFTTT)

* For the setting of IFTTT, please refer to "Chapter 4: Cloud Control micro:bit by IFTTT"



If This Add

This: Date&time

Then That

That: Smarthon IoT (micro:bit)

•If it is at 06:00pm, send WAN control command: light_on to the micro:bit

Trigger (This): Date&time Every day at	Action (That): Smarhon IoT (micro:bit) Control Command
6:00pm	Device ID lighton

•If it is at 06:00am, send WAN control command: light_off to micro:bit

Trigger (This): Date&time Every day at	Action (That): Smarhon IoT (micro:bit) Control Command
6:00am	Device ID lightoff

* Load the p
micro:bit an
devic

Step 1. Create applet in IFTTT

Result

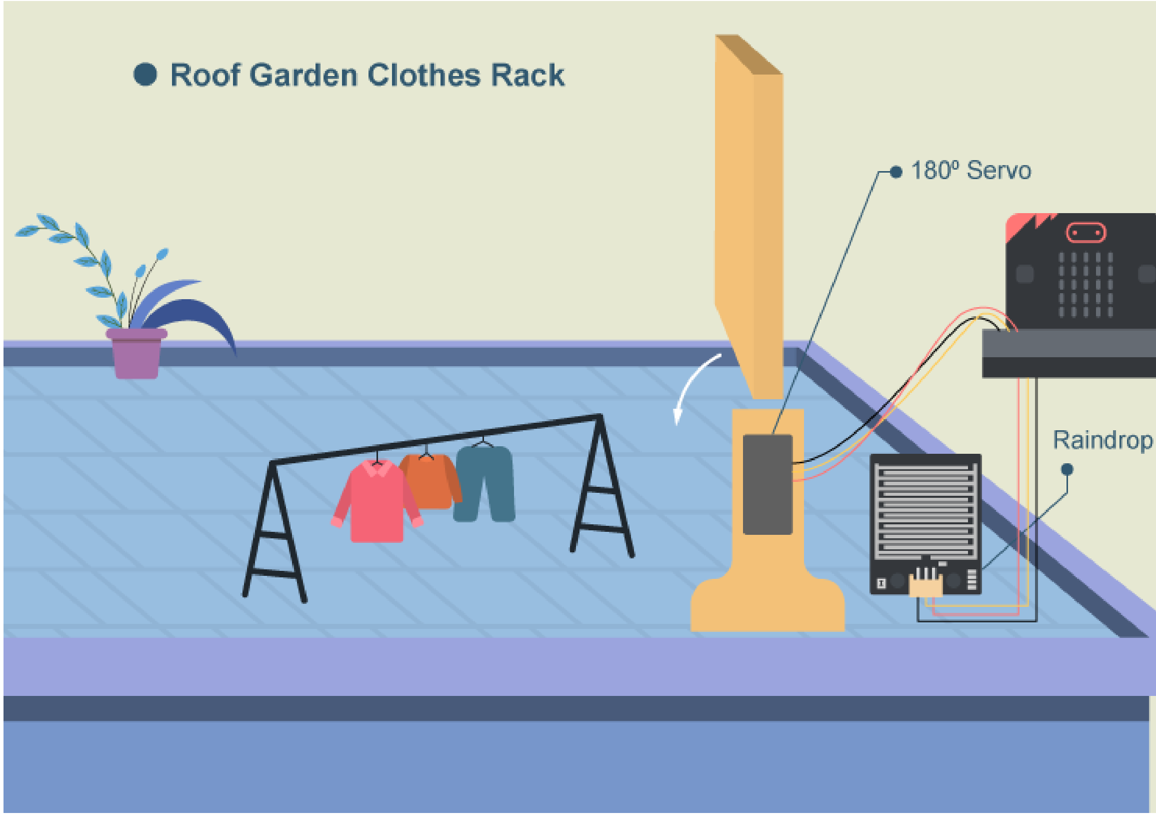
The micro:bit is controlled by IFTTT (trigger by date&time). The LED light will be turned on at 6pm and turned off at 6am every day.

Think

Q1. How to turn on the light automatically if the today's weather is cloudy reported by IFTTT.

1.1.20 IoT Case 11: Roof garden clothes rack

Level: ★ ★ ★ ★ ☆

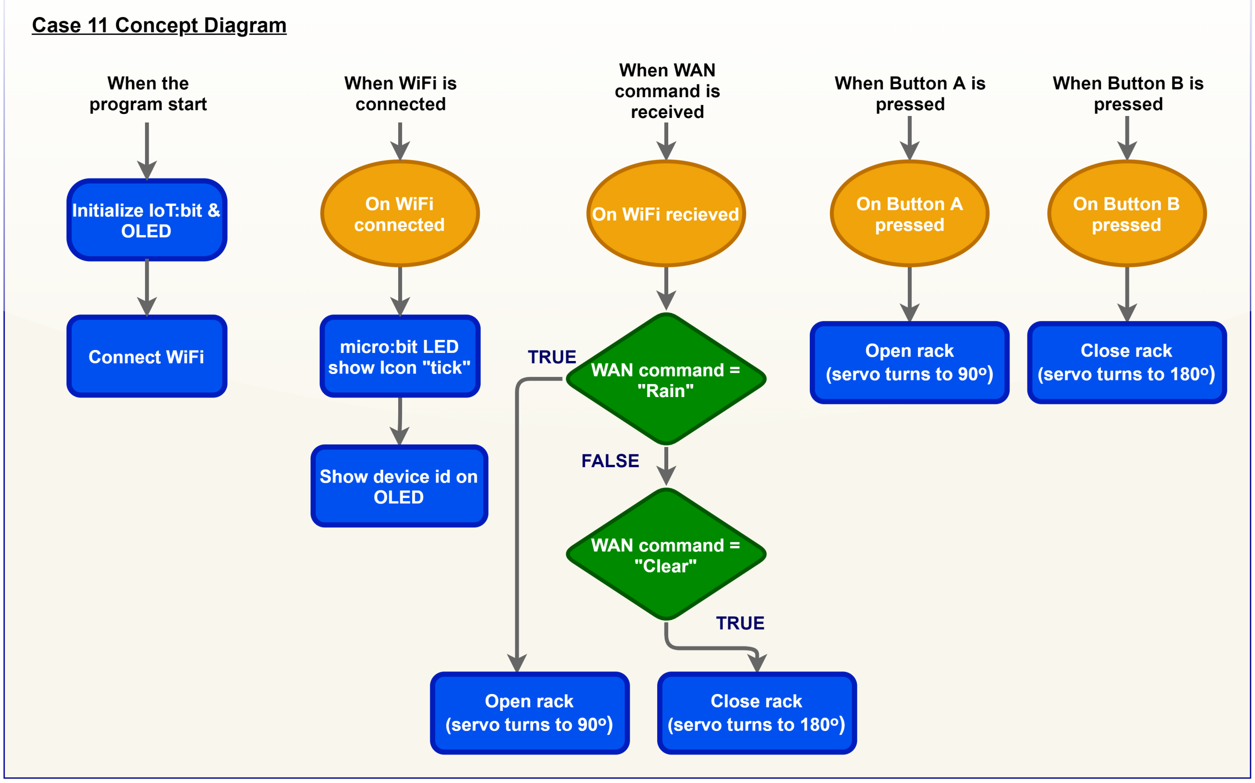


Goal

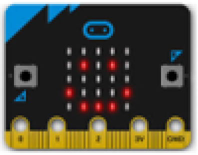
Make a smart roof garden clothes rack, once the weather condition is changed, the rack can be opened/ closed automatically.

Background

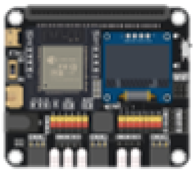
What is Roof garden clothes rack? People no long need to rush up to the roof when raining as the clothes rack can be closed automatically even when house owner is not at home. Roof garden clothes rack operation Micro:bit can receive WAN commands from WAN (eg. IFTTT), the the micro:bit will get the command name. If the command name is "Rain", the servo will turn to 90 and the rack will be opened. If the command name is "Clear", the servo will turn to 180 and the rack will be closed.



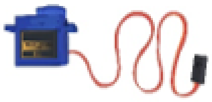
Part List



micro:bit (1)



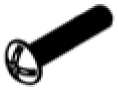
IoT:bit with OLED (1)



180° Servo (1)

3-pin module wire (1)

Servo M2*7.5mm sharp screw (1) (Included in servo pack)



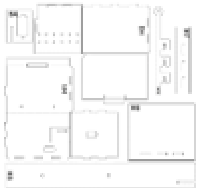
M2*10mm Screw (2)

M2 Nut (2)

M4*10mm Screw (4)

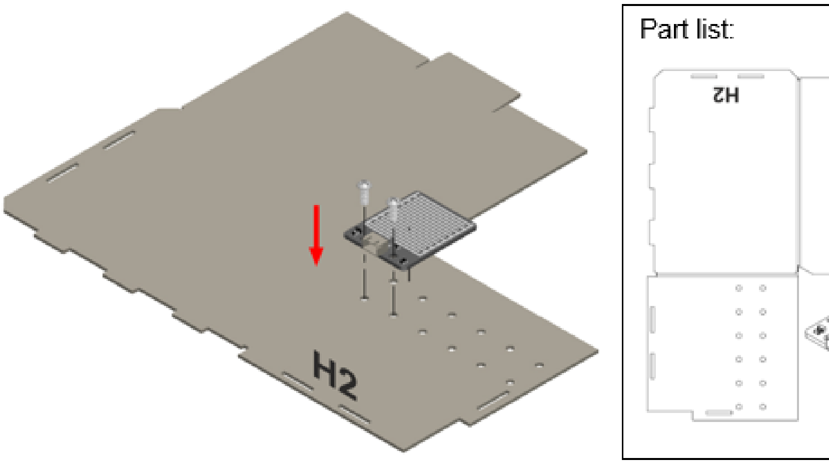
M4 Nut (4)

Screwdriver (1)

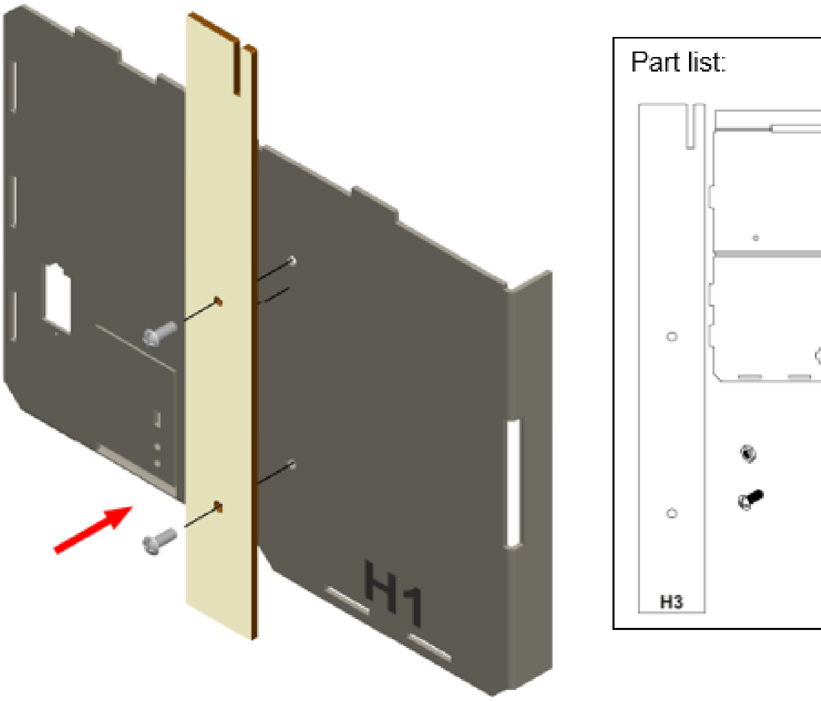


H1-H7 Model (1)

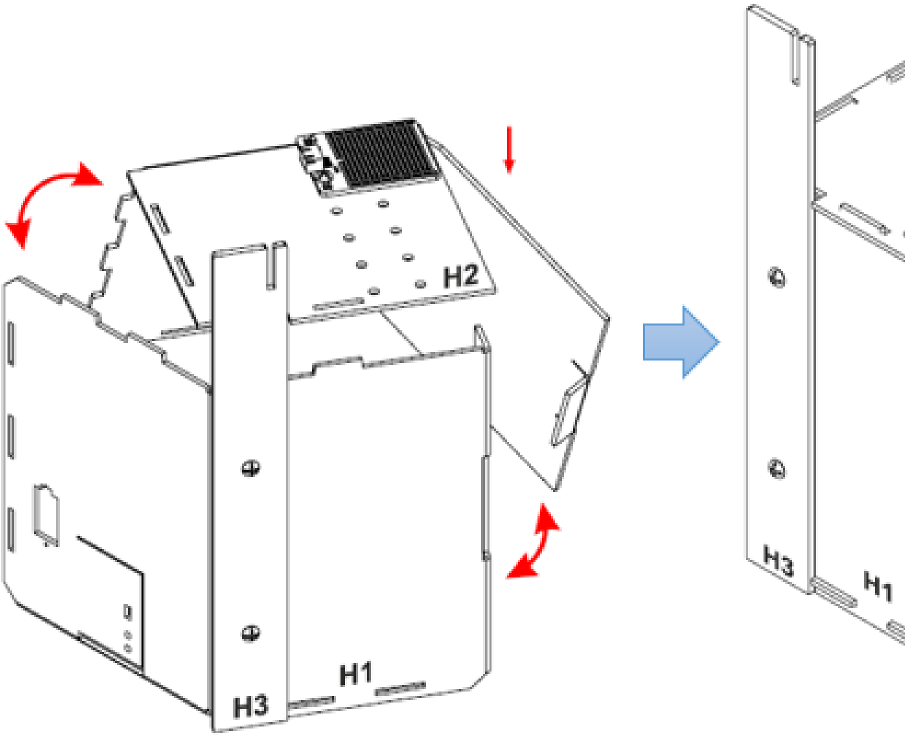
Assembly step



Step 1 Step 1. Attach the raindrop sensor to the model H2

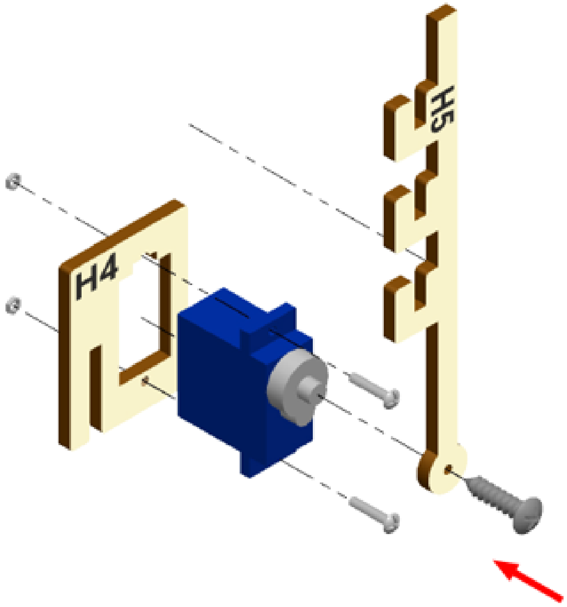


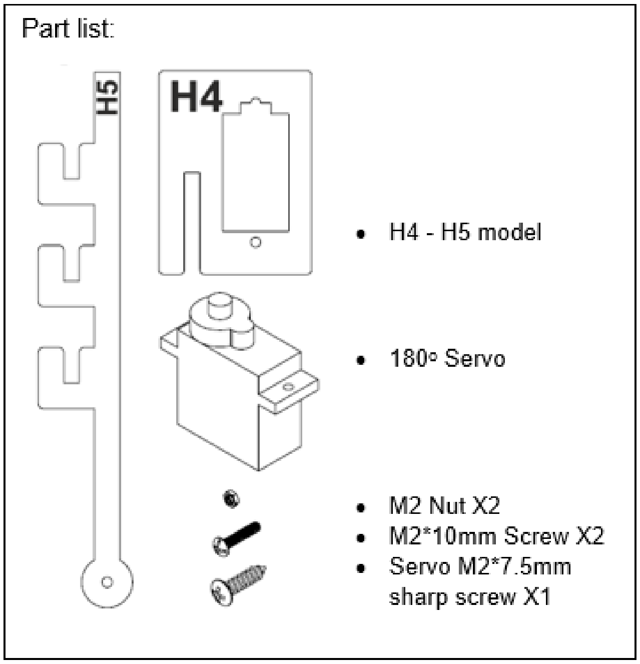
Step 2 Step 2. Attach the long stand (H3) to the model H1



Step 3 Step 3. Put together all the cardboard parts.

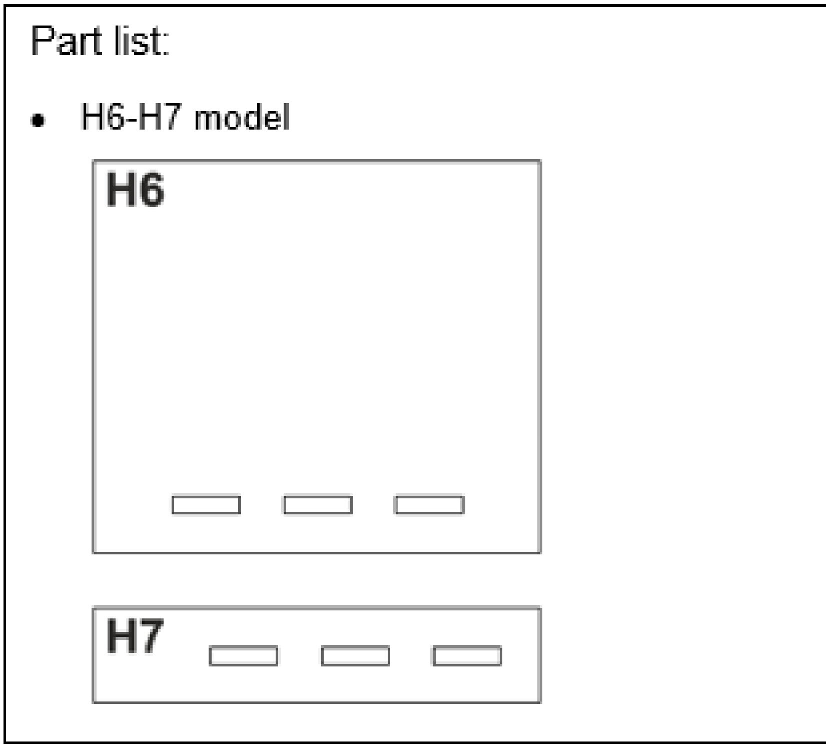
Step 4 Step 4. Screw the cloth rack (H5) onto the 180 servo using M2 screws



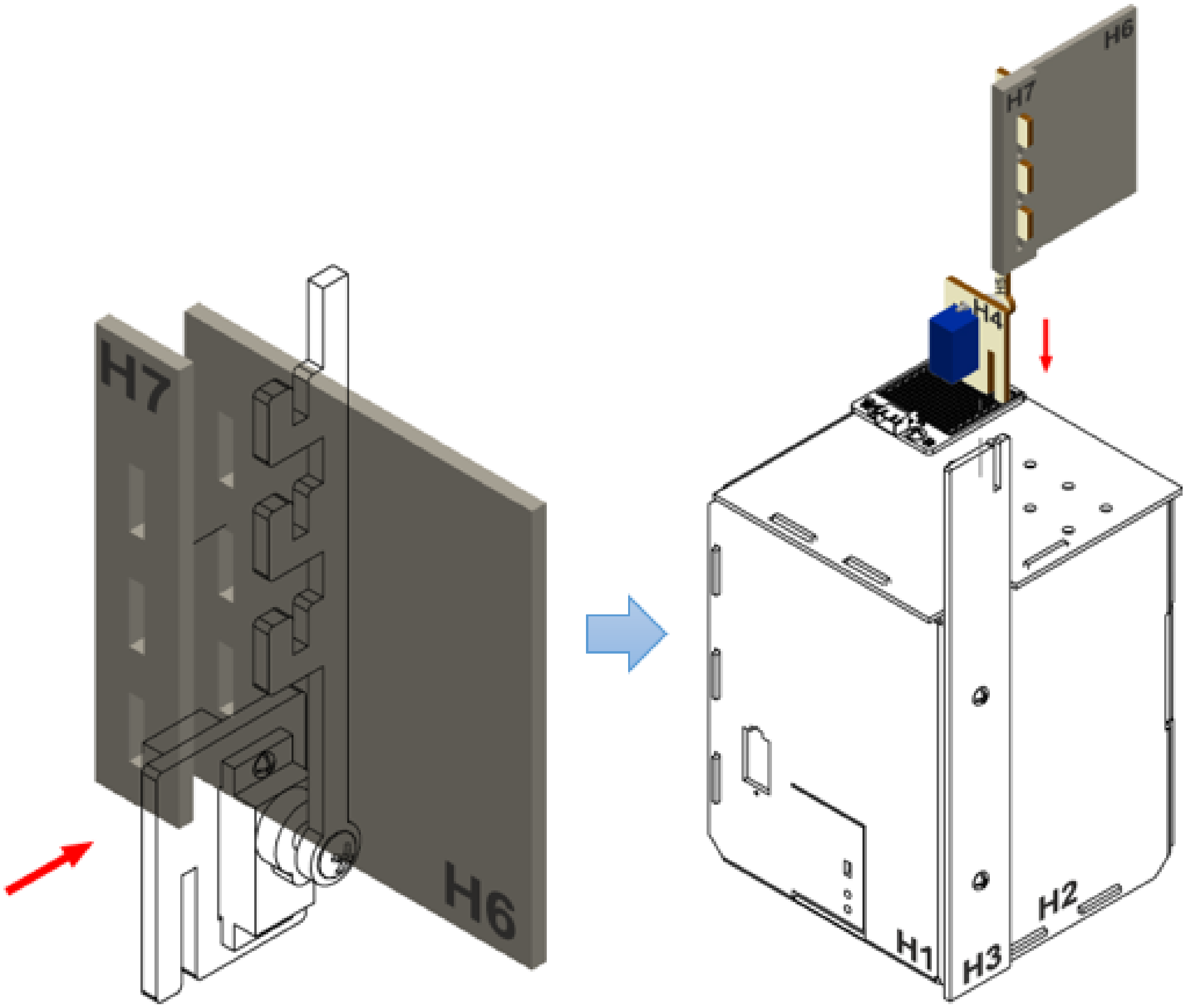


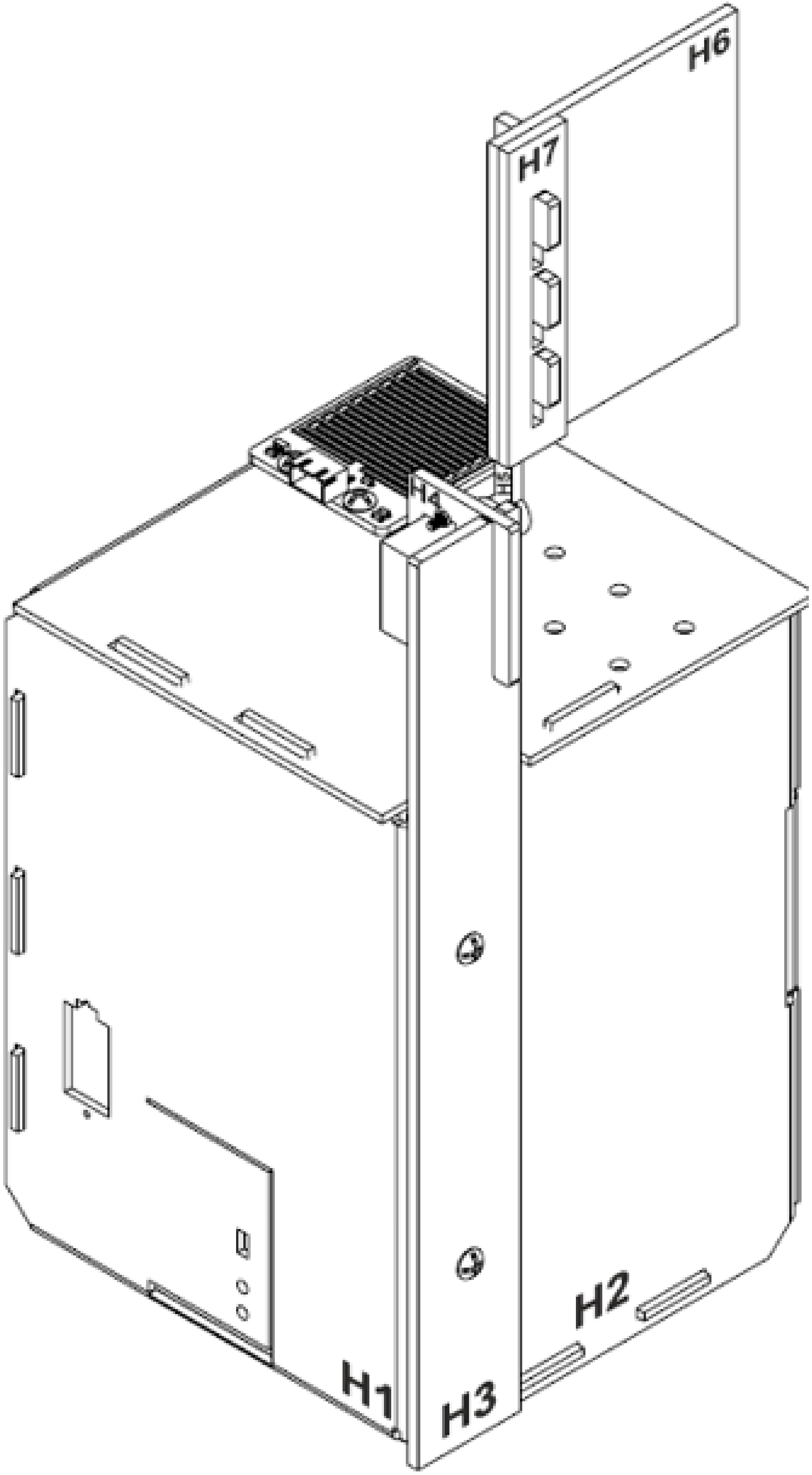
Part list:

- H4 - H5 model
- 180° Servo
- M2 Nut X2
- M2*10mm Screw X2
- Servo M2*7.5mm sharp screw X1



Step 5 Step 5. Put together the models H4-H5 and H6-H7.

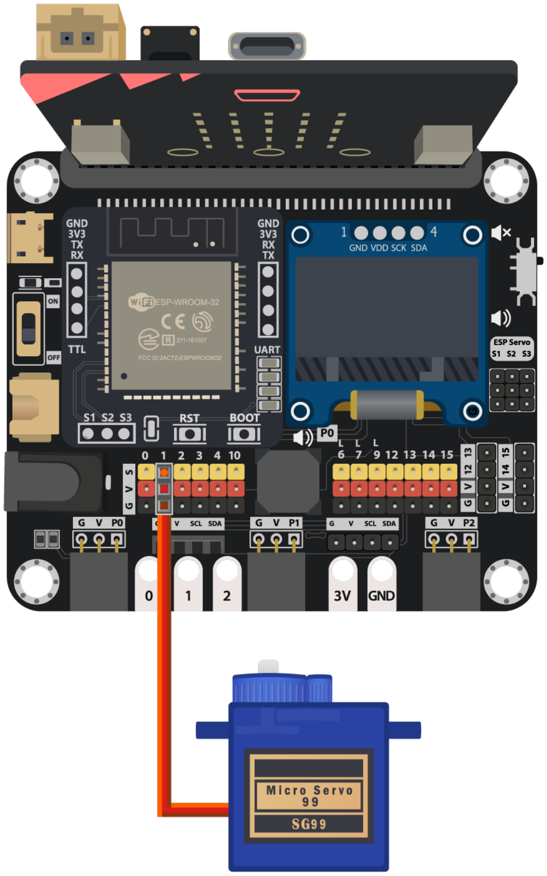

H7
H6
H6
H7
H4
H1
H3
H2



Step 6 Step 6. Assembly completed!

Hardware connect

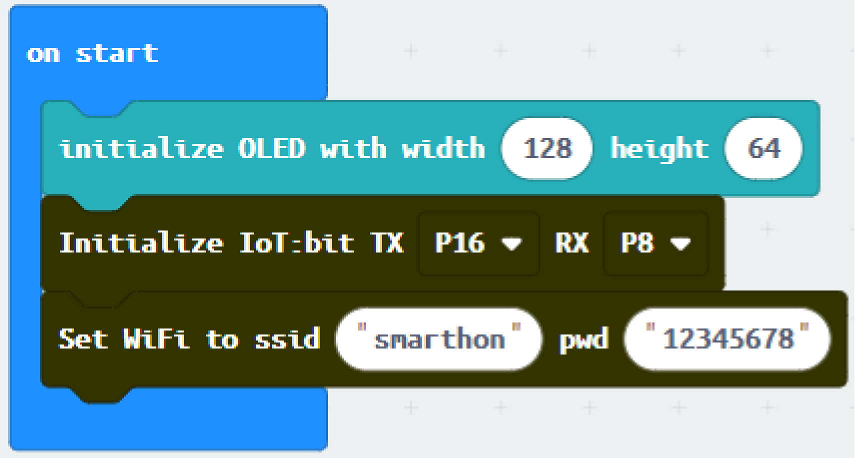
Connect the 180 Servo to P1 port of IoT:bit



Programming (MakeCode)

Step 1. Initialize OLED, IoT:bit and connect to WiFi

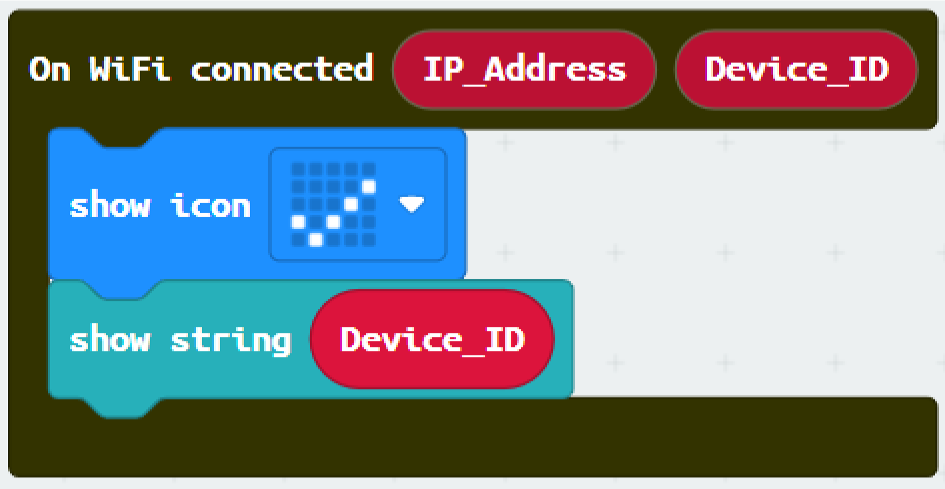
- Snap `Initialize OLED with width:128, height: 64` to `on start`
- Snap `Initialize IoT:bit TX P16 RX P8` from `IoT:bit` to `on start`
- Snap `Set Wi-Fi to ssid pwd` from `IoT:bit`
- Enter your Wi-Fi name and password. Here we set `smarthon` as `SSID` and `12345678` as `password`



```
on start
    initialize OLED with width 128 height 64
    Initialize IoT:bit TX P16 ▾ RX P8 ▾
    Set WiFi to ssid "smarthon" pwd "12345678"
```

Step 2. Show icon "tick" after WiFi connection

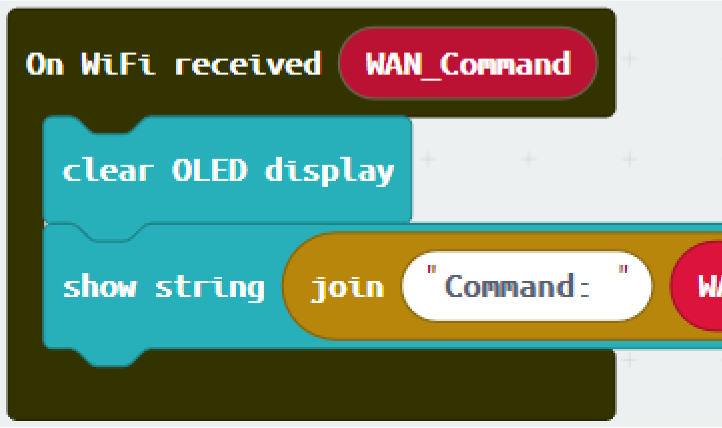
- Snap `show icon` from `basic` to `On WiFi connected` and select icon `tick`
- Draw the `Device ID` variable from `On WiFi connected` to the `show string` block placeholder



```
On WiFi connected IP_Address Device_ID
    show icon ▾
    show string Device_ID
```

Step 3. Receive WAN command

- Go to OLED
- Snap the `clear OLED display` to `On WiFi received` to avoid overlap
- Snap the `show string` to `On WiFi received`

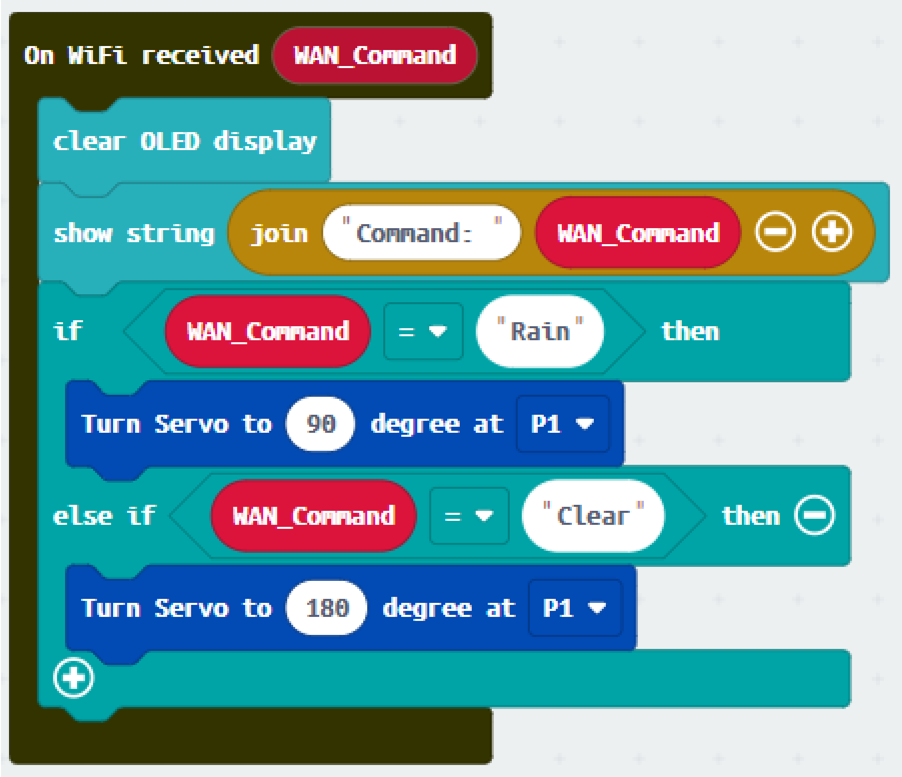


```
On WiFi received WAN_Command
    clear OLED display
    show string join "Command: " WA
```

- Draw the WAN_Command variable to show string placeholder

Step 4. Control rack open/close by WAN command

- Snap if-condition
- Set variable WAN_Command = Rain into if-condition
- Snap Turn Servo to ... degree from SmartCity > Output
- Set Servo degree to 90 at P1 (control the servo to open the rack)
- Set variable WAN_Command = Clear into else-if-condition
- Snap Turn Servo to ... degree from SmartCity > Output
- Set Servo degree to 180 at P1 (control the servo to close the rack)

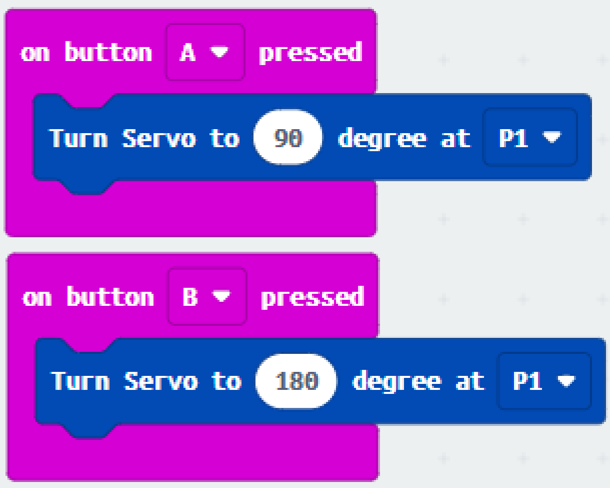


```
On WiFi received WAN_Command
    clear OLED display
    show string join "Command: " WAN_Command ⊖ ⊕
    if WAN_Command = ▾ "Rain" then
        Turn Servo to 90 degree at P1 ▾
    else if WAN_Command = ▾ "Clear" then ⊖
        Turn Servo to 180 degree at P1 ▾
    ⊕
```

Step 5. Press A to open clothes rack

- Snap on button ... pressed from Input, set button A

- Snap `Turn Servo to...` from `SmartCity` > `Output`

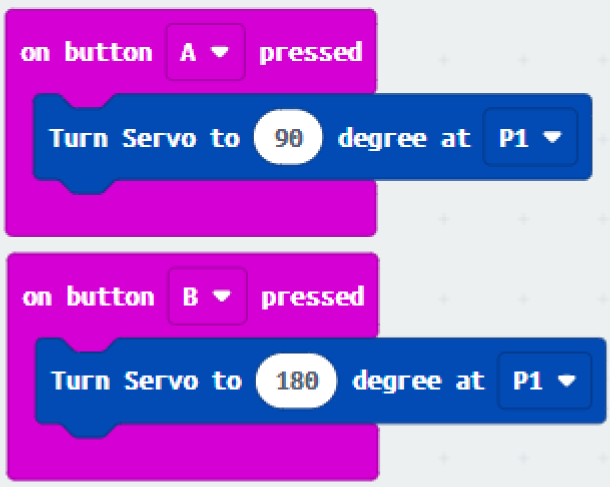


```
on button A ▾ pressed
    Turn Servo to (90) degree at P1 ▾

on button B ▾ pressed
    Turn Servo to (180) degree at P1 ▾
```

- `Set servo to 90 degree at P1` (open clothes rack)

Step 6. Press B to close clothes rack

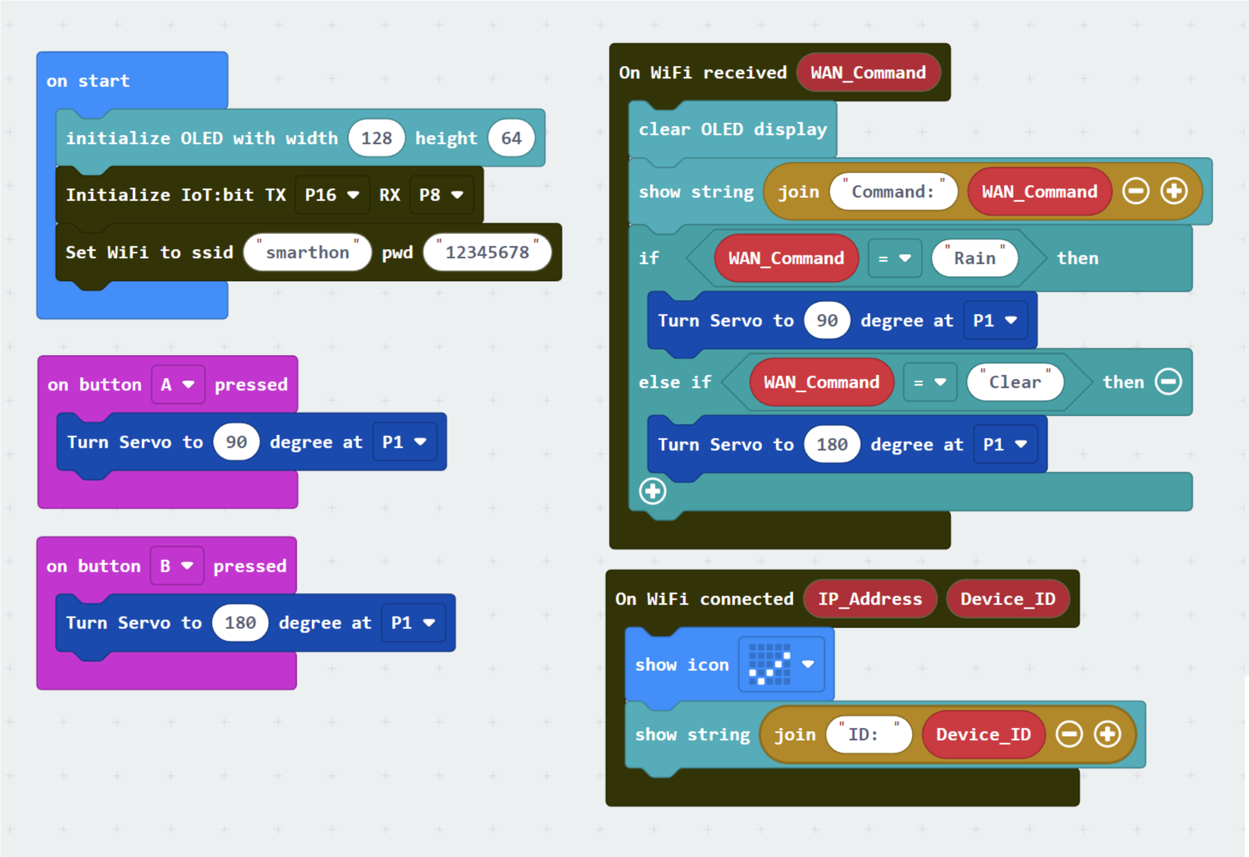
- Snap `on button ... pressed` from `Input`, set button `B`
- Snap `Turn Servo to...` from `SmartCity` > `Output`



```
on button A ▾ pressed
    Turn Servo to (90) degree at P1 ▾

on button B ▾ pressed
    Turn Servo to (180) degree at P1 ▾
```

- `Set servo to 180 degree at P1` (close clothes rack)

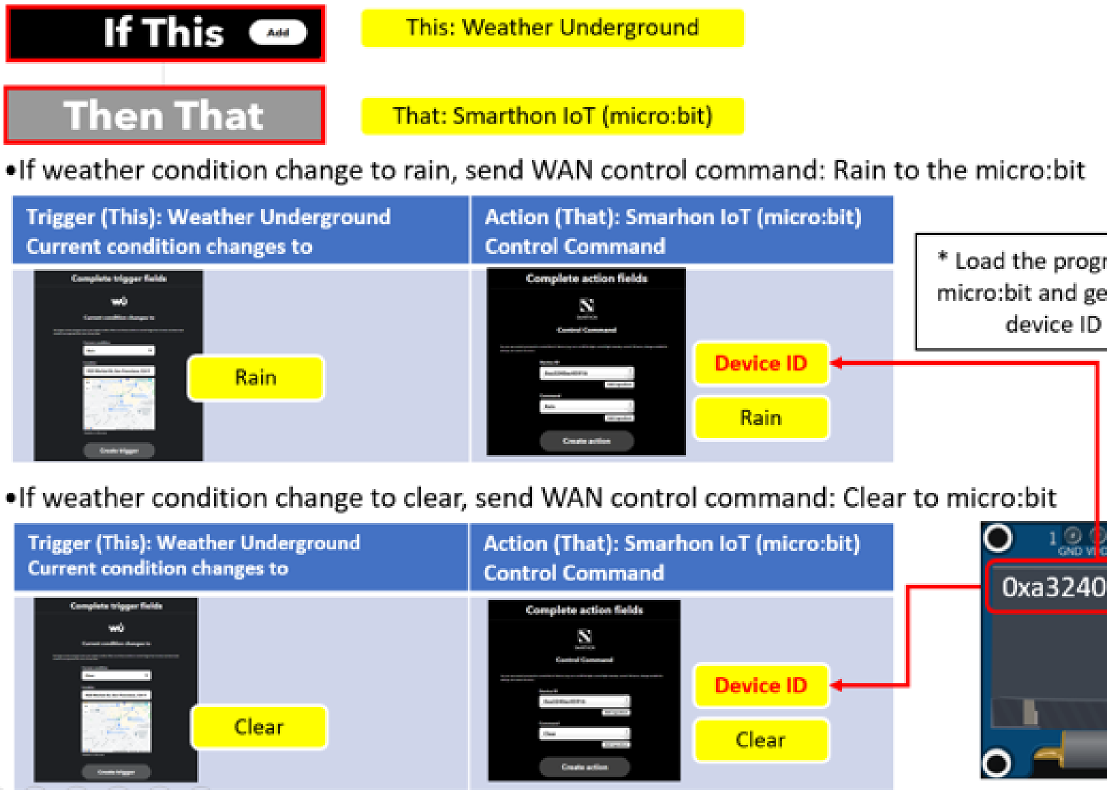
Full Solution MakeCode: https://makecode.microbit.org/_6KyfVPJEd10A



```
on start
    initialize OLED with width 128 height 64
    Initialize IoT:bit TX P16 RX P8
    Set WiFi to ssid "smarthon" pwd "12345678"

on button A pressed
    Turn Servo to 90 degree at P1

on button B pressed
    Turn Servo to 180 degree at P1

On WiFi received WAN_Command
    clear OLED display
    show string join "Command: " WAN_Command
    if WAN_Command = "Rain" then
        Turn Servo to 90 degree at P1
    else if WAN_Command = "Clear" then
        Turn Servo to 180 degree at P1

On WiFi connected IP_Address Device_ID
    show icon
    show string join "ID: " Device_ID
```

IoT (IFTTT)

* For the setting of IFTTT, please refer to “Chapter 4: Cloud Control micro:bit by IFTTT”



If This
Add
This: Weather Underground
Then That
That: Smarthon IoT (micro:bit)
Trigger (This): Weather Underground
Current condition changes to
Action (That): Smarhon IoT (micro:bit)
Control Command
Complete trigger fields
Create trigger
Complete action fields
Create action
Rain
Device ID
Rain
* Load the progr
micro:bit and ge
device ID
Trigger (This): Weather Underground
Current condition changes to
Action (That): Smarhon IoT (micro:bit)
Control Command
Clear
Device ID
Clear
0xa3240
GND

•If weather condition change to rain, send WAN control command: Rain to the micro:bit

•If weather condition change to clear, send WAN control command: Clear to micro:bit

Step 1. Create applet in IFTTT

Result

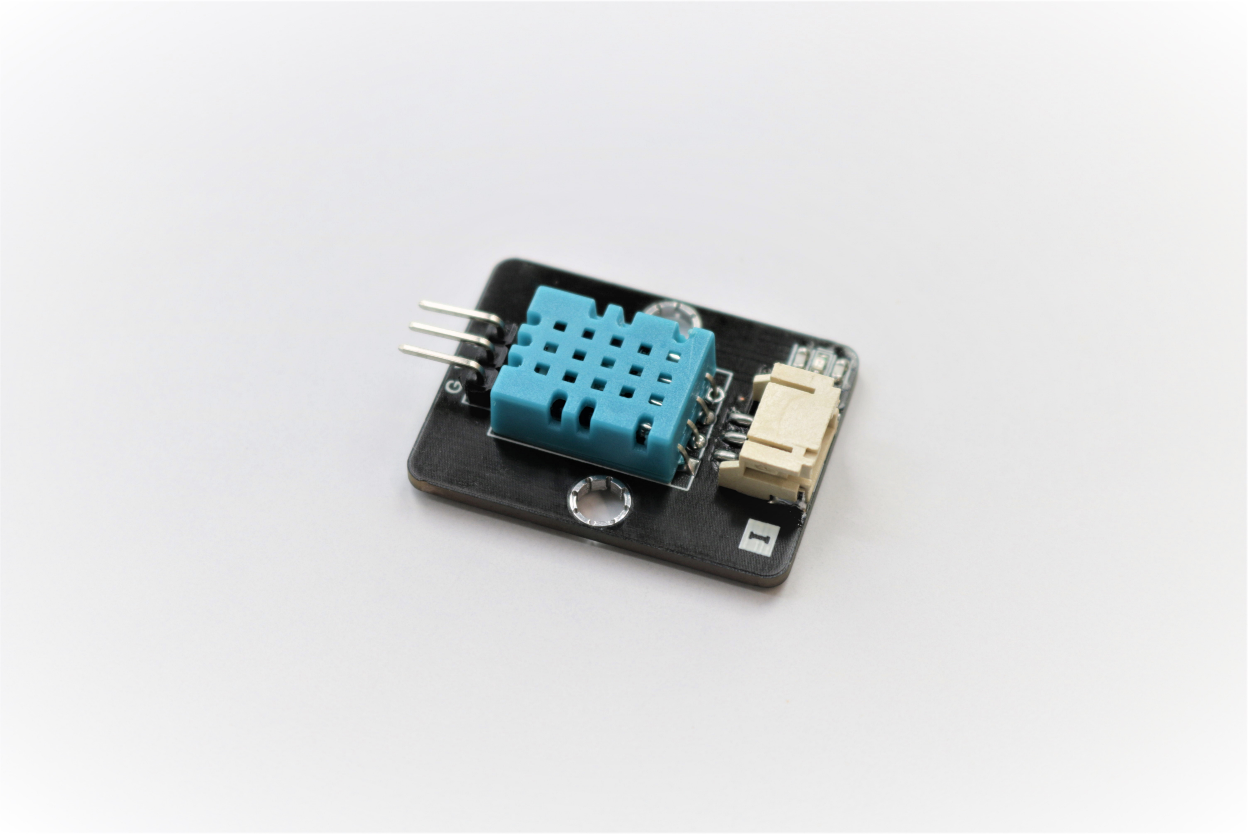
The micro:bit is controlled by IFTTT (trigger by weather open data). When the weather condition change to “Rain”, the cloth rack will be opened. When the weather condition change to “Clear”, the cloth rack will be closed.

Think

Q1. Can you control the clothes rack by other weather conditions? (e.g. sunlight)

1.2 Sensors and actuators

1.2.1 Temperature and Humidity sensor (DHT11)

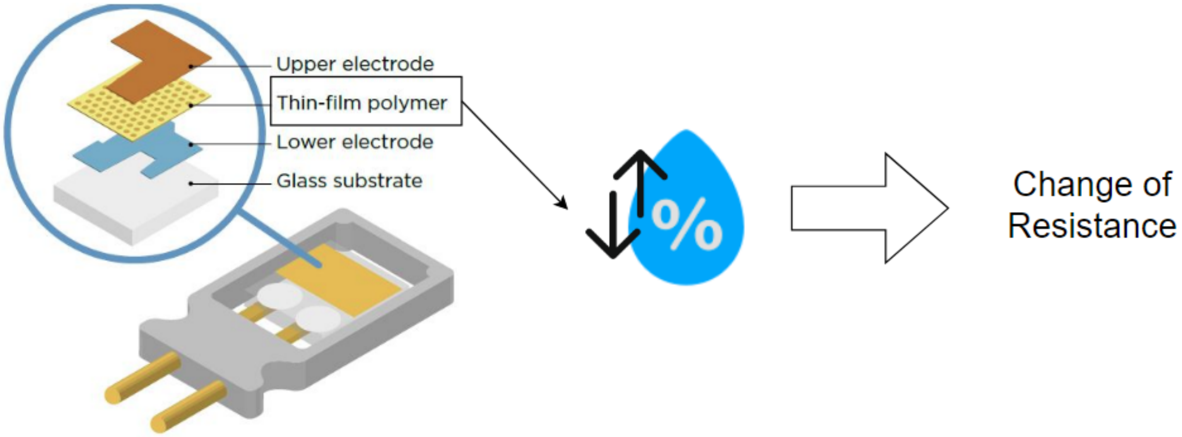


Introduction

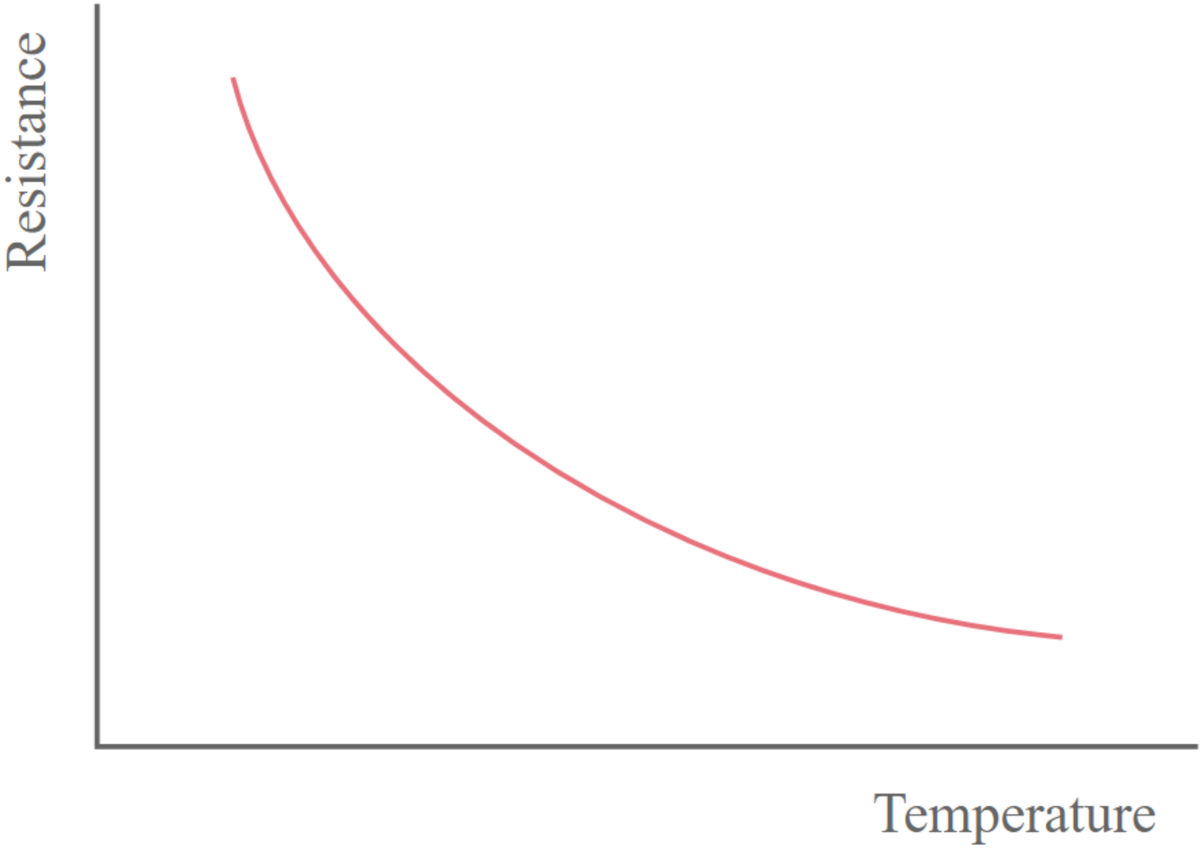
DHT11 is a commonly used sensor to detect the humidity and temperature in a lot of STEM project, it contain different advantage such low cost, easy to use and digital output.

The principle

The DHT11 contain two major components to detect humidity and temperature respectively. The first component is using the Conductivity to measure the environment humidity. When humidity is changing, the polymer between two electrodes will change the conductivity, causing the resistance between electrodes drop or rise. The chip on DHT11 will use it to determine correct humidity.



For the second component on DHT11, it is a Negative Temperature Coefficient(NTC) thermistor, when the temperature is rising, the resistance will decrease as follow. The chip will use that to determine the environment temperature.

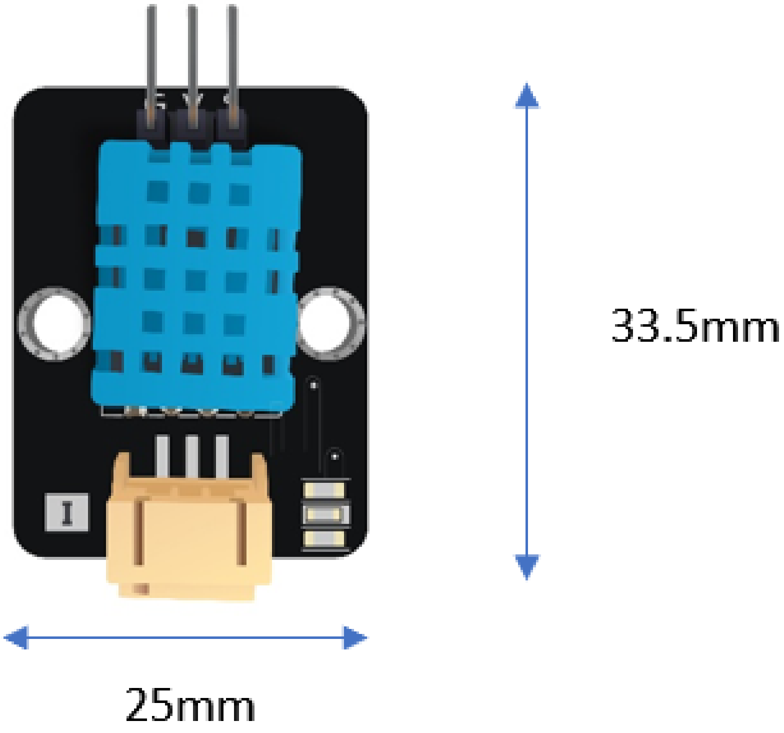


Specification

- Supply Voltage: 3.3V to 5V
- Range of Temperature reading: 0-50°C with +- 2°C accuracy
- Range of Humidity reading: 20%~80% with +- 5% accuracy
- Sampling Rate: 1Hz

Pinout Diagram

Outlook and Dimension

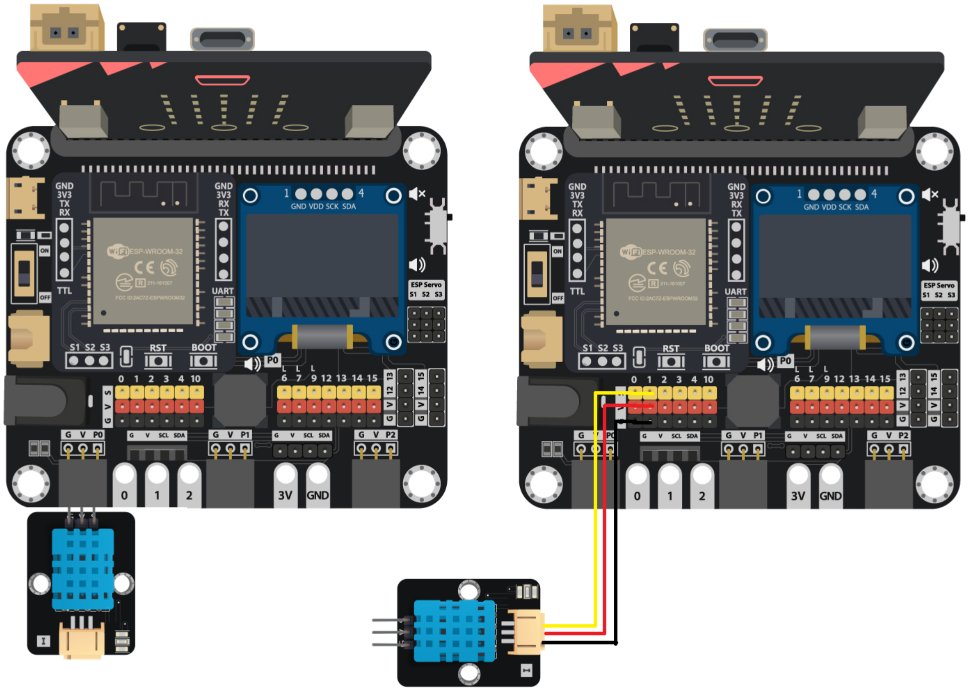


Size: 33.5mm X 25mm

Quick to Start/Sample

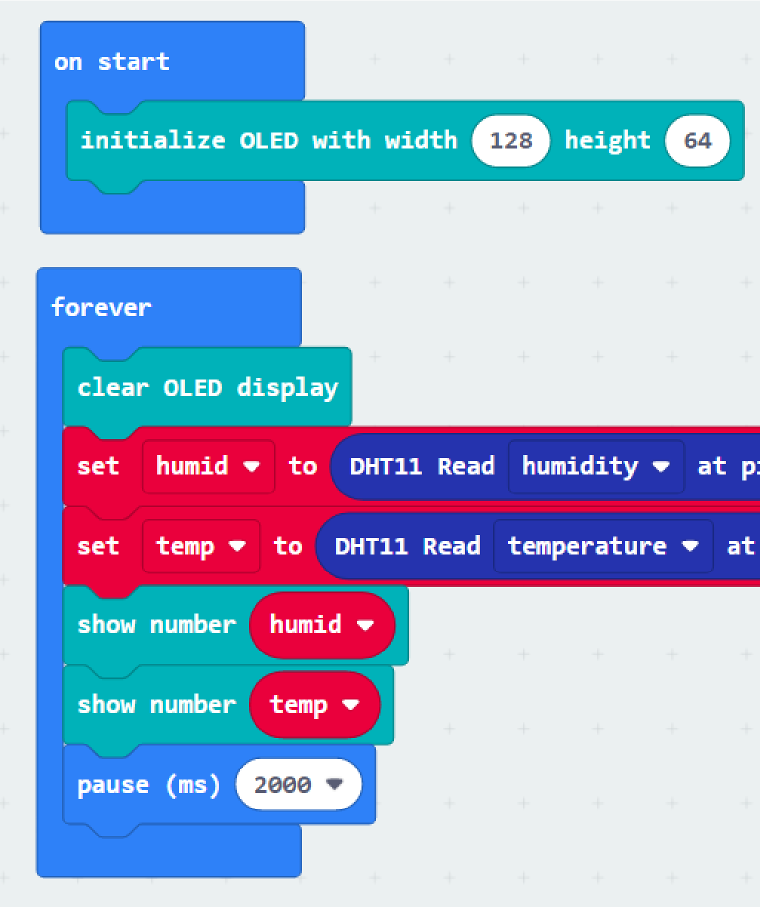
Best practice

- Connect the DHT11 to development board (direct plugin or using wire)



- Open Makecode, using the https://github.com/smarthon/pxt-smartcity PXT
- After Initial the OLED, setup two variables to store the reading value, then load the value from the DHT11 function to variable
- The DHT11 function have internal 1s delay for enough sampling time to ensure DHT11 working properly.

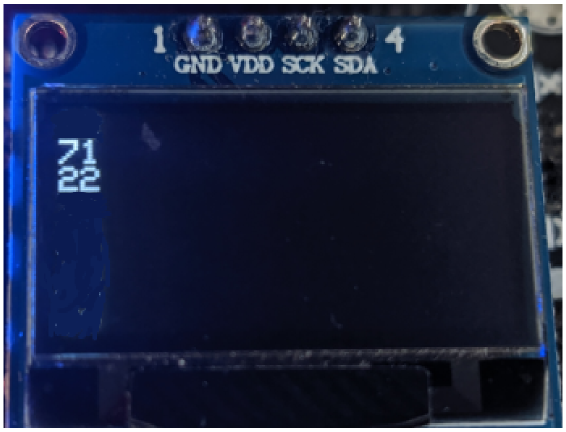
- Show the temperature and humidity variable on the OLED



```
on start
    initialize OLED with width 128 height 64

forever
    clear OLED display
    set humid ▾ to DHT11 Read humidity ▾ at p
    set temp ▾ to DHT11 Read temperature ▾ at
    show number humid ▾
    show number temp ▾
    pause (ms) 2000 ▾
```

Result

The Room Temperature and Humidity showing on the OLED screen



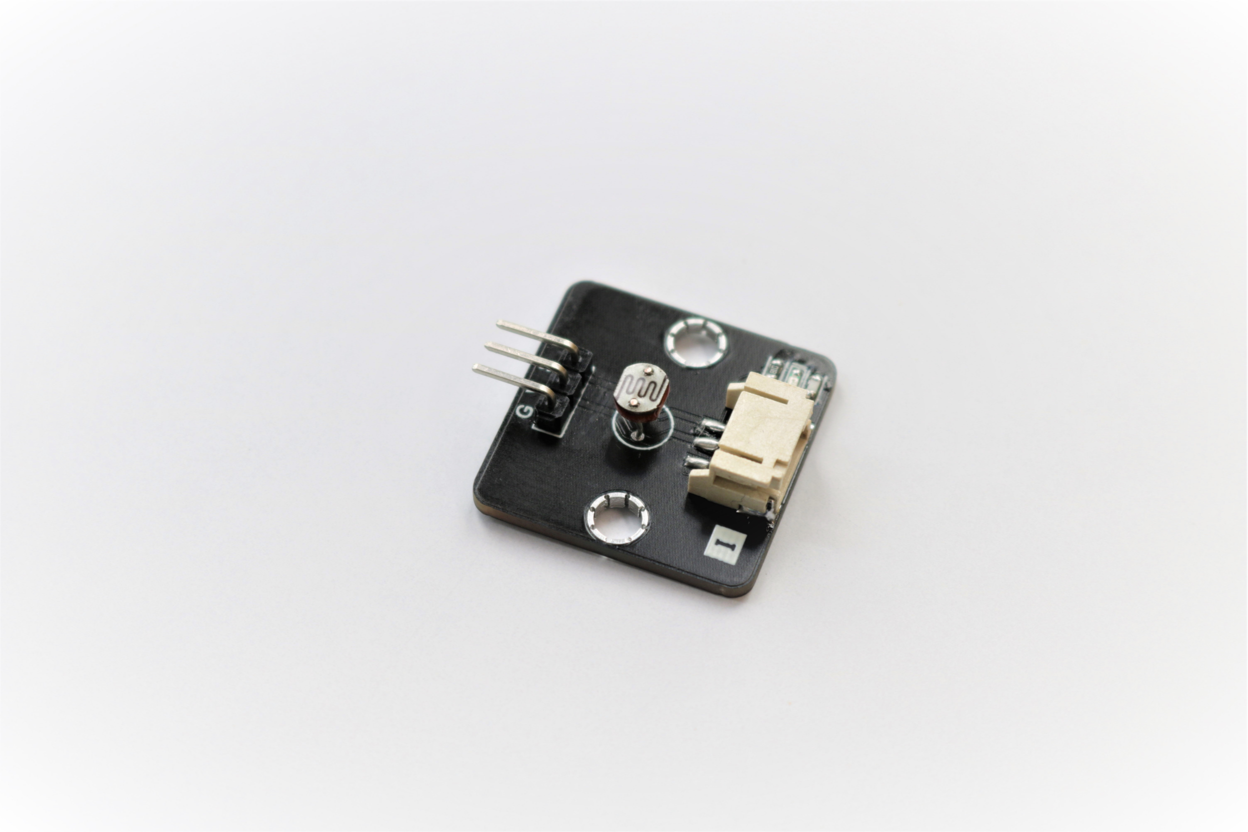
FAQ

Q: Why the termperature and humidity reaction on DHT11 have delay? A: The DHT11 sensor required some time to reach the accuracy value, form the official datasheet, the minimum response time of the Temperature and Humidity is 10 second and 6 second respectively.

Datasheet

DHT11_Datasheet

1.2.2 Light Sensor

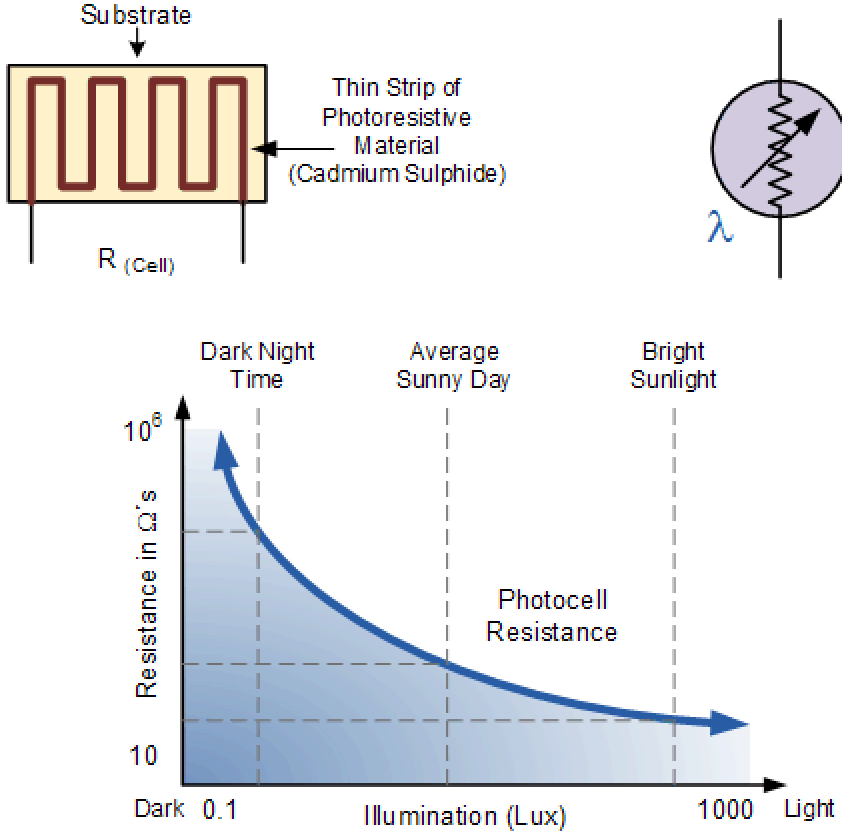


Introduction

The Light sensor use one photoresistor to detect the luminance of environment. The darker of the environment, the lower voltage will be provided.

The principle

The Light sensor also call as Light Dependent Resistor(LDR), the physical property will change under different light source as the name implied.



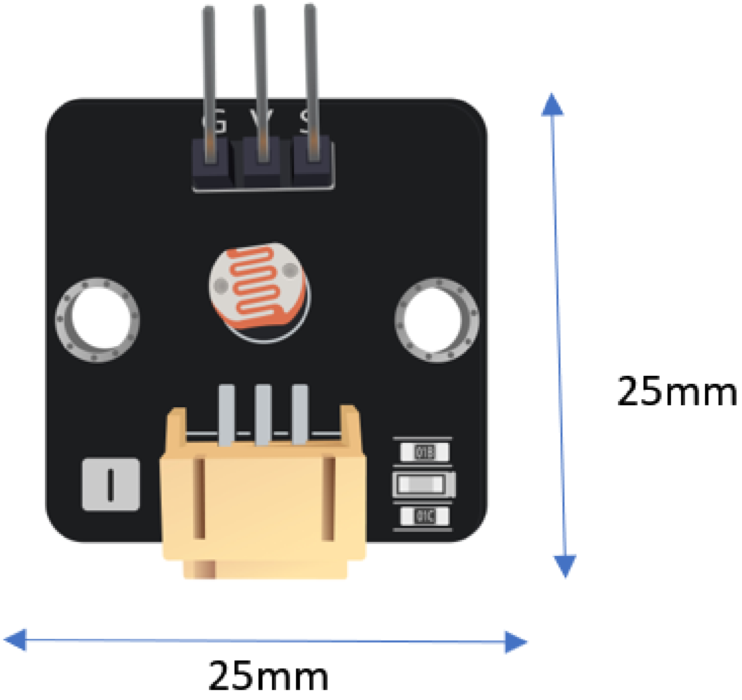
Depend on the model of different LDR, the resistance value and curves have difference, but all follow the principle: When the Illumination is increase, the resistance of the resistor will reduce. When the Illumination is decrease, the resistance of resistor will increase.

Specification

- Supply Voltage: 3.3V to 5V
- Interface: Analog
- Output Voltage Range: 1%~90% of supply Voltage
- Reading in the PXT(with Micro:bit): 1~100(0.04V~3.3V)

Pinout Diagram

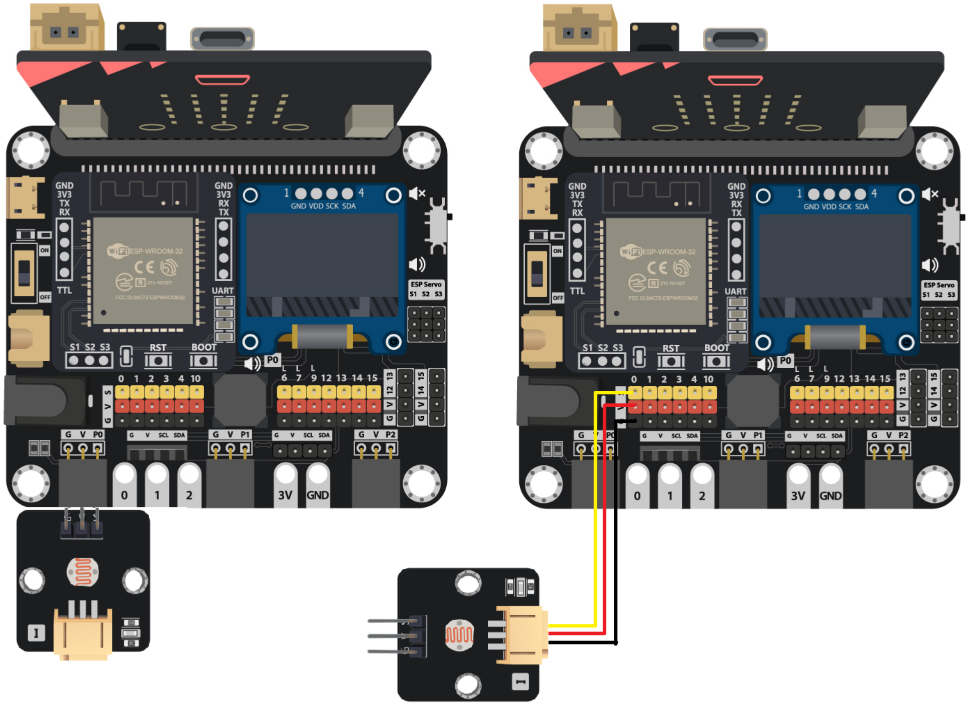
Outlook and Dimension



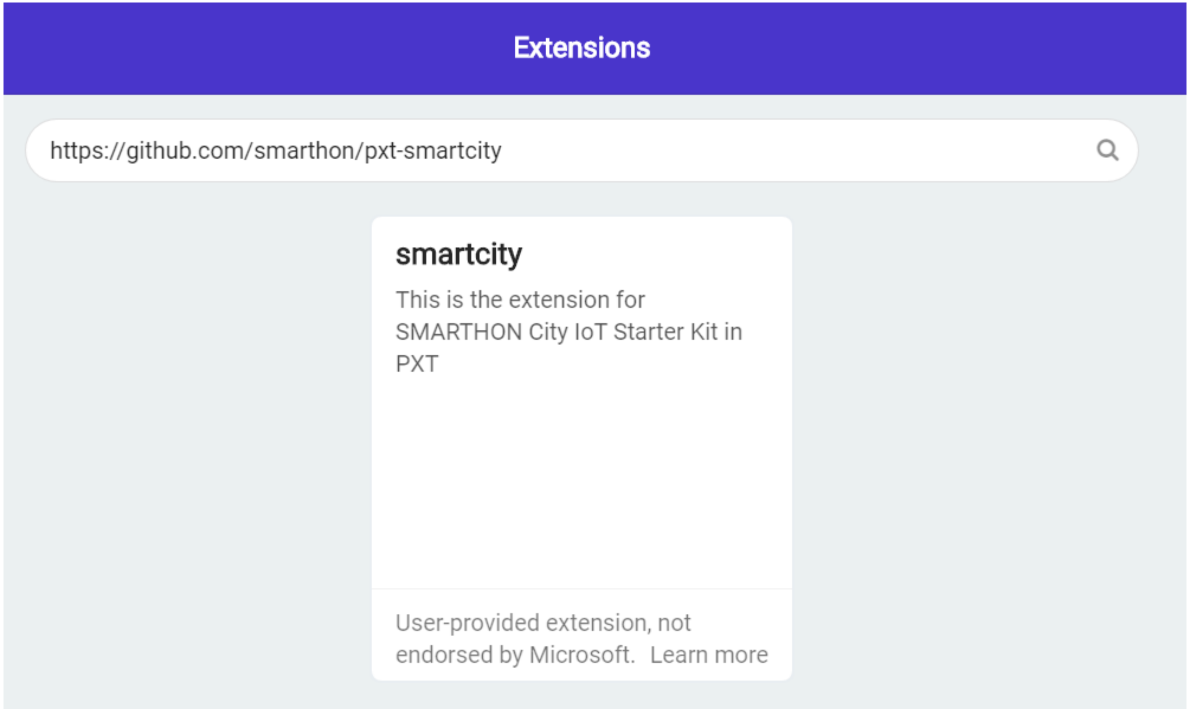
Size: 25mm X 25mm

Quick to Start/Sample

- Connect the sensor to development board (direct plugin or using wire)



- Open Makecode, using the https://github.com/smarthon/pxt-smartcity PXT



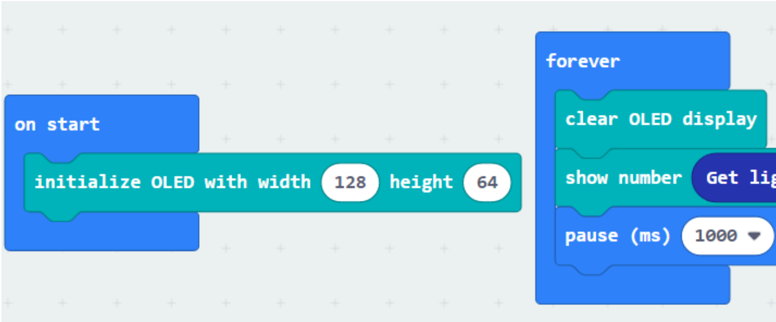
Extensions

https://github.com/smarthon/pxt-smartcity

smartcity

This is the extension for SMARTHON City IoT Starter Kit in PXT

User-provided extension, not endorsed by Microsoft. Learn more



```
on start
    initialize OLED with width 128 height 64

forever
    clear OLED display
    show number Get lig
    pause (ms) 1000
```

- Initial the OLED and show the reading of Voltage Output

The value will in range 0 to 100

Result

The reading of the luminance will show according to the environment.

FAQ

Q: Why the reading mostly stays in the middle range? A: Generally, using in a in-door environment (with electric light source) will have a middle range value. The close to limit value will only appear in the extreme cases, like under the sun/ close to light bulb, or stay in a full dark environment.

Datasheet

Photoresistor-5516-datasheet

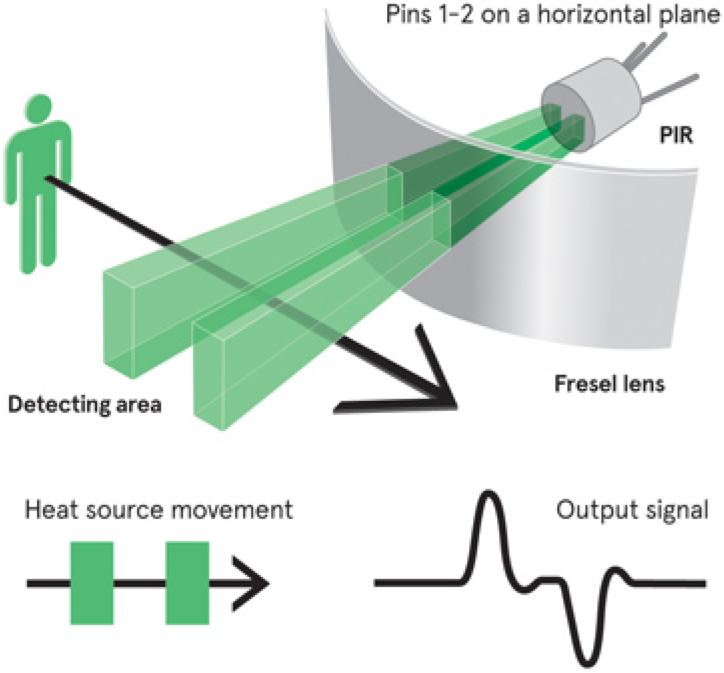
1.2.3 PIR Motion Sensor

Introduction

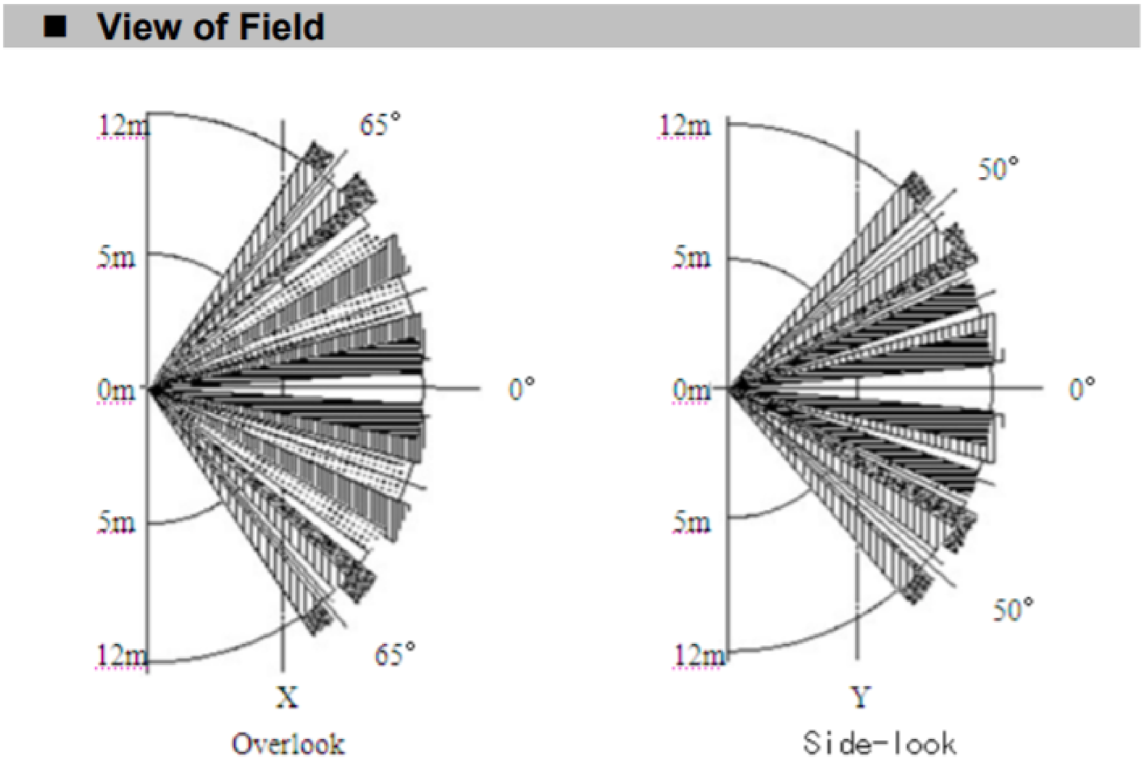
The PIR Motion Sensor use a pyroelectric infrared sensor to detect any moving object at front.When the infrared signal have rapid change, the sensor will return high to user, otherwise, return low.

The principle

The pyroelectric infrared sensor will keeping record the infrared signal inside target area. Human will emit the infrared light cause by the body temperature, when they pass thought the detection area, the signal received by sensor will have fluctuation which can determine the movement of object.



Generally, the maximum range of the detection is 5 to 12 meters , create a spherical sector with 65 degree from normal line.

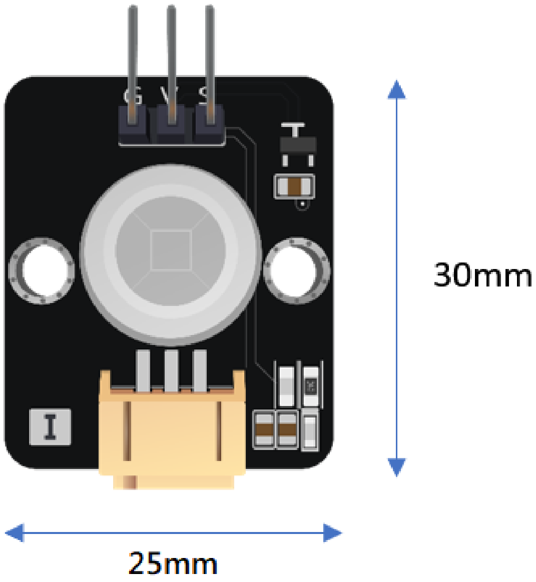


Specification

- Supply Voltage: 3.3V
- Interface: Analog
- Operation Temperature: -20 ~ 85°C
- Angle of detection: <65° from normal of center

Pinout Diagram

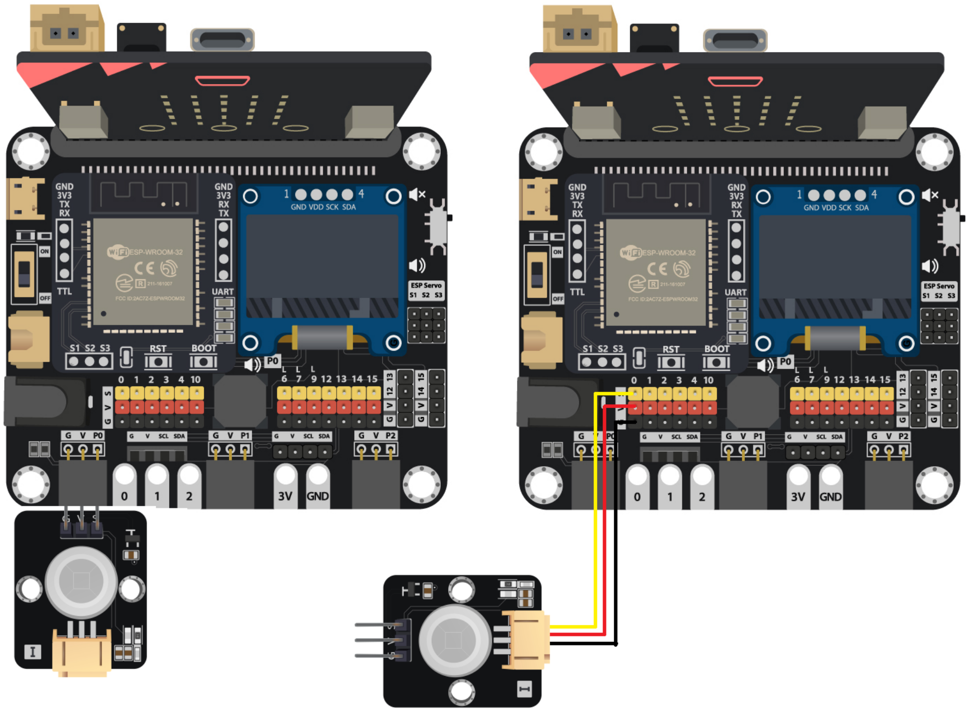
Outlook and Dimension



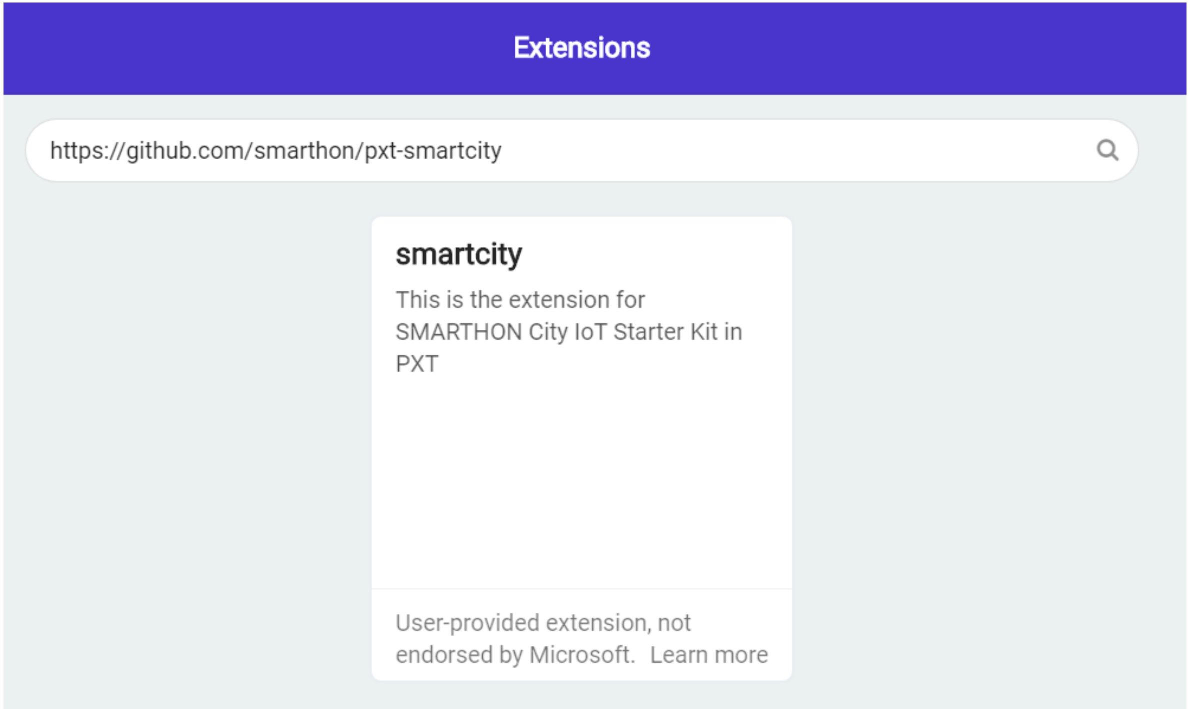
Size: 25mm X 30mm

Quick to Start/Sample

- Connect the module to the development board (direct plugin or using wire)



- Open Makecode, using the https://github.com/smarthon/pxt-smartcity PXT



Extensions

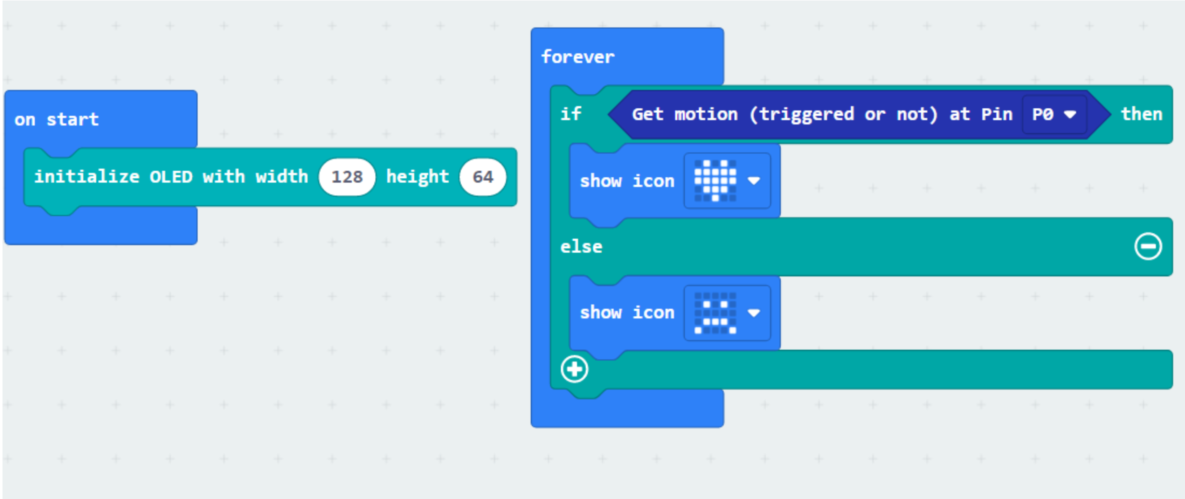
https://github.com/smarthon/pxt-smartcity

smartcity

This is the extension for SMARTHON City IoT Starter Kit in PXT

User-provided extension, not endorsed by Microsoft. Learn more

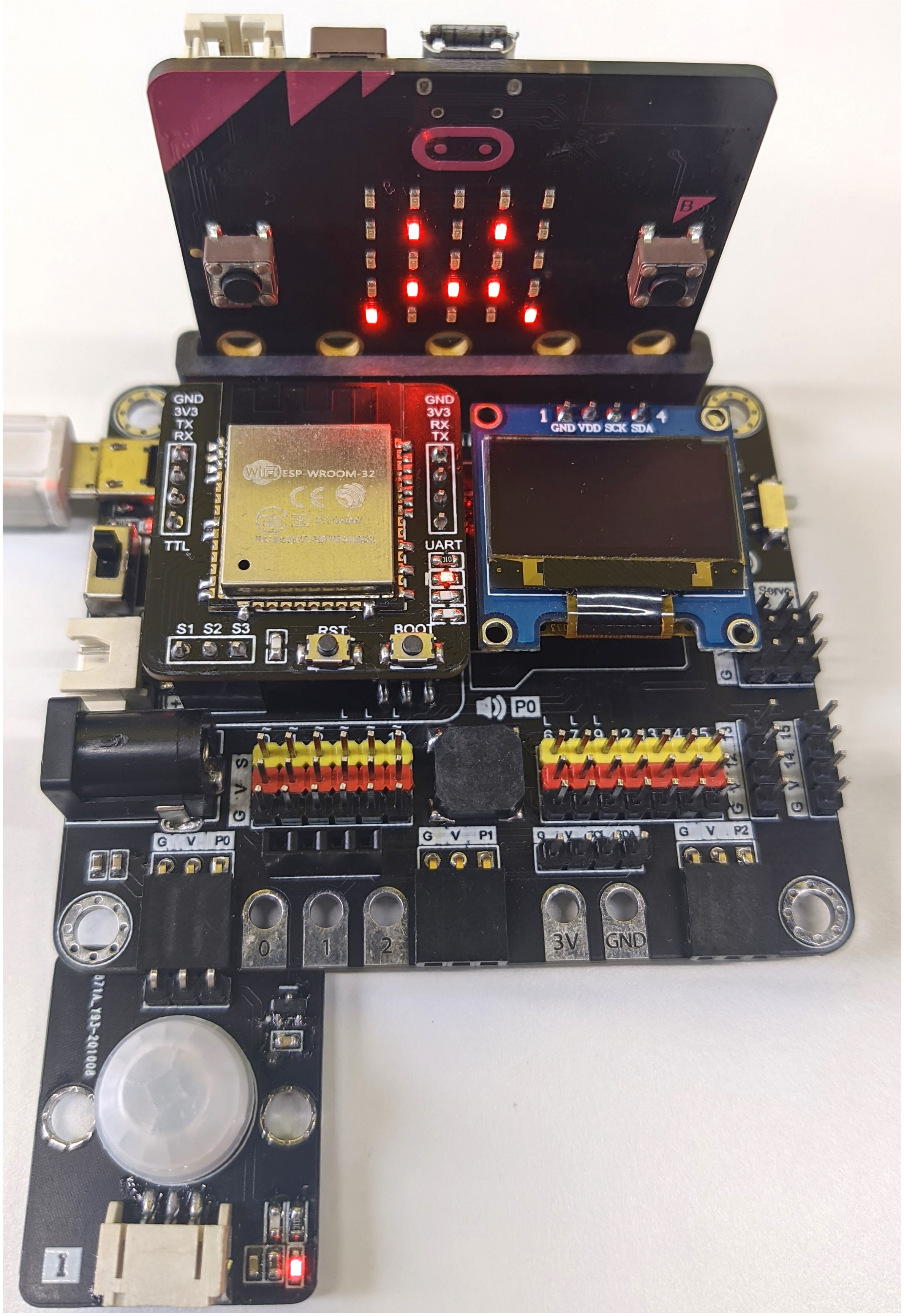
- Use to PXT library to read the detection result, and make the LEDs change according the result



```
on start
    initialize OLED with width 128 height 64

forever
    if Get motion (triggered or not) at Pin P0 then
        show icon
    else
        show icon
```

Result

The LEDs on the micro:bit will change when hand is over the sensor


GND
3V3
TX
RX
TTL
ESP-WROOM-32
GND
3V3
RX
TX
UART
S1 S2 S3
RST
BOOT
1 GND VDD SCK SDA 4
P0
G V P0
G V P1
G V P2
0
1
2
3V
GND

Datasheet

PIR-am312-datasheet

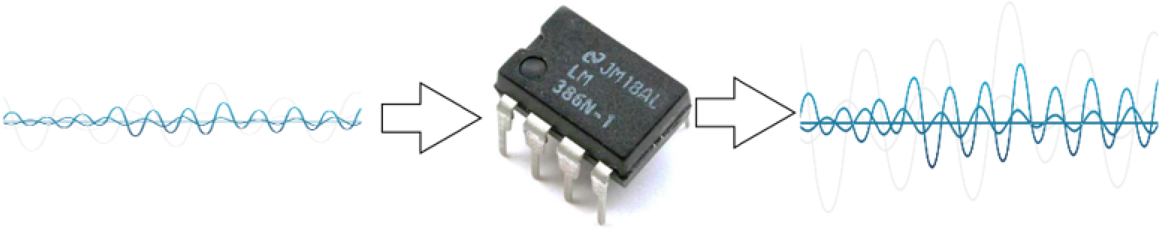
1.2.4 Noise Sensor

Introduction

Noise sensor use a microphone with an amplifier to detect the sound level of the environment. It will capture the original sound wave, enlarge, and reproduce it in voltage form. When using the sensor to calculate the sound level, use the amplitude to decide the result.

The principle

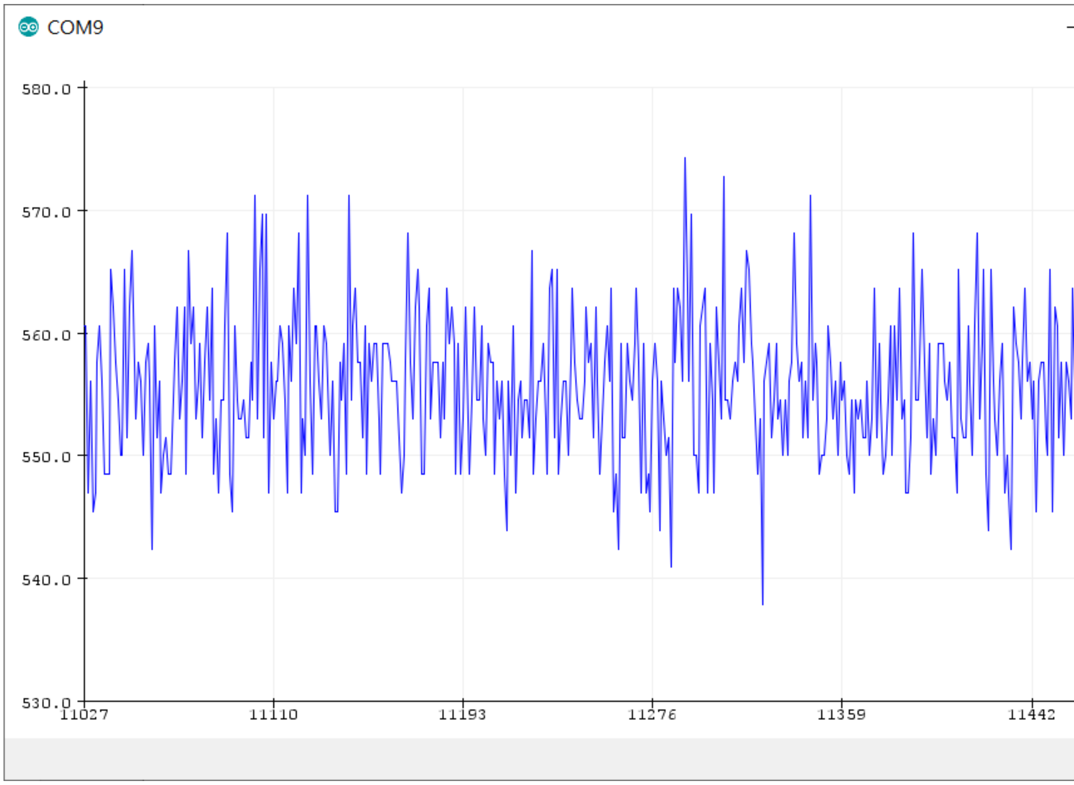
The Noise Sensor is a combine of the microphone and amplifier, the microphone will capture the raw signal form the environment and let the amplifier chip lm386 to produce the amplified signal to the output.

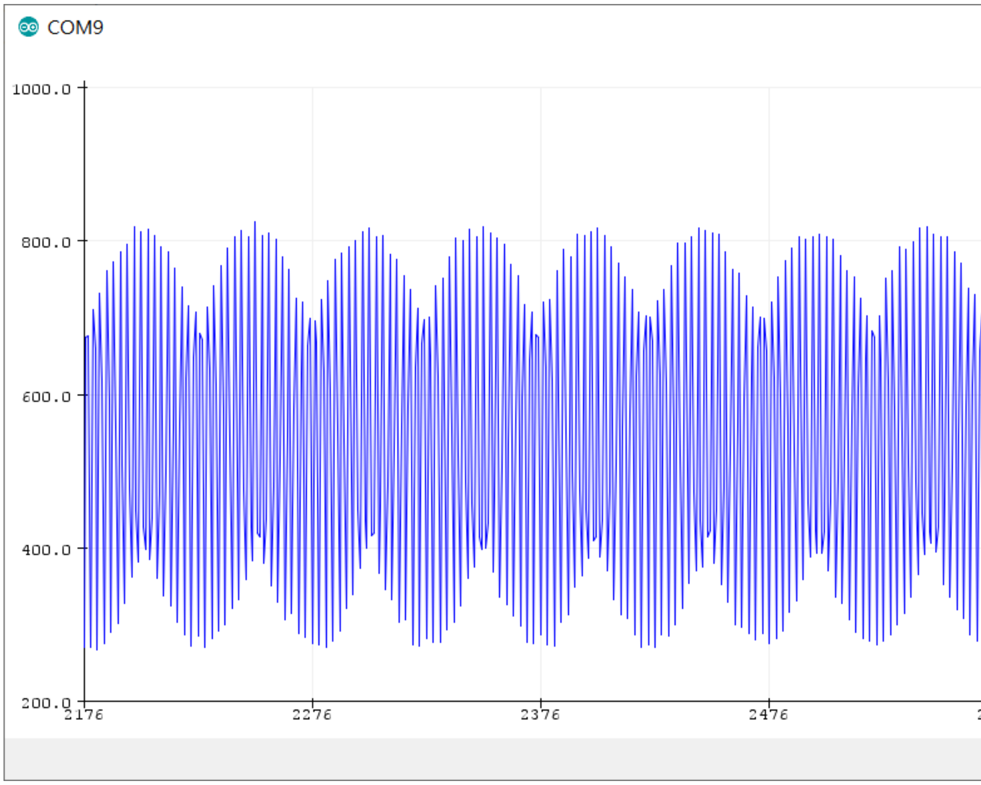


Specification

- Supply voltage: 3.3V – 5V
- Interface: Analog
- Output Range: 0~3.5V

Voltage when silent: (1.74~1.83V)

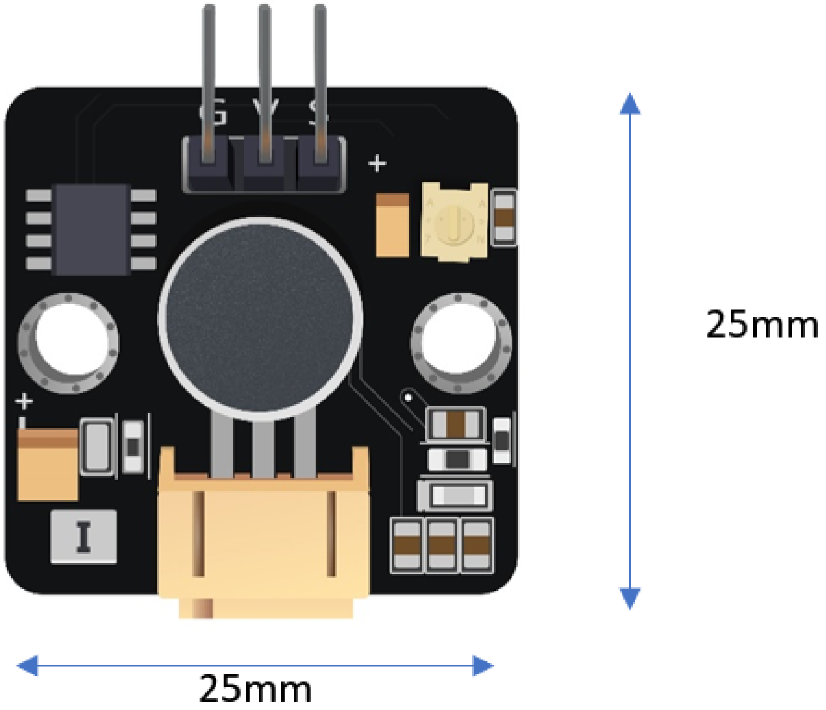




Voltage Under 440Hz sound:(0.98 ~ 2.58V)

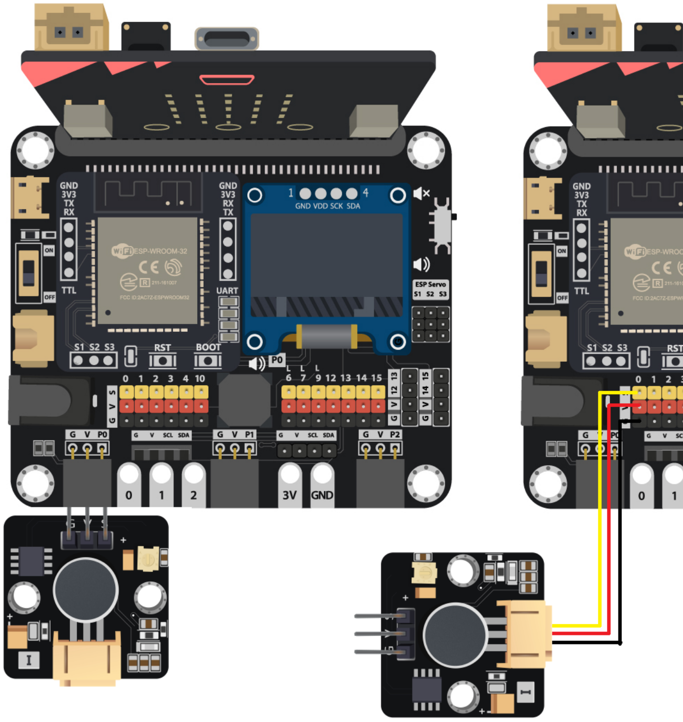
Pinout Diagram

Outlook and Dimension

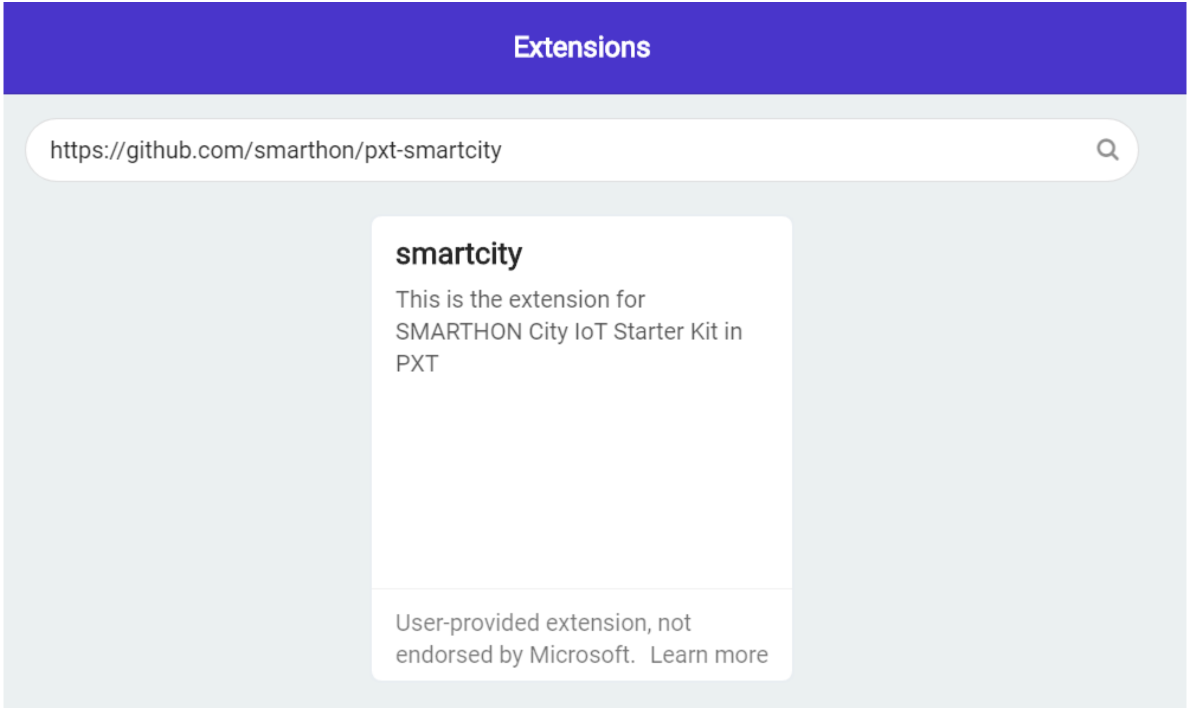


Size: 25mm X 25mm

Quick to Start/Sample



- Connect the sensor to development board (using wire)
- Open Makecode, using the https://github.com/smarthon/pxt-smartcity PXT


Extensions

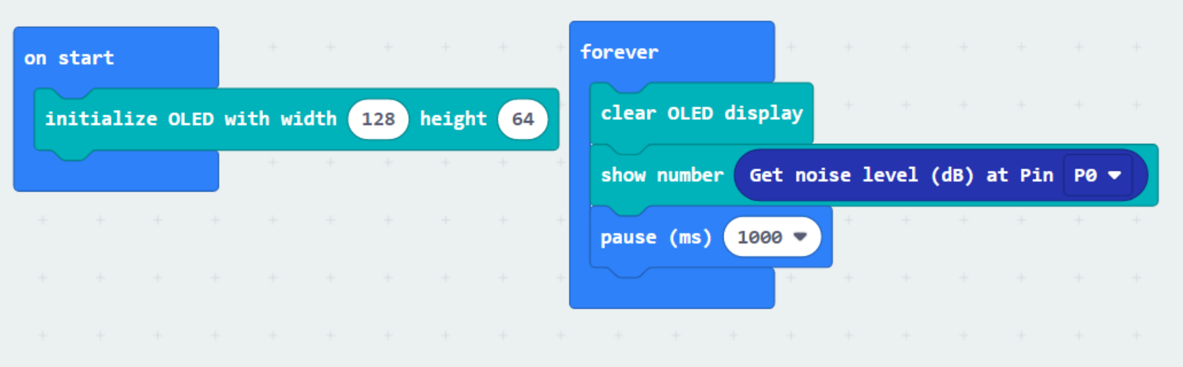
https://github.com/smarthon/pxt-smartcity

smartcity

This is the extension for SMARTHON City IoT Starter Kit in PXT

User-provided extension, not endorsed by Microsoft. Learn more

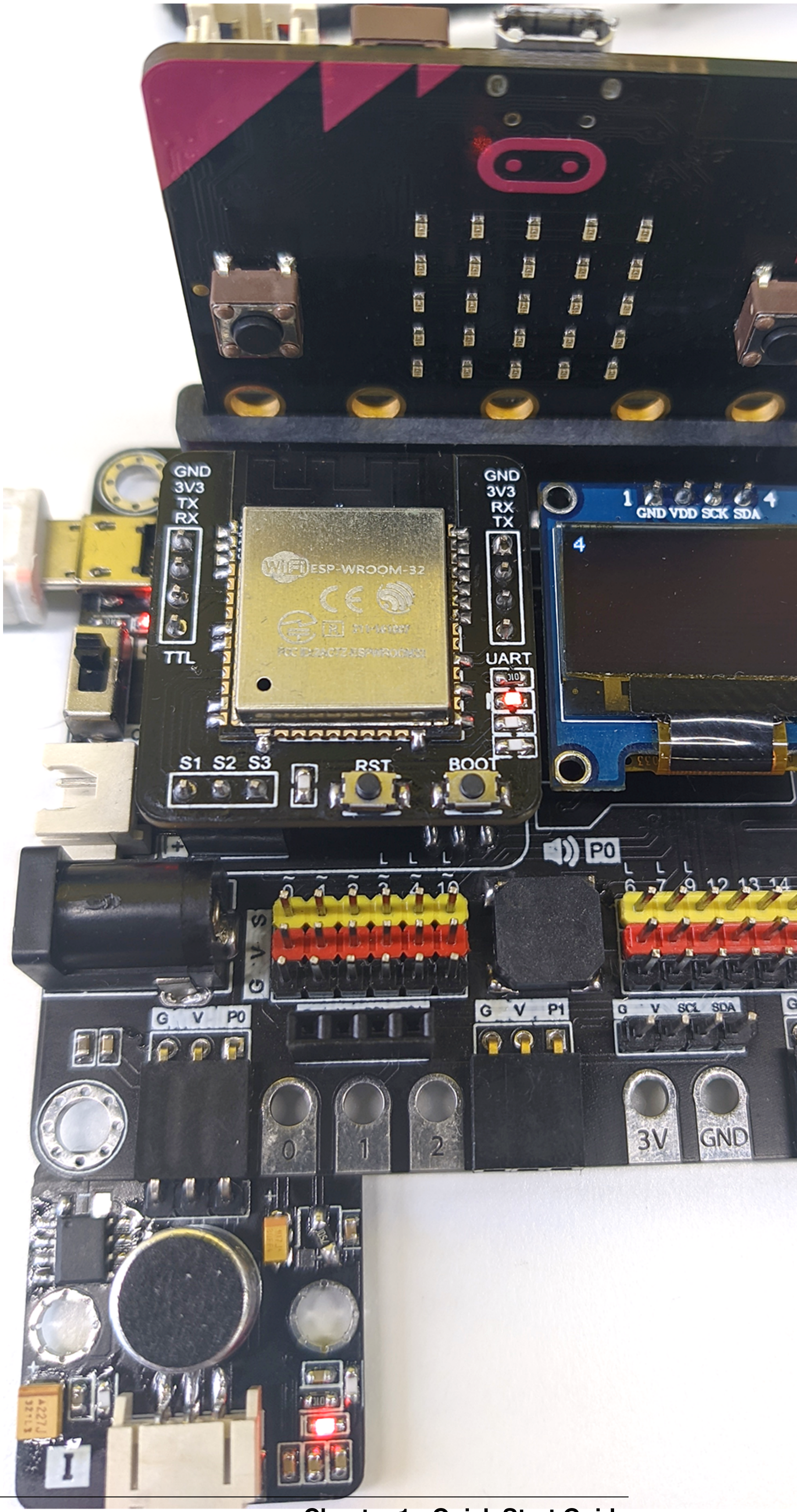
- Initial the OLED screen and show the reading of the noise level.



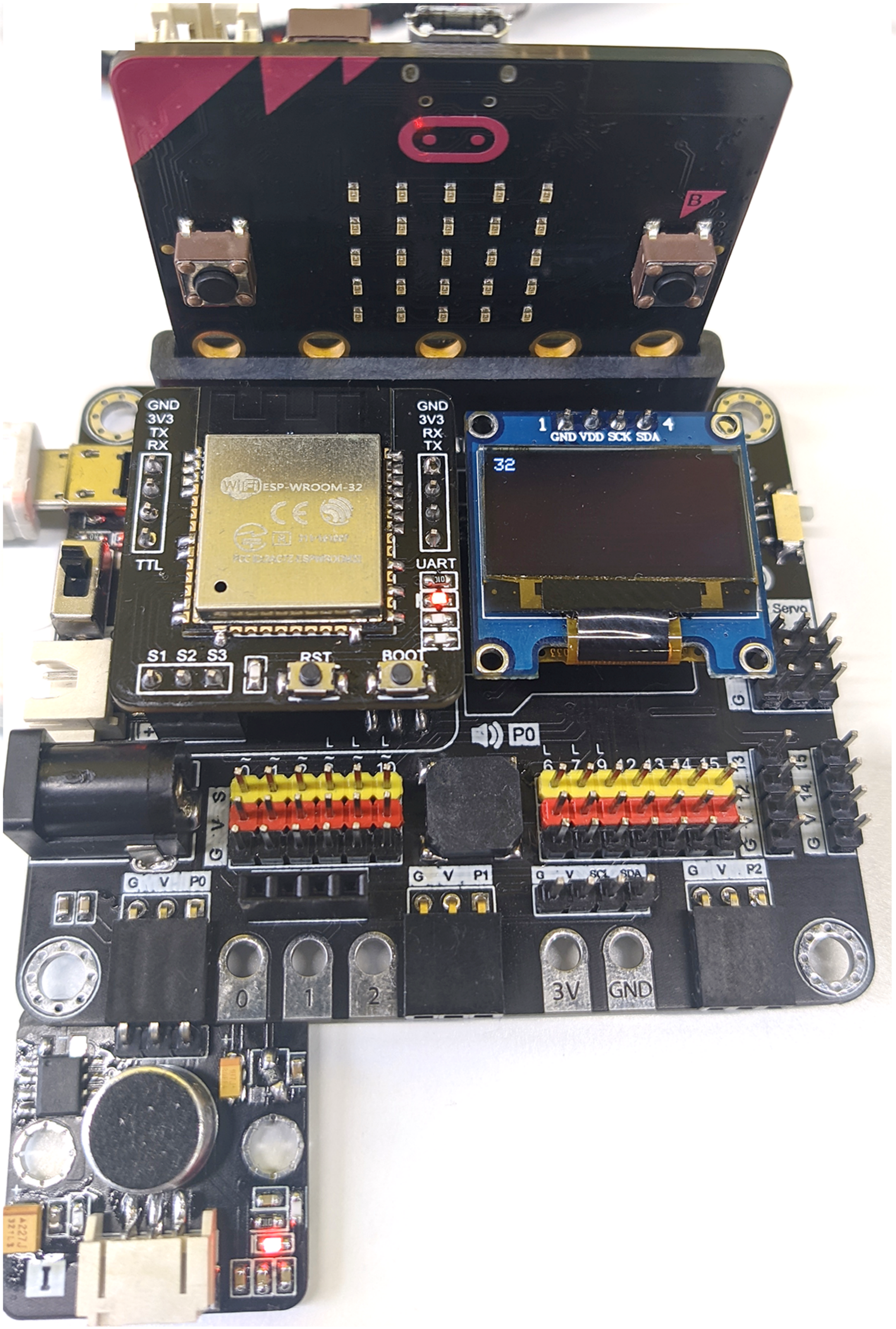
```
on start
    initialize OLED with width 128 height 64

forever
    clear OLED display
    show number Get noise level (dB) at Pin P0
    pause (ms) 1000
```

The value will be in range of 0 to 100

Result



When the environment is quite silent without noise



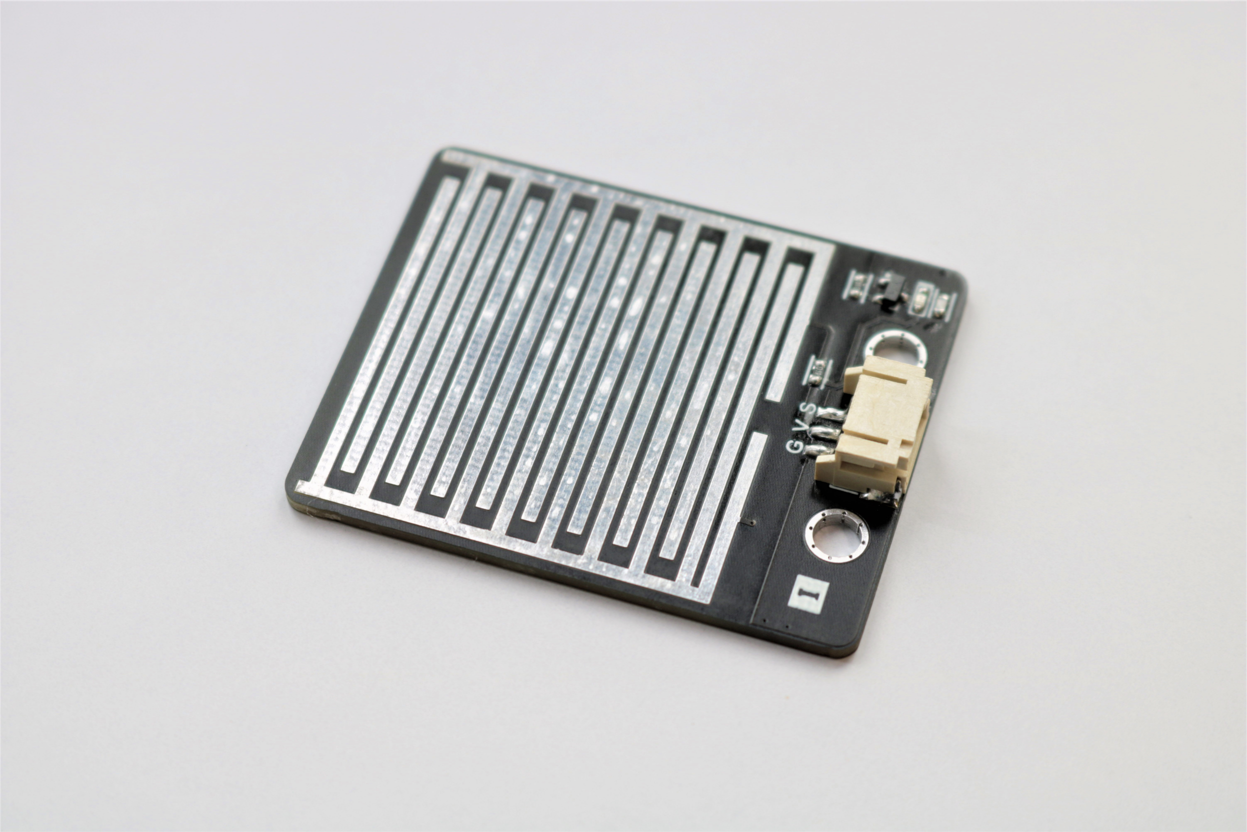
When people speak louder

FAQ

Datasheet

LM386-datasheet

1.2.5 RainDrop Sensor

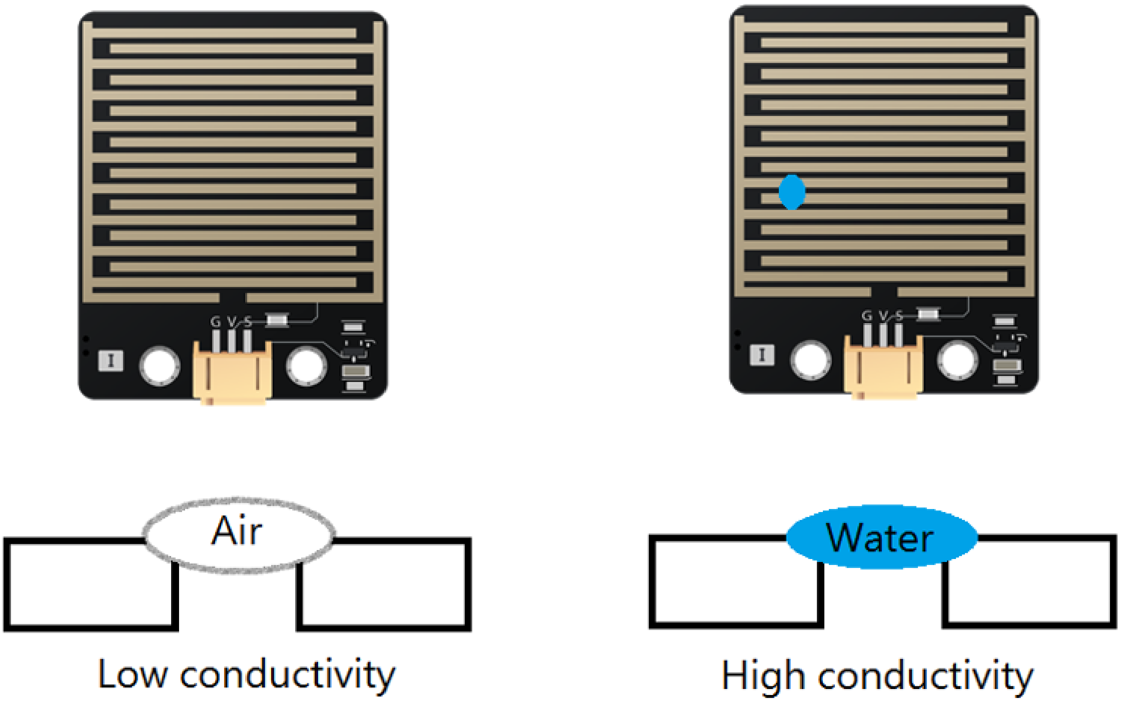


Introduction

Raindrop sensor use the Conductivity to determine the rain drops is on the board or not, when there are no liquid on the board, the reading will be very small (0 if no liquid), however even one drop of rain, can increase the conductivity very much and make the reading to extremely high (100).

The principle

The RainDrop Sensor use the conductivity between the two electrodes to determine the change of the water drop on the plane. When there is no water drop on the plane, the material between the two electrodes is air, result in very low conductivity. However, when the water drop is on the plane, the conductivity between the electrodes will become very high since water is a good conductor.

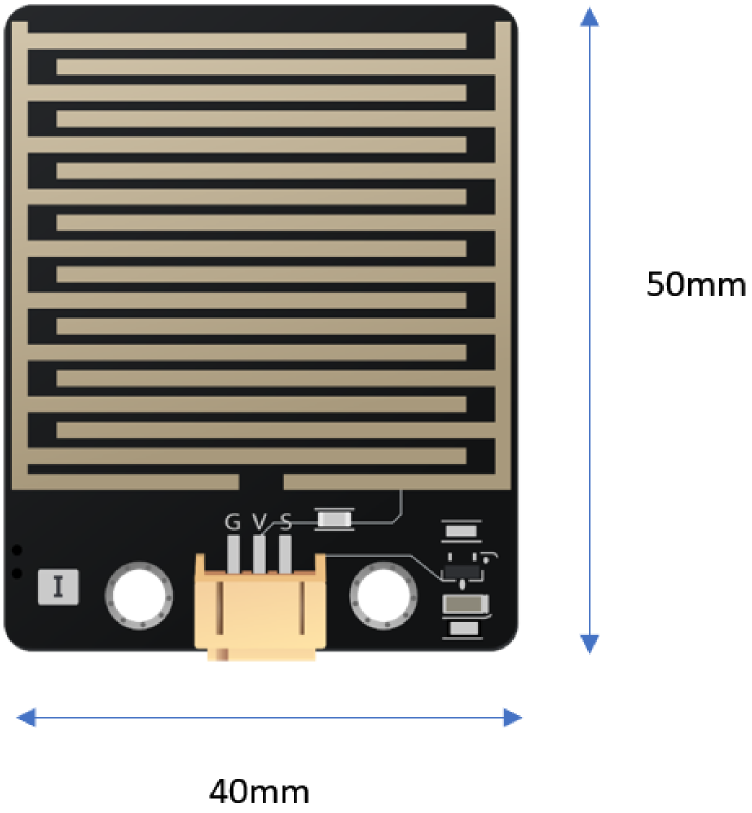


Specification

- Supply Voltage: 5V
- Interface: Analog
- Range: 0(Dry surface) ~ 100(WaterDrop on surface)

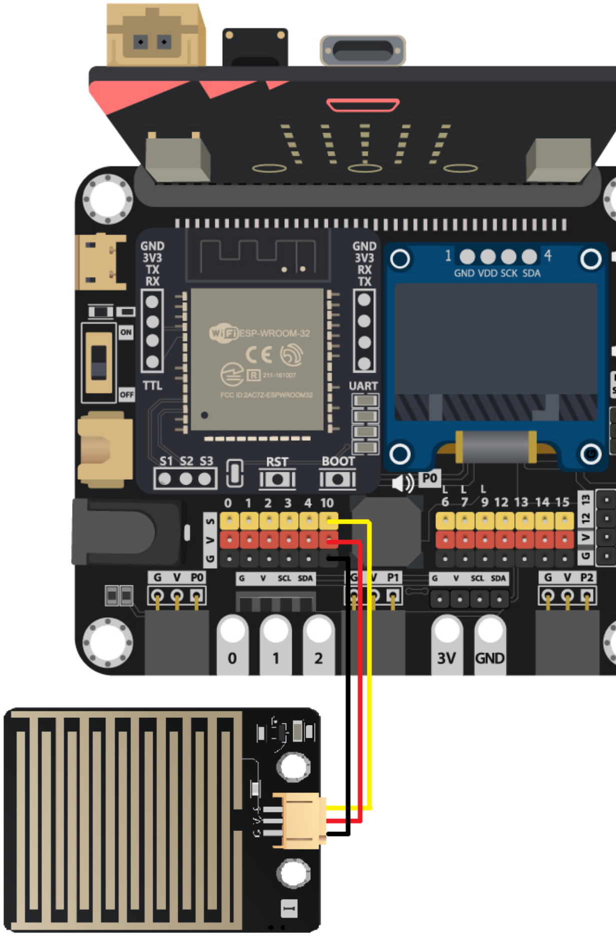
Pinout Diagram

Outlook and Dimension

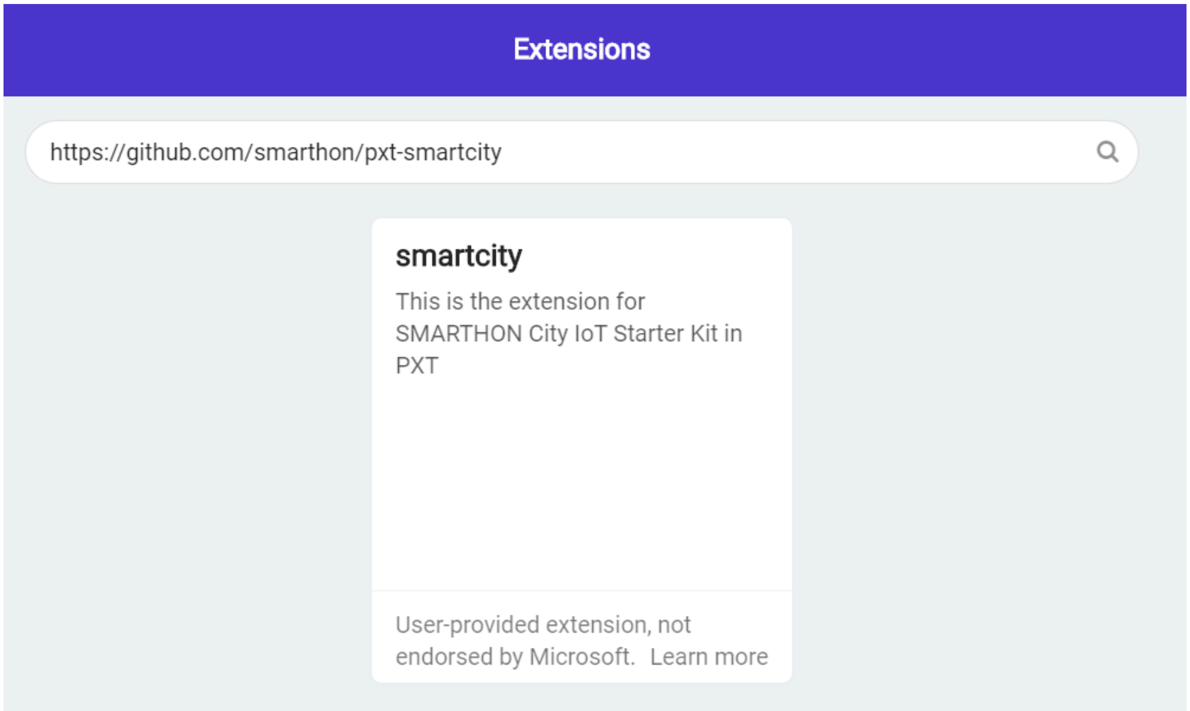


Size: 50mm X 40mm

Quick to Start/Sample



- Connect the sensor to development board (using wire)
- Open Makecode, using the https://github.com/smarthon/pxt-smartcity PXT



Extensions

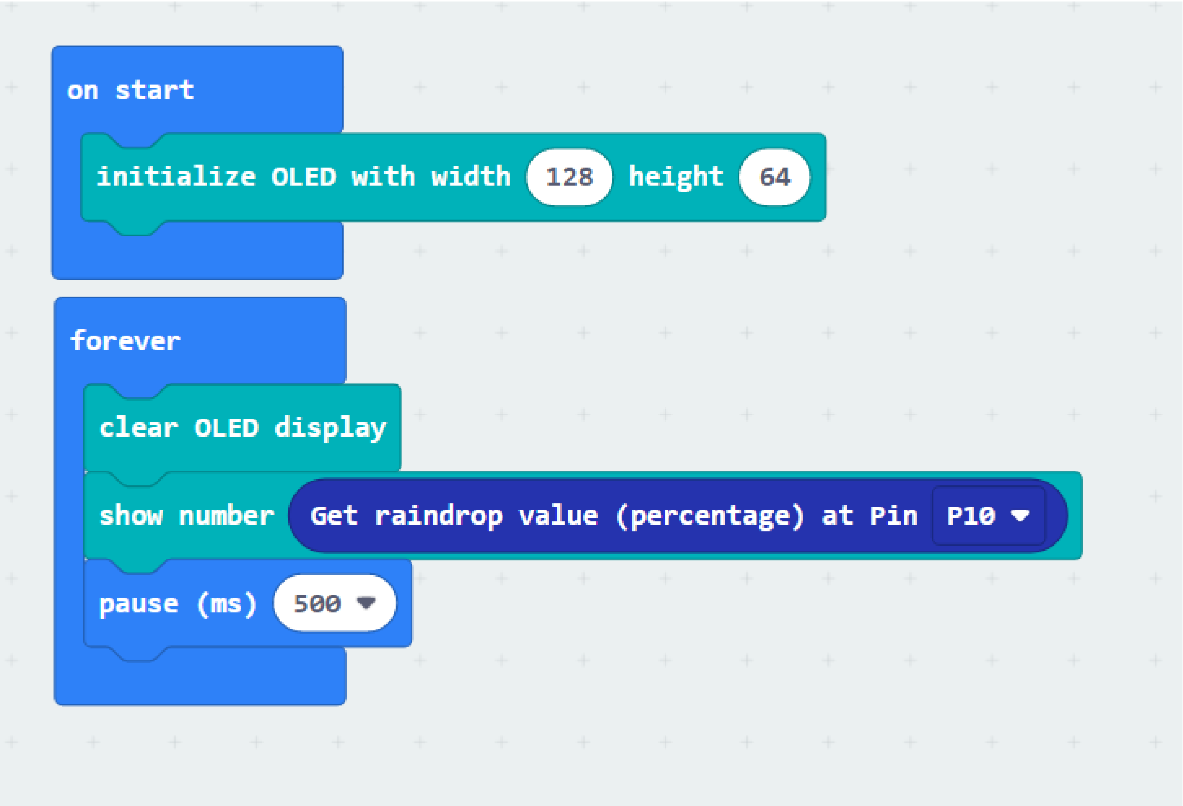
https://github.com/smarthon/pxt-smartcity

smartcity

This is the extension for SMARTHON City IoT Starter Kit in PXT

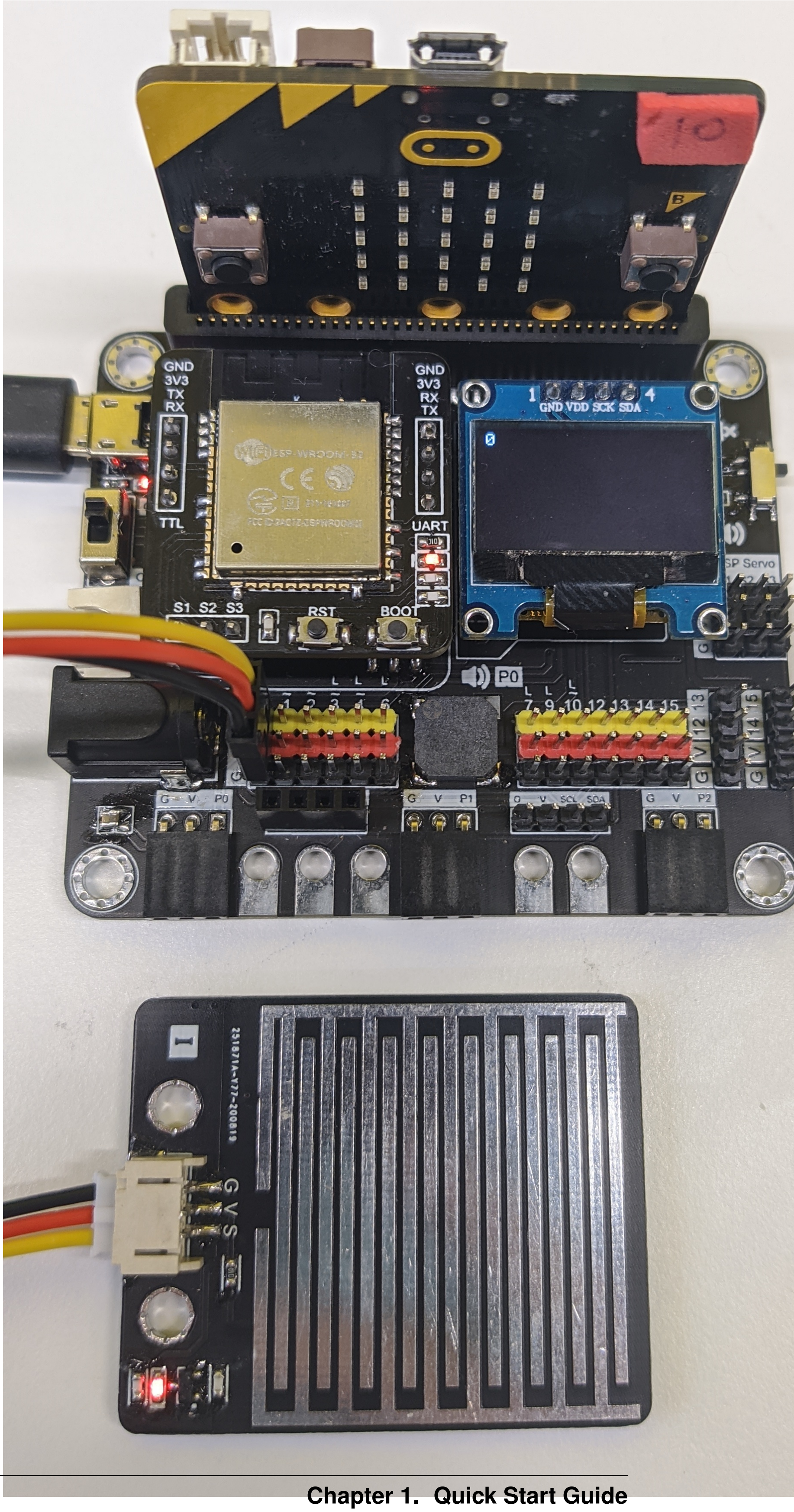
User-provided extension, not endorsed by Microsoft. Learn more

- Initial the OLED screen and show the reading of the raindrop sensor

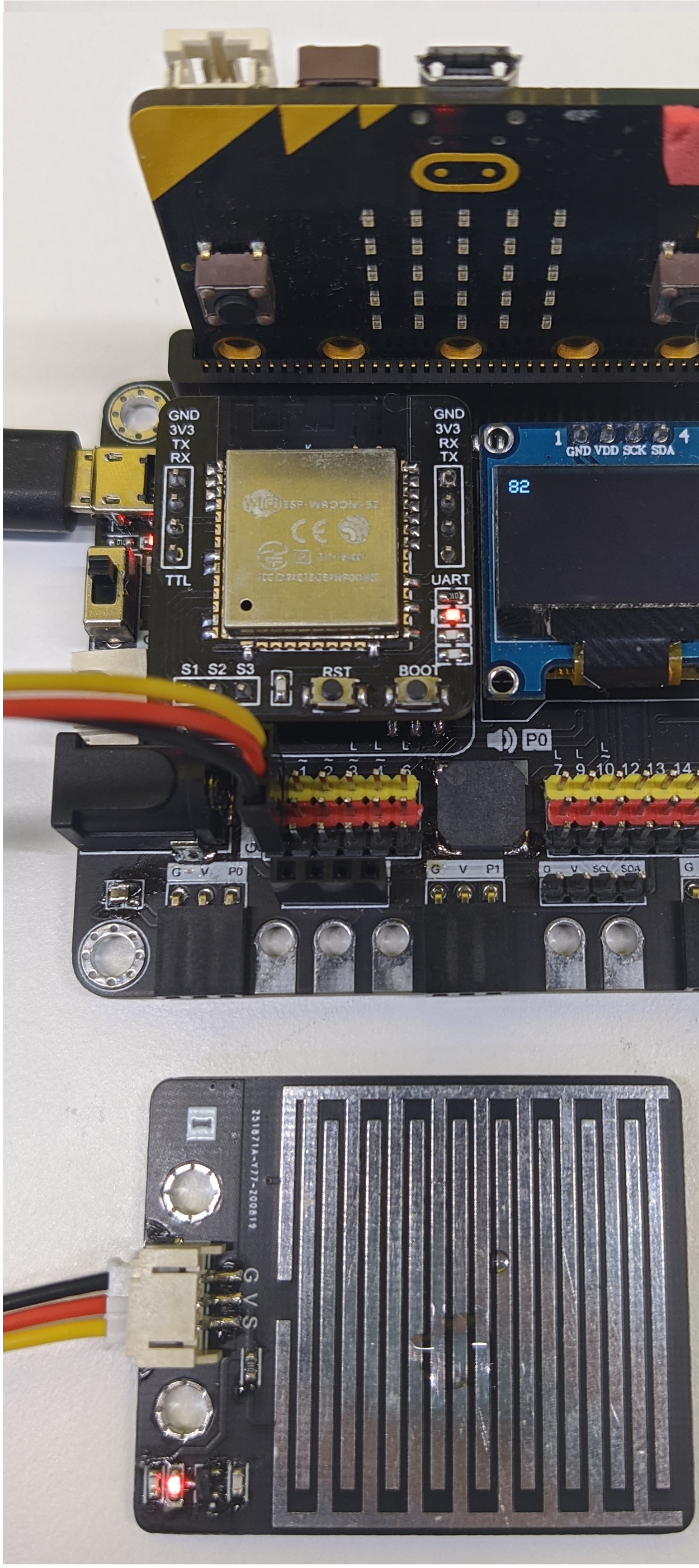


```
on start
    initialize OLED with width 128 height 64
forever
    clear OLED display
    show number Get raindrop value (percentage) at Pin P10
    pause (ms) 500
```

Result



When there are no water on board, the reading is 0



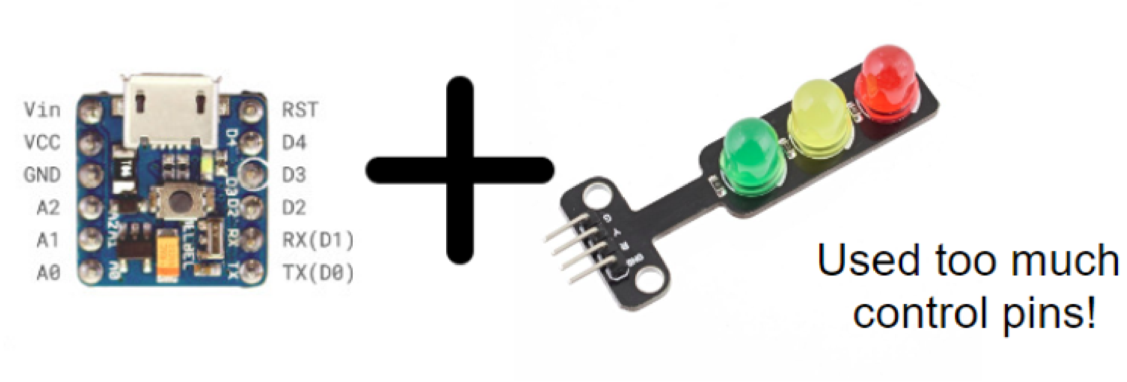
After put one waterdrop on that, the reading getting close to 100

1.2.6 Traffic Light

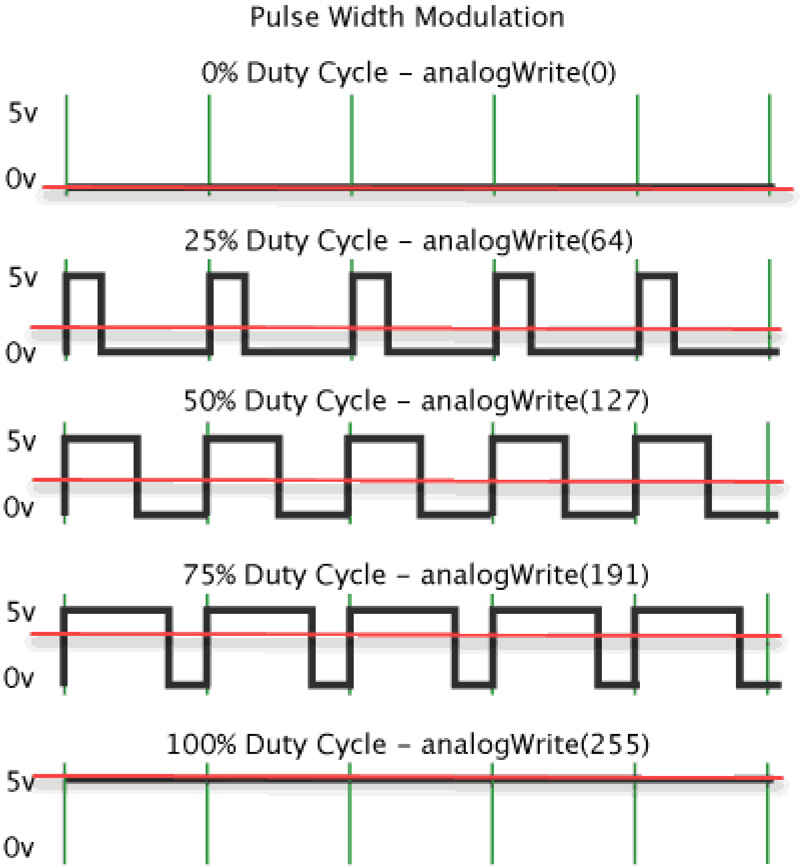
Introduction

The traffic light module including three different colors LED which are red, yellow and green color, like the real-life traffic light. When received different frequency of PWM signal, it will provide different pattern of light.

The principle



Most of the traffic light module on market will using the concept of common anode to make it easy to use. However, it required 4 pin on the module(3 signal and 1 ground) and consume 3 control pin from the controller, make it become heavy and hard to use on some tiny device.When using PWM method instead of common anode, it can reduce to 3pins and only consume 1 signal pin form the controller, more suitable for the tiny project.



The PWM signal can represent in the average voltage output, or duty cycle % in one complete cycle. This module will control by the duty cycle method.The chip on the traffic light module keep recording the incoming PWM signal and distinguish the "states" of traffic light by the preset duty cycle % level table. Then control the turn on /off states of the three LED lights.

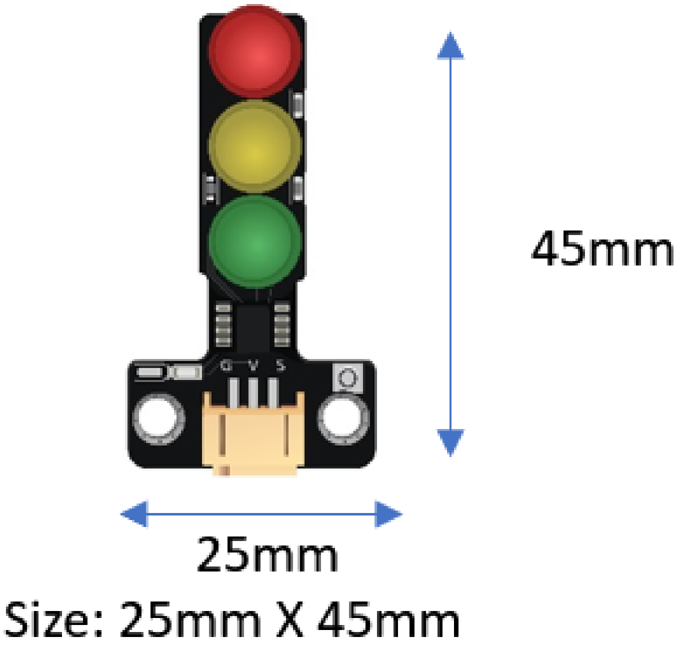
Specification

- Supply voltage: 3.3V~5V
- Interface: Analog/Digital PWM

Duty Cycle:

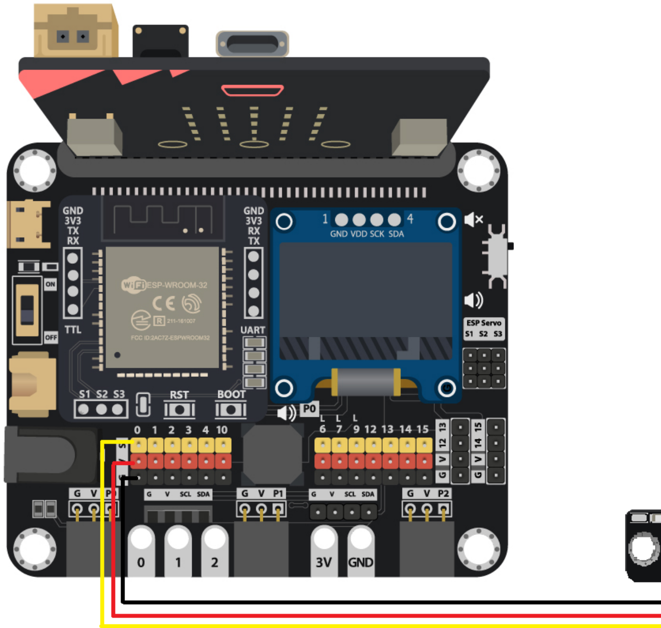
Pinout Diagram

Outlook and Dimension

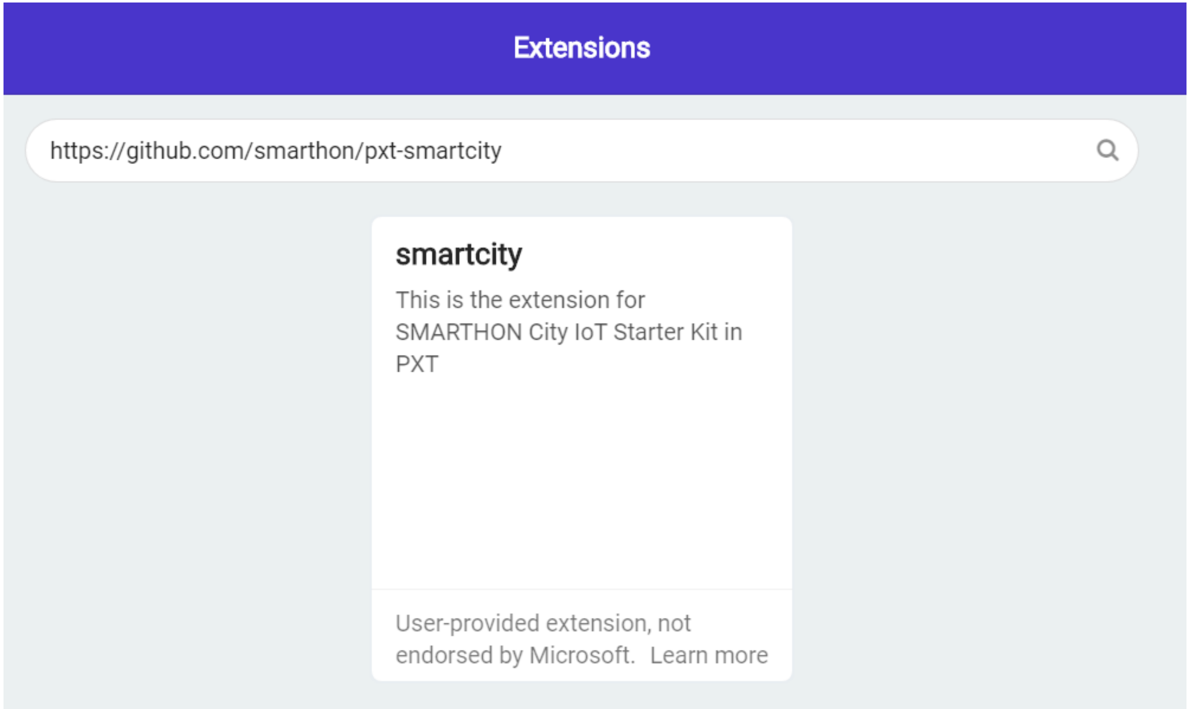


Size: 25mm X 45mm

Quick to Start/Sample



- Connect the sensor to development board (using wire)
- Open Makecode, using the https://github.com/smarthon/pxt-smartcity PXT



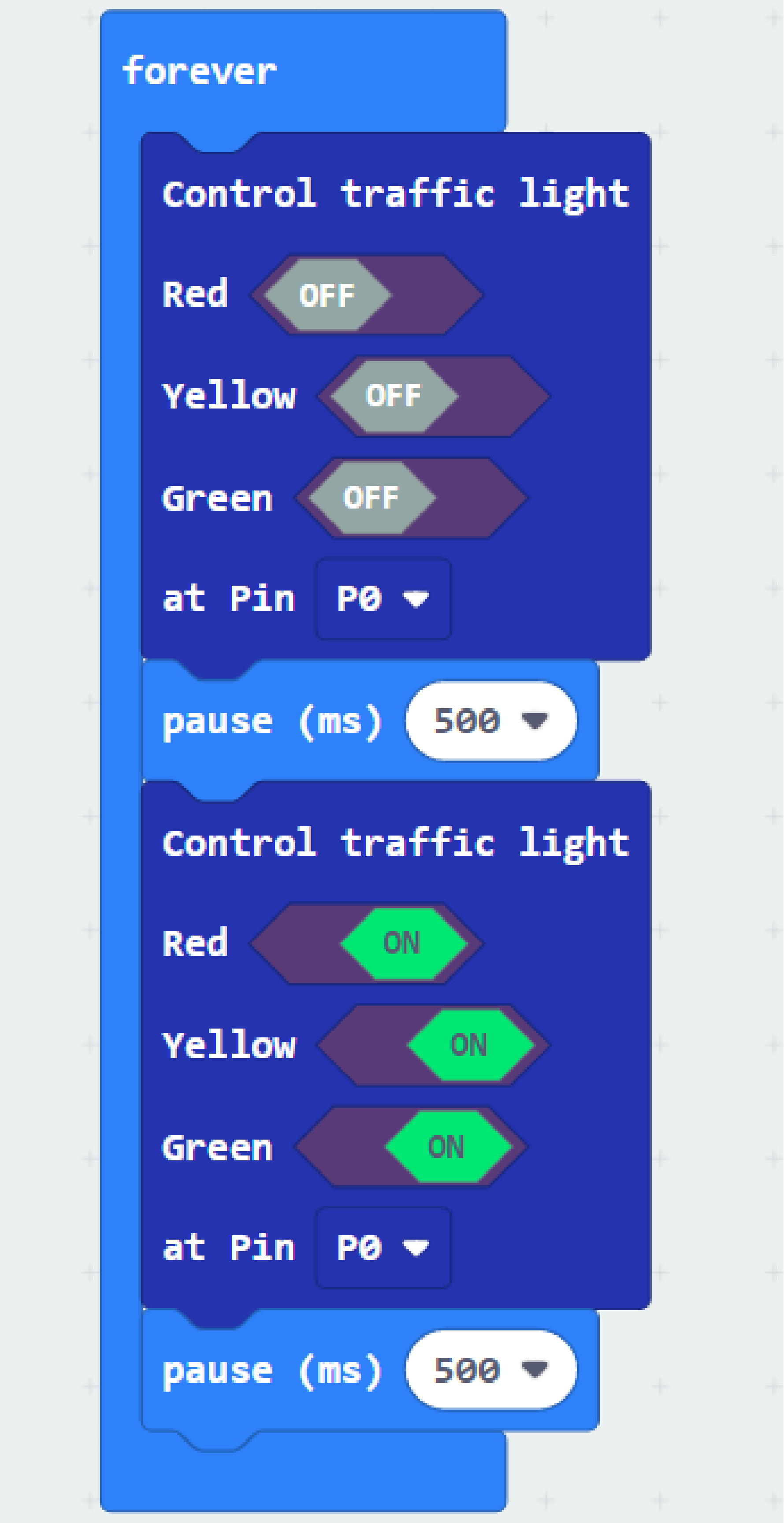
Extensions

https://github.com/smarthon/pxt-smartcity

smartcity

This is the extension for SMARTHON City IoT Starter Kit in PXT

User-provided extension, not endorsed by Microsoft. Learn more



```
forever
    Control traffic light
    Red OFF
    Yellow OFF
    Green OFF
    at Pin P0
    pause (ms) 500
    Control traffic light
    Red ON
    Yellow ON
    Green ON
    at Pin P0
    pause (ms) 500
```

- Provide the corresponding PWM signal

Result

The traffic light will turn on and off

FAQ

Q: Why the lights is out of control? A: Check the PWM signal duty cycle is fit the requirement or not. Also be care the change of light status should not too quick, at least wait for 1 second.

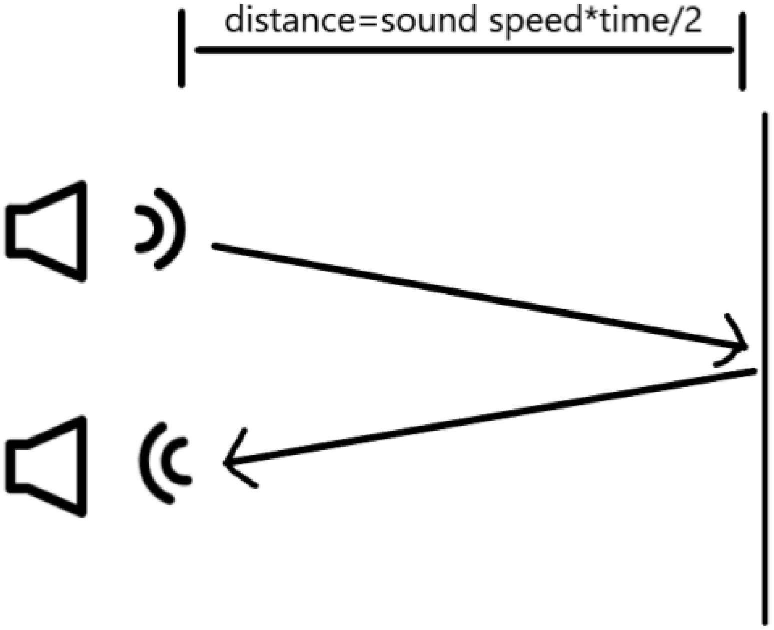
1.2.7 Ultrasonic Distance Sensor

Introduction

The Distance sensor using two ultrasonic sensors to detect the distance between current position and object. In most situation, the availability of ultrasonic allow it provided quick and accurate result.

The principle

The Distance sensor use the ultrasonic to measure the distance between the sensor and object. When the sensor emit a wave of ultrasonic wave, it will start passing thought the air until it hit the object, then rebound and back to the receiver. The sound wave travel in the air is approximately 340meter per second, with it, use the interval time between sent and received can calculate the distance.

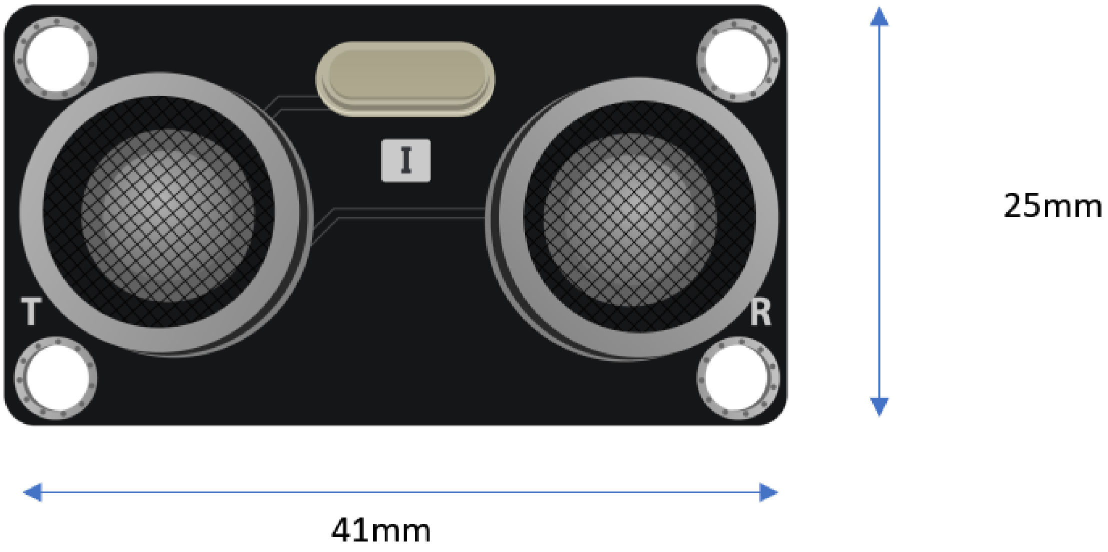


Specification

- Supply voltage: 5V
- Working Frequency: 40kHz
- Range: 3cm to 4m
- MeasuringAngle: <15°

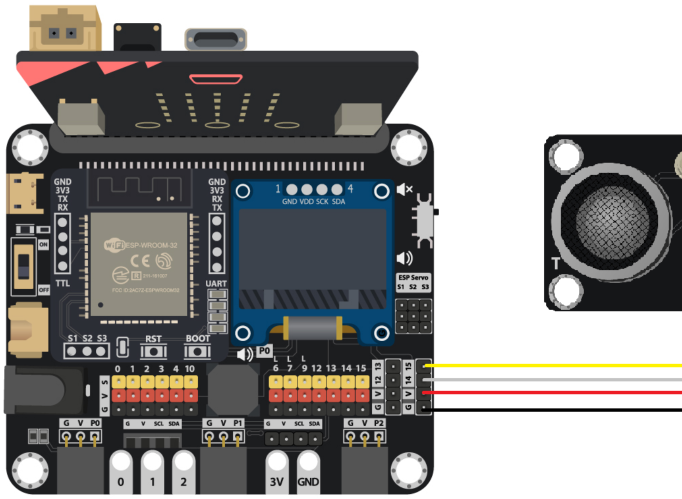
Pinout Diagram

Outlook and Dimension

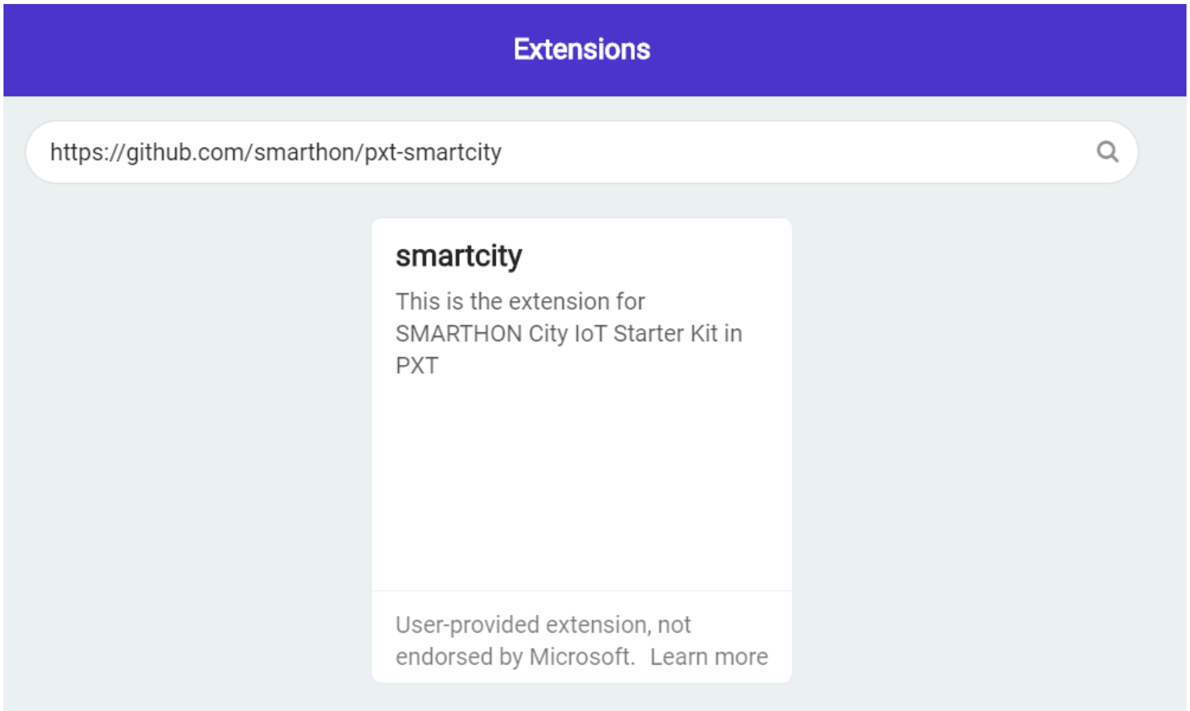


Size: 41 X 25mm

Quick to Start/Sample



- Connect the sensor to development board (using wire)
- Open Makecode, using the https://github.com/smarthon/pxt-smartcity PXT



Extensions

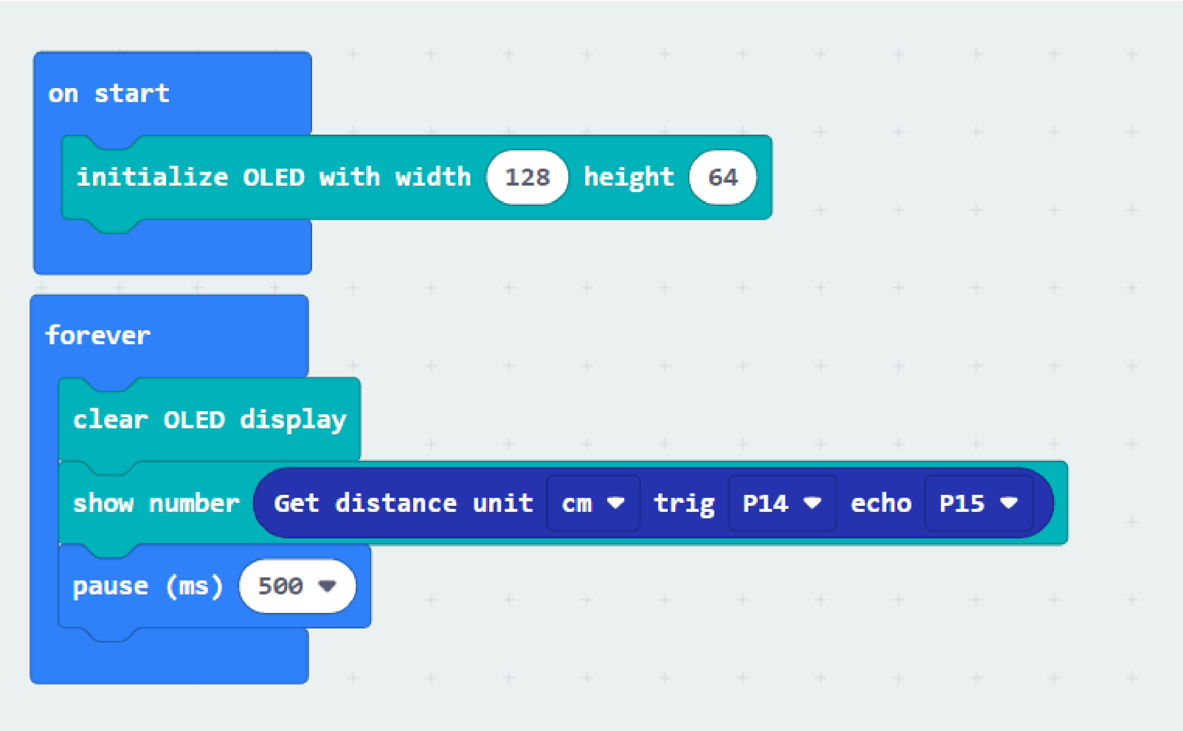
https://github.com/smarthon/pxt-smartcity

smartcity

This is the extension for SMARTHON City IoT Starter Kit in PXT

User-provided extension, not endorsed by Microsoft. Learn more

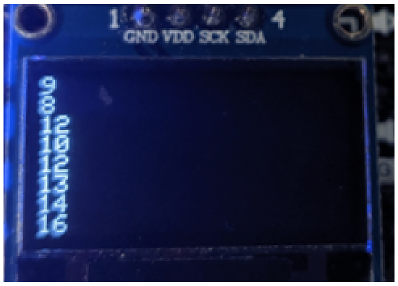
- Initial the OLED and show the distance value on the OLED screen



```
on start
    initialize OLED with width 128 height 64
forever
    clear OLED display
    show number Get distance unit cm ▾ trig P14 ▾ echo P15 ▾
    pause (ms) 500 ▾
```

Result

The Distance will be increasing when moving out the hand from sensor

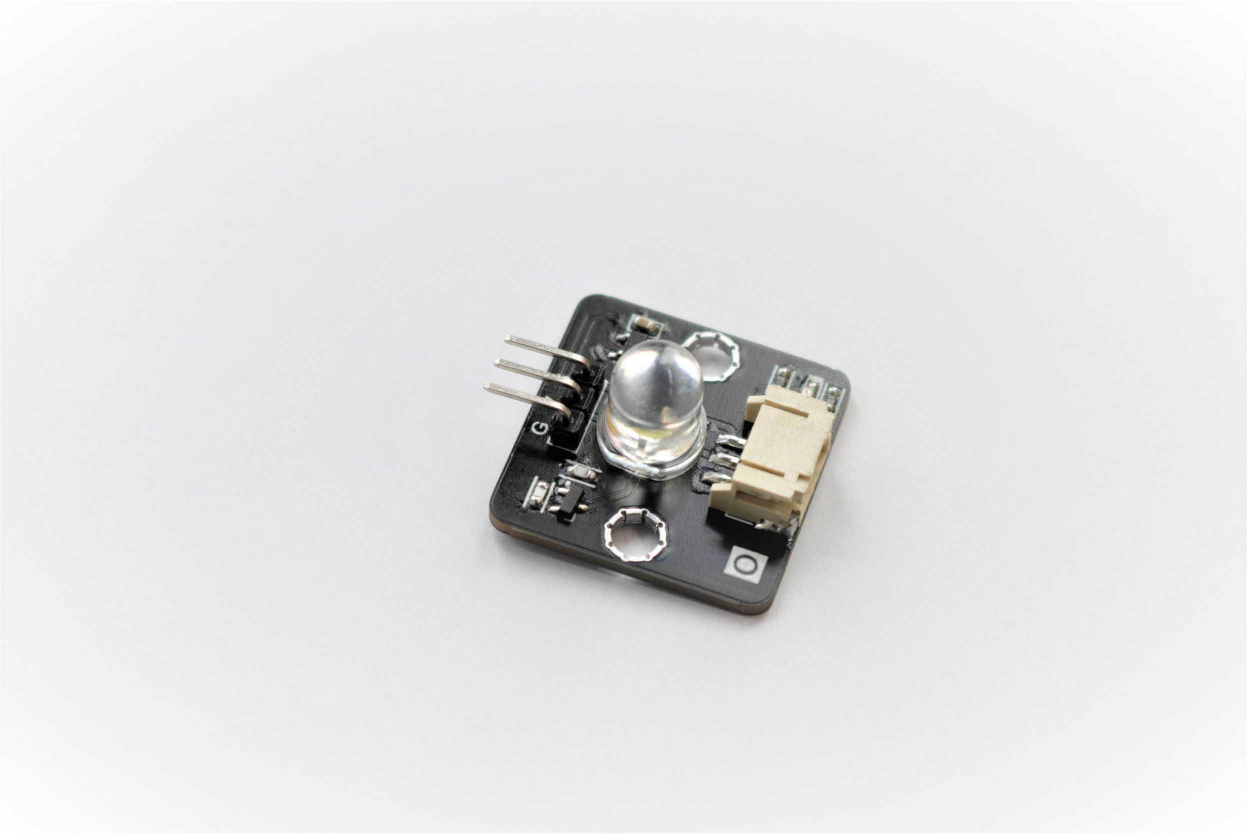


```
9
8
12
10
12
13
14
16
```

FAQ

Q: Why I cannot get the correct reading in far distance? A: Because of the ultrasonic sender and receiver both on the board, limited by the physical reason, the reflection angle of the object will affect the result correctly, especially when the distance is far, little reflection angle can create large displacement of mainstream, so please adjust the angle of detection.

1.2.8 White LED

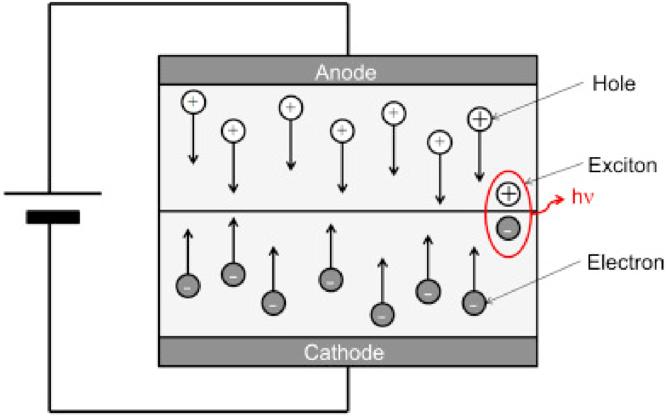


Introduction

White LED module build in a high-performance LED for different usage, such illuminate, express program status, or make visual presentation. User just need to provide voltage, then can control the brightness by the corresponding input.

The principle

The LED is a two-lead semiconductor light source. Under electroluminescence theory, when a suitable voltage is applied to the leads, electrons are able to release energy in the form of photons, which means emit of light. The color of light emitted depend on the energy level drop when doing electroluminescence, it is relative to material of the semiconductor.

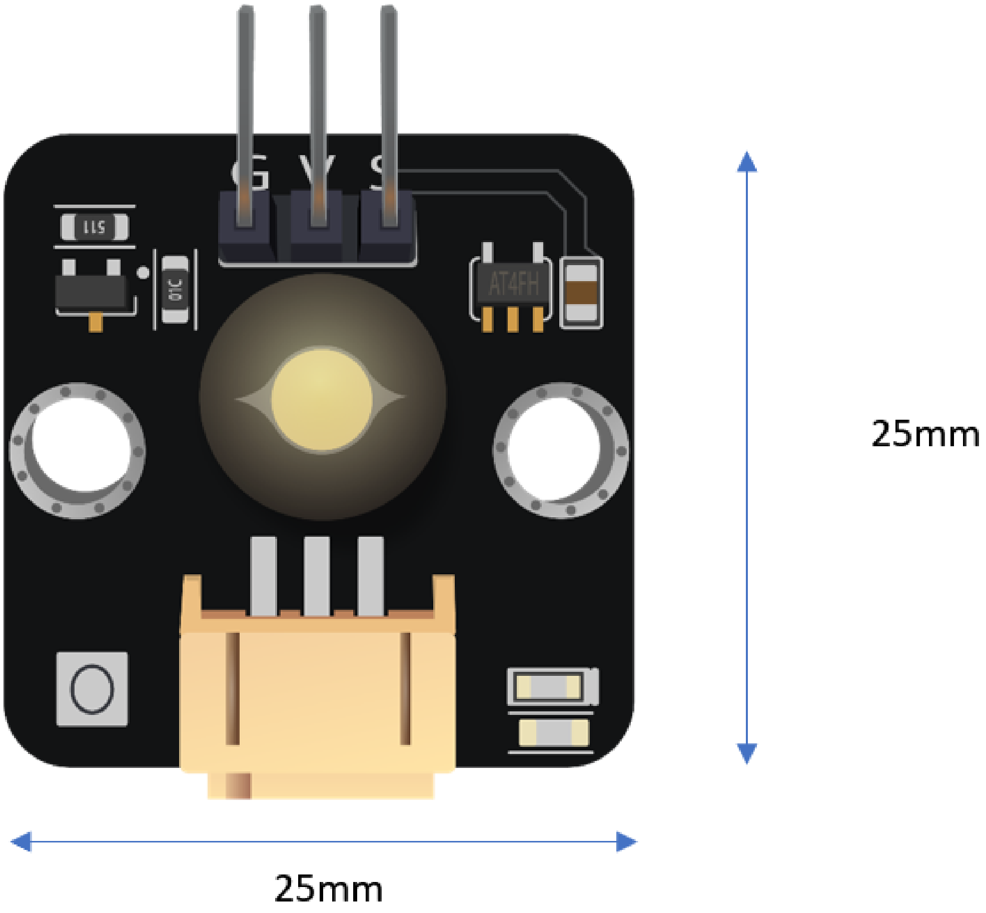


Specification

- Operate voltage: 3 - 5V
- Power consumption: 0.06W
- Efficiency: 90 lm/W

Pinout Diagram

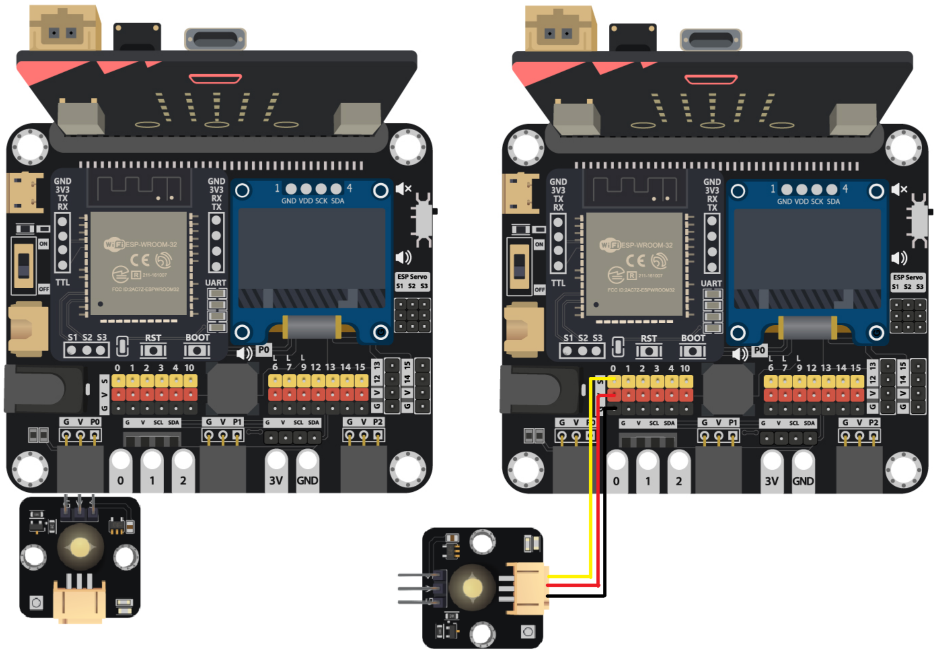
Outlook and Dimension



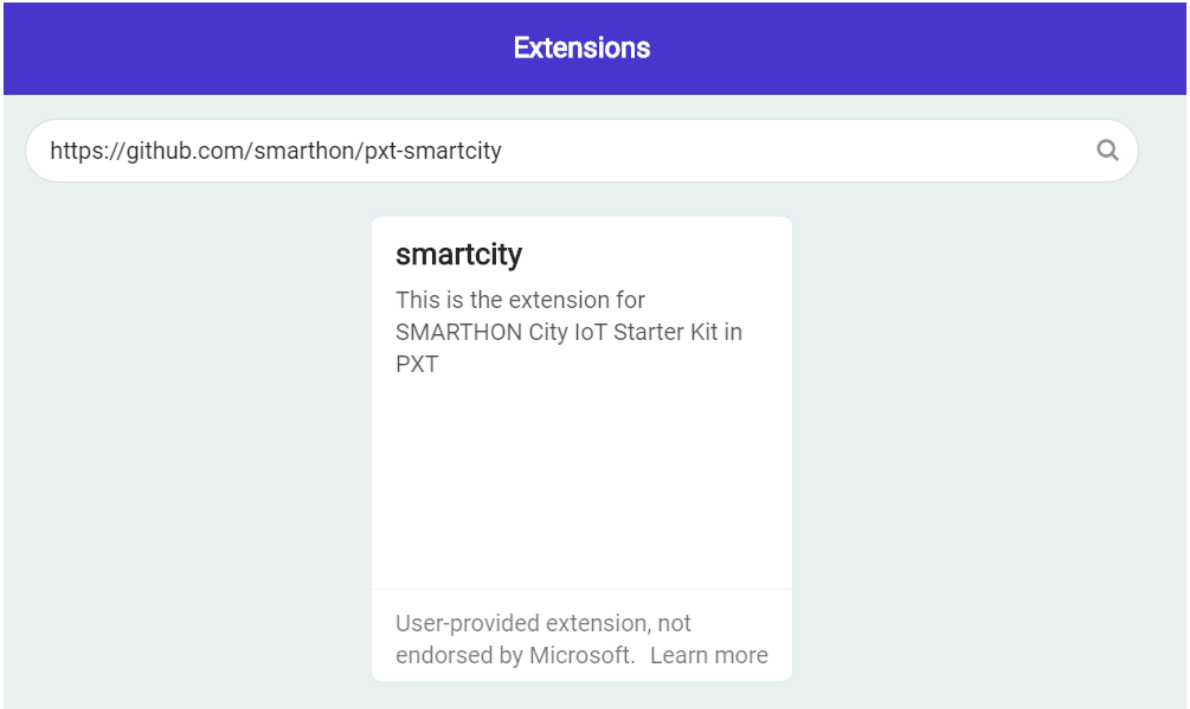
Size: 25mm X 25mm

Quick to Start/Sample

- Connect the sensor to development board (direct plugin or using wire)



- Open Makecode, using the https://github.com/smarthon/pxt-smartcity PXT



Extensions

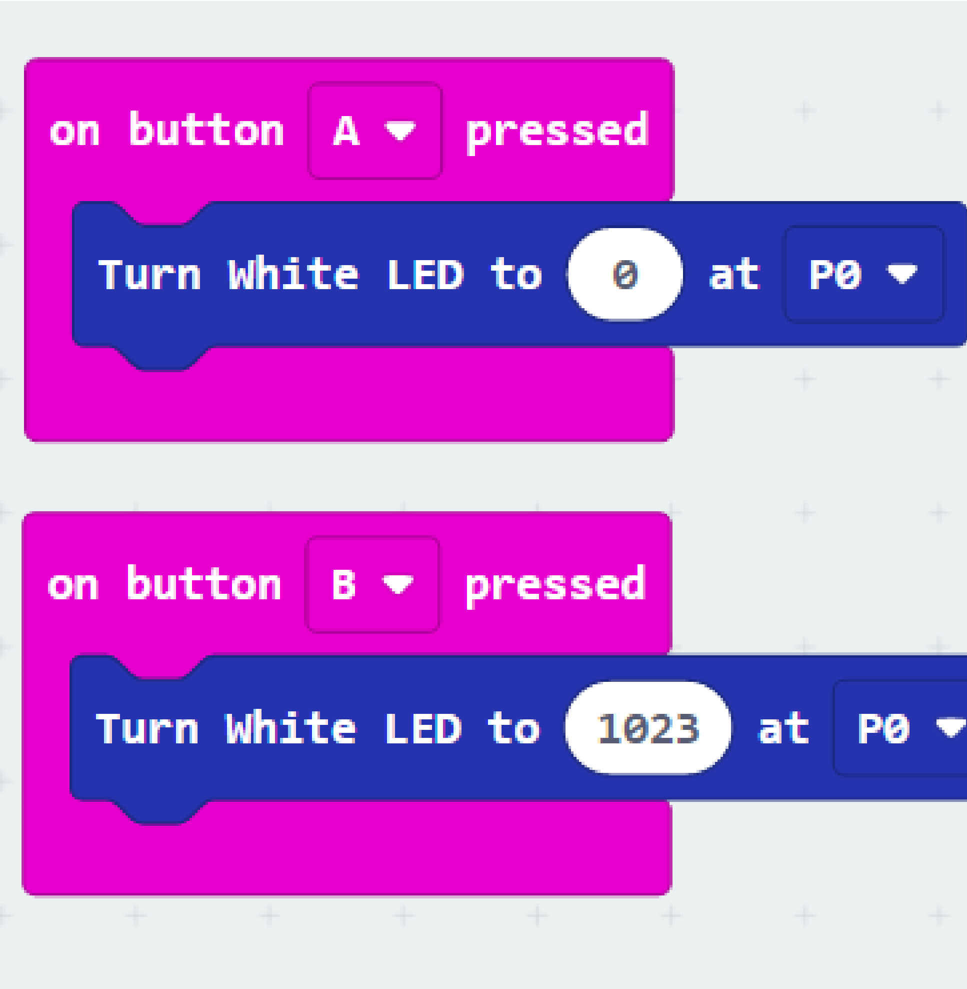
https://github.com/smarthon/pxt-smartcity

smartcity

This is the extension for SMARTHON City IoT Starter Kit in PXT

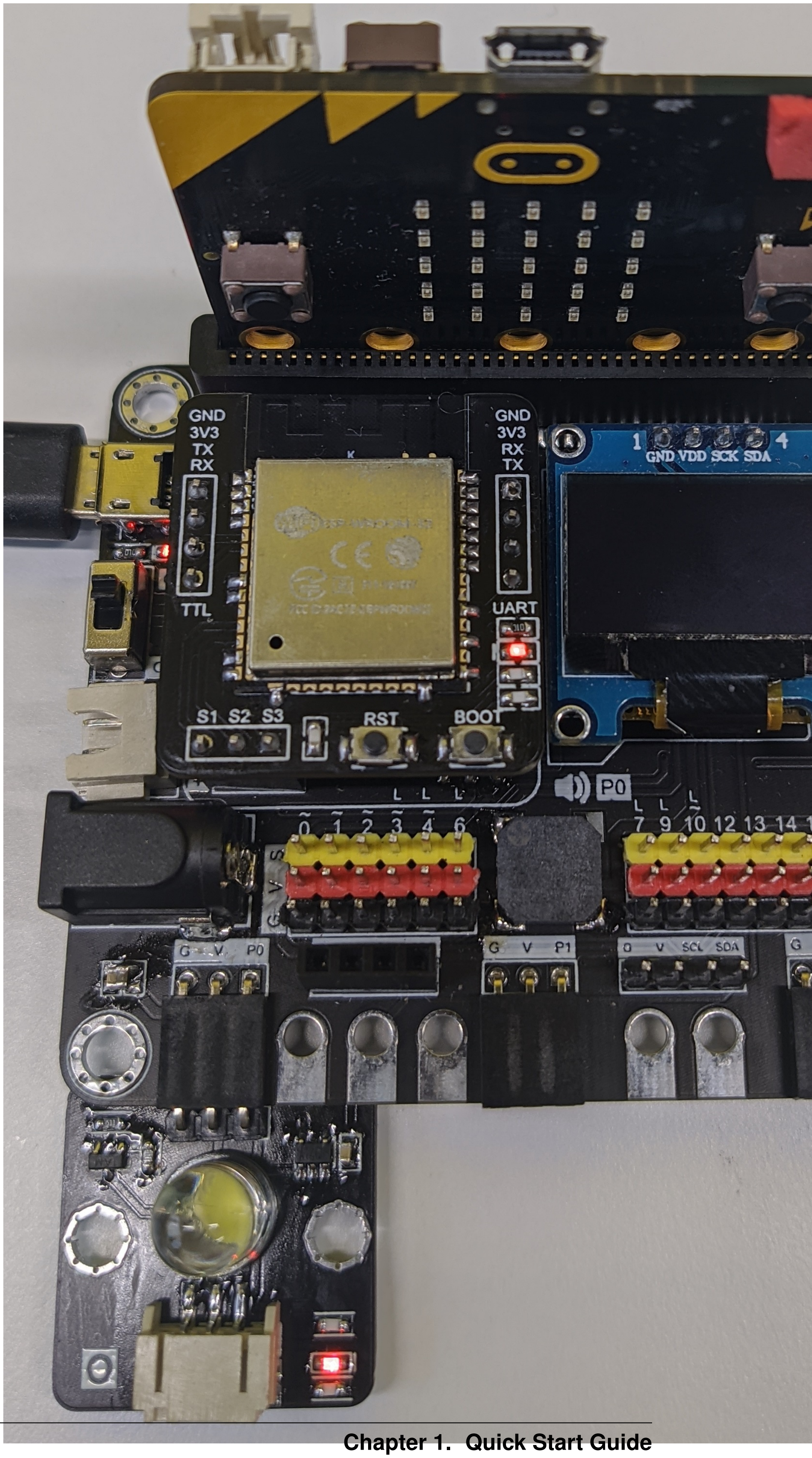
User-provided extension, not endorsed by Microsoft. Learn more

- Set different buttons to on/off the LED

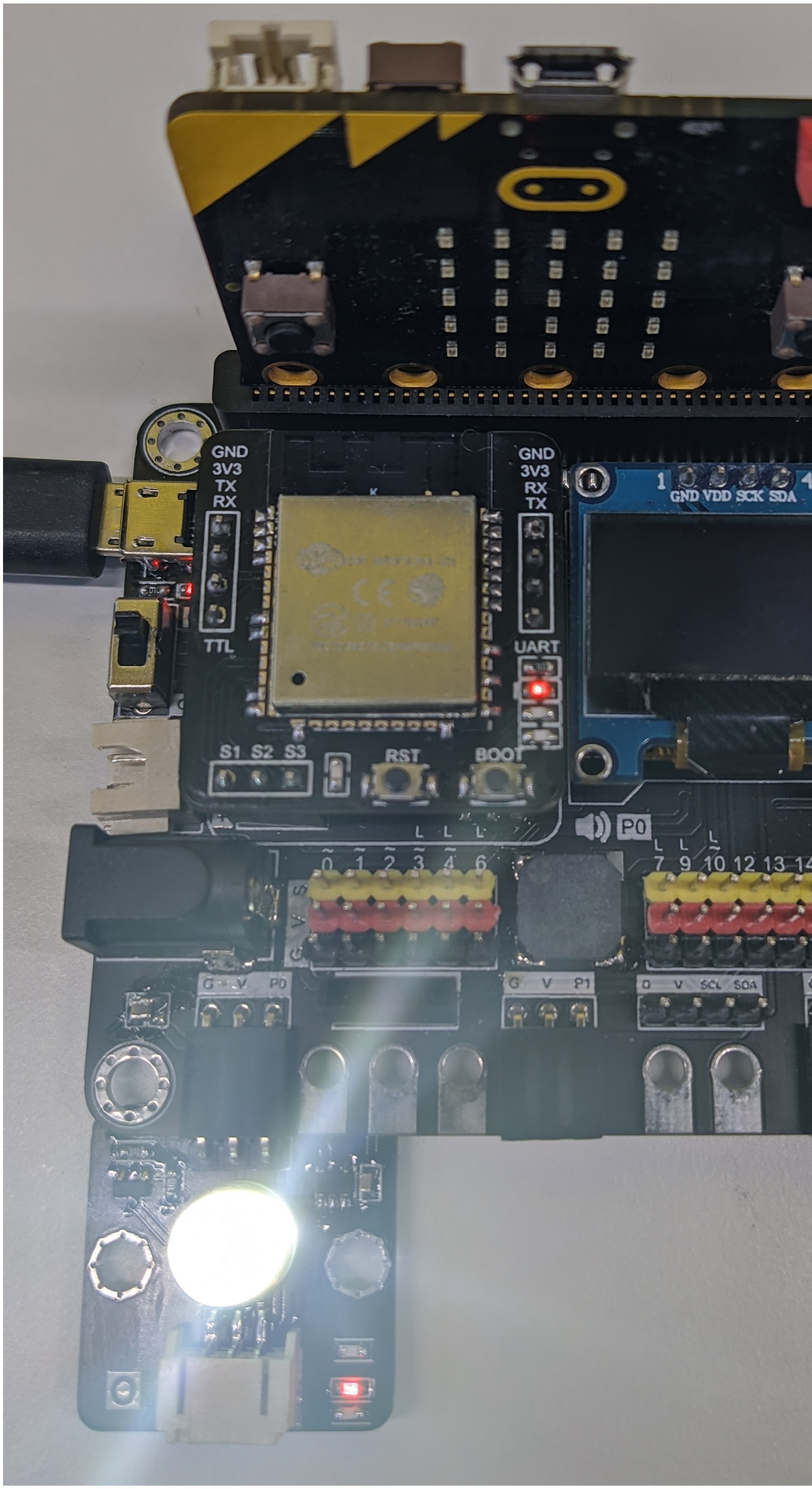


```
on button A ▾ pressed
    Turn White LED to 0 at P0 ▾

on button B ▾ pressed
    Turn White LED to 1023 at P0 ▾
```

Result

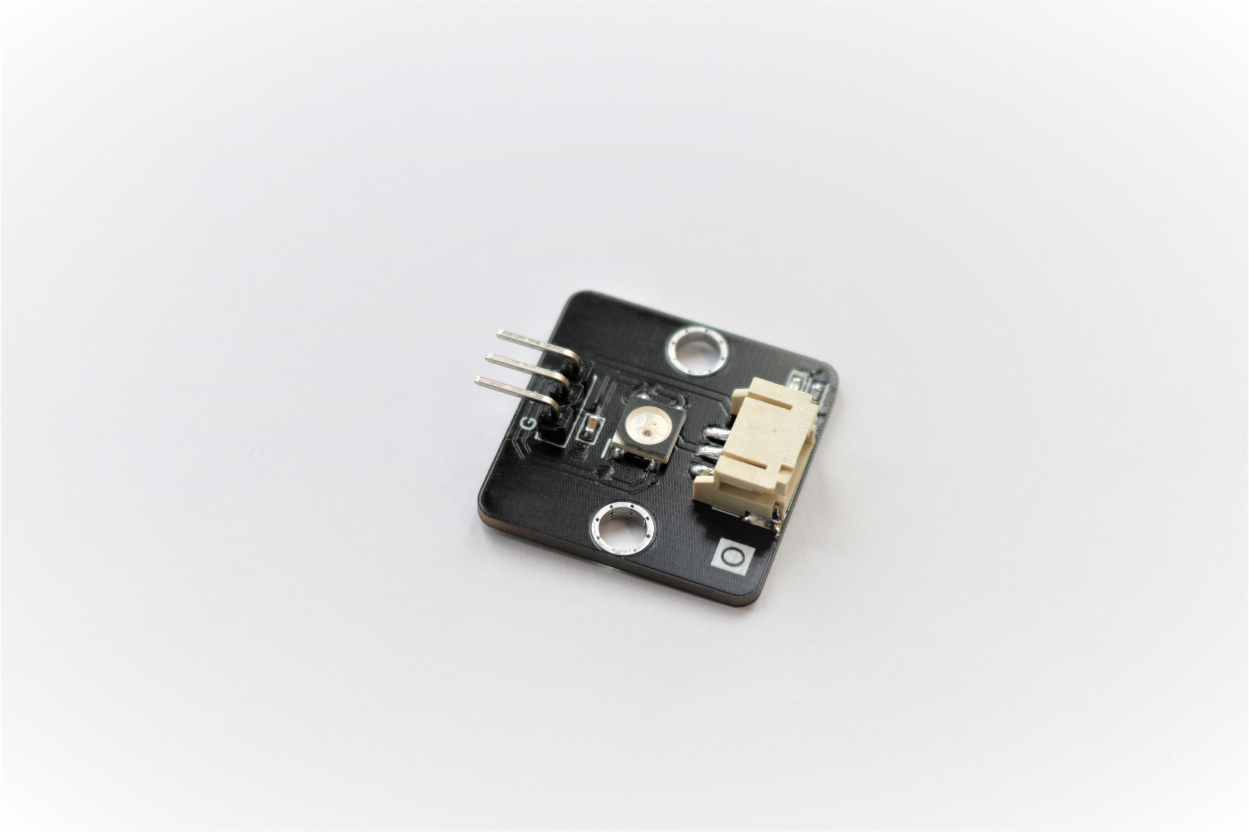


Press A button the turn off the LED (set to 0)



Press B button the turn on the LED (set to 1023)

1.2.9 Multi-color LED (WS2812)

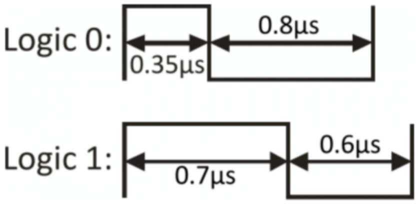


Introduction

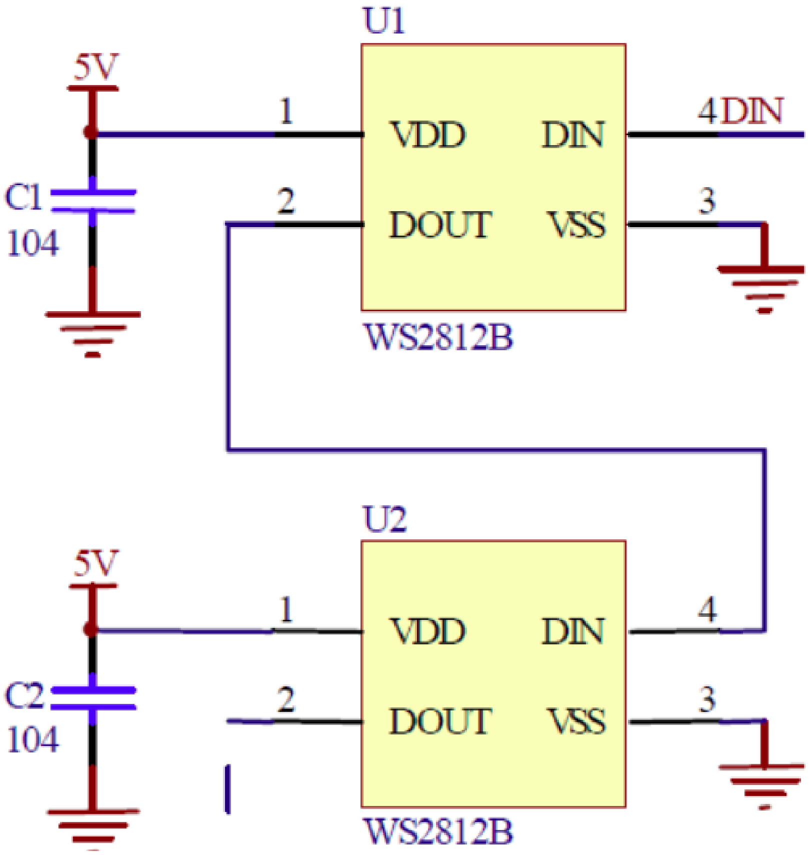
Multi-color LED (WS2812) is an integrated LED chip which can create different RGB color. Using programming, user can pick any one of color in 16 million color and showing it.

The principle

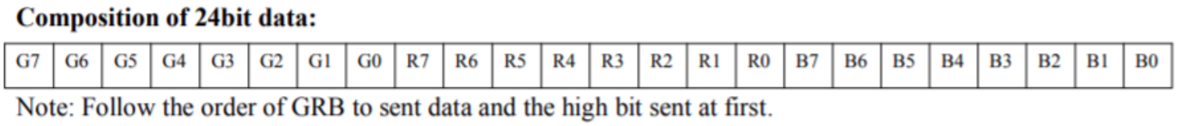
The WS2812 have a small chip inside the module, which handle the incoming data and communicate with the next WS2812B in the serial. WS2812B operation under digital, user need to follow the special logic representation to transmits the data.



WS2812 allow connect to other WS2812B and share the signal line, but benefit from the data transmission policy, it can use the single line to control all the light individually.



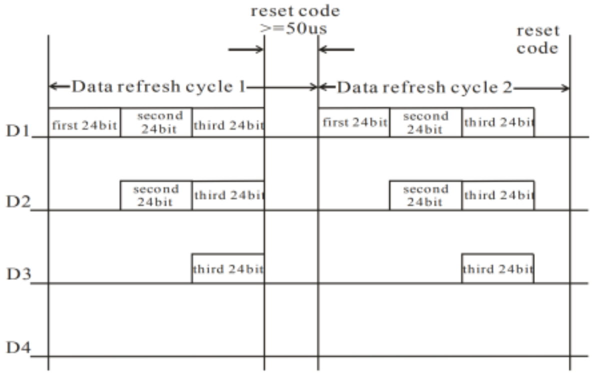
For each WS2812, there are three 8bit color LED (Green, Red, Blue) inside, so each WS2812 required total 24bit of data in once action.


Composition of 24bit data:

G7	G6	G5	G4	G3	G2	G1	G0	R7	R6	R5	R4	R3	R2	R1	R0	B7	B6	B5	B4	B3	B2	B1	B0

Note: Follow the order of GRB to sent data and the high bit sent at first.

After the 24 bits data read by WS2812, the remaining data bit will pass to next WS2812(if need), so the next WS2812 is possible to read another set of data to produce different color. When all the data bit is sent, another 50us reset code is required to let chip understand the transmission is end and control the led to emit the light.

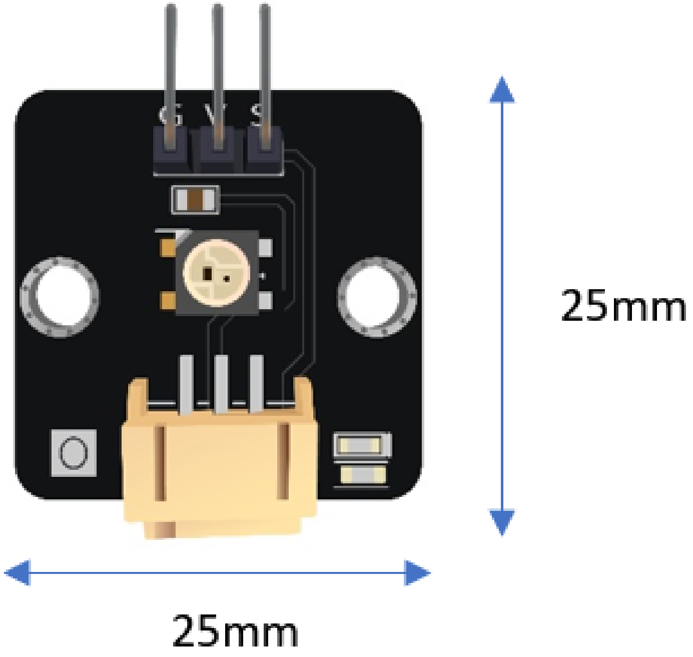


Specification

- Supply Voltage: 3.3V to 5V
- Interface: Digital
- Range of Color: (0,0,0) to (255,255,255)

Pinout Diagram

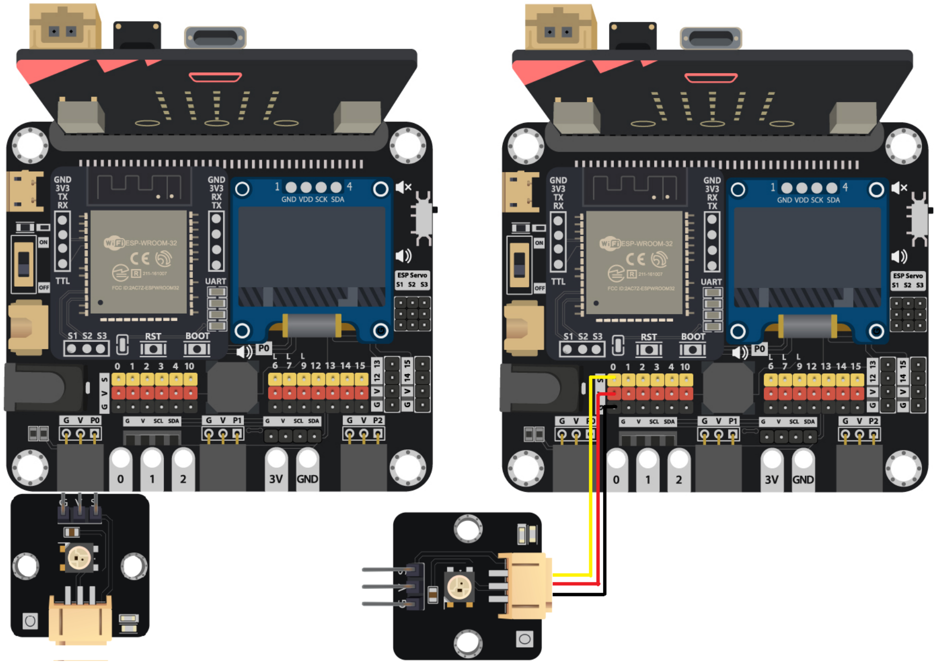
Outlook and Dimension



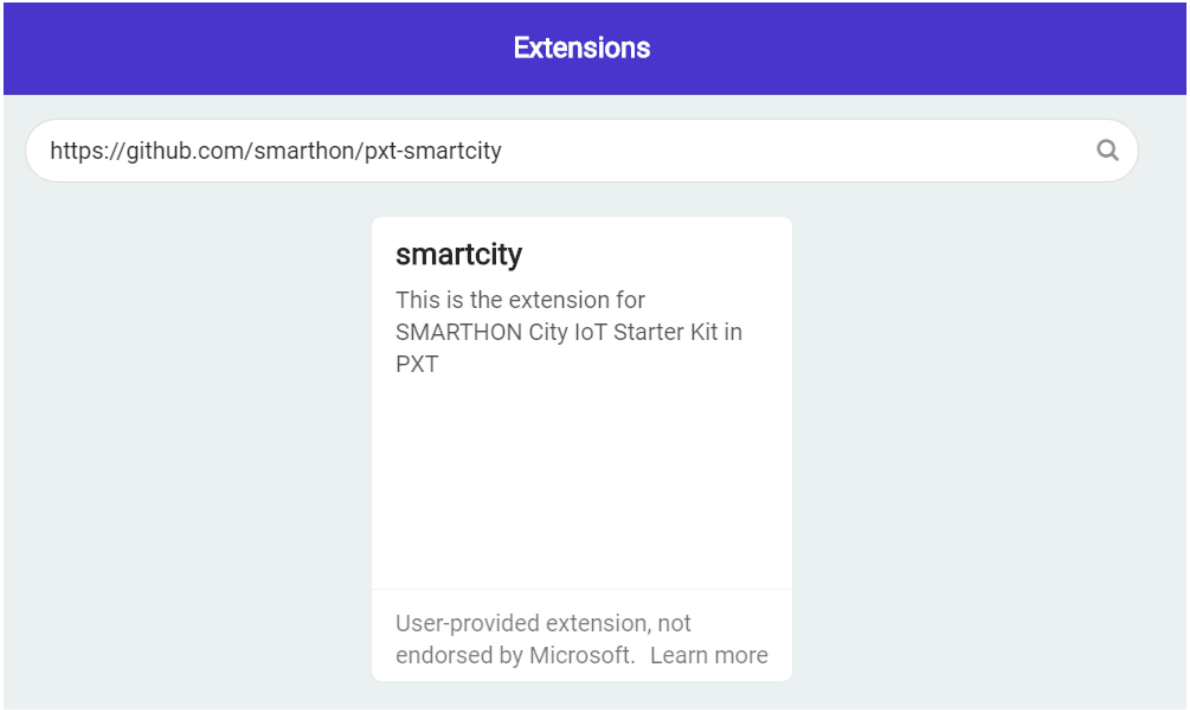
Size: 25mm X 25mm

Quick to Start/Sample

- Connect the sensor to development board (direct plugin or using wire)



- Open Makecode, using the https://github.com/smarthon/pxt-smartcity PXT



Extensions

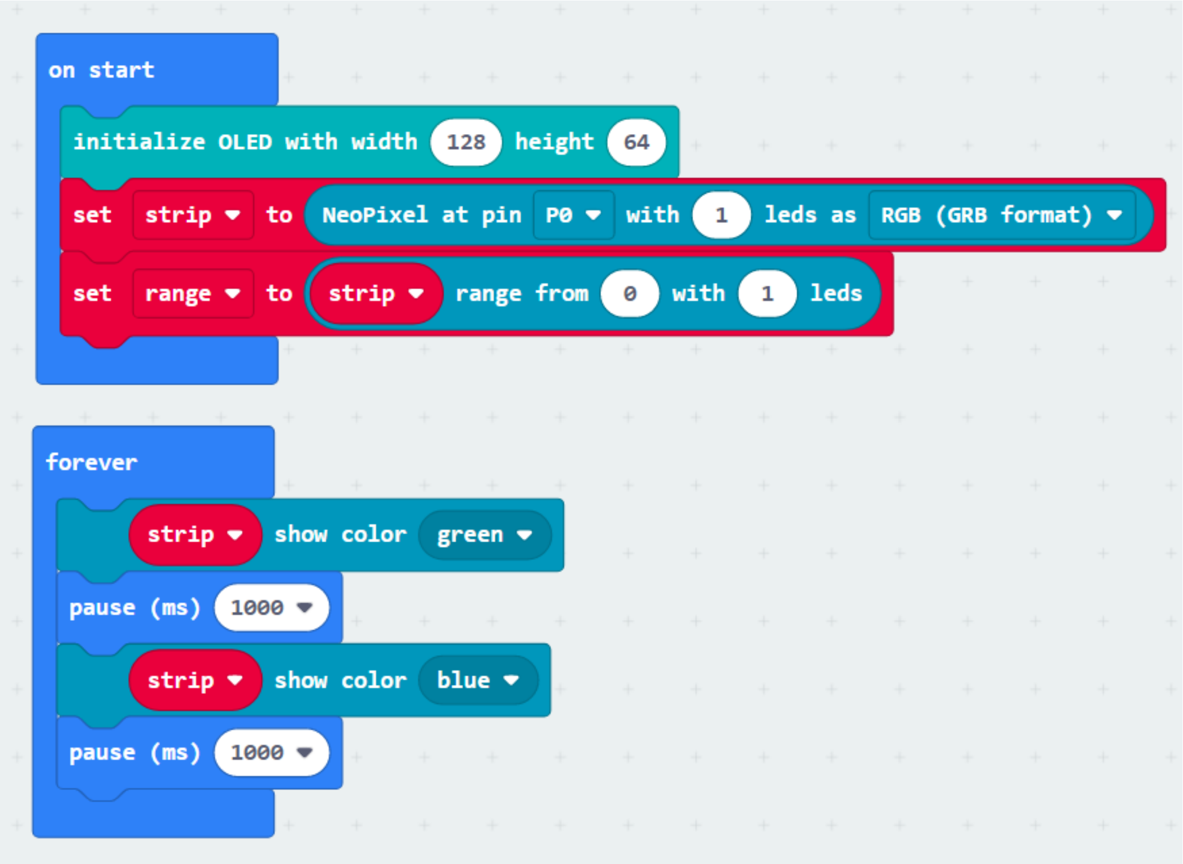
https://github.com/smarthon/pxt-smartcity

smartcity

This is the extension for SMARTHON City IoT Starter Kit in PXT

User-provided extension, not endorsed by Microsoft. Learn more

- Use to PXT library to control the LED showing different colors



```
on start
    initialize OLED with width 128 height 64
    set strip to NeoPixel at pin P0 with 1 leds as RGB (GRB format)
    set range to strip range from 0 with 1 leds

forever
    strip show color green
    pause (ms) 1000
    strip show color blue
    pause (ms) 1000
```

Result

The WS2812 following the program to change the color

1.3 Smarthon Smart House Kit for micro:bit

1.3.1 Product Introduction

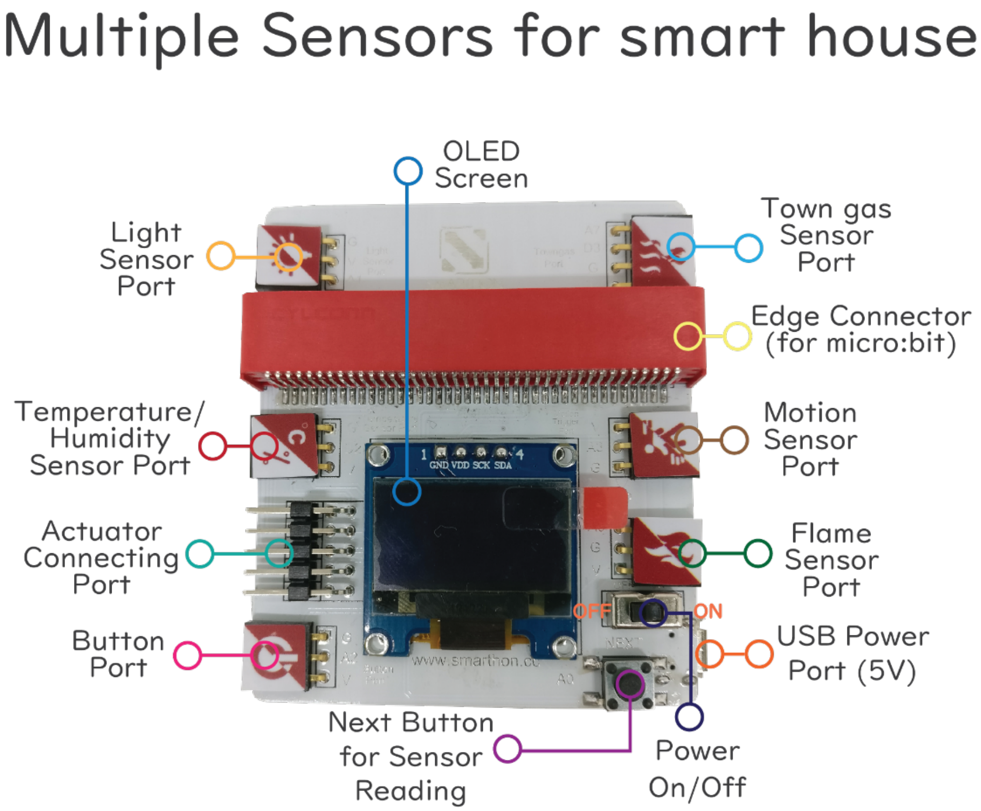
About Smarthon Smart House Kit

Smarthon Smart House Kit for micro:bit is a collaboration of computing and engineering with art for users to design and build a smart house model. This educational tool contains 7 sensors and 7 actuators that cover all the basic areas of the smart house, including security, home automation, and comfortable living as well as energy management features. Let's start to develop your prototype.

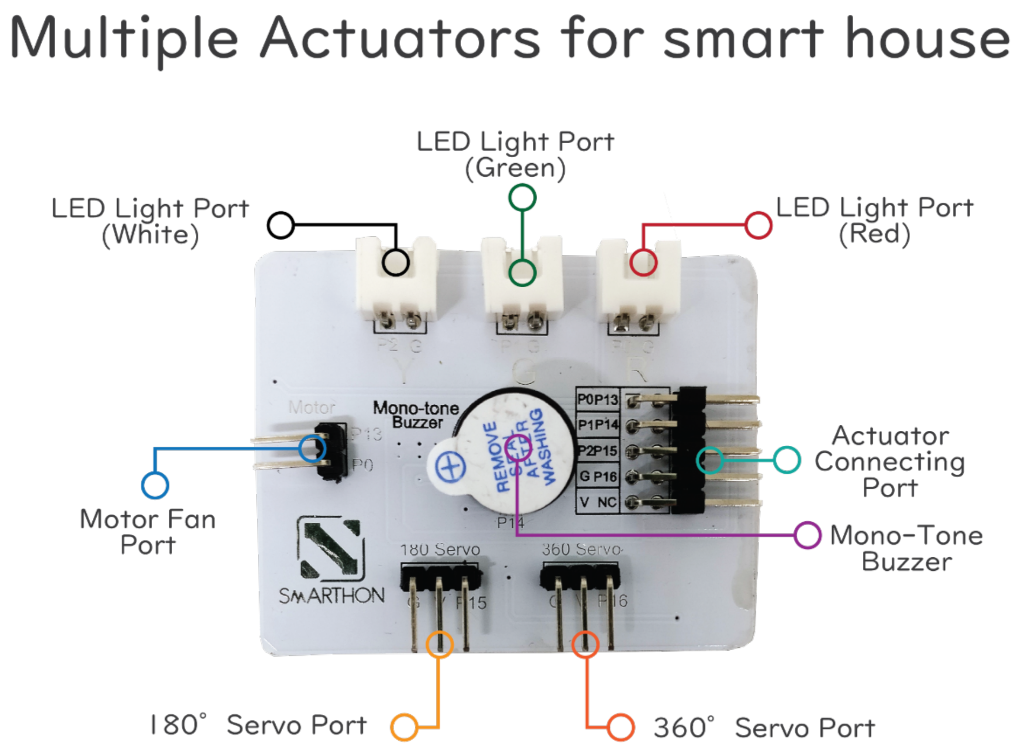
Kit components

1.3.2 Smarthon Smart House Kit

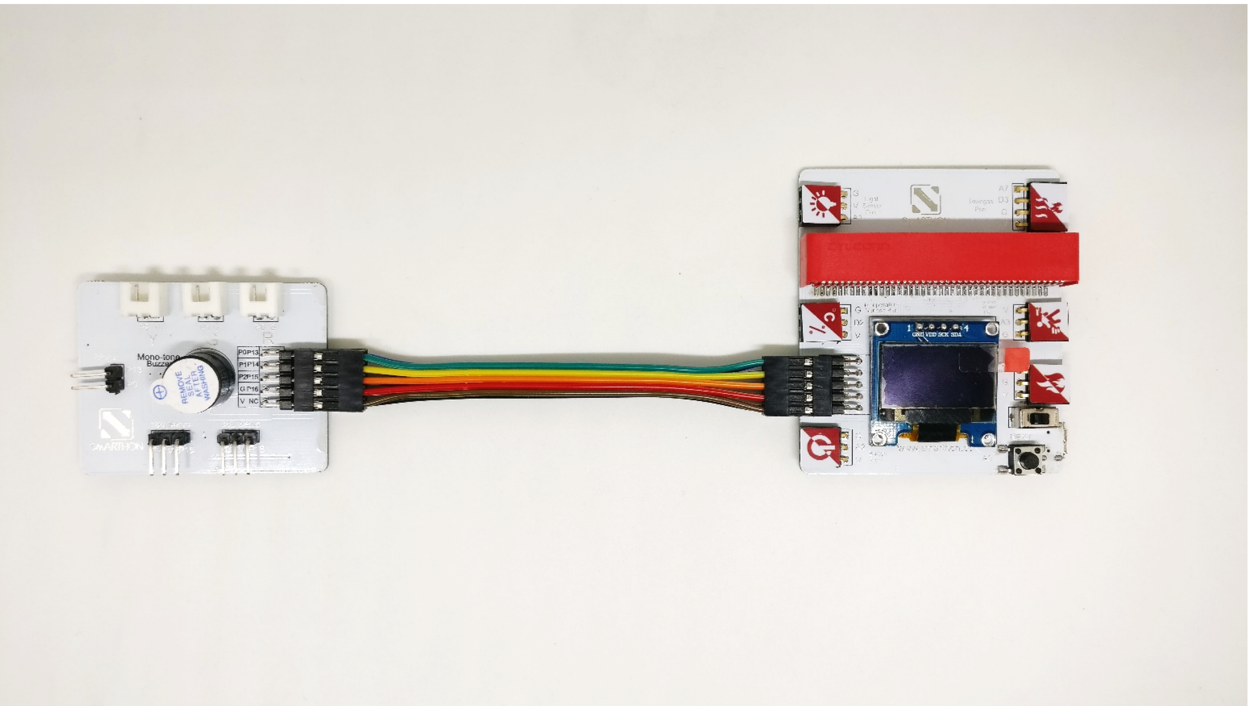
Introduction for Multiple Sensors for smart house



Introduction for Multiple Actuators for smart house



Basic Connection Between Multiple Sensors and Multiple Actuators



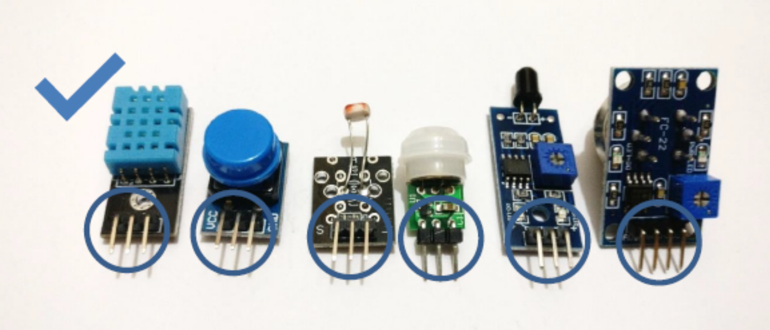
*To make sure the connecting wire is on the same direction.

1.3.3 Hardware User Guide

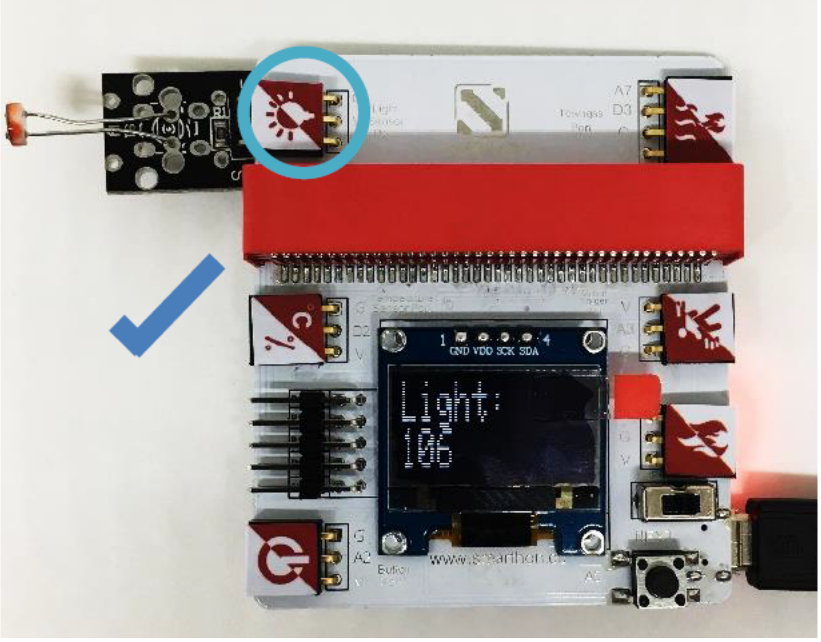
A. The Sensors

How to connect the sensor modules and Multiple Sensors for smart house?

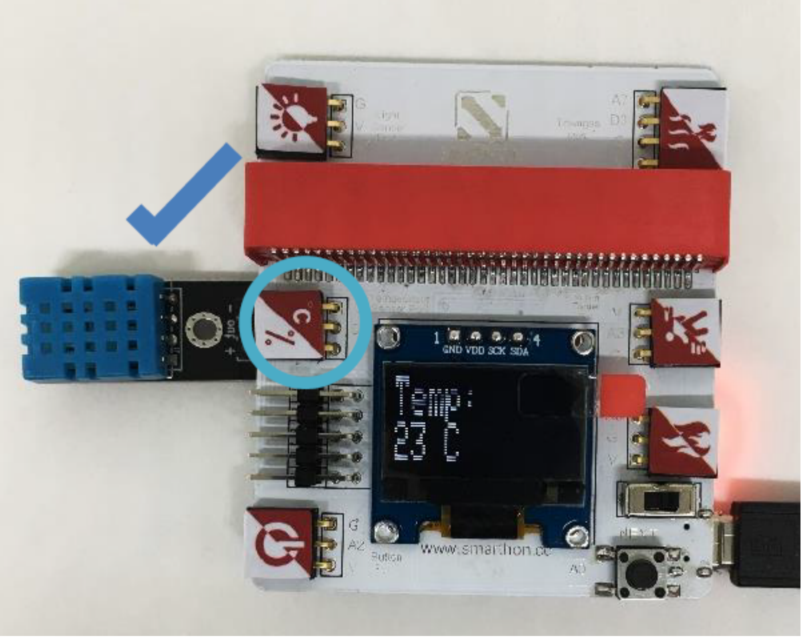
Please follow the instructions shown below to plug sensor modules into Multiple Sensors correctly to ensure all the modules work effectively.

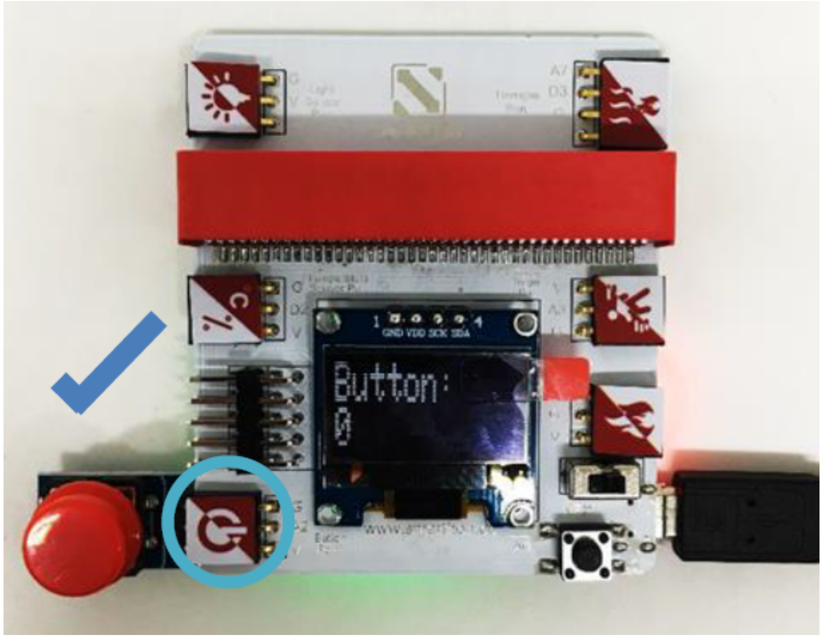
All Sensors: 

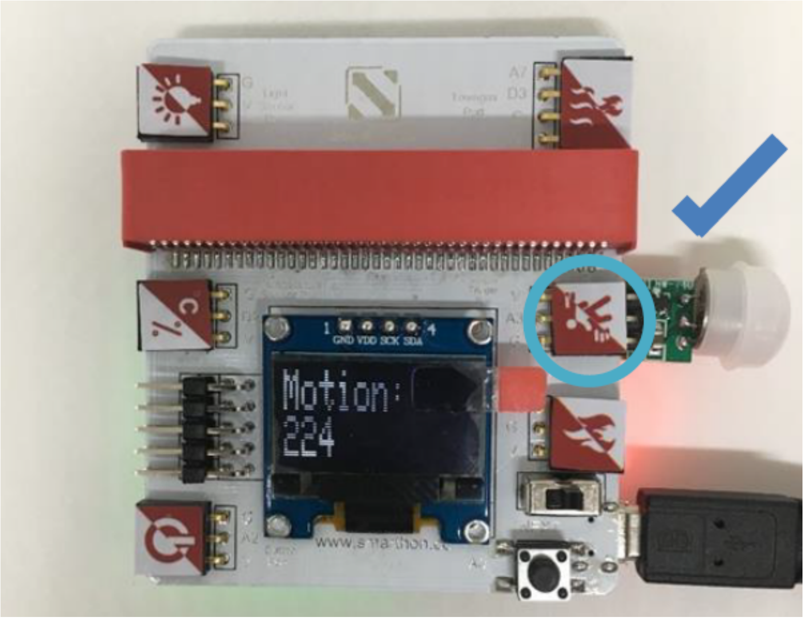
1. Light Sensor

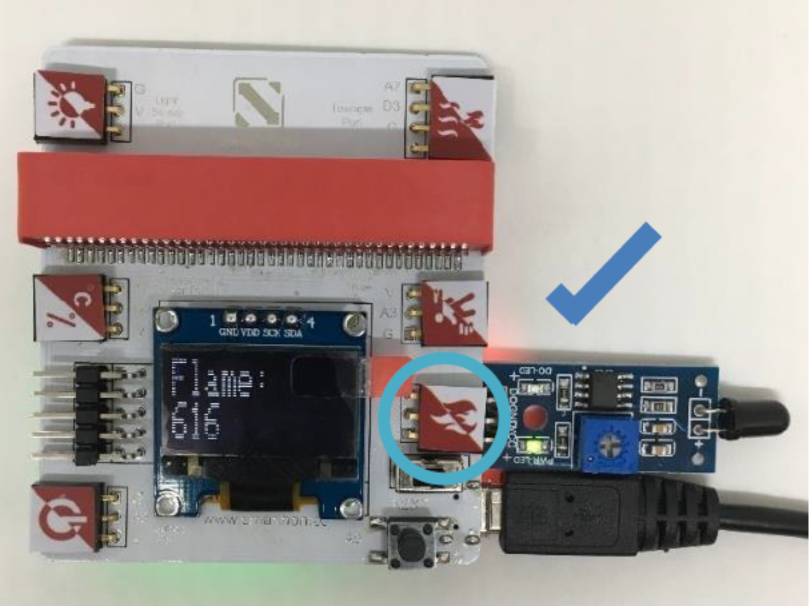


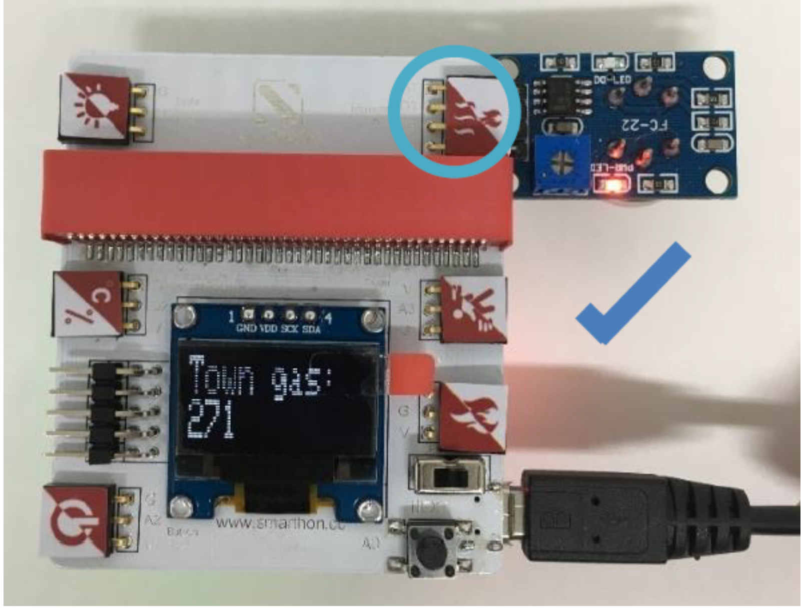
2. Temperature and Humidity Sensor



3. Button 

4. Motion Sensor 

5. Flame Sensor 

6. Town Gas Sensor 

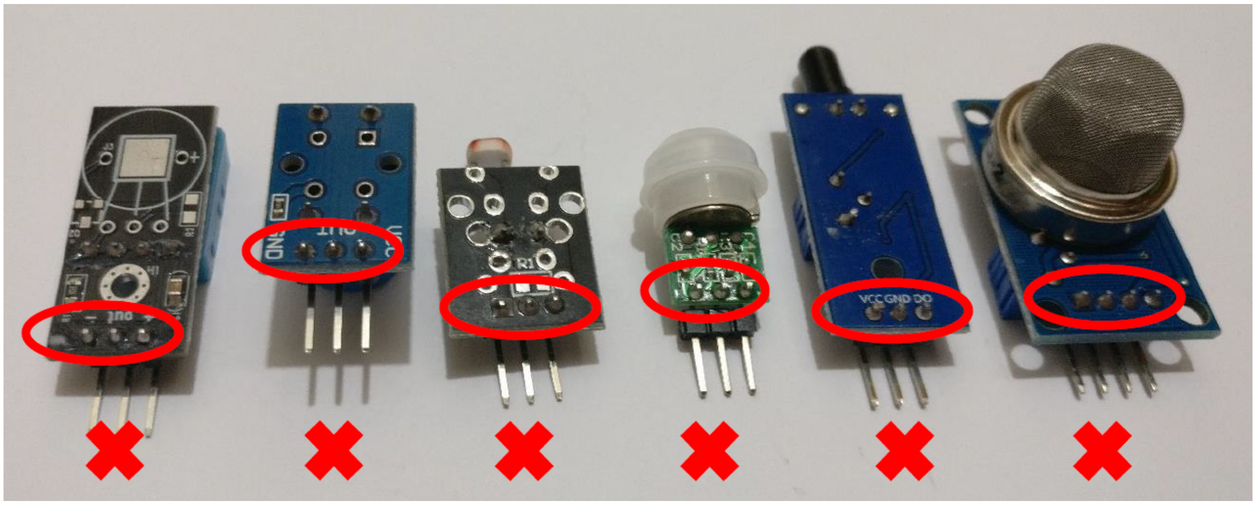
Caution!

Incorrect Plug-in:

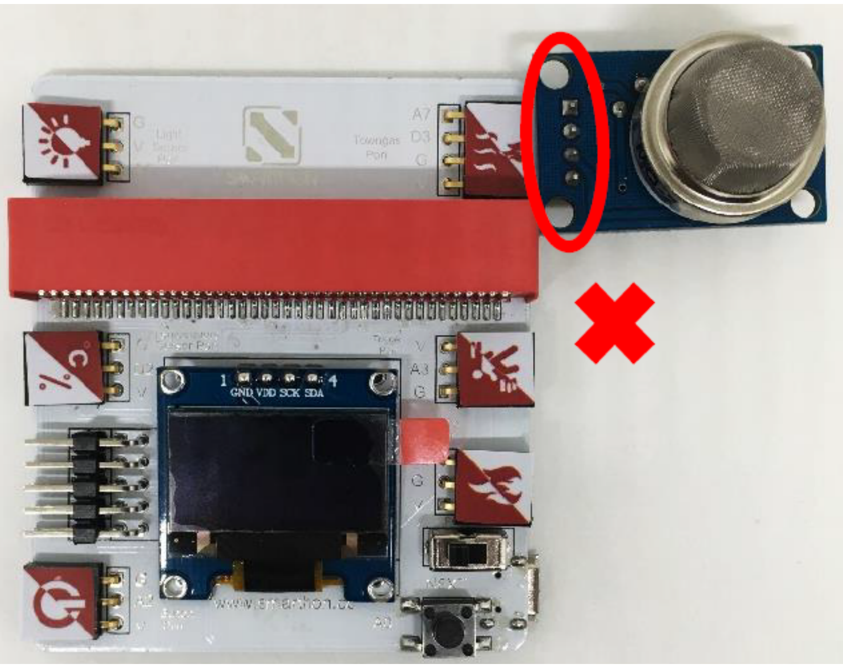
User should **NOT** plug the sensors into the Multiple Sensors for smart house at incorrect format as shown in below. Incorrectly plugging in the Multiple Sensors will cause:

1. an error of the sensor value
2. **electrical short circuit** as well as damage the sensors or Multiple Sensors for smart house

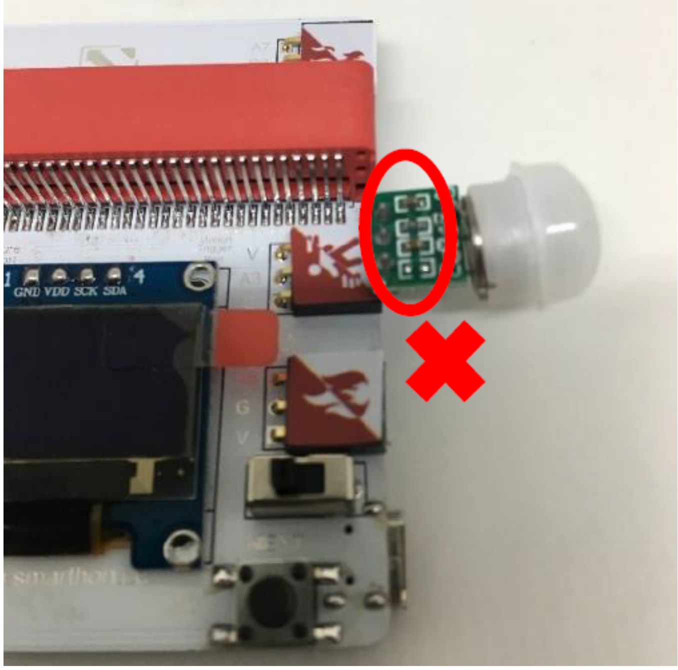
You must always be concerned with your own safety first and cut-off power supply and remove sensors under safe conditions



Incorrect Plug-in Example 1:



Incorrect Plug-in Example 2:

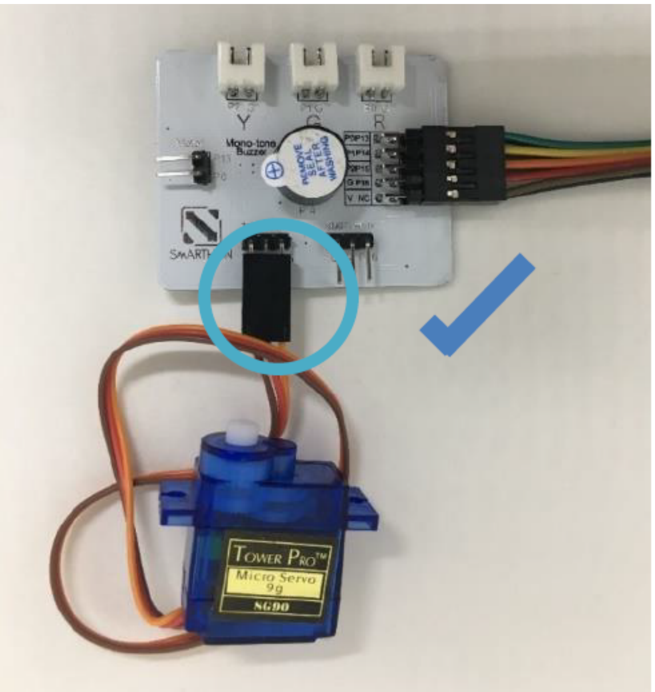


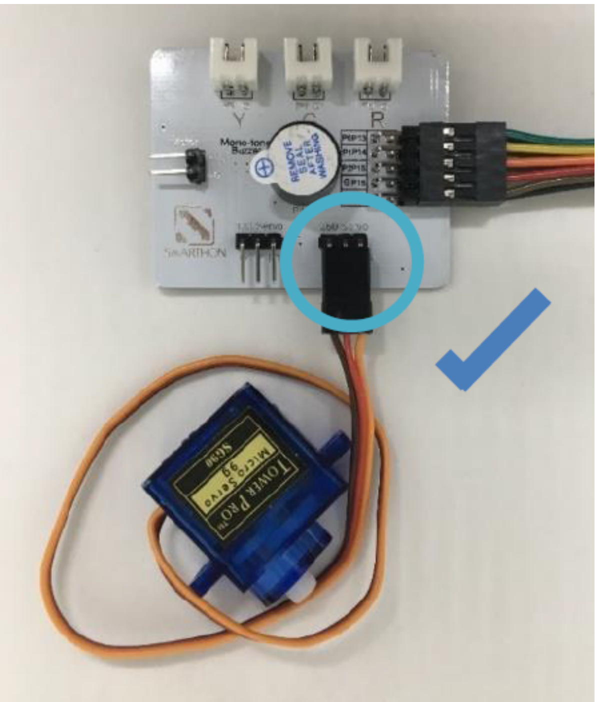
The Actuators

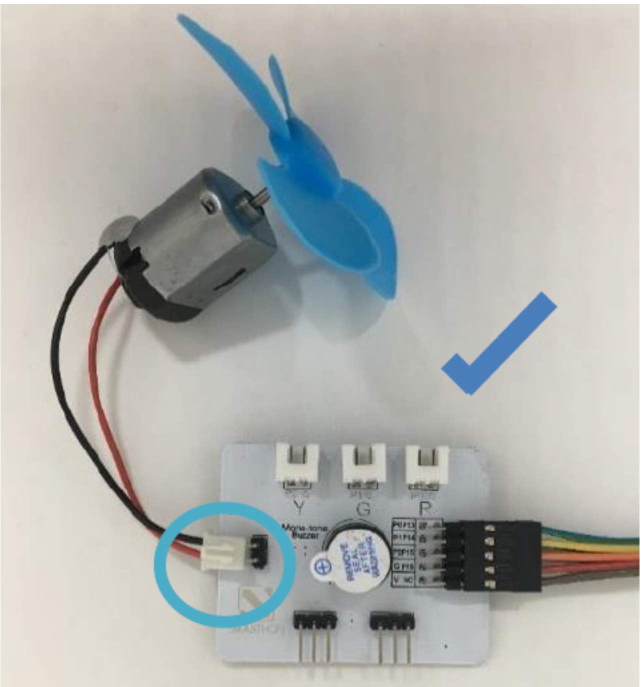
How to connect the actuator modules and Multiple Actuators for smart house?

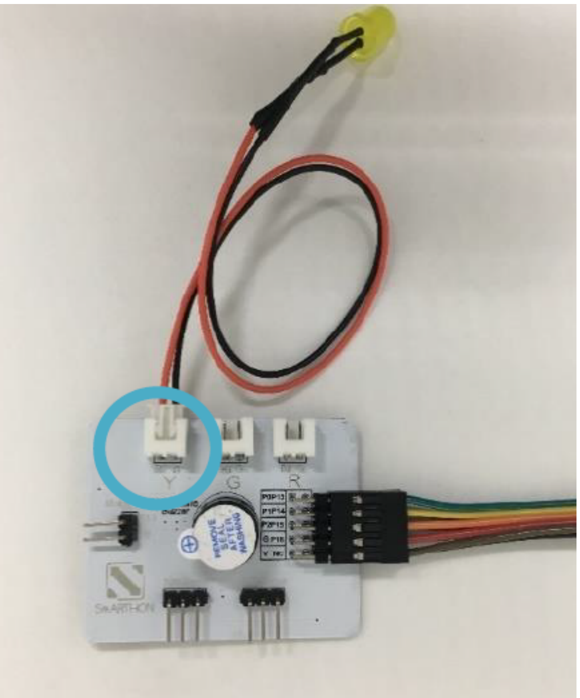
Please follow the instructions shown below to plug actuator modules into Multiple Actuators correctly to ensure all the modules work effectively.

1. 180° Servo

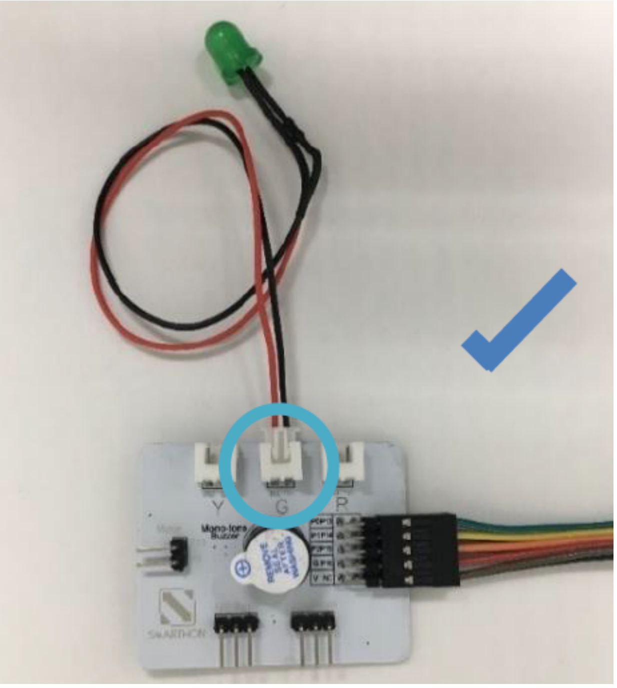


2. 360° Servo 

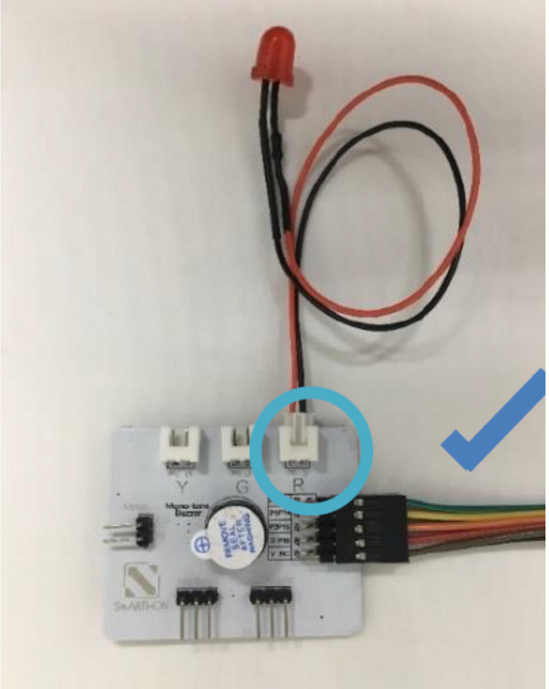
3. Motor Fan 



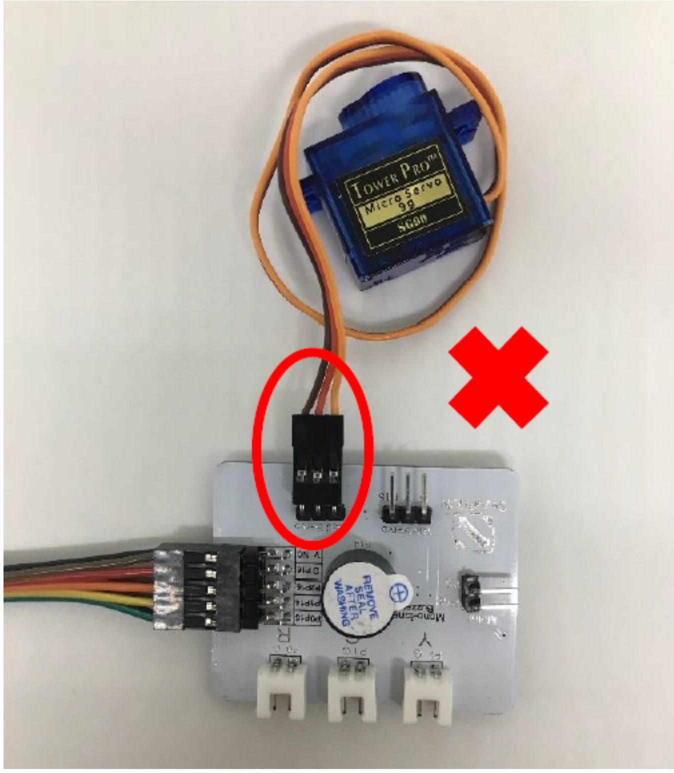
4. LED Light (Yellow)



5. LED Light (Green)

6. LED Light (Red) 

Caution! Actuators will not work properly if users connect it with the Multiple Actuators incorrectly.

Incorrect Plug-in Example: 

1.3.4 Reading all the sensors on screen: Button, Temperature & Humidity sensor, Light sensor, Motion sensor, Flame sensor and Town gas sensor

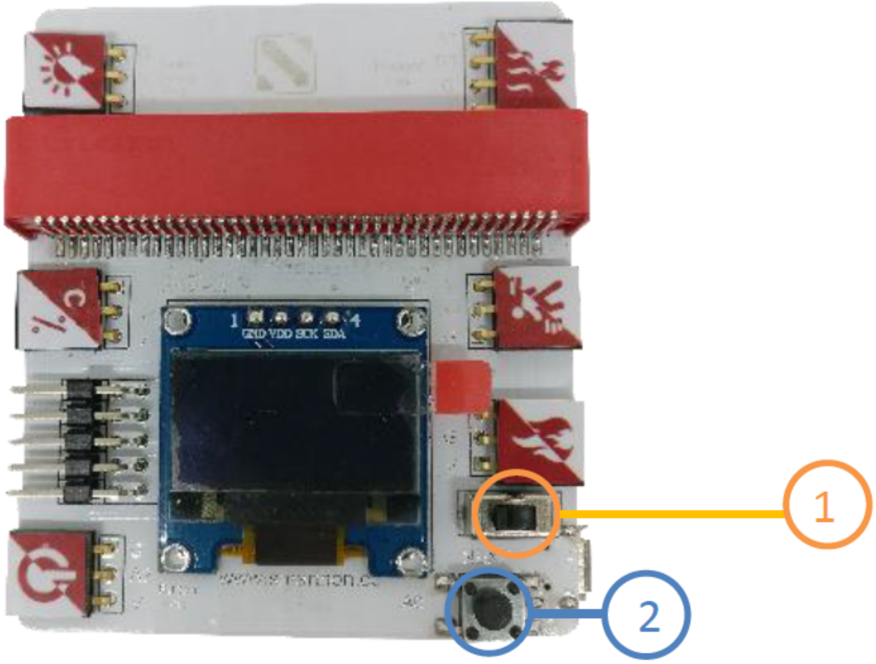
Part List:

1. Multiple Sensors for plant X1
2. Button X1
3. Temperature & Humidity sensor X1
4. Light sensor X1
5. Motion sensor X1
6. Flame sensor X1
7. Town gas sensor X1
8. USB Cable

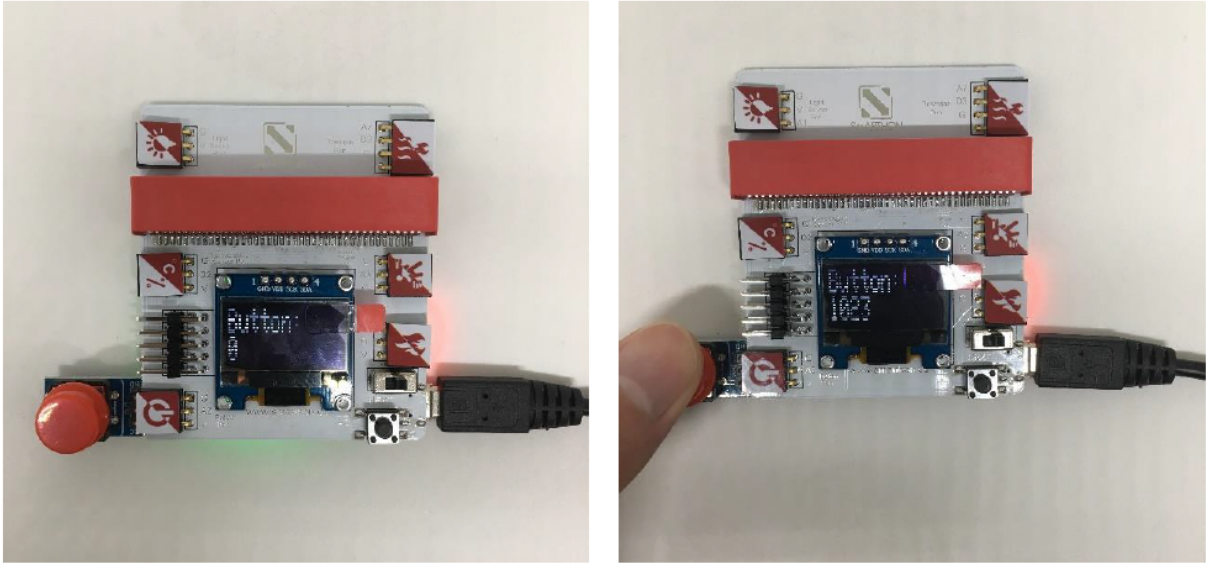
You should learn how to read the sensor value which is the foundation of the other projects.

1. Switch on the power button first (orange colour)
2. Pressed the button in the right bottom corner of the Multiple Sensors for smart house (blue colour)

It shows the instant value of the digital light sensor, temperature, humidity, button, motion, flame as well as town gas sensor.

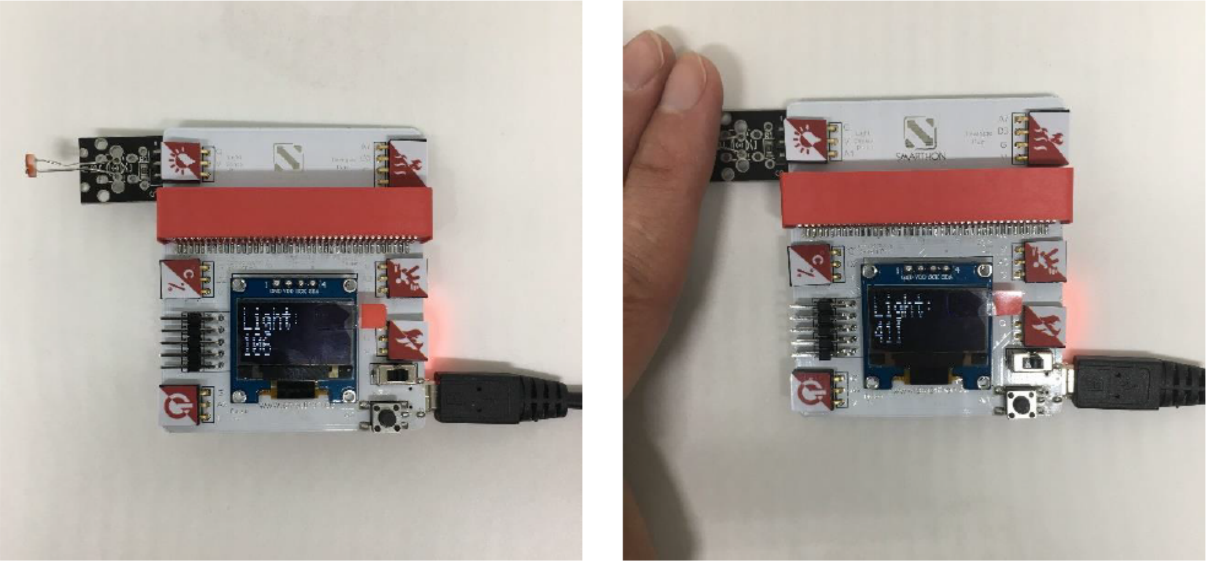


Button, Light Sensor and Town gas sensor are shown as examples:



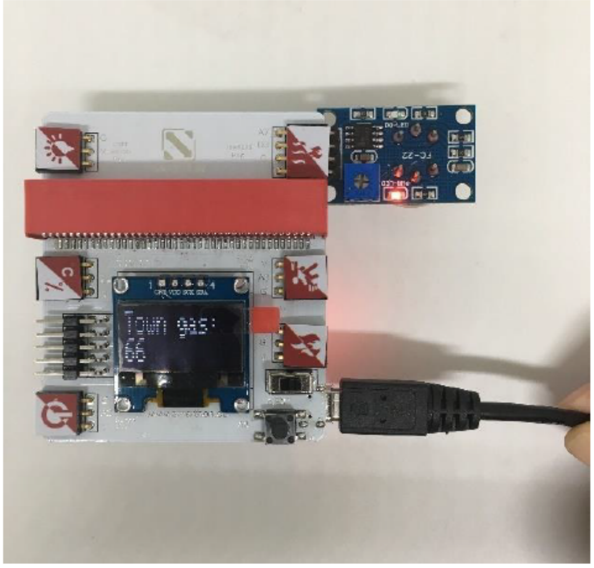
Button

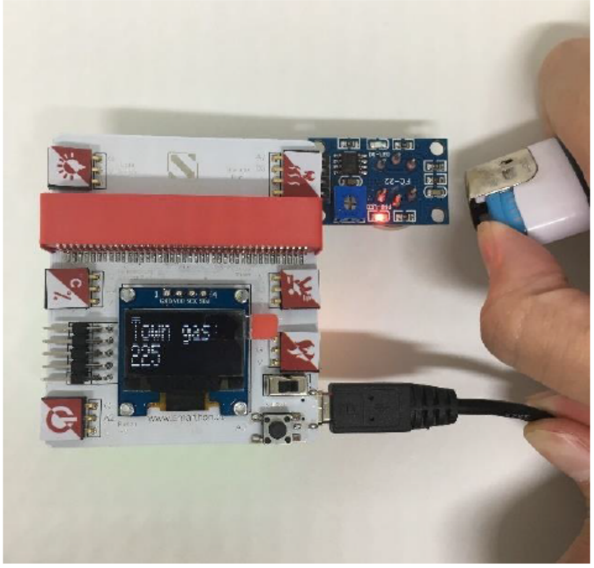
When you plug in the button, the screen shows 0 or 1 (small number); while when you pressed the button, the screen shows around 1020 - 1023 (large number). It indicates that a large number will be shown when you pressed the button.



Light Sensor

When there is a light turning on, the light sensor value is around 280-300; otherwise when you use a finger to cover the light sensor, it has increased into around 450-470. It reveals that the ambient lighting are getting brighter, the number of the light sensor value shown on the screen is getting smaller and vice versa.





Town Gas Sensor
In the normal condition, the town gas sensor value is around 30-50; while there is a town gas leakage, the number has increased from 200 – 850 subject to the concentration of different leakage velocities.

* Please conduct a gas test in a well-ventilated area.

Reading the other three sensors:

Temperature & Humidity:____

Motion sensor:____

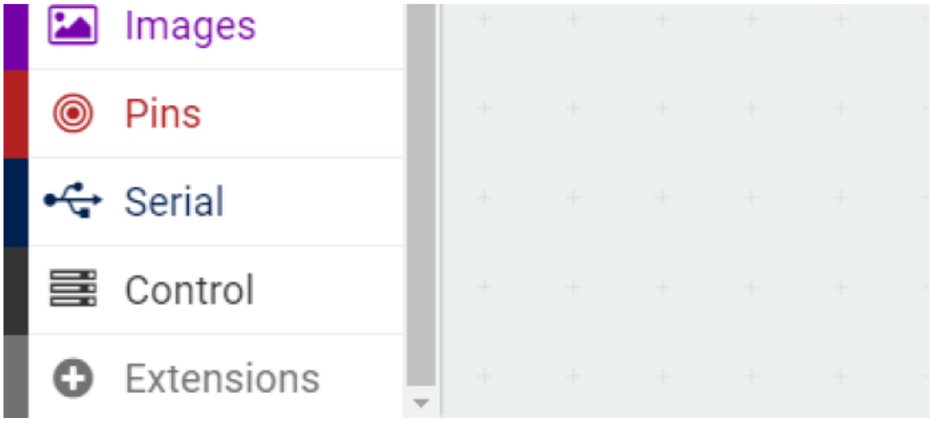
Flame sensor:____

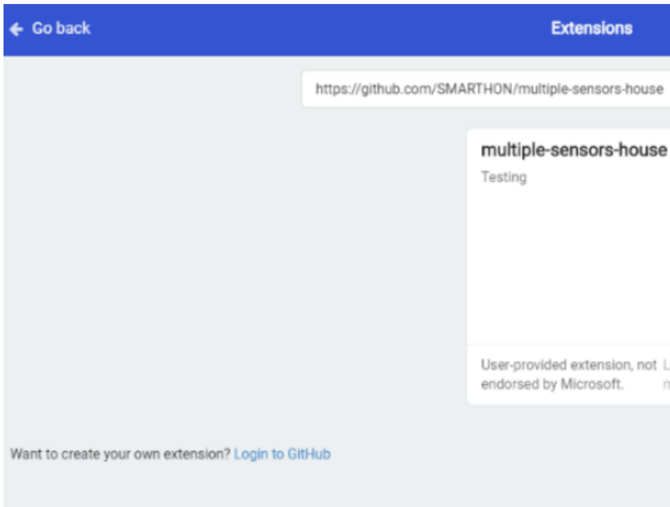
1.3.5 Getting started: How to program Smarthon Smart House Kit on micro:bit

To Program Smarthon modules with micro:bit, you need to add the Smarthon Smart House Kit packages to the BBC micro:bit makecode platform and initialize it first.

Steps:

1. Find the "Extensions" block at the bottom

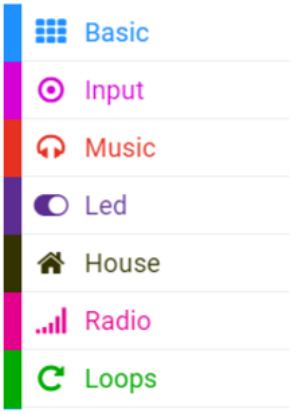


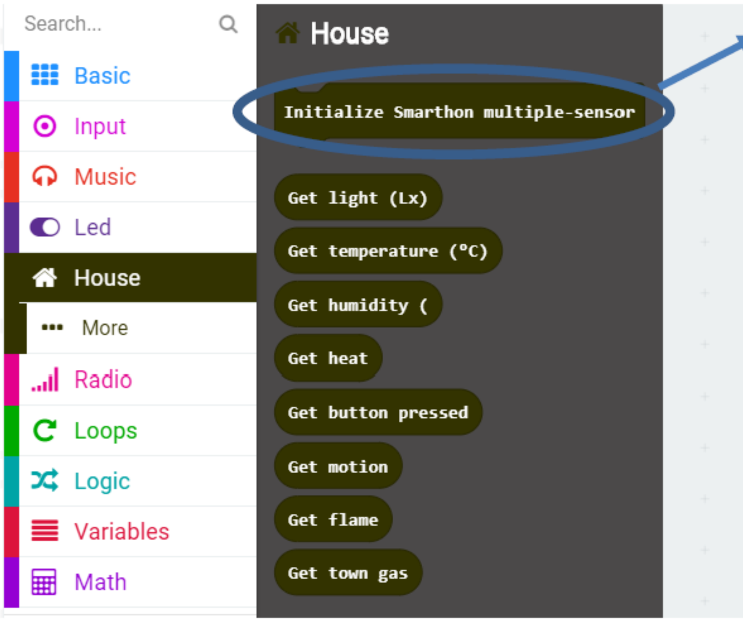


User-provided extension, not endorsed by Microsoft.

Want to create your own extension? Login to GitHub

2. Search “https://github.com/SMARTHON/multiple-sensors-house”
3. Clicked “multiple-sensors-house” and you can see “House” package has been successfully added under “Led”.





4. Initialize Smarthon Smart House Kit to add all the functions

1.3.6 Project 1: Read Sensors on micro:bit 25 LED lights

Level: ★ ☆ ☆ ☆ ☆

Part List

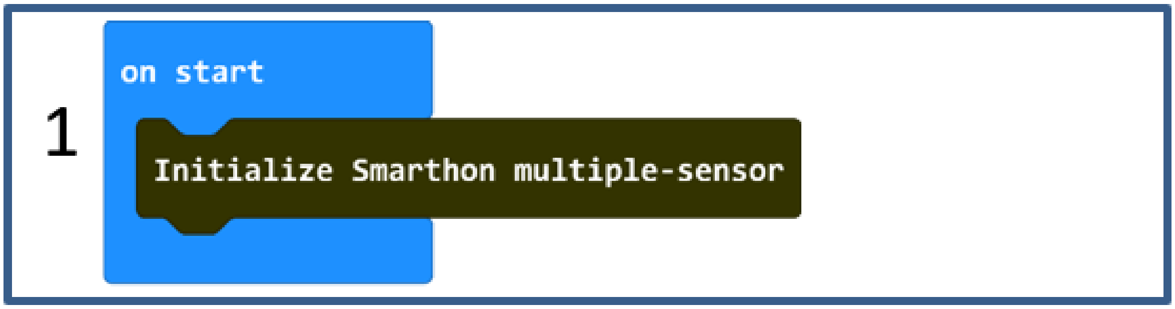
1. Temperature/Humidity Sensor X1
2. Multiple Sensors for smart house X1
3. Multiple Actuators for smart house X1
4. Connecting Wire X1
5. micro:bit X1
6. USB Cable X1

Description

You can learn how to get the value of the sensor and show on micro:bit LED. Temperature/Humidity sensor will be illustrated in this demo.

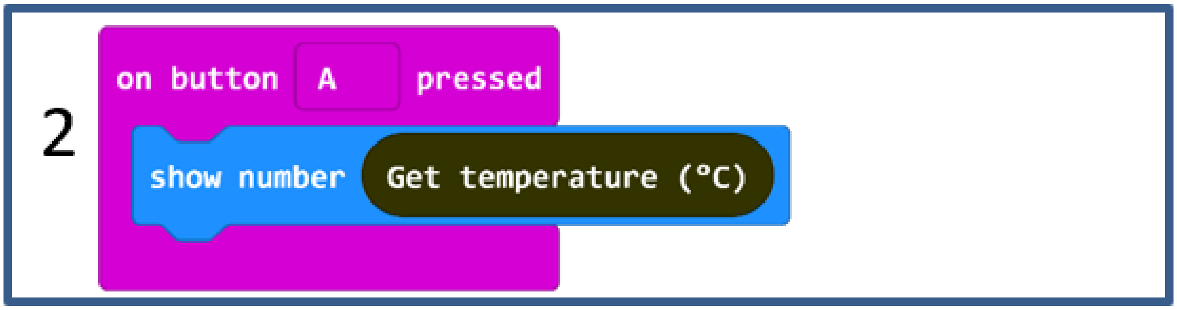
Steps

1. Basic (Add block on start) > House (Add block Initialize Smarthon multiple-sensor)
2. Input (Add block on button A is pressed) > Basic (Add block show number) > House (Add block Get temperature (°C))
3. Input (Add block on button A is pressed, change “A” into “B”) > Basic (Add block show number) > House (Add block Get humidity)



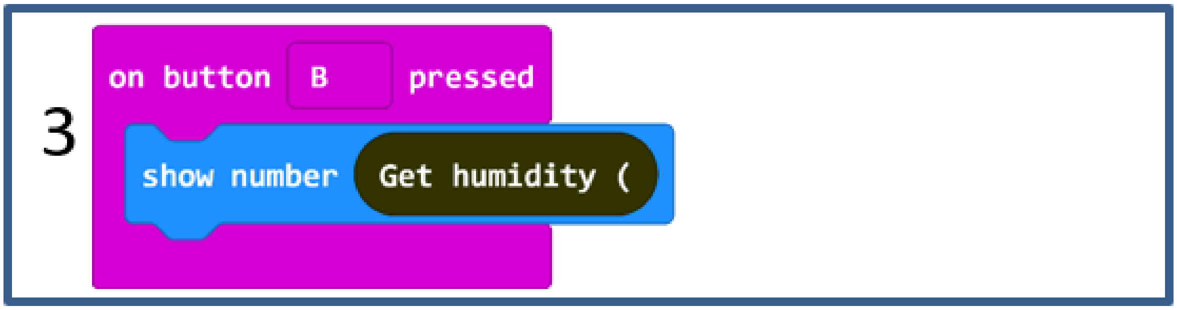
1

```
on start
    Initialize Smarthon multiple-sensor
```



2

```
on button A pressed
    show number Get temperature (°C)
```

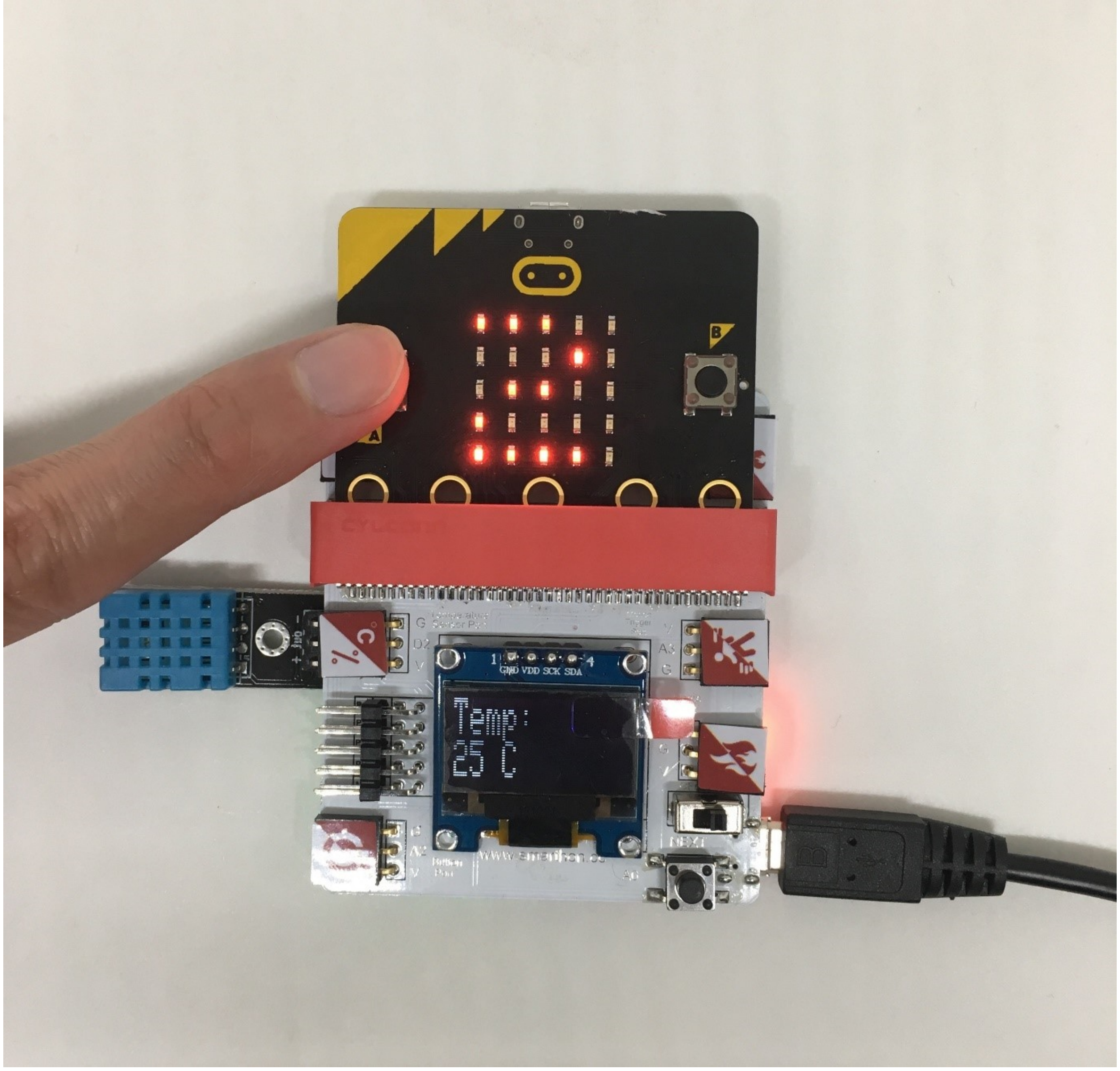


3

```
on button B pressed
    show number Get humidity (
```

Hardware

1. Connect the Temperature/Humidity sensor to Temperature Sensor port of Multiple Sensors
2. Download the code and transfer to micro:bit
3. Plug the micro:bit into Multiple Sensors
4. Pressed micro:bit button A to see the temperature and button B to see the humidity



* Try to show the other sensors value on micro:bit LED.

1.3.7 Project 2: Control Multiple Actuators Lv1

Level: ★ ☆ ☆ ☆ ☆

Part List

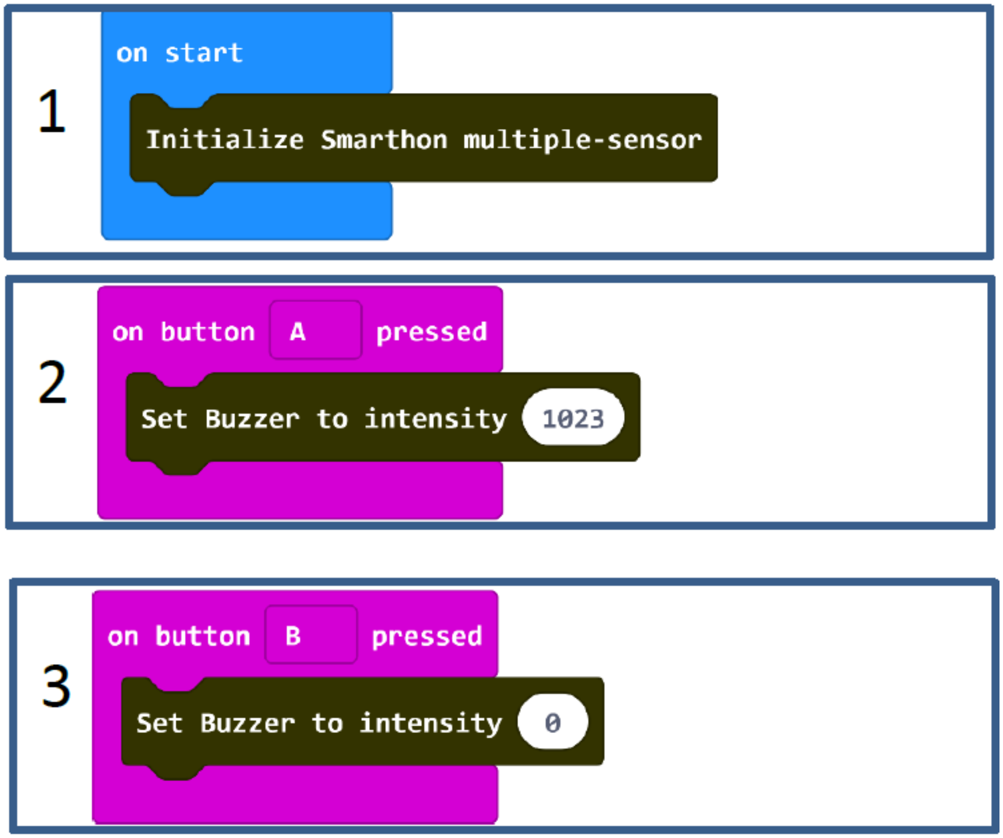
1. Mono-tone buzzer X1 (placed on Multiple Actuators for smart house)
2. Multiple Sensors for smart house X1
3. Multiple Actuators for smart house X1
4. Connecting Wire X1
5. micro:bit X1
6. USB Cable X1

Description

In project 2, you can learn how to use micro:bit button A & B to control mono-tone buzzer as an example in this demo.

Steps

1. Basic (Add block on start) > House (Add block Initialize Smarthon multiple-sensor)
2. Input (Add block on button A is pressed) > House-More (Add block Set Buzzer to intensity 0, change “0” to “1023”)
3. Input (Add block on button A is pressed, change “A” into “B”) > House (Add block Set Buzzer to intensity “0”)



1

```
on start
    Initialize Smarthon multiple-sensor
```

2

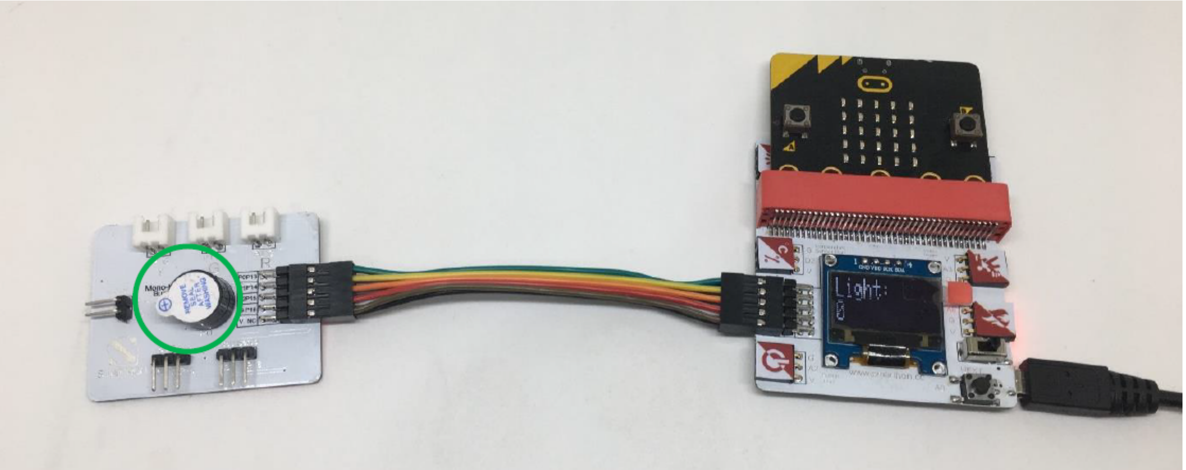
```
on button A pressed
    Set Buzzer to intensity 1023
```

3

```
on button B pressed
    Set Buzzer to intensity 0
```

Hardware

1. **Download** the code and transfer to micro:bit.
2. Plug the micro:bit into Multiple Sensors.
3. Connect Multiple Sensors and Multiple Actuators by connecting wire.
4. Pressed micro:bit button A to turn to the buzzer and button B to turn off the buzzer.



* Try to use on shake to turn on and off buzzer.

1.3.8 Project 3: Control Multiple Actuators Lv2

Level: ★ ★ ★ ☆ ☆

Part List

1. Mono-tone buzzer X1 (placed on Multiple Actuators for smart house)
2. Multiple Sensors for smart house X1
3. Multiple Actuators for smart house X1
4. Connecting Wire X1
5. micro:bit X1
6. USB Cable X1

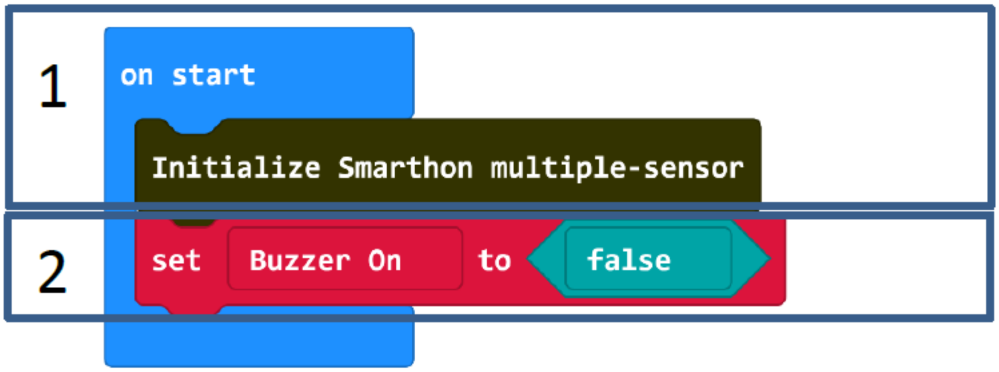
Description

In project 3, this is level two to control multiple actuators and you can learn how to use micro:bit on shake to control mono-tone buzzer on and off together.

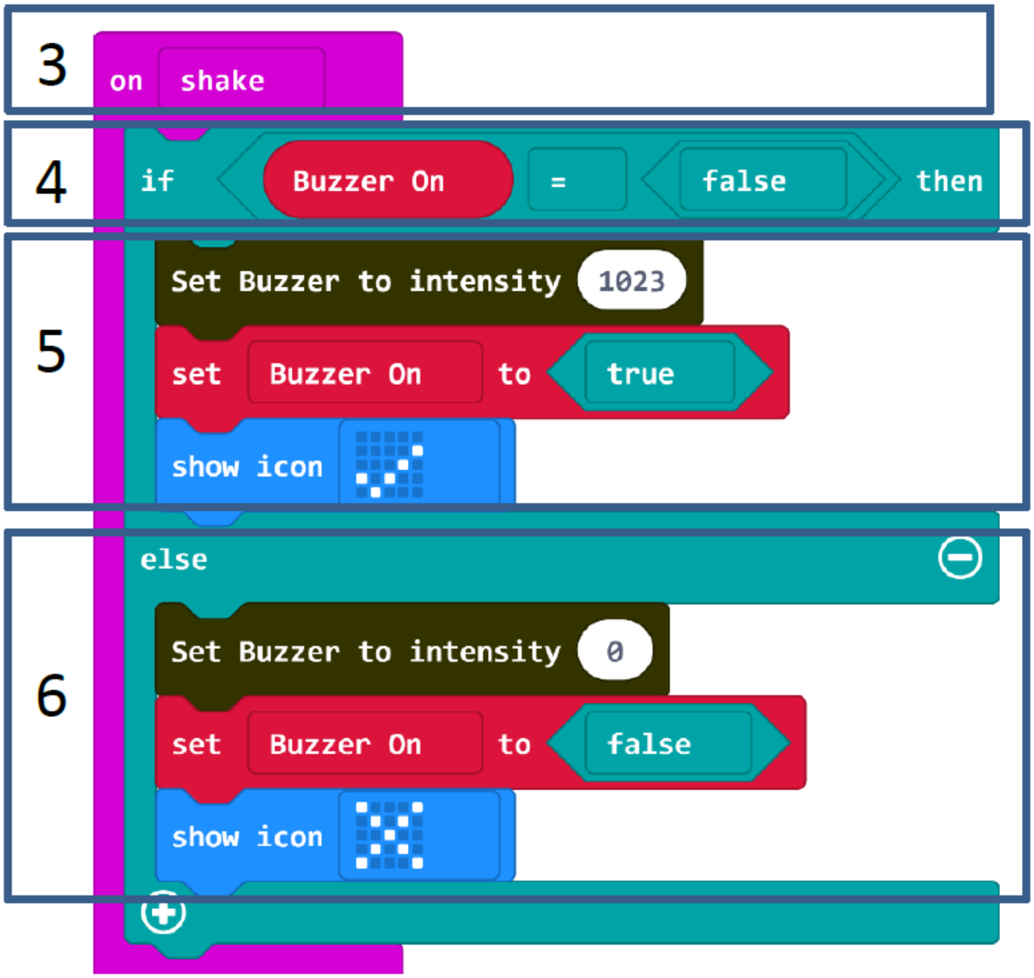
Steps

1. Basic (Add block on start) > House (Add block Initialize Smarthon multiple-sensor)
2. Variables (Make a Variable "Buzzer On") > Variables (Add block set item to 0, change "item" to "Buzzer On"; "0" to Logic "false")
3. Input (Add block on shake)
4. Logic (Add block if. . . then. . . else) > For if : Logic (Add block 0 = 0, change "0" to Variables "Buzzer On"; "0" to Logic "false")
5. For then: House-More (Add block Set Buzzer to intensity 0, change "0" to "1023") > Variables (Add block set item to 0, change "item" to "Buzzer On"; "0" to Logic "true") > Basic (Add block show icon, change "heart" to "yes")

6. For else: House-More (Add block Set Buzzer to intensity 0) > Variables (Add block set item to 0, change “item” to “Buzzer On”; “0” to Logic “false”) > Basic (Add block show icon, change “heart” to “no”)



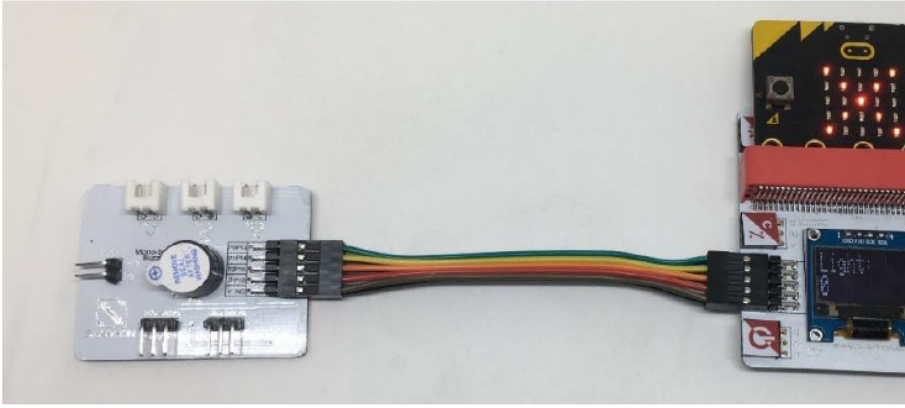
```
1 on start
    Initialize Smarthon multiple-sensor
2   set Buzzer On to false
```



```
3 on shake
4   if Buzzer On = false then
5     Set Buzzer to intensity 1023
      set Buzzer On to true
      show icon
6   else
      Set Buzzer to intensity 0
      set Buzzer On to false
      show icon
```

Hardware

1. **Download** the code and transfer to micro:bit
2. Plug the micro:bit into Multiple Sensors
3. Connect Multiple Sensors and Multiple Actuators by connecting wire



4. Shook the micro:bit to turn on/off the buzzer
   * Try to use on shake to turn on/off other actuators.

1.3.9 Project 4: Open the door

Level: ★ ★ ☆ ☆ ☆

Part List

1. Button X1
2. 180° servo X1
3. Multiple Sensors for smart house X1
4. Multiple Actuators for smart house X1
5. Connecting Wire X1
6. micro:bit X1
7. USB Cable X1
8. Lolly Sticks X2

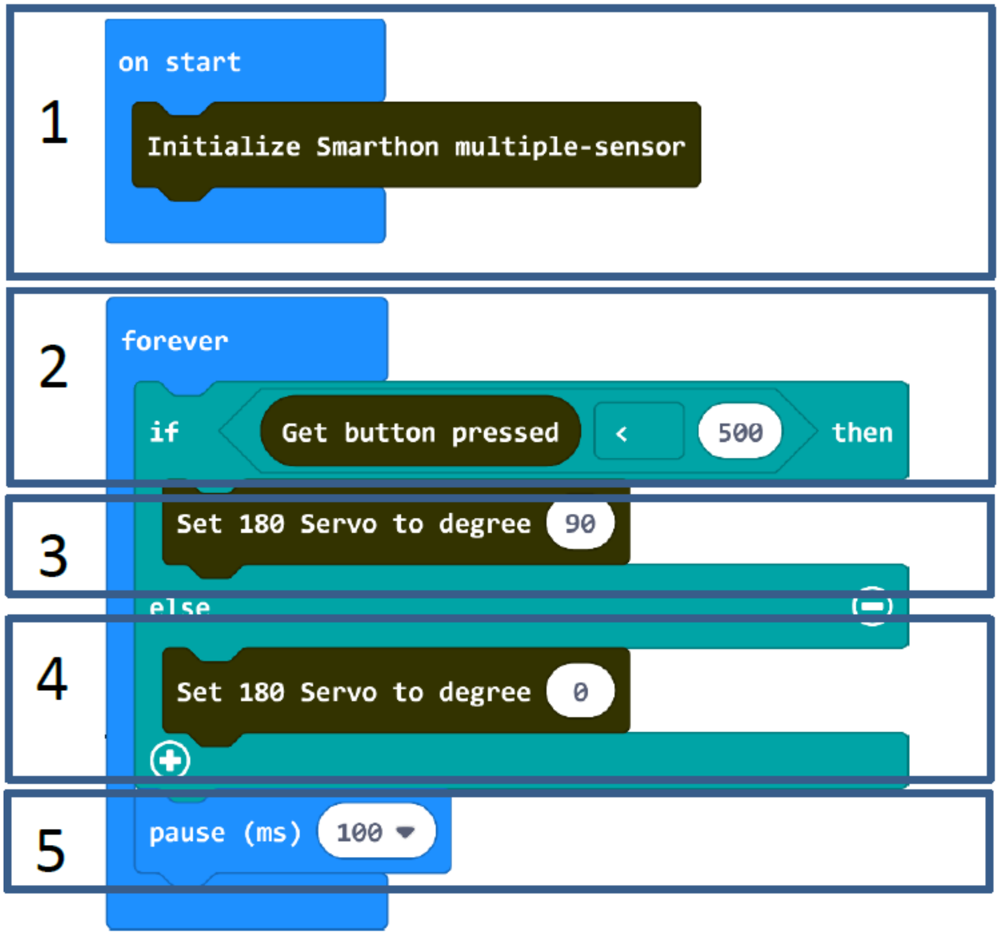
Description

In project 4, you can learn how to use the button to open the door.

Steps

1. Basic (Add block on start) > House (Add block Initialize Smarthon multiple-sensor)
2. Basic (Add block forever) > Logic (Add block if…then…else) > For if : Logic (Add block 0 = 0, change "0" to House "Get button pressed"; "=" to "<"; "0" to "500")
3. For then: House (Add block Set 180 Servo to degree 0, change "0" to "90")
4. For else: House (Add block Set 180 Servo to degree 0)
5. Basic (Add block pause (ms) 100)

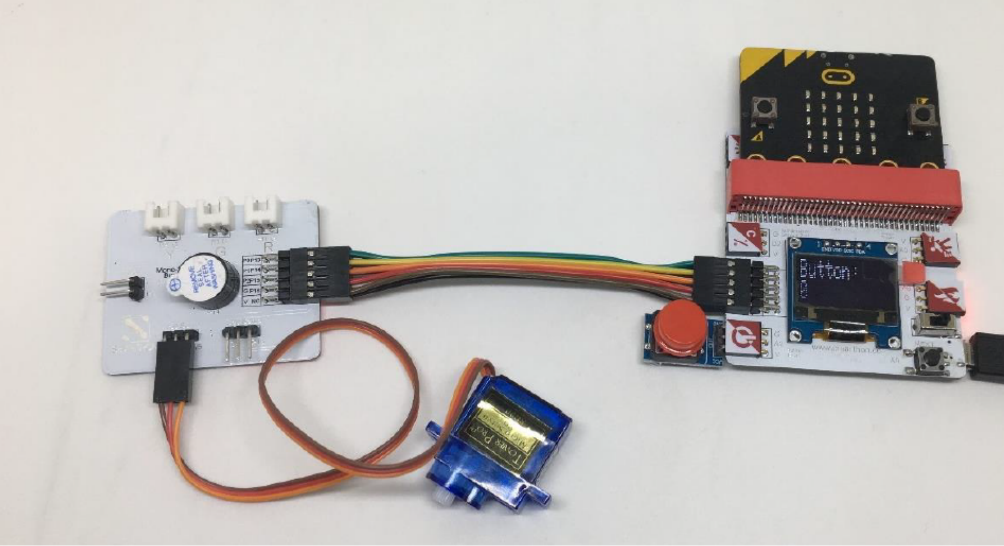
*degrees of the servo have to be adjusted under the different condition



```
on start
    Initialize Smarthon multiple-sensor

forever
    if Get button pressed < 500 then
        Set 180 Servo to degree 90
    else
        Set 180 Servo to degree 0
    pause (ms) 100
```

1. **Download** the code and transfer to micro:bit
2. Plug the micro:bit into Multiple Sensors
3. Connect Multiple Sensors and Multiple Actuators by connecting wire
4. Put the horn into 180° servo
5. Stick "+" with the lolly sticks and stick it up on the horn
6. Connect the module to 180° servo port of the Multiple Actuators and install it to the door

7. Pressed the button to open the door



   * Try to use the other sensors, such as motion sensor to control the door.

1.3.10 Project 5: Alarm System

Level: ★ ★ ☆ ☆ ☆

Part List

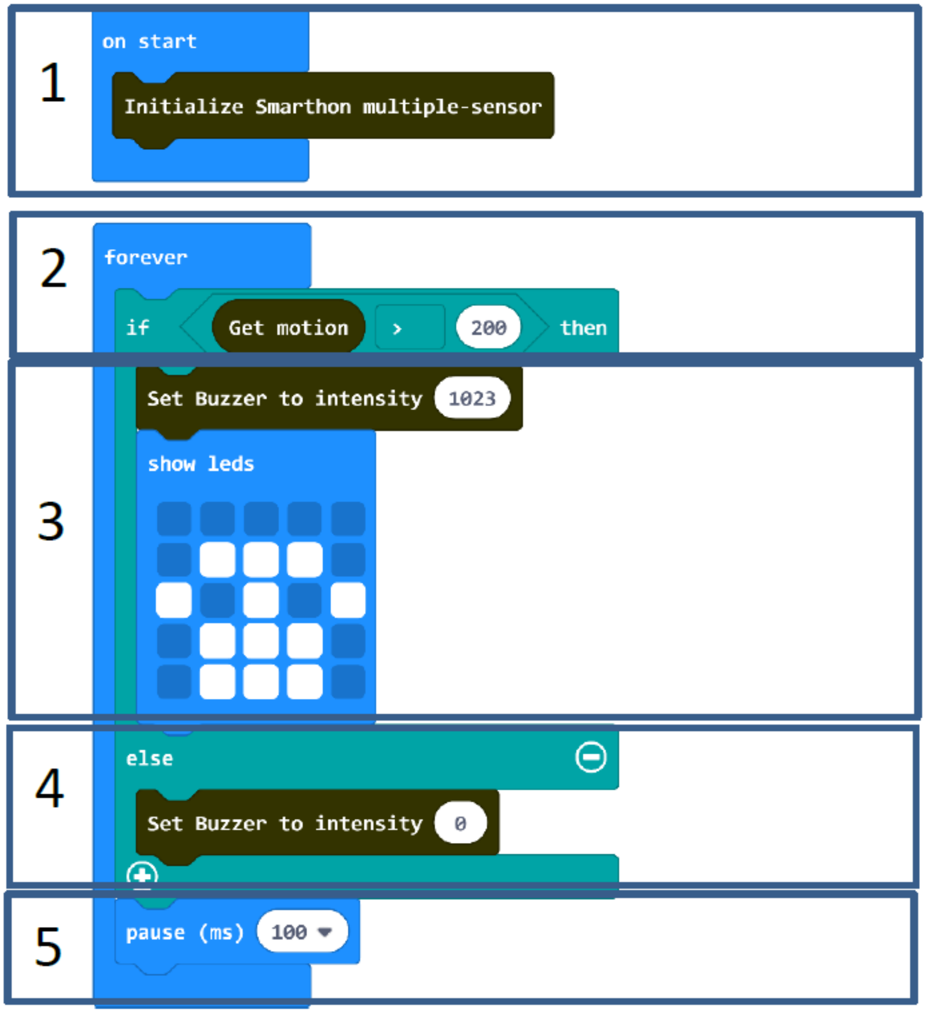
1. Motion sensor X1
2. Mono-tone buzzer X1 (placed on Multiple Actuators for smart house)
3. Multiple Sensors for smart house X1
4. Multiple Actuators for smart house X1
5. Connecting Wire X1
6. micro:bit X1
7. USB Cable X1

Description

In project 5, you can learn how to use motion sensor and buzzer to design an alarm system to maintain the house security.

Steps

1. Basic (Add block on start) > House (Add block Initialize Smarthon multiple-sensor)
2. Basic (Add block forever) > Logic (Add block if…then…else) > For if : Logic (Add block 0 = 0, change “0” to House “Get motion”; “=” to “>”; “0” to “200”)
3. For then: House-More (Add block Set Buzzer to intensity 0, change “0” to “1023”) > Basic (Add block show leds – create a telephone symbol)
4. For else: House-More (Add block Set Buzzer to intensity 0)

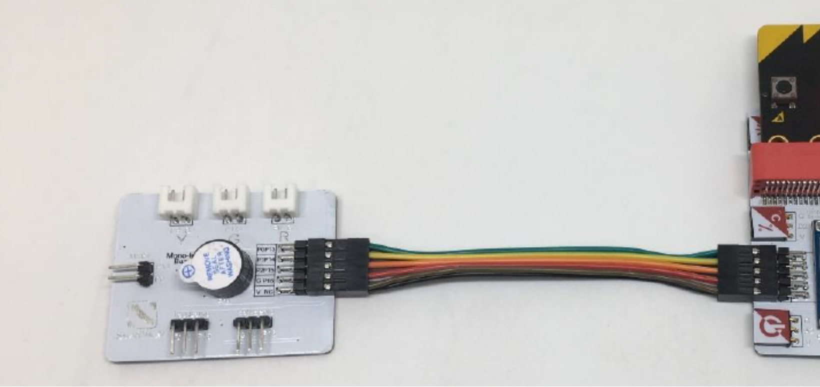


```
on start
    Initialize Smarthon multiple-sensor

forever
    if  Get motion  >  200  then
        Set Buzzer to intensity 1023
        show leds
    else
        Set Buzzer to intensity 0
    pause (ms) 100
```

5. Basic (Add block pause (ms) 100)

Hardware

1. Connect the **Motion sensor** to Motion Trigger port of Multiple Sensors
2. **Download** the code and transfer to micro:bit
3. Plug the micro:bit into Multiple Sensors
4. Connect Multiple Sensors and Multiple Actuators by connecting wire



5. Try shake your hand nearby the motion sensor to test
   * Try to use other actuators, such as LED lights (green and red) to make an alarm system.

1.3.11 Project 6: Light Automation

Level: ★ ★ ★ ☆ ☆

Part List

1. Light sensor X1
2. Yellow LED X1
3. Multiple Sensors for smart house X1
4. Multiple Actuators for smart house X1
5. Connecting Wire X1
6. micro:bit X1
7. USB Cable X1

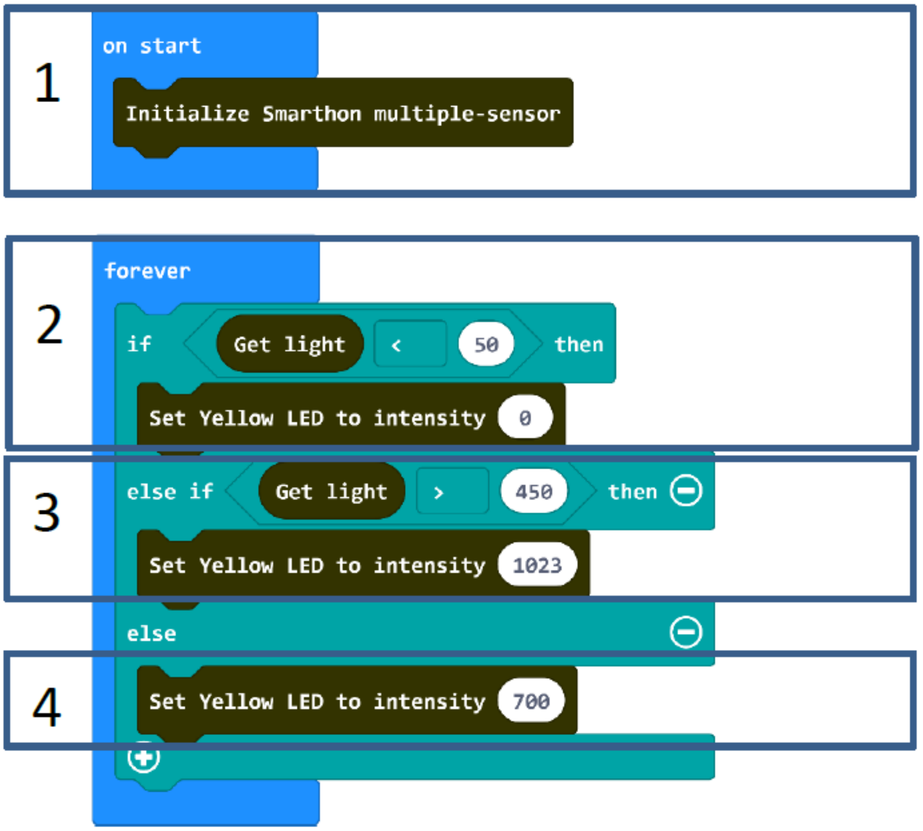
Description

In project 6, you can learn how to build a light automation in the house subject to the sunshine.

Steps

1. Basic (Add block on start) > House (Add block Initialize Smarthon multiple-sensor)
2. Basic (Add block forever) > Logic (Add block if…then…else, add 1 more if…then…else) > For 1st if : Logic (Add block 0 = 0, change “0” to House “Get light”; “=” to “<”; “0” to “50”) > For then: House-More (Add block Set Yellow LED to intensity 0)
3. For else if: Logic (Add block 0 = 0, change “0” to House “Get light”; “=” to “>”; “0” to “450”) > For then: House-More (Add block Set Yellow LED to intensity 0, change “0” to “1023”)

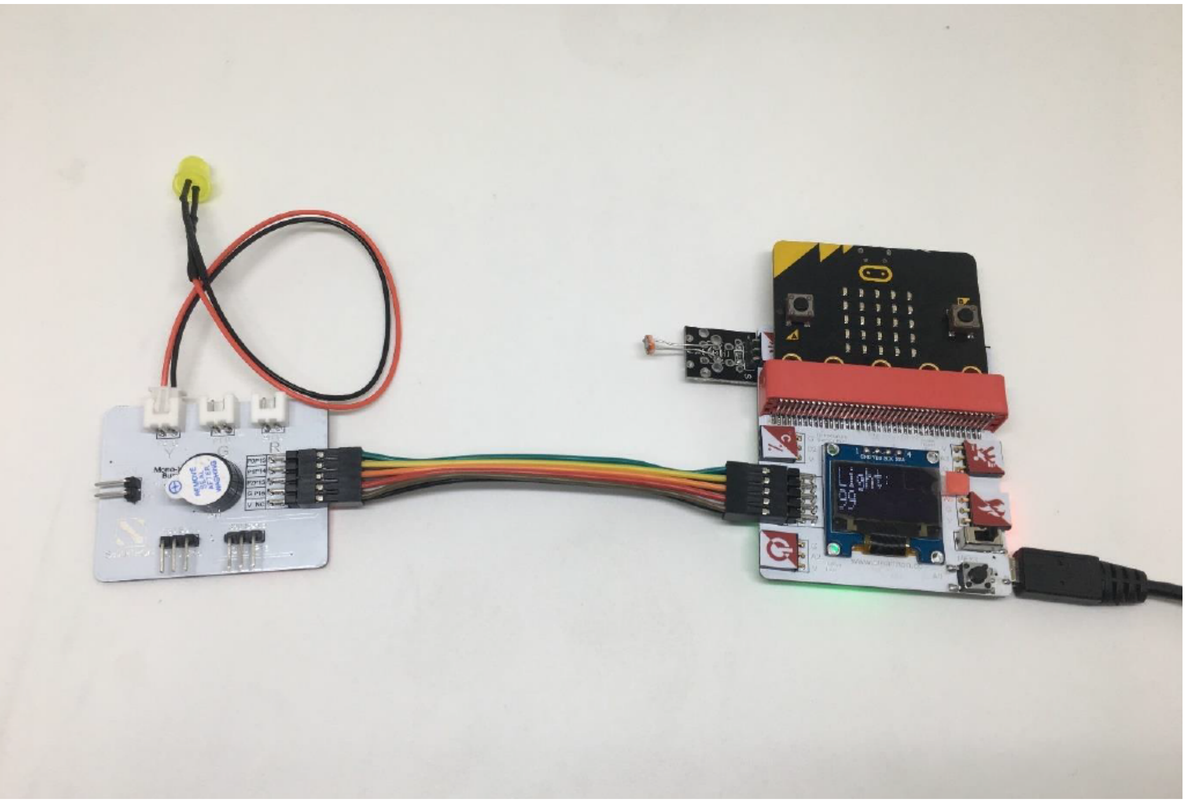
4. For else: House-More (Add block Set Yellow LED to intensity 0, change “0” to “700”)



```
on start
    Initialize Smarthon multiple-sensor

forever
    if Get light < 50 then
        Set Yellow LED to intensity 0
    else if Get light > 450 then
        Set Yellow LED to intensity 1023
    else
        Set Yellow LED to intensity 700
```

Hardware

1. Connect the **Light sensor** to Light Sensor port of Multiple Sensors
2. **Download** the code and transfer to micro:bit
3. Plug the micro:bit into Multiple Sensors
4. Connect Multiple Sensors and Multiple Actuators by connecting wire
5. Connect the Yellow LED to Y port of the Multiple Actuators
6. Use the finger to cover the light sensor to see if the LED light can be turn on / Shining a torch to the light sensor to see if the LED light can be turn off



* Try to conduct the experiment under the dark environment or sunshine and adjust the value of the sensors or LED light to fit the different projects.

1.3.12 Project 7: Smart Motor Fan

Level: ★ ★ ★ ☆ ☆

Part List

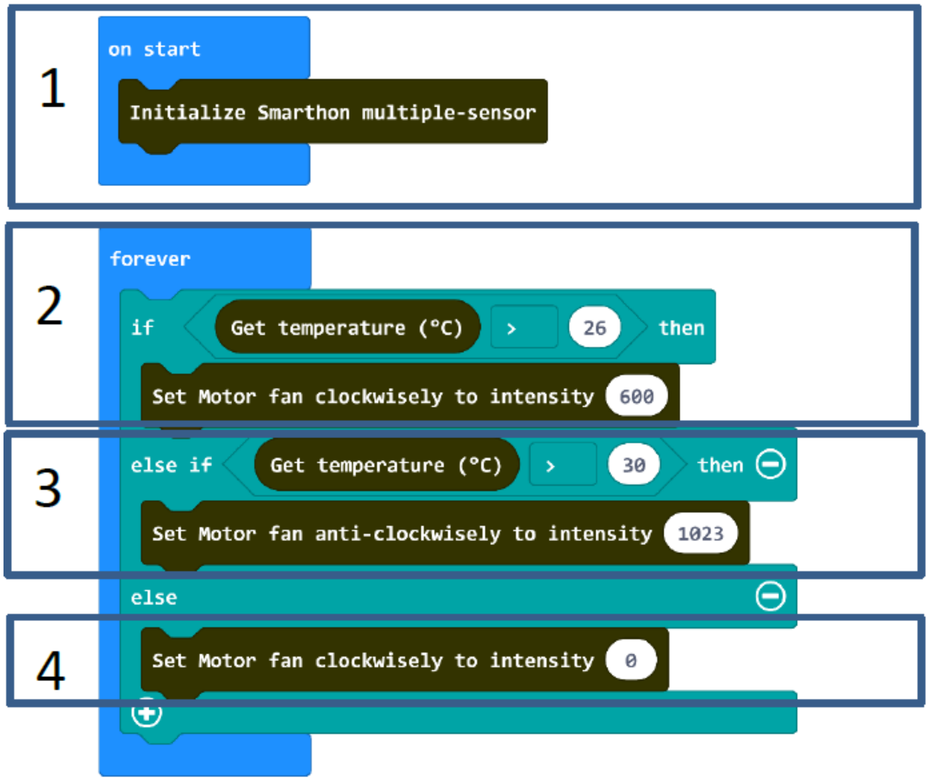
1. Temperature/Humidity sensor X1
2. Motor Fan X1
3. Multiple Sensors for smart house X1
4. Multiple Actuators for smart house X1
5. Connecting Wire X1
6. micro:bit X1
7. USB Cable X1

Description

In project 7, you can learn how to make an smart fan automation that allows you to design a smarter and more comfortable house.

Steps

1. Basic (Add block on start) > House (Add block Initialize Smarthon multiple-sensor)
2. Basic (Add block forever) > Logic (Add block if…then…else, add 1 more if…then…else) > For 1st if : Logic (Add block 0 = 0, change “0” to House “Get temperature(°C)”; “=” to “>”; “0” to “26”) > For then: House-More (Add block Set Motor fan clockwisely to intensity 0, change “0” to “600”)
3. For else if: Logic (Add block 0 = 0, change “0” to House “Get temperature(°C)”; “=” to “>”; “0” to “30”) > For then: House-More (Add block Set Motor fan anti-clockwisely to intensity 0, change “0” to “1023”)
4. For else: House-More (Add block Set Motor fan clockwisely to intensity 0)

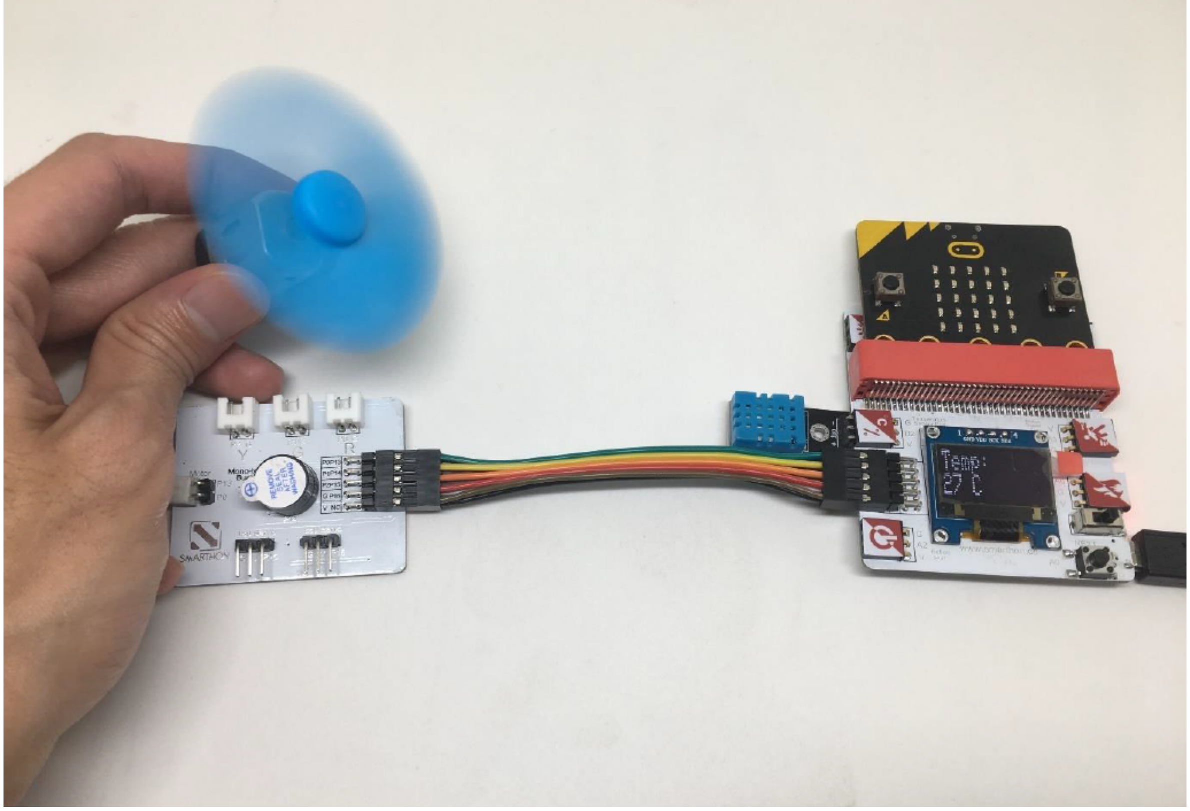


```
on start
    Initialize Smarthon multiple-sensor

forever
    if Get temperature (°C) > 26 then
        Set Motor fan clockwisely to intensity 600
    else if Get temperature (°C) > 30 then
        Set Motor fan anti-clockwisely to intensity 1023
    else
        Set Motor fan clockwisely to intensity 0
```

Hardware

1. Connect the Temperature/Humidity sensor to Temperature Sensor port of Multiple Sensors
2. Download the code and transfer to micro:bit
3. Plug the micro:bit into Multiple Sensors
4. Connect Multiple Sensors and Multiple Actuators by connecting wire
5. Connect the Motor fan to Motor port of the Multiple Actuators

6. Conduct the test to see if the motor fan can be turn on > 26°C



* Can you think of other possible combination for turning on the motor fan? Try to do it.

1.3.13 Project 8: Fire Alarm

Level: ★ ★ ☆ ☆ ☆

Part List

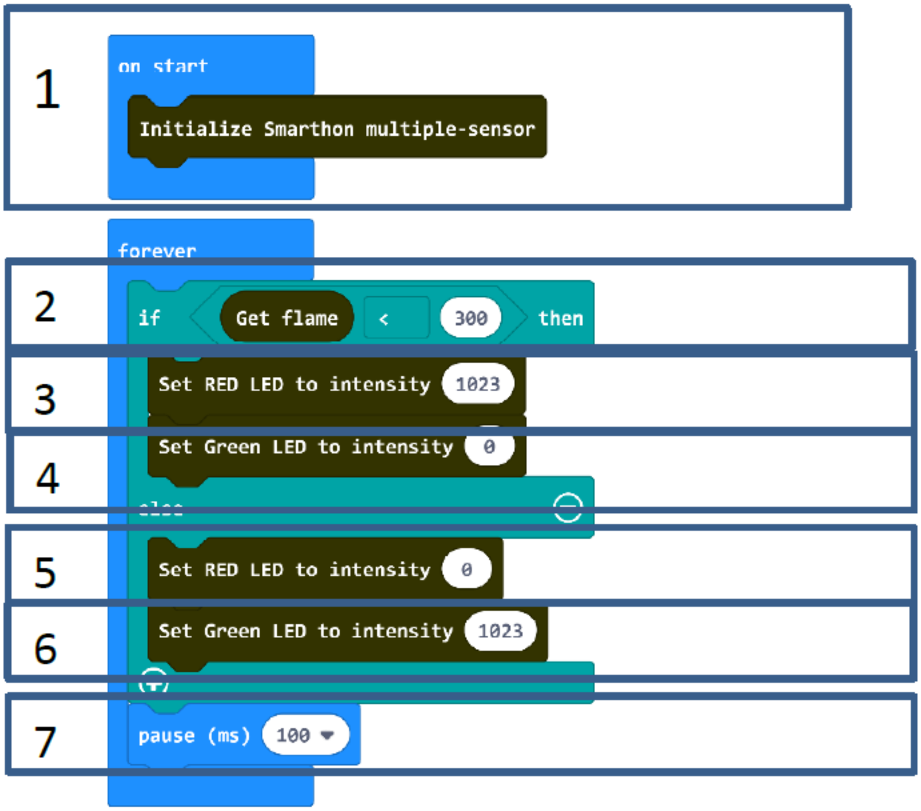
1. Flame sensor x 1
2. Red LED X1
3. Green LED X1
4. Multiple Sensors for smart house X1
5. Multiple Actuators for smart house X1
6. Connecting Wire X1
7. micro:bit X1
8. USB Cable X1

Description

In project 8, you can learn how to use micro:bit to make Alert LED (Red/Green). If the flame is detected, the Red LED will be turned on, the Green LED will be turned off. If there is no flame, Green LED is turned on and Red LED will be turned off.

Steps

1. Basic (Add block on start) > House (Add block Initialize Smarthon multiple-sensor)
2. Basic (Add block forever) > Logic (Add block if. . . then. . . else) > For if : Logic (Add block 0 = 0, change "0" to House "Get flame"; "=" to "<"; "0" to "300")
3. For then: House (Add block Set RED LED to intensity 0, change 0 to 1023)
4. For then: House (Add block Set Green LED to intensity 0)
5. For else: House (Add block Set RED LED to intensity 0)
6. For else: House (Add block Set GREEN LED to intensity 0, change 0 to 1023)

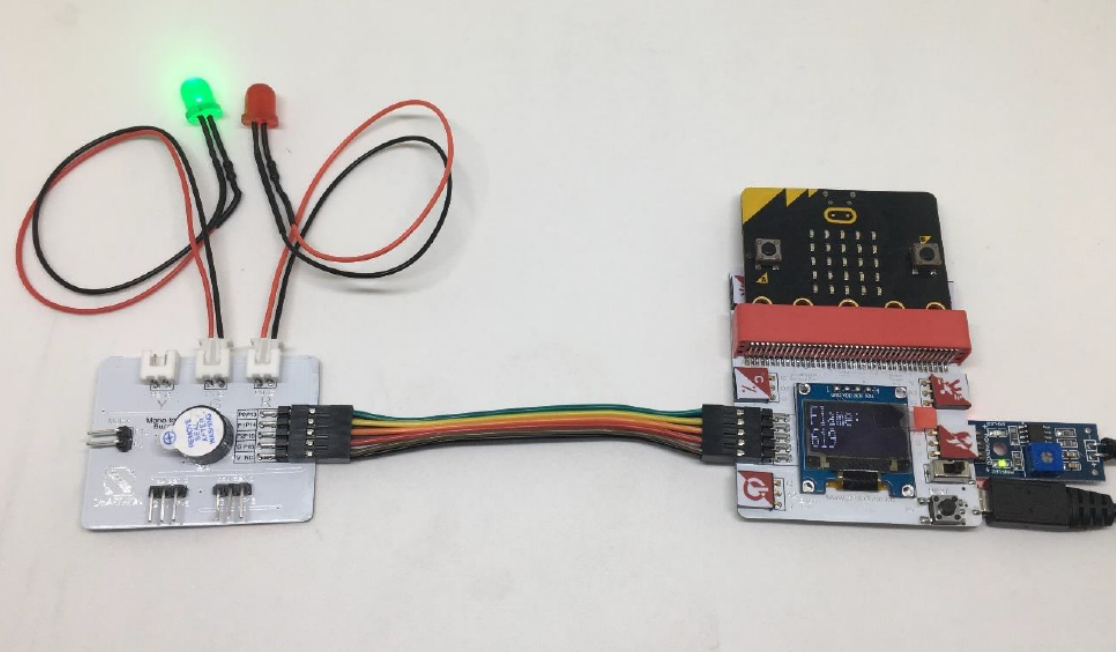


```
on start
    Initialize Smarthon multiple-sensor

forever
    if Get flame < 300 then
        Set RED LED to intensity 1023
        Set Green LED to intensity 0
    else
        Set RED LED to intensity 0
        Set Green LED to intensity 1023
    pause (ms) 100
```

7. Basic (Add block pause (ms) 100)

Hardware

1. **Download** the code and transfer to micro:bit
2. Plug the micro:bit into Multiple Sensors
3. Connect Multiple Sensors and Multiple Actuators by connecting wire
4. Connect the RED LED and GREEN LED module to the corresponding port of the Multiple Actuators



5. You can test it by lighter
   * Try to make some blinking of the RED LED and buzzer turned on if there is a fire.

1.3.14 Project 9: Kitchen Safety

Level: ★ ★ ★ ☆ ☆

Part List

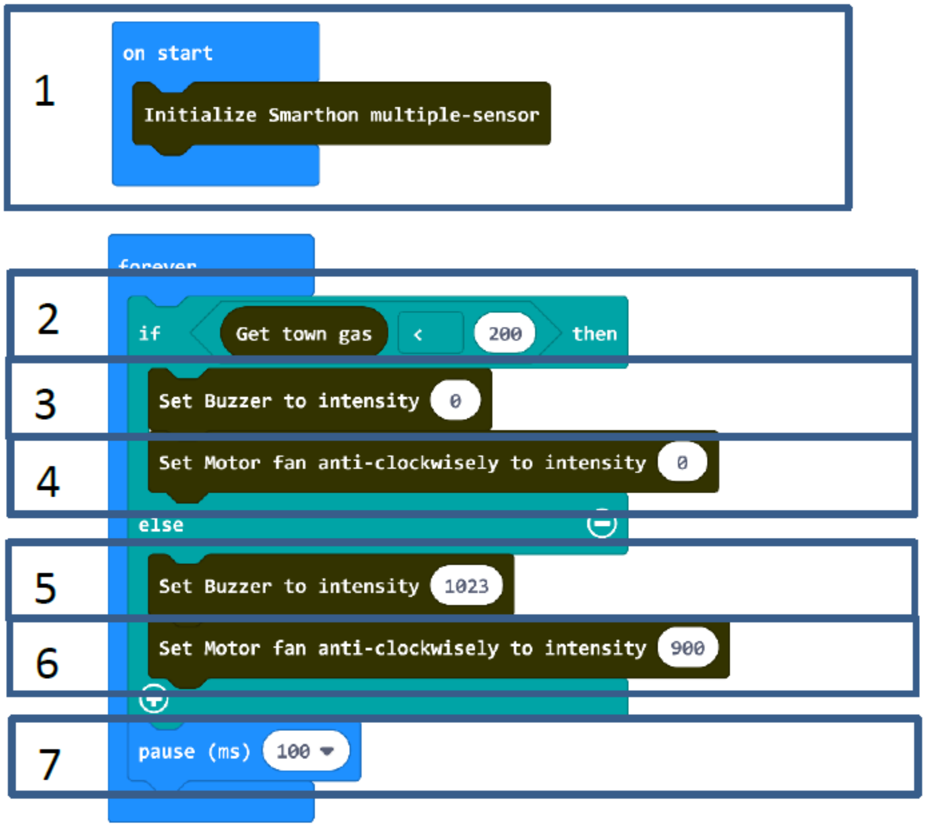
1. Town gas sensor x 1
2. Mono-tone buzzer X1 (placed on Multiple Actuators for smart house)
3. Motor fan x 1
4. Multiple Sensors for smart house X1
5. Multiple Actuators for smart house X1
6. Connecting Wire X1
7. micro:bit X1
8. USB Cable X1

Description

In project 9, you can learn how to use micro:bit to make Kitchen safety system. If the town gas leakage happened, the buzzer and motor fan will be turned on.

Steps

1. Basic (Add block on start) > House (Add block Initialize Smarthon multiple-sensor)
2. Basic (Add block forever) > Logic (Add block if…then…else) > For if : Logic (Add block 0 = 0, change "0" to House "Get town gas"; "=" to "<"; "0" to "200")
3. For then: House (Add block Set Buzzer to intensity 0)
4. For then: House (Add block Set Motor fan anti-clockwisely to intensity 0)
5. For else: House (Add block Set Buzzer to intensity 0, change 0 to 1023)
6. For else: House (Add block Set Motor fan anti-clockwisely to intensity 0, change 0 to 900)

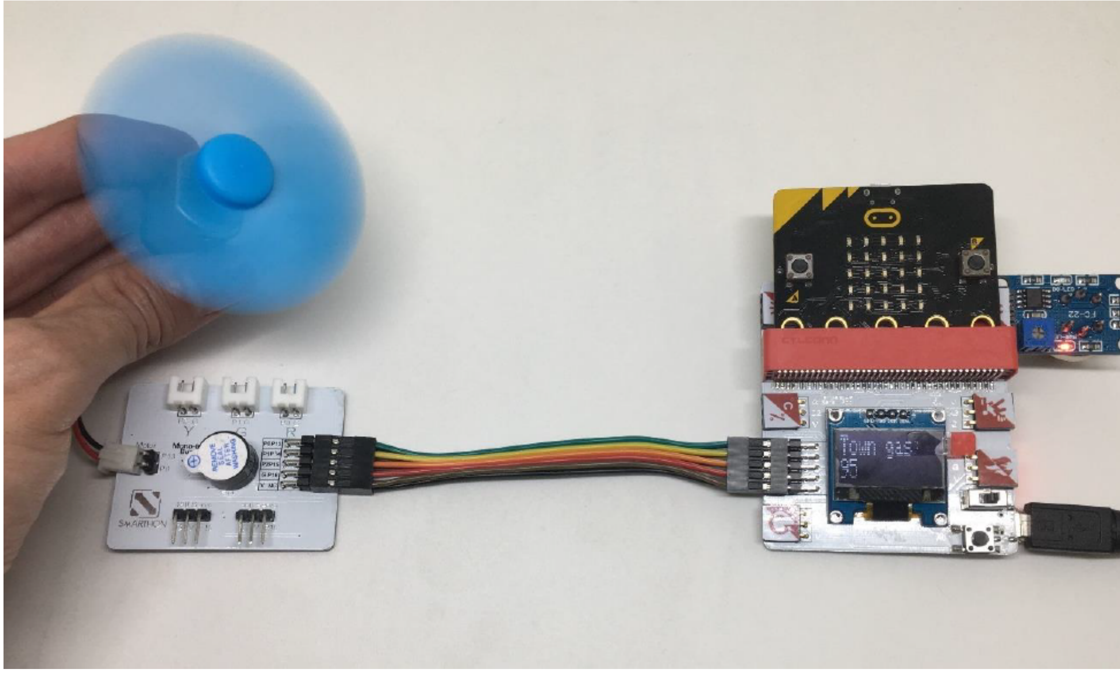


```
on start
    Initialize Smarthon multiple-sensor

forever
    if Get town gas < 200 then
        Set Buzzer to intensity 0
        Set Motor fan anti-clockwisely to intensity 0
    else
        Set Buzzer to intensity 1023
        Set Motor fan anti-clockwisely to intensity 900
    pause (ms) 100
```

7. Basic (Add block pause (ms) 100)

Hardware

1. **Download** the code and transfer to micro:bit
2. Plug the micro:bit into Multiple Sensors
3. Connect Multiple Sensors and Multiple Actuators by connecting wire
4. Connect the Motor Fan module to motor port of the Multiple Actuators and install it in the kitchen



5. You can test it by lighter
   * Try to adjust different speed of the motor fan.

1.3.15 Project 10: Remote control on the curtain

Level: ★ ★ ★ ★ ★

Part List

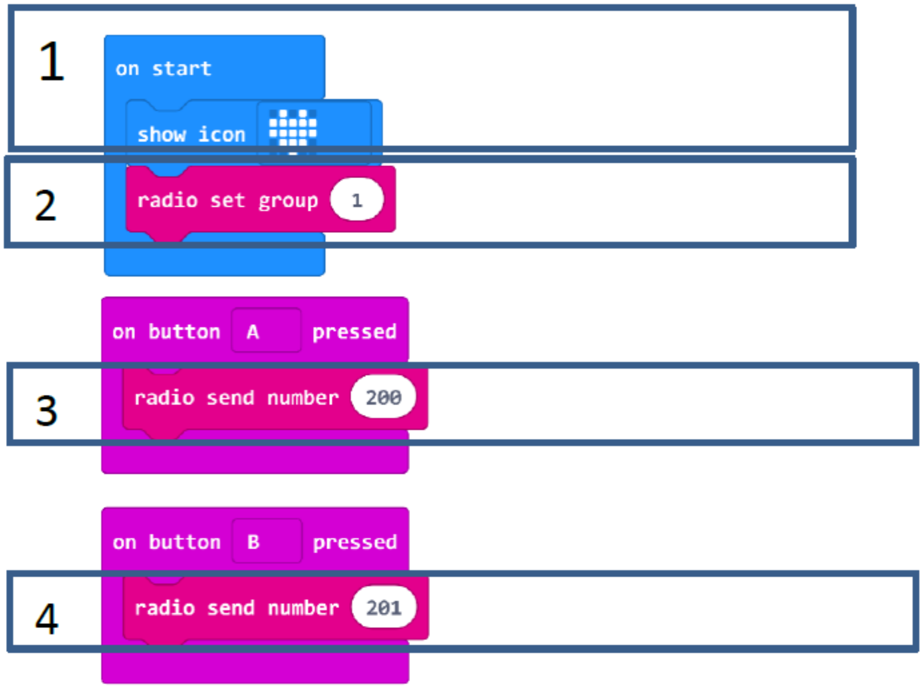
1. 360° servo X1
2. Multiple Sensors for smart house X1
3. Multiple Actuators for smart house X1
4. Connecting Wire X1
5. micro:bit X2
6. USB Cable X2

Description

In project 10, you will need 2 microbit. One is sender (controller). Another is the receiver (curtain machine). You can learn how to use micro:bit to remote control SMARTHON main board with curtain (360 servo) up and down.

Sender Steps

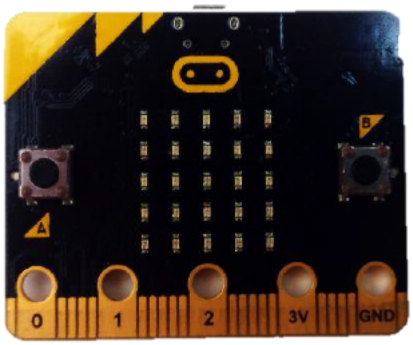
1. Basic (Add block on start) > Basic (Add block show icon - heart)
2. Basic (Add block on start) > (Add block radio set group 0, change 0 to 1)
3. Input (Add block on button A is pressed) > Radio (Add block radio send number 0, change 0 to 200)
4. Input (Add block on button A is pressed > change "A" into "B") > Radio (Add block radio send number 0, change 0 to 201)



```
on start
  show icon
  radio set group 1

on button A pressed
  radio send number 200

on button B pressed
  radio send number 201
```

Hardware

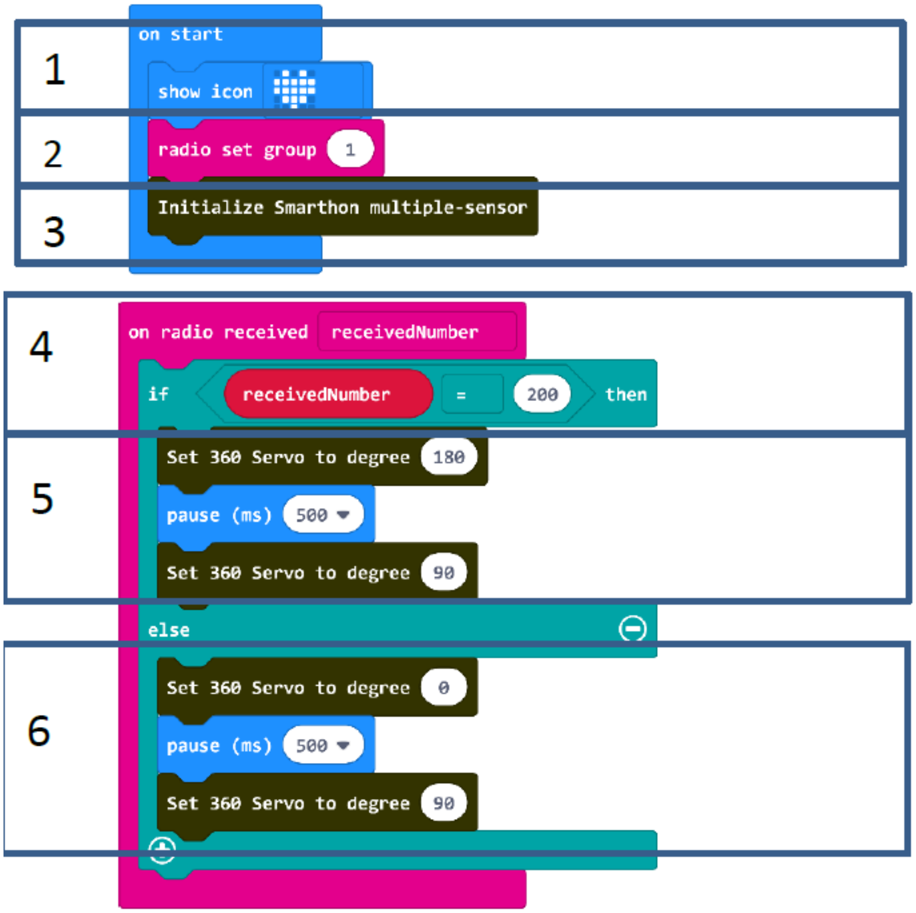
1. **Download** the code and transfer to micro:bit
2. Connect USB with micro:bit



3. Send the signal with button A/B pressed

Receiver Steps

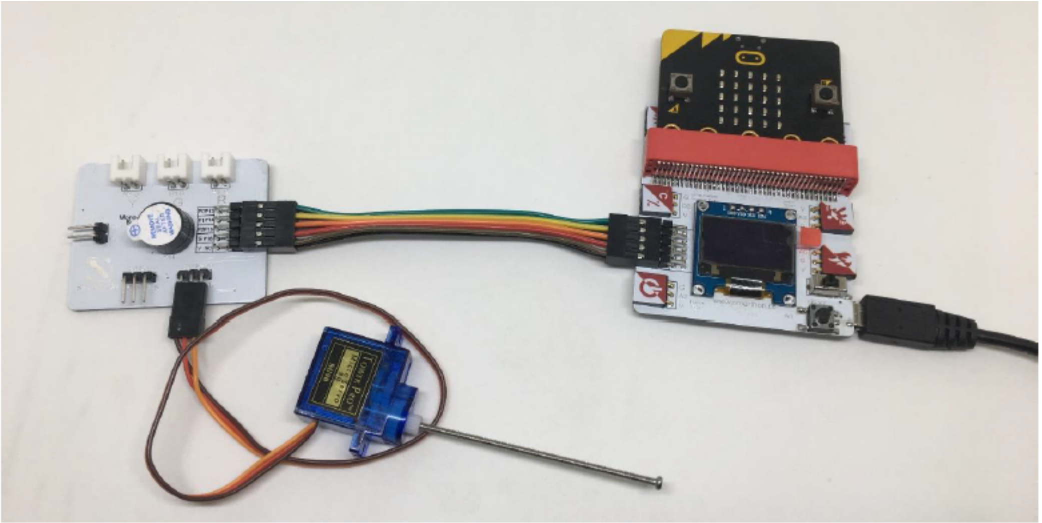
1. Basic (Add block on start) > Basic (Add block show icon - heart)
2. Basic (Add block on start) > (Add block radio set group 0, change 0 to 1)
3. Basic (Add block on start) > House (Add block Initialize Smarthon multiple-sensor)
4. Radio (Add block on radio received receivedNumber) > Logic (Add block if…then…else) > For if : Logic (Add block 0 = 0, change "0" to Variables "receivedNumber"; "0" to "200")
5. For then: House (Add block Set 360 servo to degree 0, change 0 to 180); Basic (Add block pause (ms) 100, change "100" to "500"); House (Add block Set 360 servo to degree 0, change 0 to 90)
6. For else: House (Add block Set 360 servo to degree 0); Basic (Add block pause (ms) 100, change "100" to "500"); House (Add block Set 360 servo to degree 0, change 0 to 90)Basic (Add block pause (ms) 100)

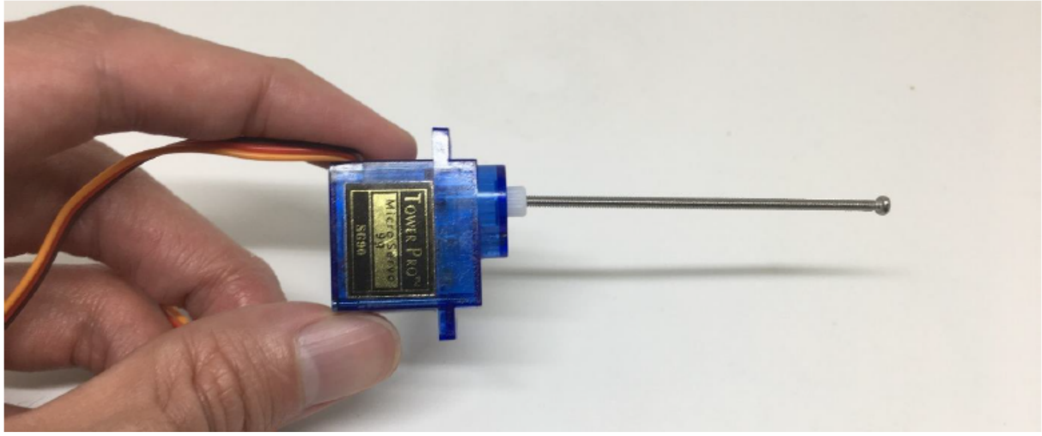


```
on start
  show icon
  radio set group 1
  Initialize Smarthon multiple-sensor

on radio received receivedNumber
  if receivedNumber = 200 then
    Set 360 Servo to degree 180
    pause (ms) 500
    Set 360 Servo to degree 90
  else
    Set 360 Servo to degree 0
    pause (ms) 500
    Set 360 Servo to degree 90
```

Hardware

1. **Download** the code and transfer to micro:bit
2. Plug the micro:bit into Multiple Sensors
3. Connect Multiple Sensors and Multiple Actuators by connecting wire
4. Connect 360° servo module to 360 servo port of the Multiple Actuators





* Let's put on the hardware into the house model.

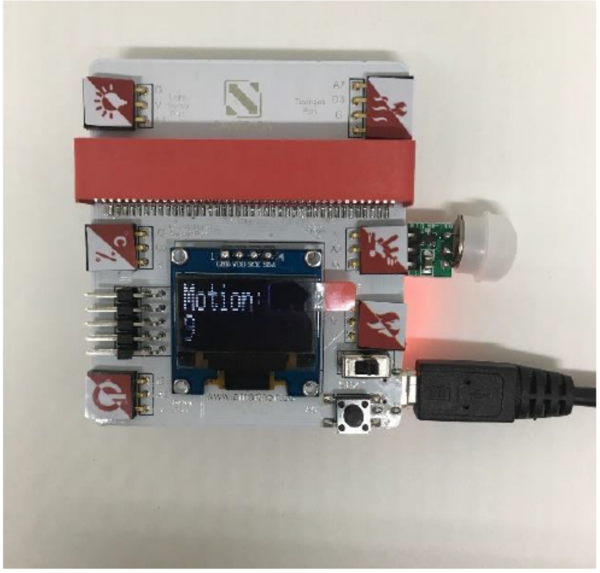
1.3.16 Appendix I

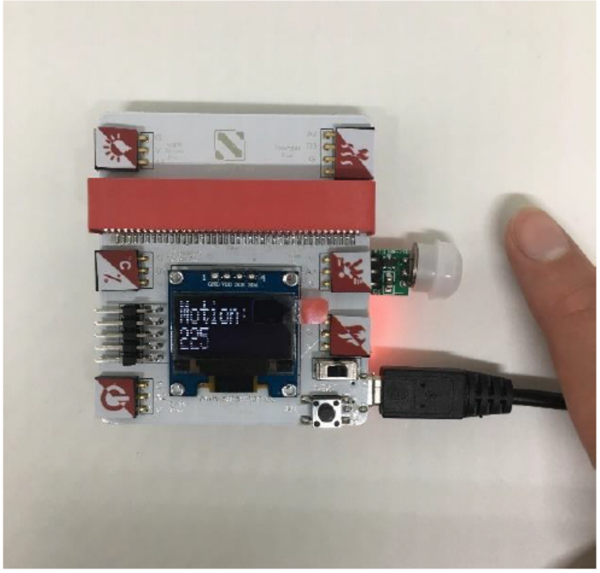
Reading Sensors Answer:

Temperature & Humidity:

Please feel free to write down the value of the current condition.

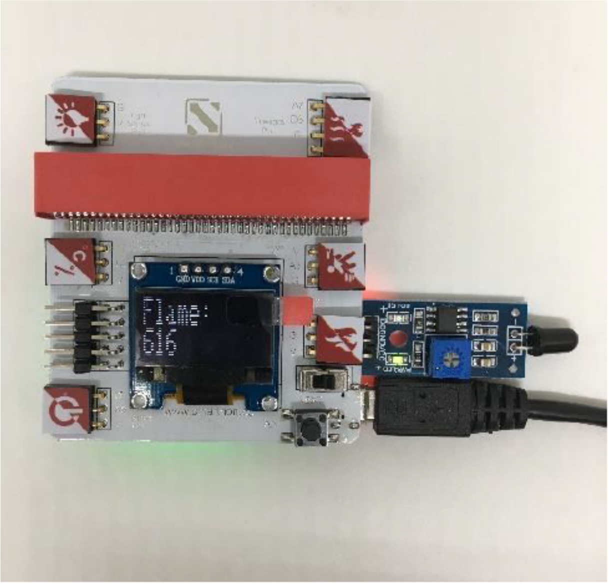
Motion Sensor:

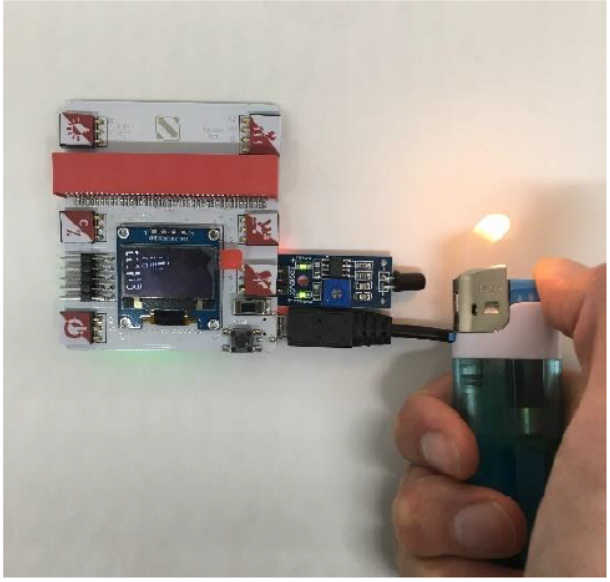




If this sensor detects any movements, the value of the motion sensor will increase from 0 to 200, as shown in the picture.

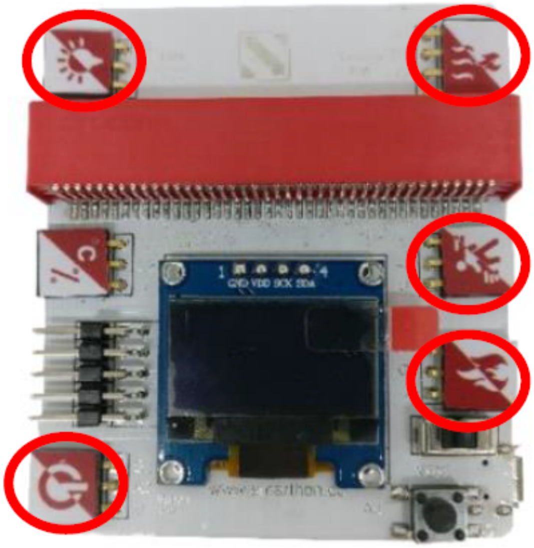
Flame Sensor:



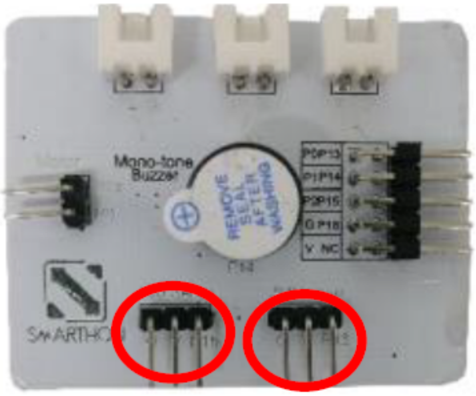


If there is a fire, the number of the flame sensor will decrease from 600 to 30.

1.3.17 Maker: Control other actuators and sensors



1. Connect to more other sensors (Input)!



2. Connect to more other actuators (Output)!

1.4 Smarthon Plant Kit for micro:bit

1.4.1 Product Introduction

About Smarthon Plant Kit for micro:bit

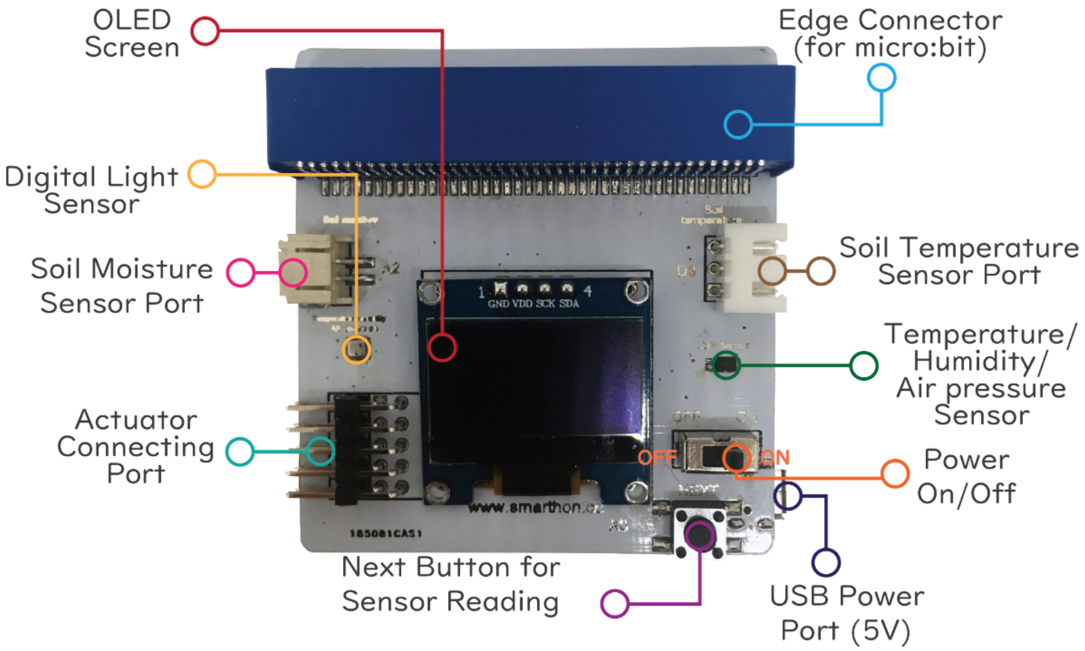
Plants are tremendously important to life on earth. They are the foundations of our ecosystem, producing life-giving oxygen, absorbing carbon dioxide and influencing climate as a carbon sink as well as providing food to us.

Smarthon Plant Kit is a scientific aid with Multiple Sensors and Actuators for students to learn and conduct experiment on plant studies. Simply using the graphical programming on BBC micro:bit makecode platform, it would be much easier for students to set up plant experiment. Not only can students learn about plant biology, they are also encouraged to ignite their creativity to design and study the particular projects on plant growth in deep.

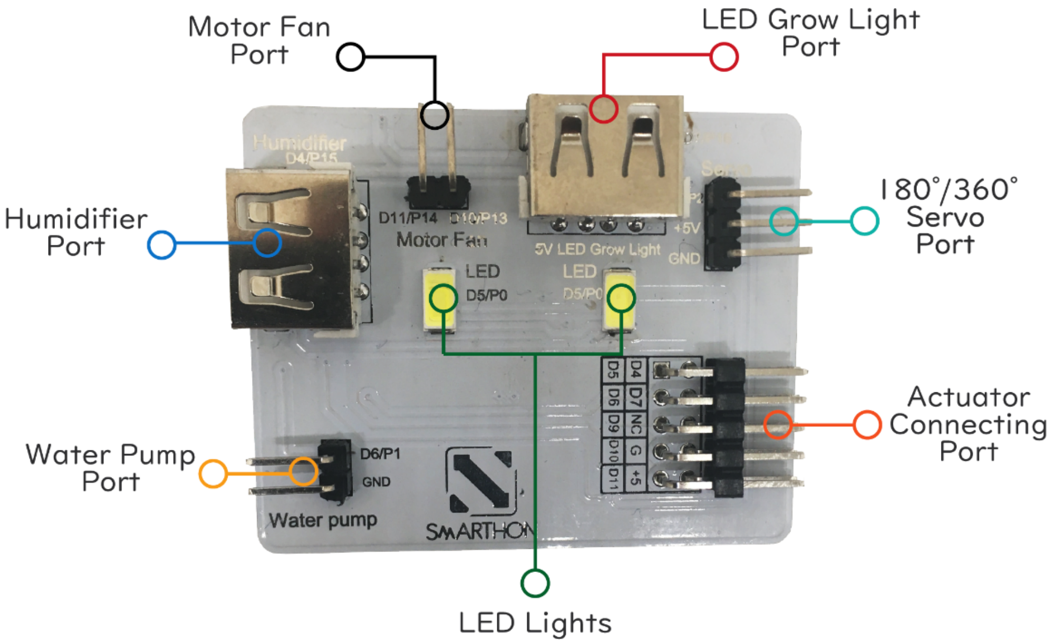
Kit components

1.4.2 Smarthon Plant Kit

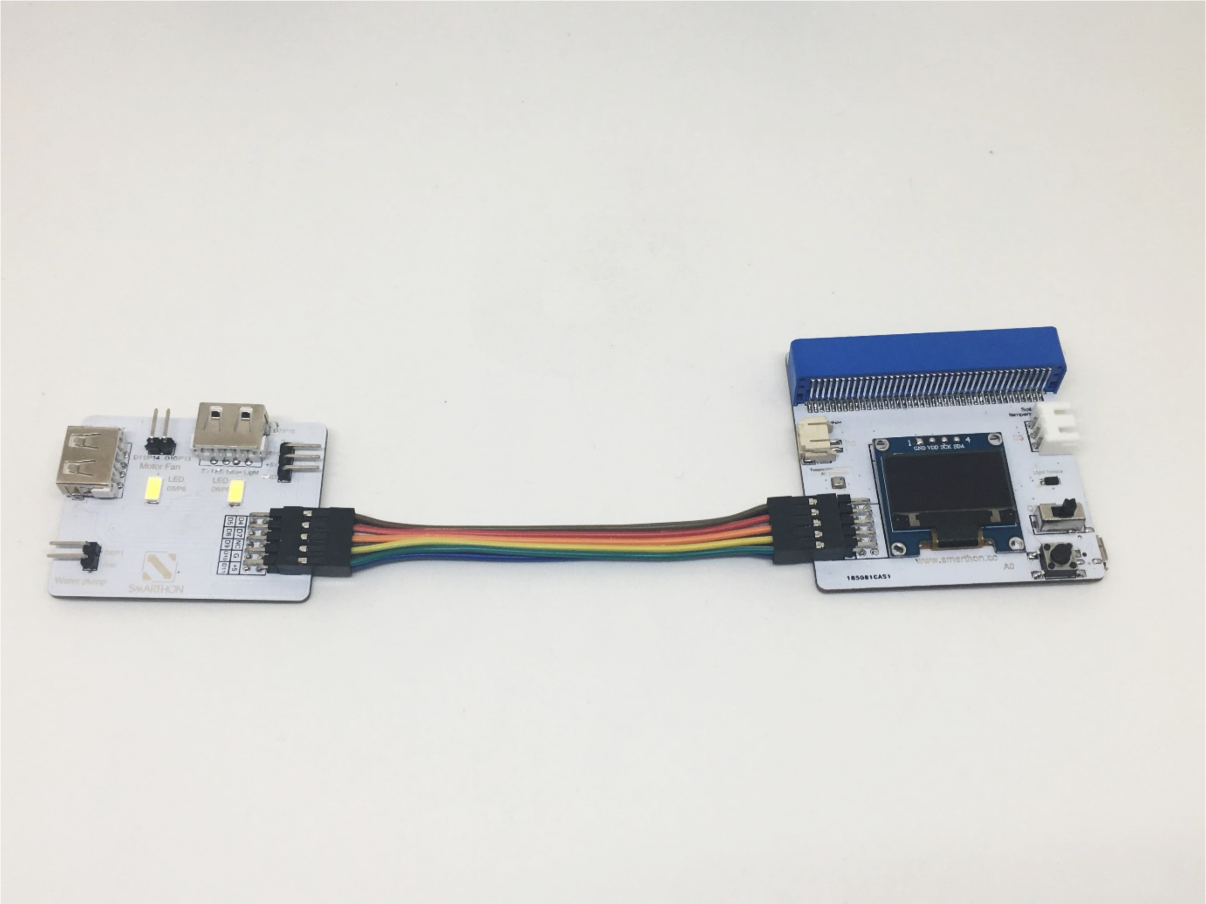
Introduction for Multiple Sensors for plant



Introduction for Multiple Actuators for plant



Basic Connection Between Multiple Sensors and Multiple Actuators

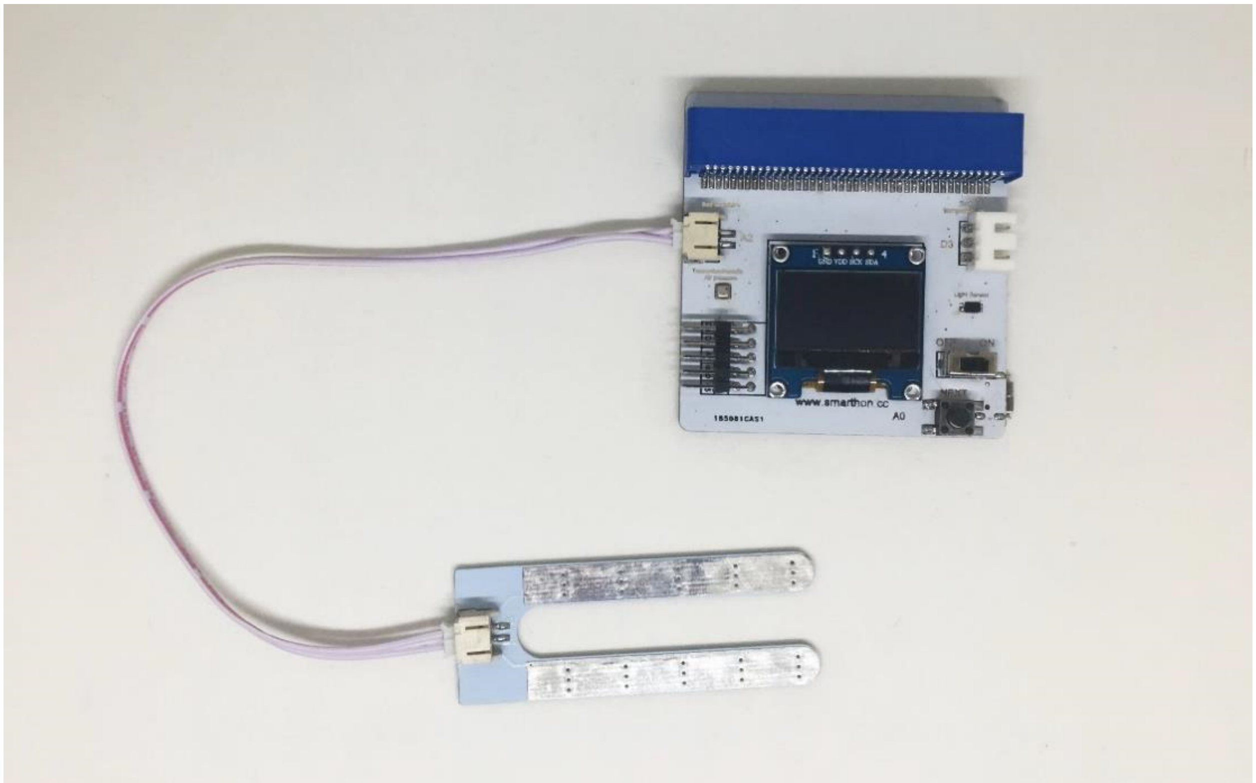


*To make sure the connecting wire is on the same direction.

1.4.3 Hardware User Guide

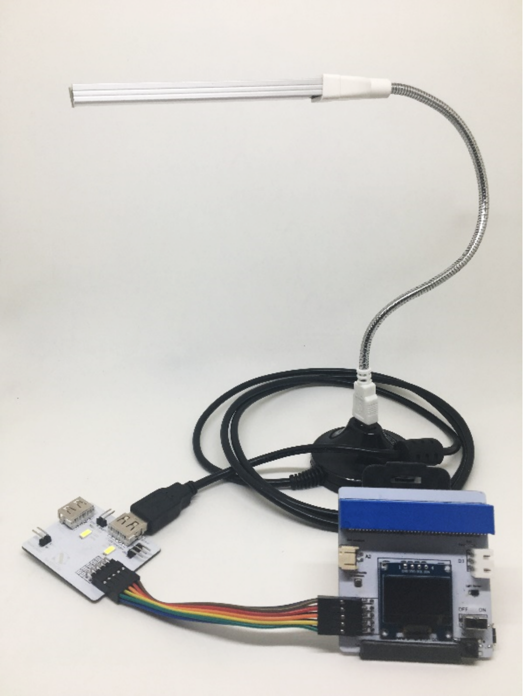
Sensors and Multiple Sensors and Plant

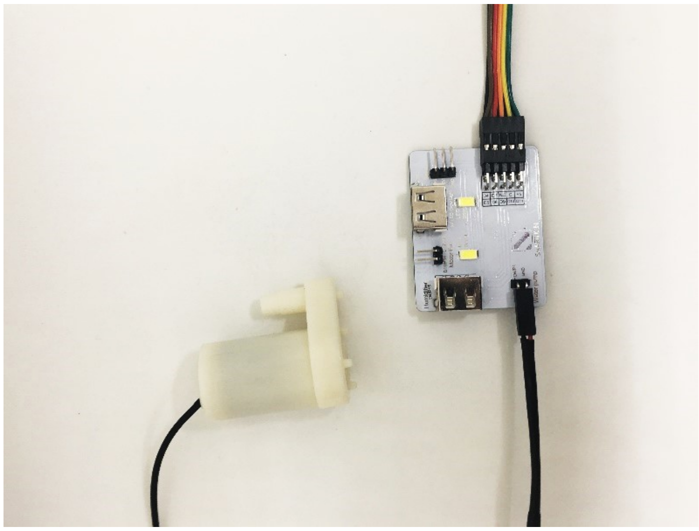
Most of the sensors are placed on the board of Multiple Sensors, namely temperature/humidity/air-pressure sensor and digital light sensor. Soil moisture sensor will be shown as a demo.

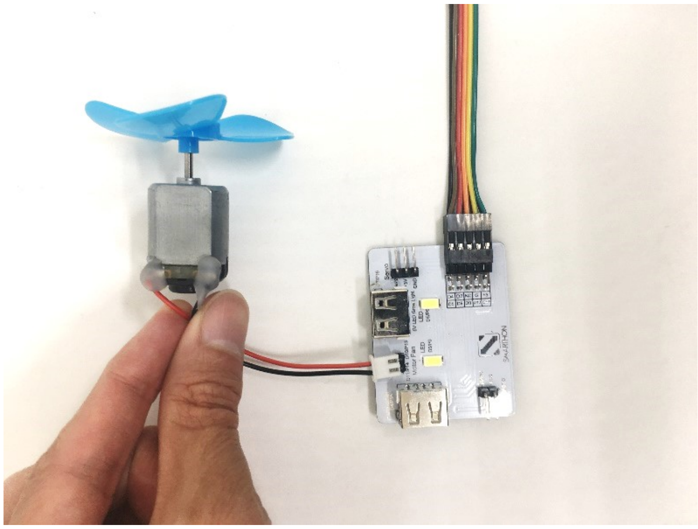


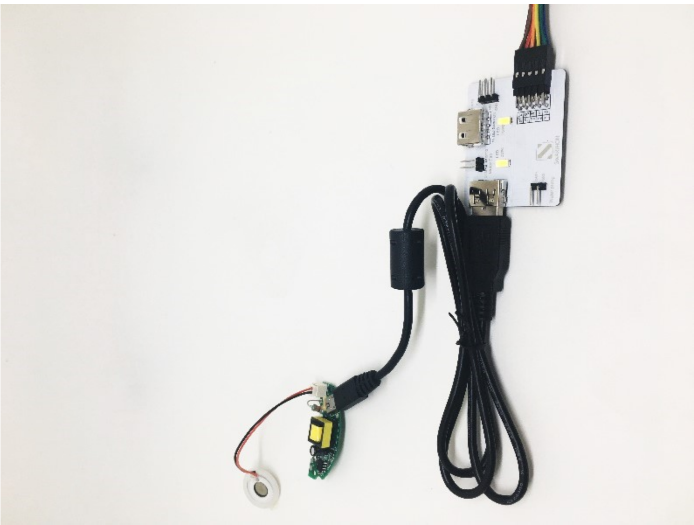
Actuators and Multiple Actuators for plant

How to connect the actuator modules and Multiple Actuators for plant? Please follow the instructions shown below to plug actuator modules into Multiple Actuators correctly to ensure all the modules work effectively.

1.LED grow light 

2.Water pump 

3.Motor fan 

4.Humidifier 

1.4.4 Reading all the sensors on screen

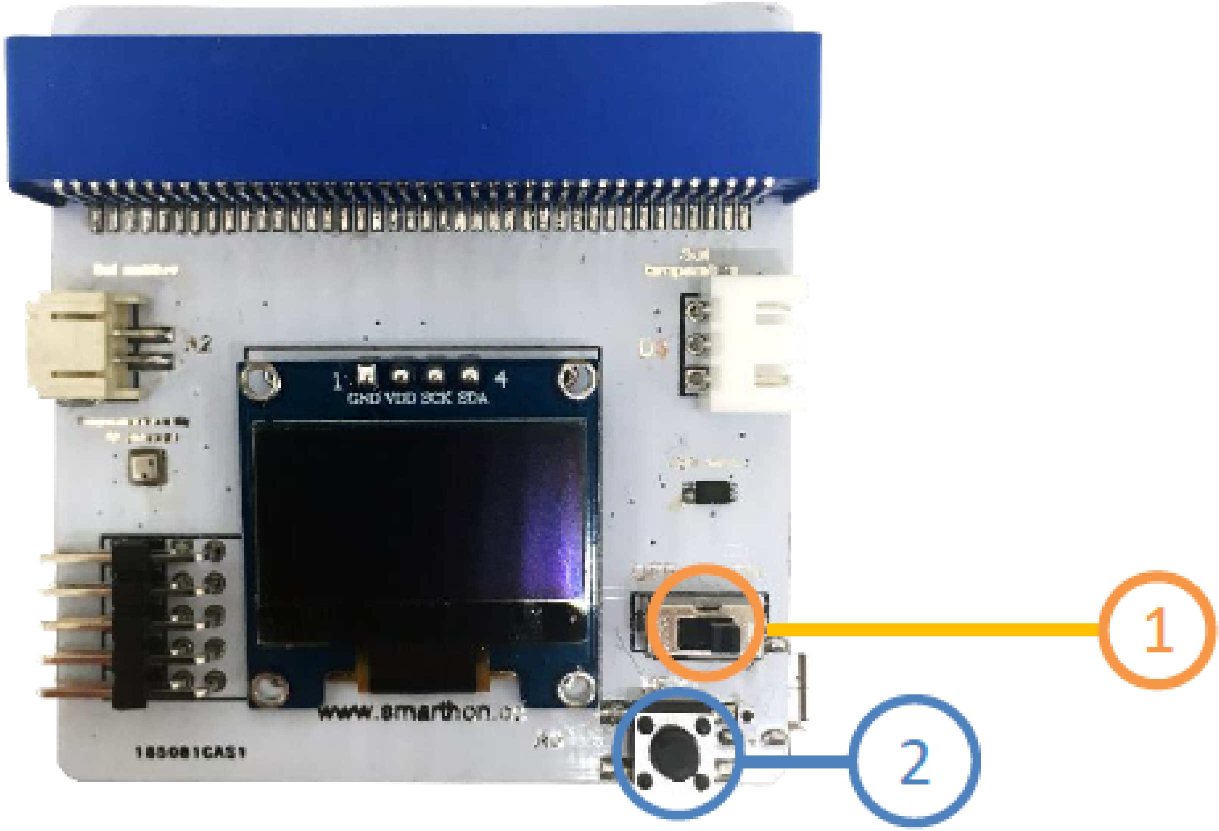
Part List:

1. Multiple Sensors for plant X1
2. Temperature, Humidity and Air pressure Sensor X1 (placed on the Multiple Sensors)
3. Digital Light Sensor X1 (placed on the Multiple Sensors)
4. Soil humidity sensor X1

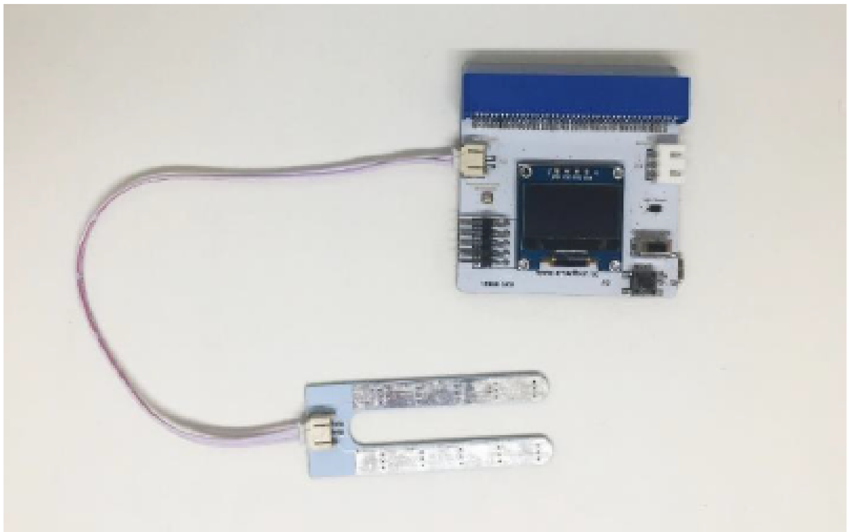
You can learn reading sensors and its' interruption which can help you to conduct other projects.

1. Switch on the power button
2. Pressed the button in the right bottom corner of the Multiple Sensors for plant

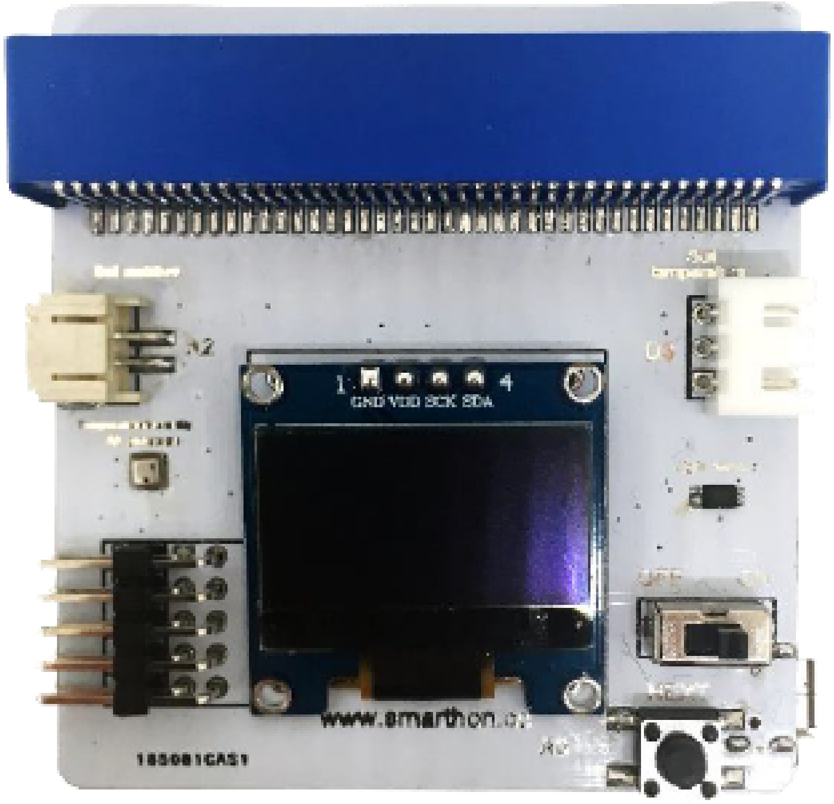
It shows the temperature, humidity, air Pressure, light as well as soil humidity.



Soil moisture sensor will be shown as an example: 1.Soil Moisture Sensor



When you plug the soil moisture sensor into the port on Multiple Sensors for micro:bit, the screen shows 0.10%; while when you put the soil moisture into the plant with watering, the screen shows around 94%. It indicates that the larger number of the screen shown, the soil moisture level is higher. Reading the other three sensors:



Temperature:___________________

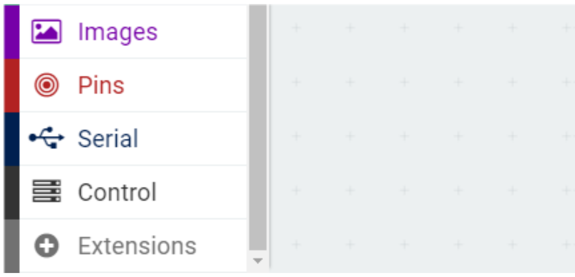
Humidity:______________________

Digital light sensor:___________________

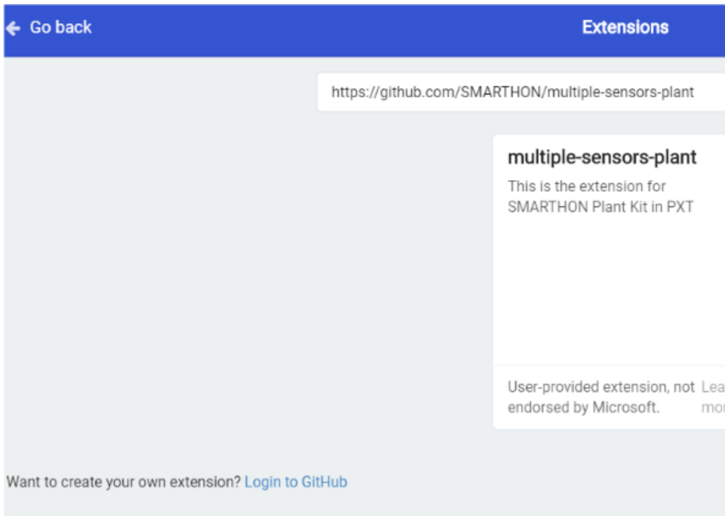
1.4.5 How to program Smarthon Plant Kit on micro:bit

To Program Smarthon modules with micro:bit, you need to add the Smarthon Plant Kit packages to the BBC micro:bit makecode platform and initialize it first.

Steps: 1.Find the “Extensions” block at the bottom



2.Search “https://github.com/SMARTHON/multiple-sensors-plant”



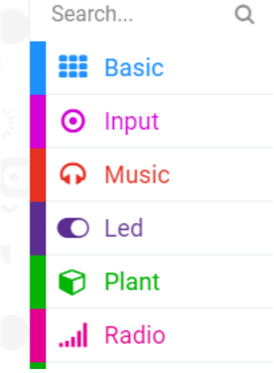
https://github.com/SMARTHON/multiple-sensors-plant

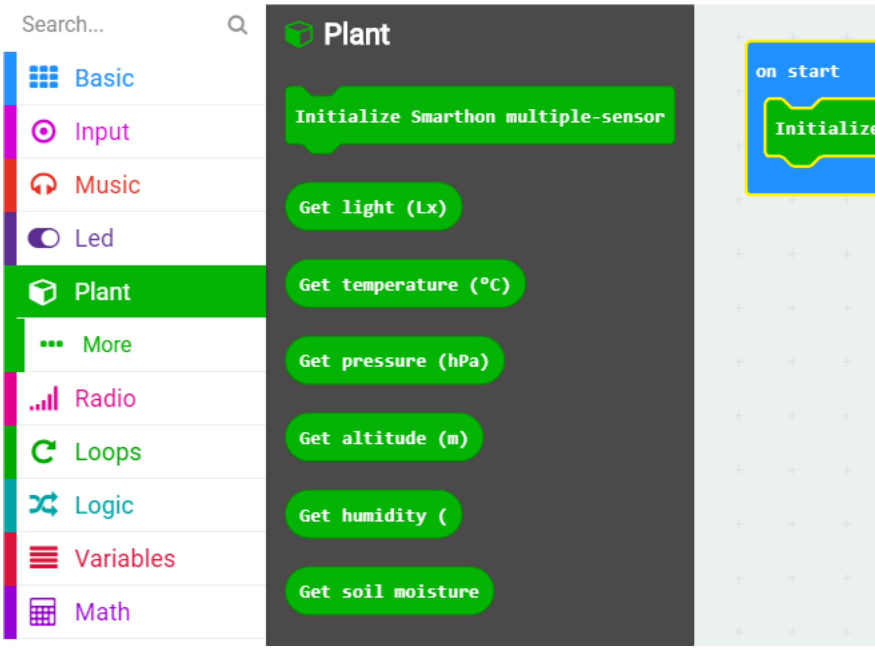
This is the extension for SMARTHON Plant Kit in PXT

User-provided extension, not endorsed by Microsoft.

Want to create your own extension? Login to GitHub

3.Clicked “multiple sensors” and you can see “Plant” package has been successfully added under “Led”.





4.Initialize Smarthon Plant Kit to add all the functions

1.4.6 Project 1: Read Sensors on micro:bit LED lights

Level: ★ ☆ ☆ ☆ ☆

Part List:

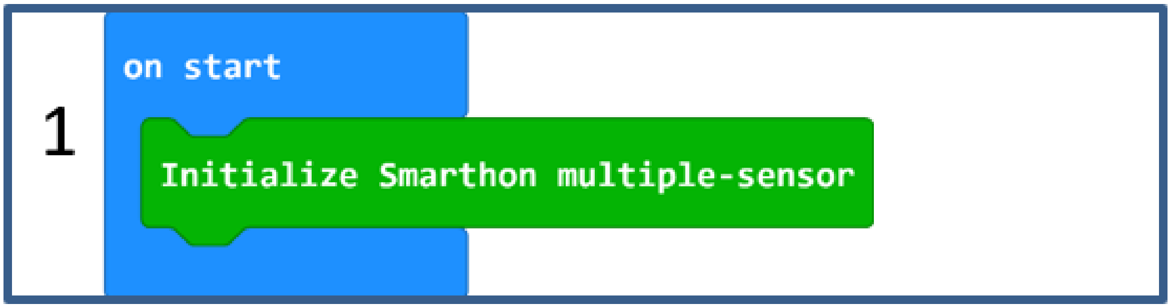
1. Temperature/Humidity Sensor X1
2. Multiple Sensors for plant X1
3. Multiple Actuators for plant X1
4. Connecting Wire X1
5. micro:bit X1
6. USB Cable X1

Description:

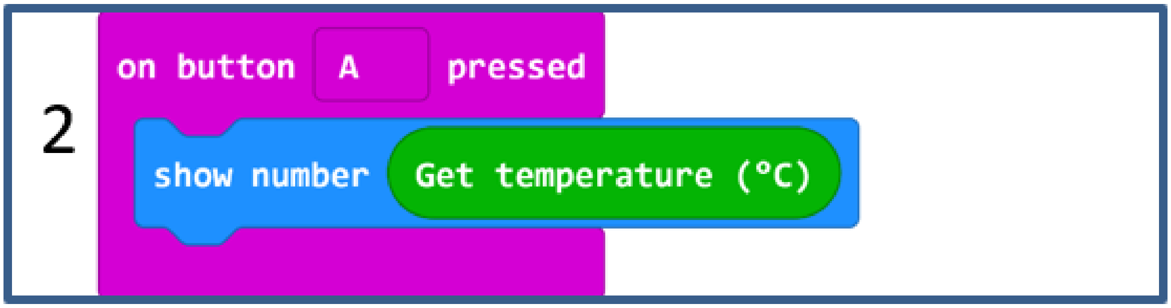
You can learn how to get the value of the sensor and show on micro:bit LED. Temperature/Humidity sensor will be illustrated in this demo.

Steps:

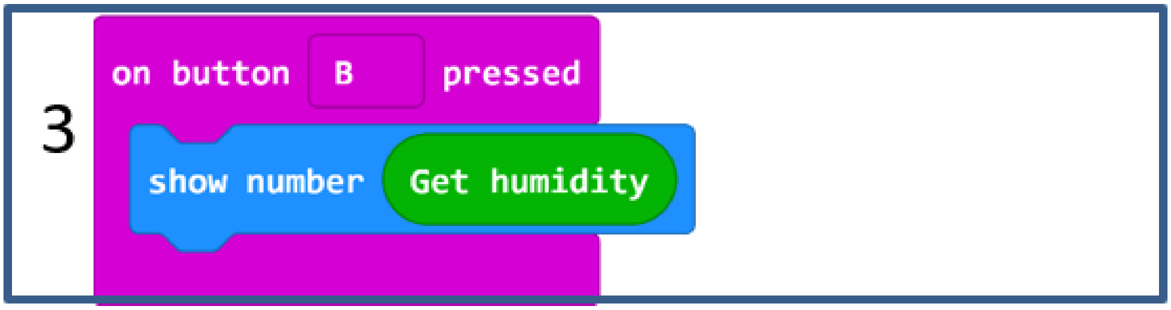
1. Basic (Add block on start) > Plant (Add block Initialize Smarthon multiple-sensor)
2. Input (Add block on button A is pressed) > Basic (Add block show number) > Plant (Add block Get temperature (°C))
3. Input (Add block on button A is pressed, change “A” into “B”) > Basic (Add block show number) > House (Add block Get humidity)



```
on start
    Initialize Smarthon multiple-sensor
```



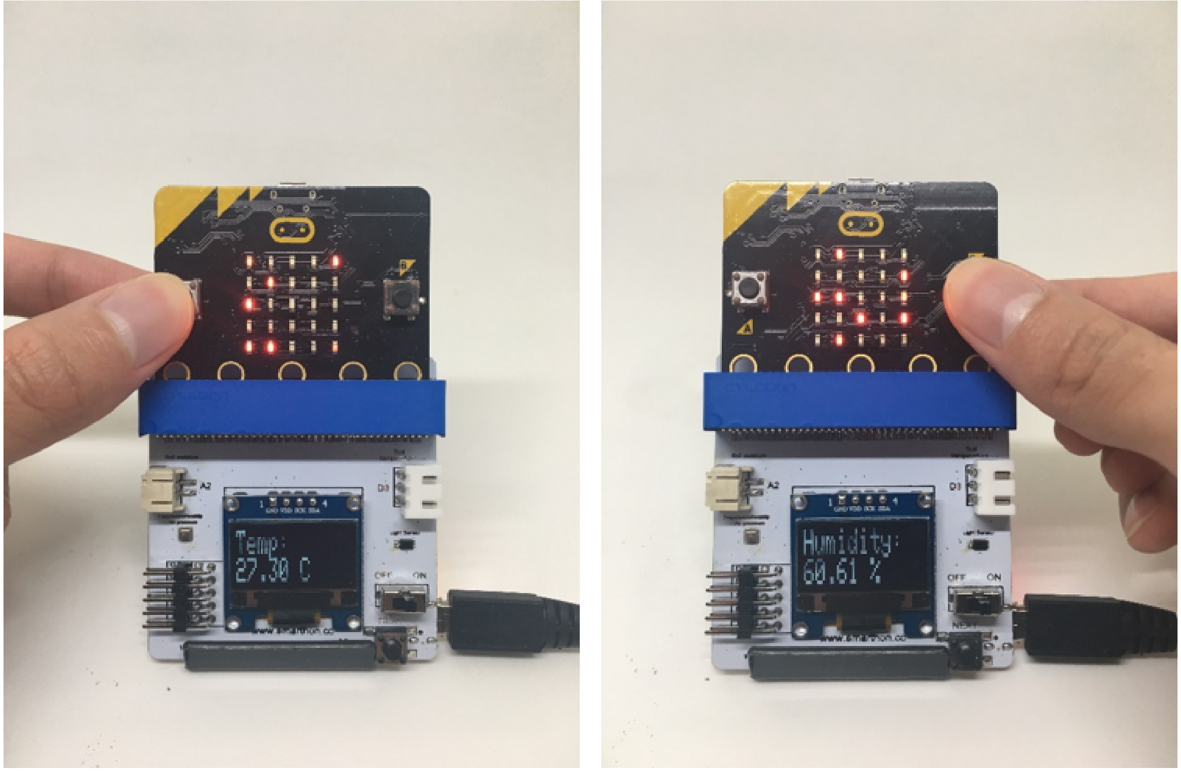
```
on button A pressed
    show number Get temperature (°C)
```



```
on button B pressed
    show number Get humidity
```

Hardware:

1. Download the code and transfer to micro:bit
2. Plug the micro:bit into Multiple Sensors
3. Pressed micro:bit button A to see the temperature and button B to see the humidity



* Try to show the other sensors value on micro:bit LED.

1.4.7 Project 2: Control Actuators - button A/B

Level: ★ ☆ ☆ ☆ ☆

Part List:

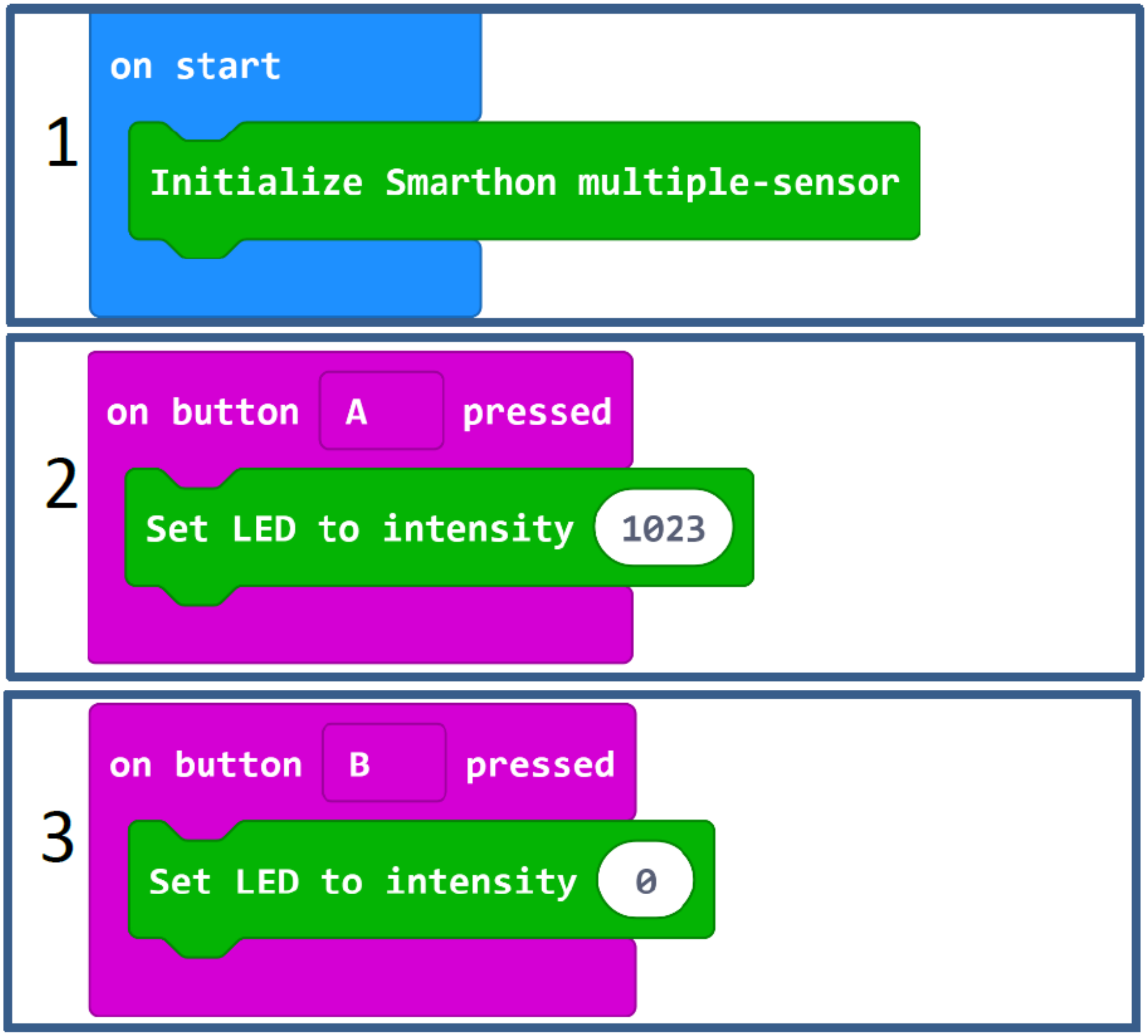
1. LED lights X2 (placed on the Multiple Actuators for plant)
2. Multiple Sensors for plant X1
3. Multiple Actuators for plant X1
4. Connecting wire X1
5. micro:bit X1
6. USB Cable X1

Description:

In project 2, you can learn how to use micro:bit Button A/B to control LED lights on Multiple Actuators.

Steps:

1. Basic (Add block on start) > Plant (Add block Initialize Smarthon multiple-sensor)
2. Input (Add block on button A is pressed) > Plant-More (Add block Set LED to intensity 0, change “0” to “1023”)
3. Input (Add block on button A is pressed, change “A” into “B”) > Plant-More (Add block Set LED to intensity 0)



1

```
on start
    Initialize Smarthon multiple-sensor
```

2

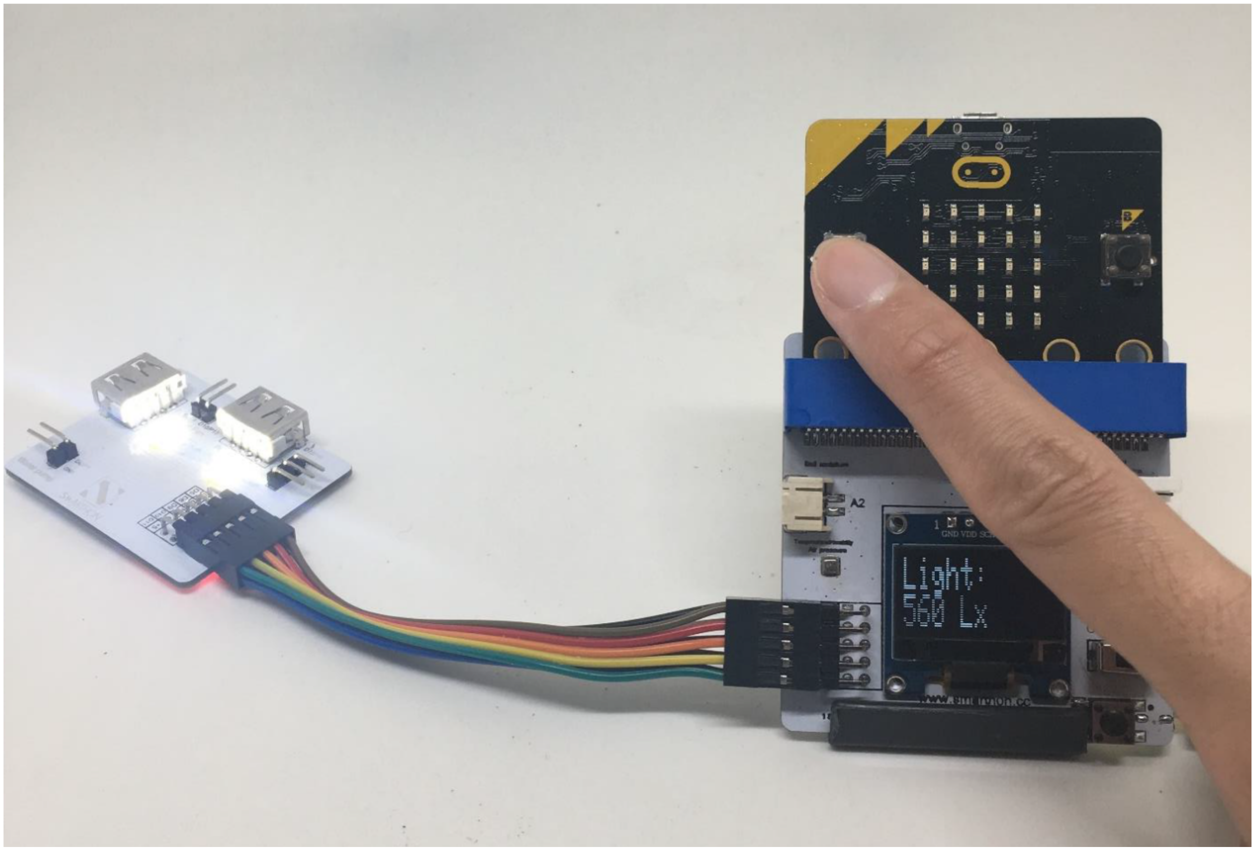
```
on button A pressed
    Set LED to intensity 1023
```

3

```
on button B pressed
    Set LED to intensity 0
```

Hardware:

1. Connect Multiple Sensors and Multiple Actuators by connecting wire
2. **Download** the code and transfer to micro:bit
3. Plug the micro:bit into Multiple Sensors
4. Pressed micro:bit button A to turn on the LED light; button B to turn off the LED light



*Try to use micro:bit Button A/B to control other outputs – motor fan, humidifier, water pump and LED grow lights

1.4.8 Project 3: Smart LED Grow Light

Level: ★ ★ ☆ ☆ ☆

Part List:

1. Digital Light Sensor X1 (placed on Multiple Sensors for plant)
2. LED Grow Light X1
3. USB Base (for LED Grow Light) X1
4. USB Stand (for LED Grow Light) X1
5. Multiple Sensors for plant X1
6. Multiple Actuators for plant X1

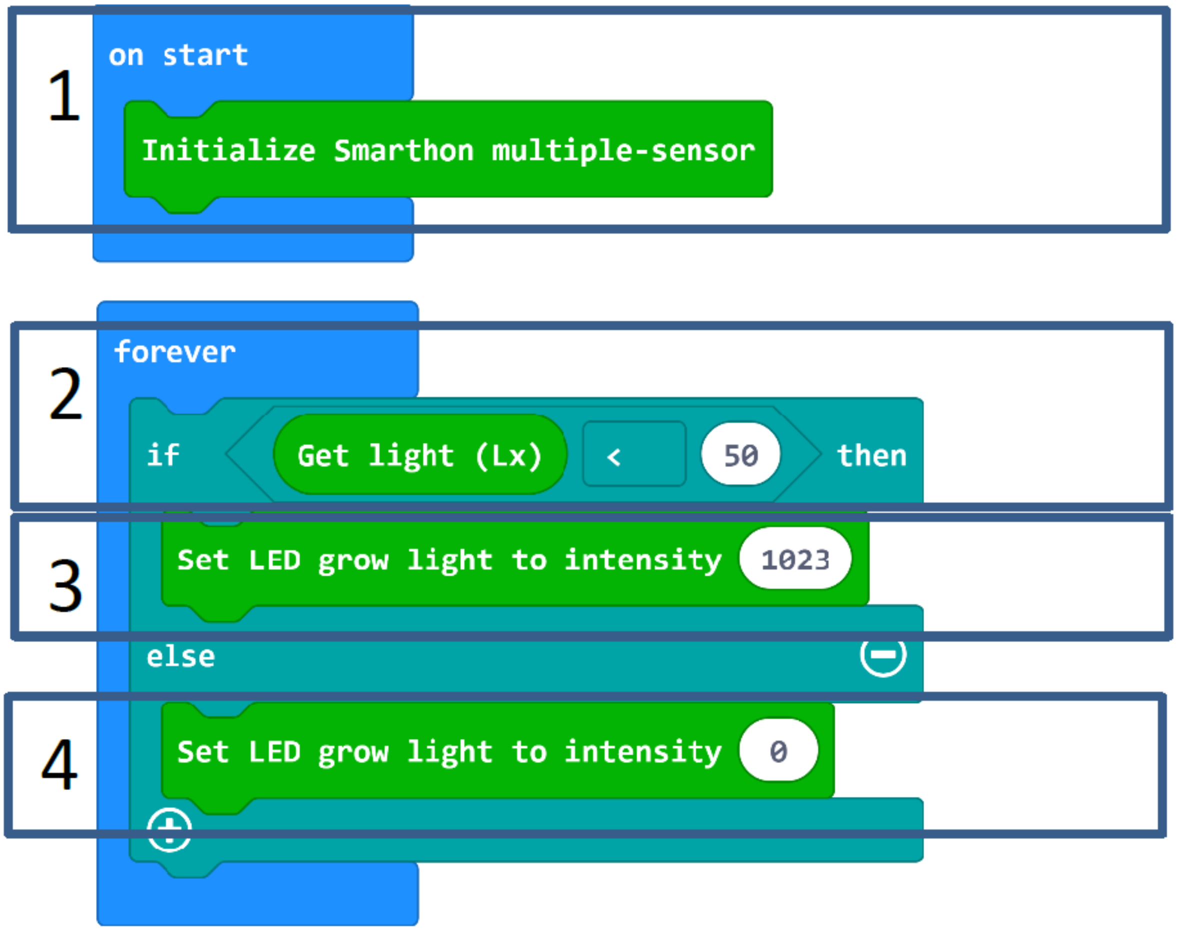
7. Connecting Wire X1
8. micro:bit X1
9. USB Cable X1

Description:

In project 3, you can learn how to control LED grow light by digital light sensor.

Steps:

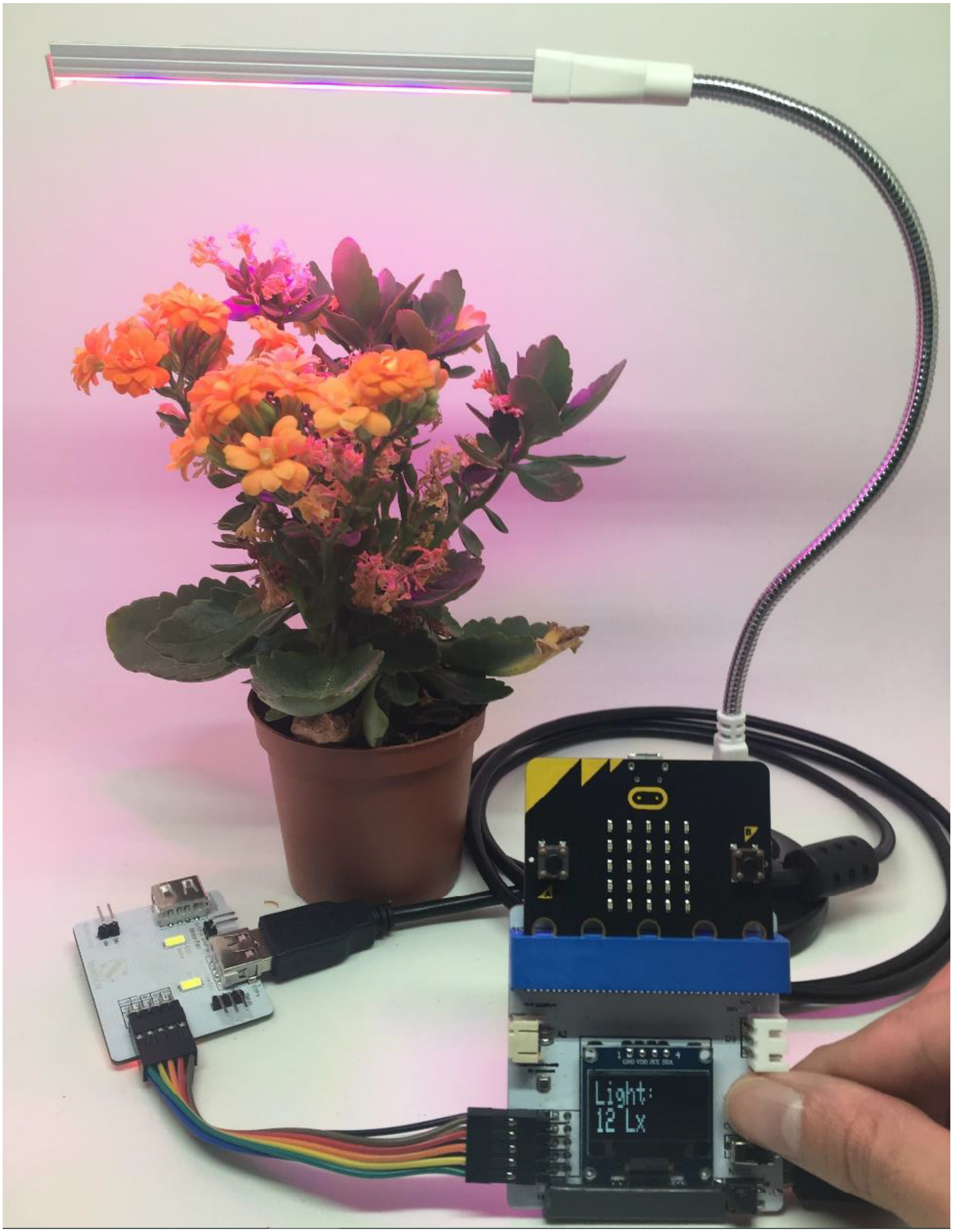
1. Basic (Add block on start) > Plant (Add block Initialize Smarthon multiple-sensor)
2. Basic (Add block forever) > Add Logic (Add block if…then…else) > For if : Logic (Add block 0 = 0, change "0" to Plant "Get light (Lx)"; "=" to ">"; "0" to "50")
3. For then: Plant-More (Add block Set LED grow light to intensity 0, change "0" to "1023")
4. For else: Plant -More (Add block Set LED grow light to intensity 0)



```
on start
    Initialize Smarthon multiple-sensor

forever
    if  Get light (Lx)  <  50  then
        Set LED grow light to intensity 1023
    else
        Set LED grow light to intensity 0
```

Hardware:

1. Connect Multiple Sensors and Multiple Actuators by connecting wire
2. Connect USB Base (for LED Grow Light) to 5V LED Grow Light (D7/P16) port of Multiple Actuators
3. Connect **LED Grow Light** and USB Base by **USB Stand** (for LED Grow Light)
4. Adjust the USB Stand to fit the plant
5. **Download** the code and transfer to micro:bit
6. Plug the micro:bit into Multiple Sensors
7. Try to cover the light sensor by hand to test if the LED Grow Light works properly


Light:
12 Lx

Experiment:

- Do different colours of LED grow lights affect the direction of plant growth?
- Do different colours of LED grow lights affect the rate of photosynthesis?
- Do different types of LED grow lights affect plant growth?
- What is the difference between sunlight and LED grow light among the plant growth?

1.4.9 Project 4: Automatic Irrigation (watering)

Level: ★ ★ ☆ ☆ ☆

Part List:

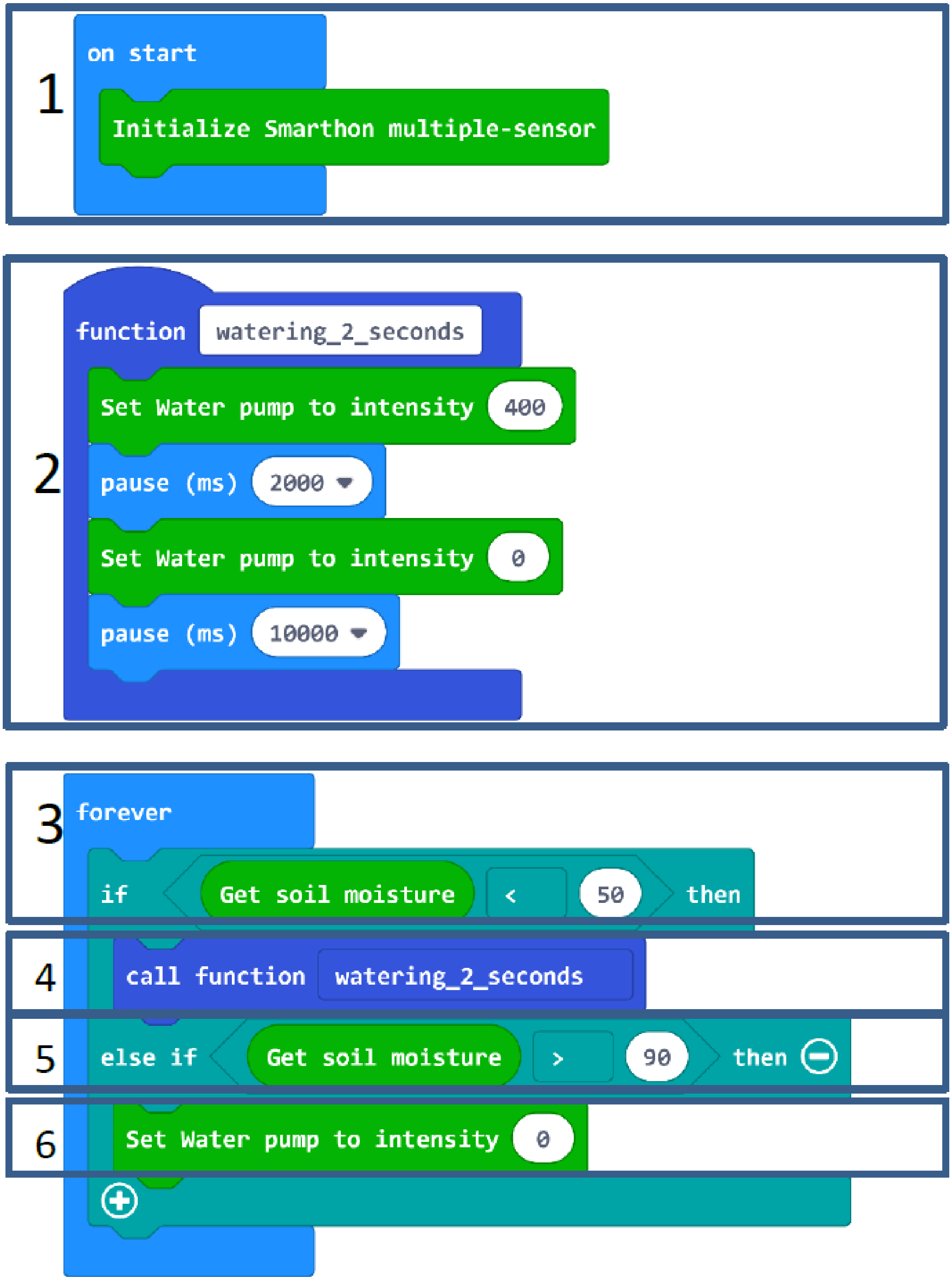
1. Soil Moisture Sensor
2. Water Pump & Pipe X1
3. Multiple Sensors for plant X1
4. Multiple Actuators for plant X1
5. Connecting Wire X1
6. micro:bit X1
7. USB Cable X1
8. A glass of water X1

Description:

In project 4, you can learn how to watering the plant automatically by soil moisture detector.

Steps:

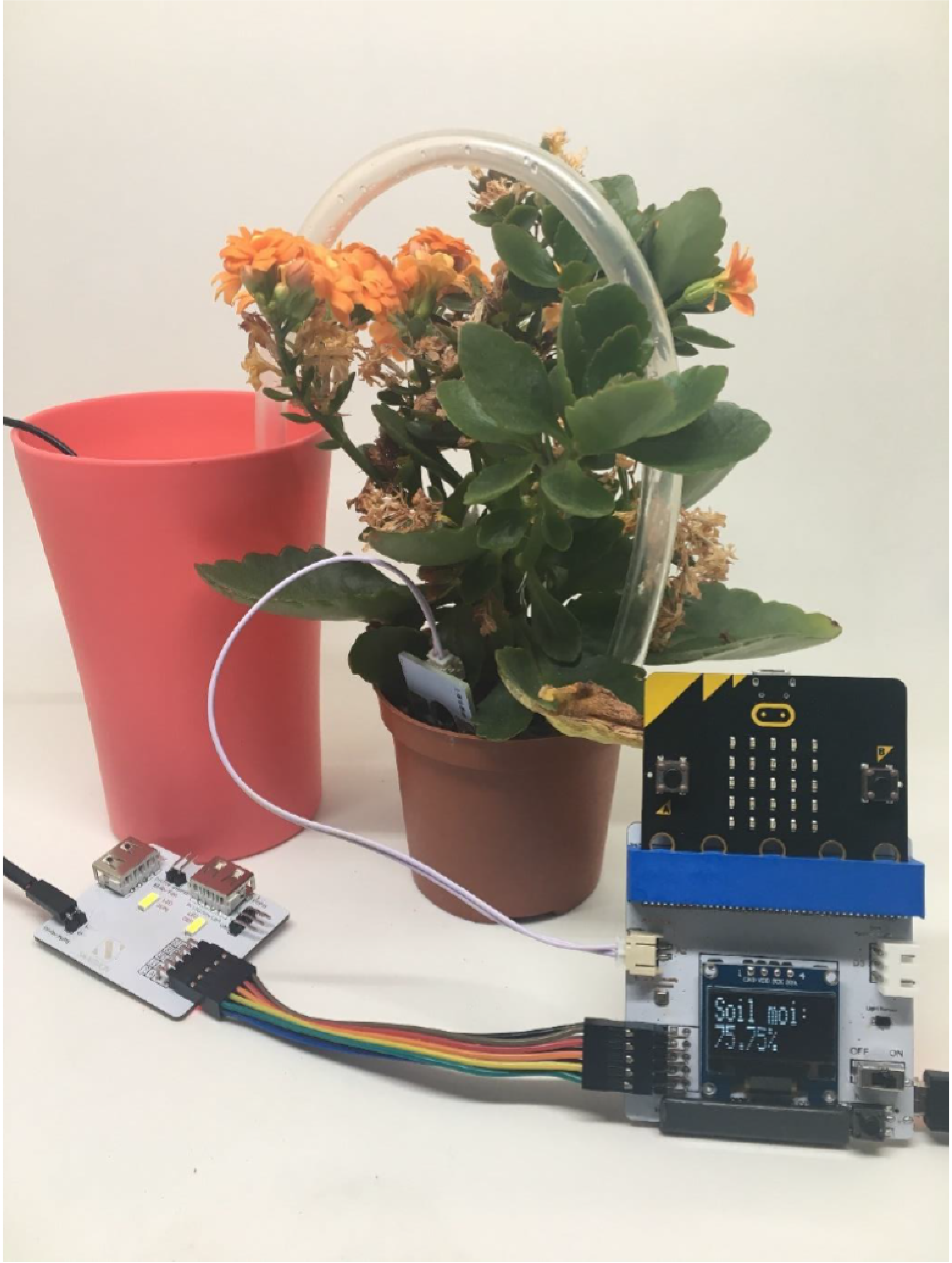
1. Basic (Add block on start) > Plant (Add block Initialize Smarthon multiple-sensor)
2. Functions (Make a Function > New function name:) > watering_2_seconds) > Plant-More (Add block Set Motor pump to intensity 0, change “0” to “400”) > Basic (Add block pause (ms) 100, change “100” to “2000”) > > Plant-More (Add block Set Motor pump to intensity 0) > Basic (Add block pause (ms) 100, change “100” to “10000”)
3. Basic (Add block forever) > Logic (Add block if…then…else, add 1 more if…then…else, delete else) > For 1st if : Logic (Add block 0 = 0, change “0” to Plant “Get soil moisture”; “=” to “<”; “0” to “50”)
4. For then: Advanced>Functions (Add block call function watering_2_seconds)
5. For else if: Logic (Add block 0 = 0, change “0” to Plant “Get soil moisture”; “=” to “>”; “0” to “90”)
6. For then: Plant-More (Add block Set Water pump to intensity 0)



```
on start
    Initialize Smarthon multiple-sensor

function watering_2_seconds
    Set Water pump to intensity 400
    pause (ms) 2000
    Set Water pump to intensity 0
    pause (ms) 10000

forever
    if Get soil moisture < 50 then
        call function watering_2_seconds
    else if Get soil moisture > 90 then
        Set Water pump to intensity 0
```

Hardware:

1. Connect Multiple Sensors and Multiple Actuators by connecting wire
2. Connect Water Pump and Pipe
3. Connect Water Pump & Pipe to Water pump (D6/P1) port of Multiple Actuators
4. **Download** the code and transfer to micro:bit
5. Plug the micro:bit into Multiple Sensors


Soil moi:
75.75%

Experiment:

- How does soil moisture level affect seed germination?
- Does soil moisture level affect the size of leave?
- Does soil moisture level affect photosynthesis or respiration of the plant?
- Does soil moisture level affect root growth?
- Does water salinity affect plant growth?

1.4.10 Project 5: Smart Environment (temperature control)

Level: ★ ☆ ☆ ☆ ☆

Part List:

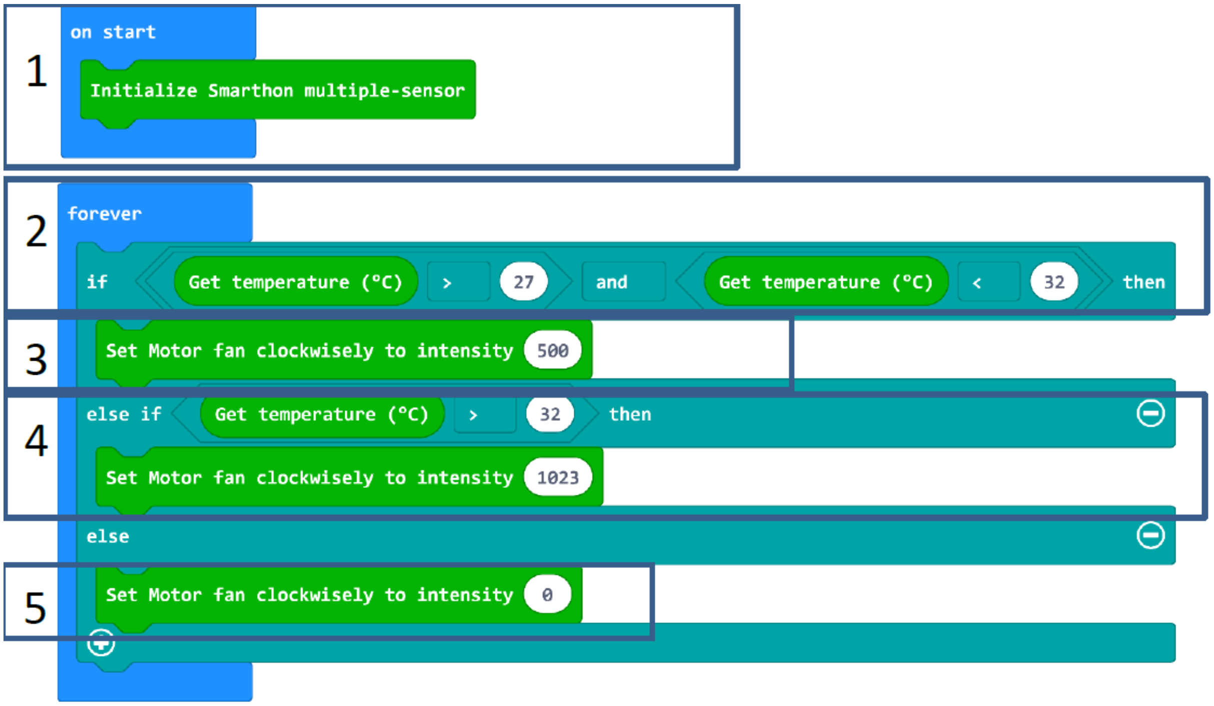
1. Temperature Sensor X1 (placed on Multiple Sensors for plant)
2. Motor Fan X1
3. Multiple Sensors for plant X1
4. Multiple Actuators for plant X1
5. Connecting Wire X1
6. micro:bit X1
7. USB Cable X1

Description:

In project 5, you can learn how to make a fan automation that allows you to control the ambient temperature.

Steps:

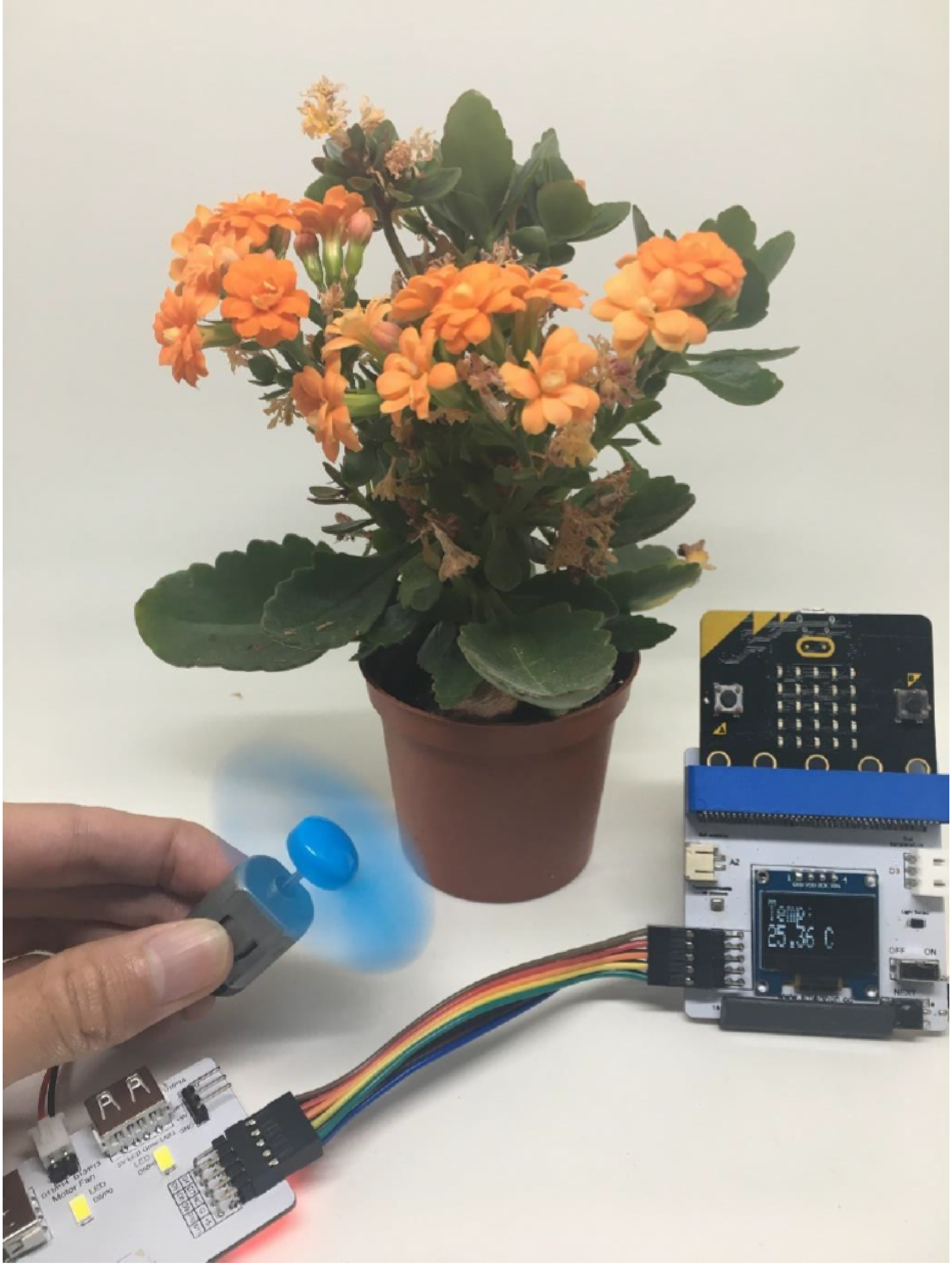
1. Basic (Add block on start) > Plant (Add block Initialize Smarthon multiple-sensor)
2. Basic (Add block forever) > Logic (Add block if. . . then. . . else, add 1 more if. . . then. . . else) > For 1st if : Logic (Add block “ ” and “ ”, 1st “ ” : Add block 0 = 0, change “0” to Plant “Get temperature(°C)”; “=” to “>”; “0” to “27” ; 2nd “ ” : Add block 0 = 0, change “0” to Plant “Get temperature(°C)”; “=” to “<”; “0” to “32”)
3. For then: Plant-More (Add block Set Motor fan clockwisely to intensity 0, change “0” to “500”)
4. For else if: Logic (Add block 0 = 0, change “0” to Plant “Get temperature(°C)”; “=” to “>”; “0” to “32”) > For then: Plant-More (Add block Set Motor fan clockwisely to intensity 0, change “0” to “1023”)
5. For else: Plant-More (Add block Set Motor fan clockwisely to intensity 0)



```
on start
    Initialize Smarthon multiple-sensor

forever
    if Get temperature (°C) > 27 and Get temperature (°C) < 32 then
        Set Motor fan clockwisely to intensity 500
    else if Get temperature (°C) > 32 then
        Set Motor fan clockwisely to intensity 1023
    else
        Set Motor fan clockwisely to intensity 0
```

Hardware:

1. Connect Multiple Sensors and Multiple Actuators by connecting wire
2. Connect Motor fan to Motor Fan (D11/P14; D10/P13) port of Multiple Actuators
3. **Download** the code and transfer to micro:bit
4. Plug the micro:bit into Multiple Sensors


Temp:
25.36 C
OFF
ON
NEXT
D3
A2
Motor Fan
LED

Experiment:

- How does temperature affect plant growth?
- Does temperature affect seed growth?
- Does motor fan affect plant transpiration rates?

1.4.11 Project 6: Automatic Humidity Control

Level: ★ ☆ ☆ ☆ ☆

Part List:

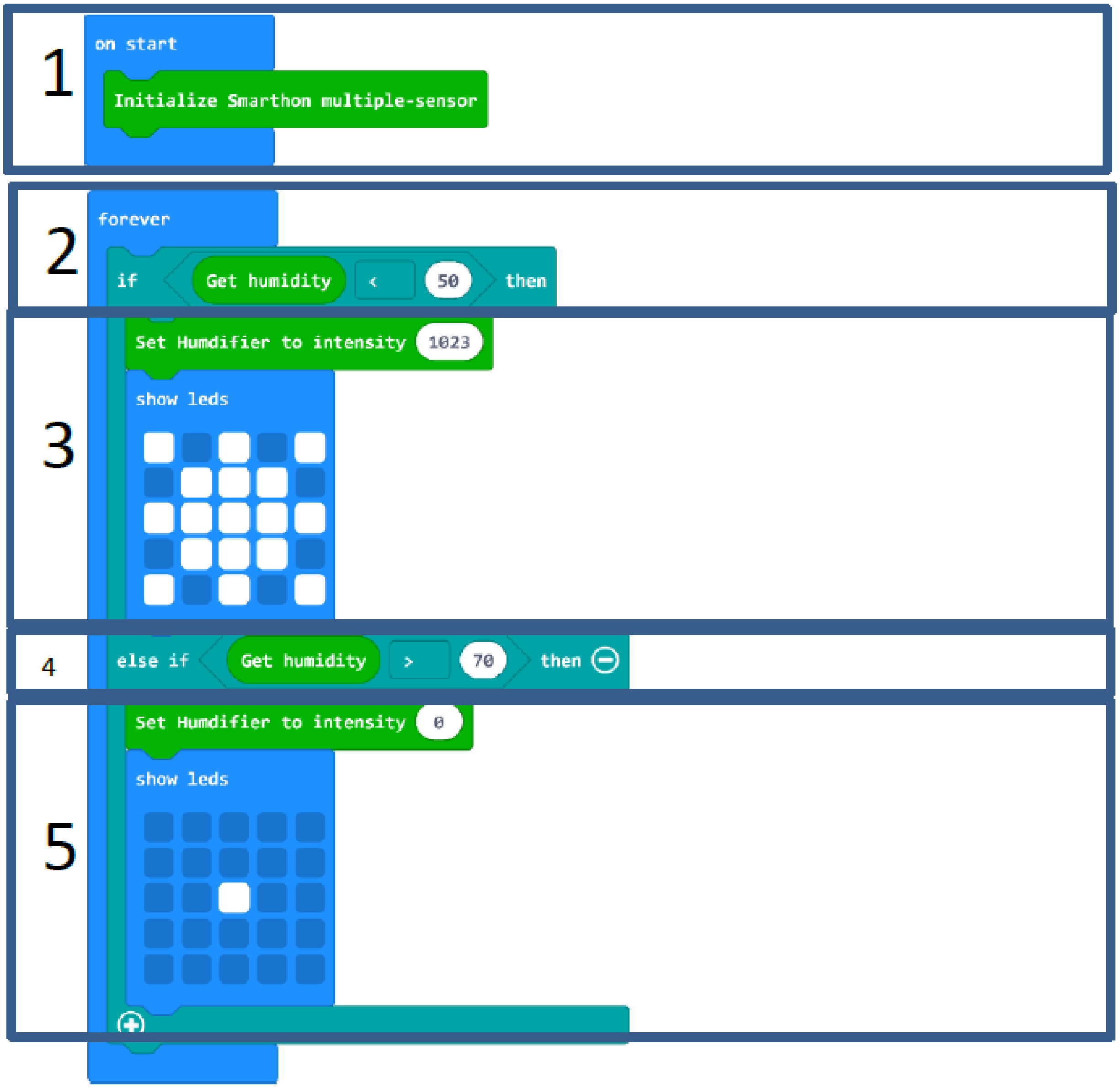
1. Humidity Sensor X1 (Placed on the Multiple Sensors)
2. Humidifier X1
3. Multiple Sensors for micro:bit X1
4. Multiple Actuators for micro:bit X1
5. Connecting Wire
6. micro:bit X1
7. USB Cable X2
8. A bottle of water X1

Description:

In project 6, you can learn how to increase humidity level by a humidity sensor.

Steps:

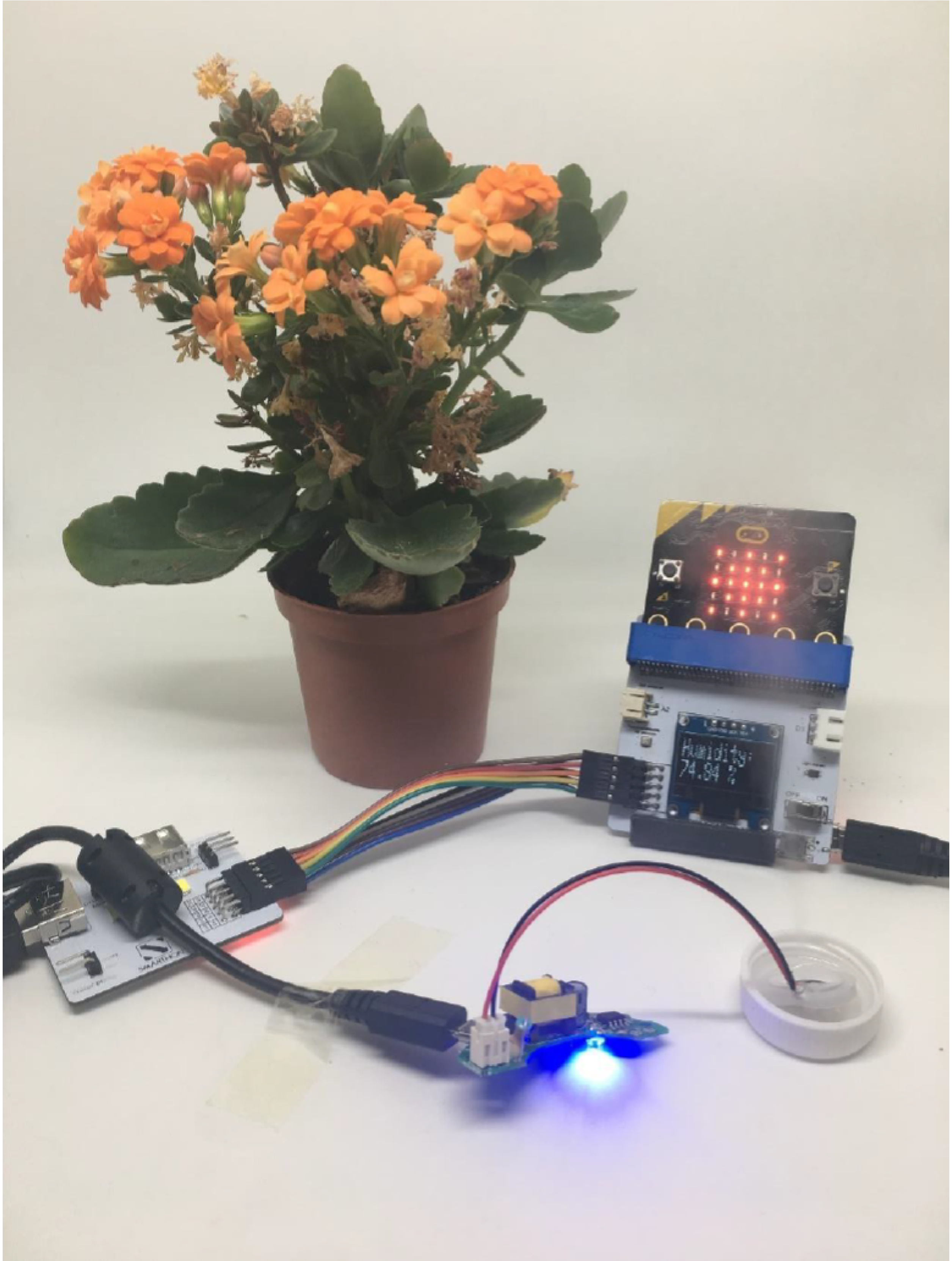
1. Basic (Add block on start) > Plant (Add block Initialize Smarthon multiple-sensor)
2. Basic (Add block forever) > Logic (Add block if…then…else, add 1 more if…then…else, delete else) > For 1st if : Logic (Add block 0 = 0, change “0” to Plant “Get humidity”; “=” to “<”; “0” to “50”)
3. For then: Plant-More (Add block Set Humidifier to intensity 0, change “0” to “1023”) > Basic (Add block show leds)
4. For else if: Logic (Add block 0 = 0, change “0” to Plant “Get humidity”; “=” to “>”; “0” to “70”)
5. For then: Plant-More (Add block Set Humidifier to intensity 0) > Basic (Add block show leds)



```
on start
  Initialize Smarthon multiple-sensor
forever
  if Get humidity < 50 then
    Set Humdifier to intensity 1023
    show leds
  else if Get humidity > 70 then
    Set Humdifier to intensity 0
    show leds
```

Hardware:

1. Connect Multiple Sensors and Multiple Actuators by connecting wire
2. Connect Humidifier module and USB Cable
3. Connect Humidifier & USB Cable to Humidifier (D4/P15) port of Multiple Actuators
4. **Download** the code and transfer to micro:bit
5. Plug the micro:bit into Multiple Sensors


Humidity:
74.84 %

Experiment:

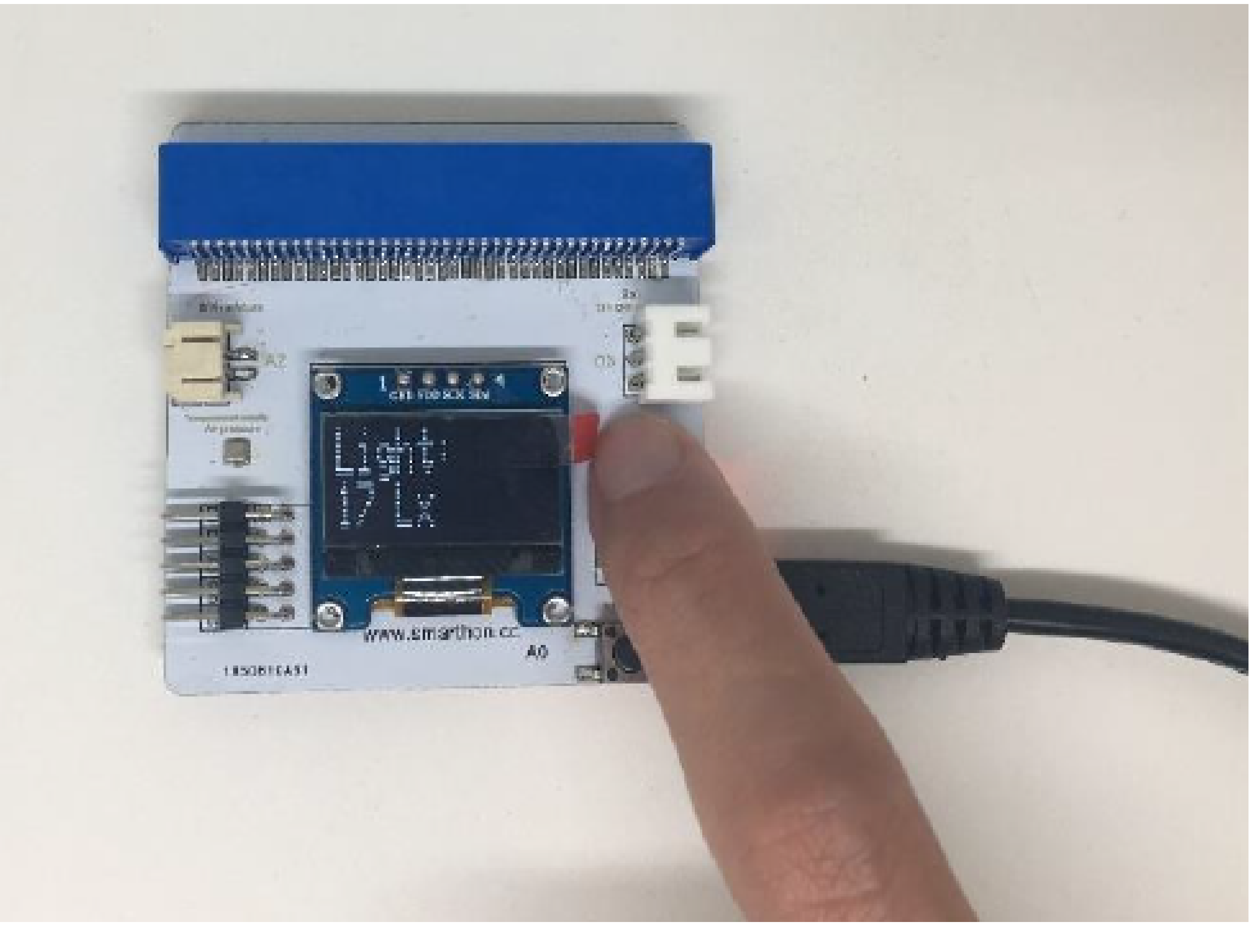
- Does humidifier affect plant transpiration rates?
- Does humidity level affect plant height?
- Does humidifier extend plant life during low soil moisture level?

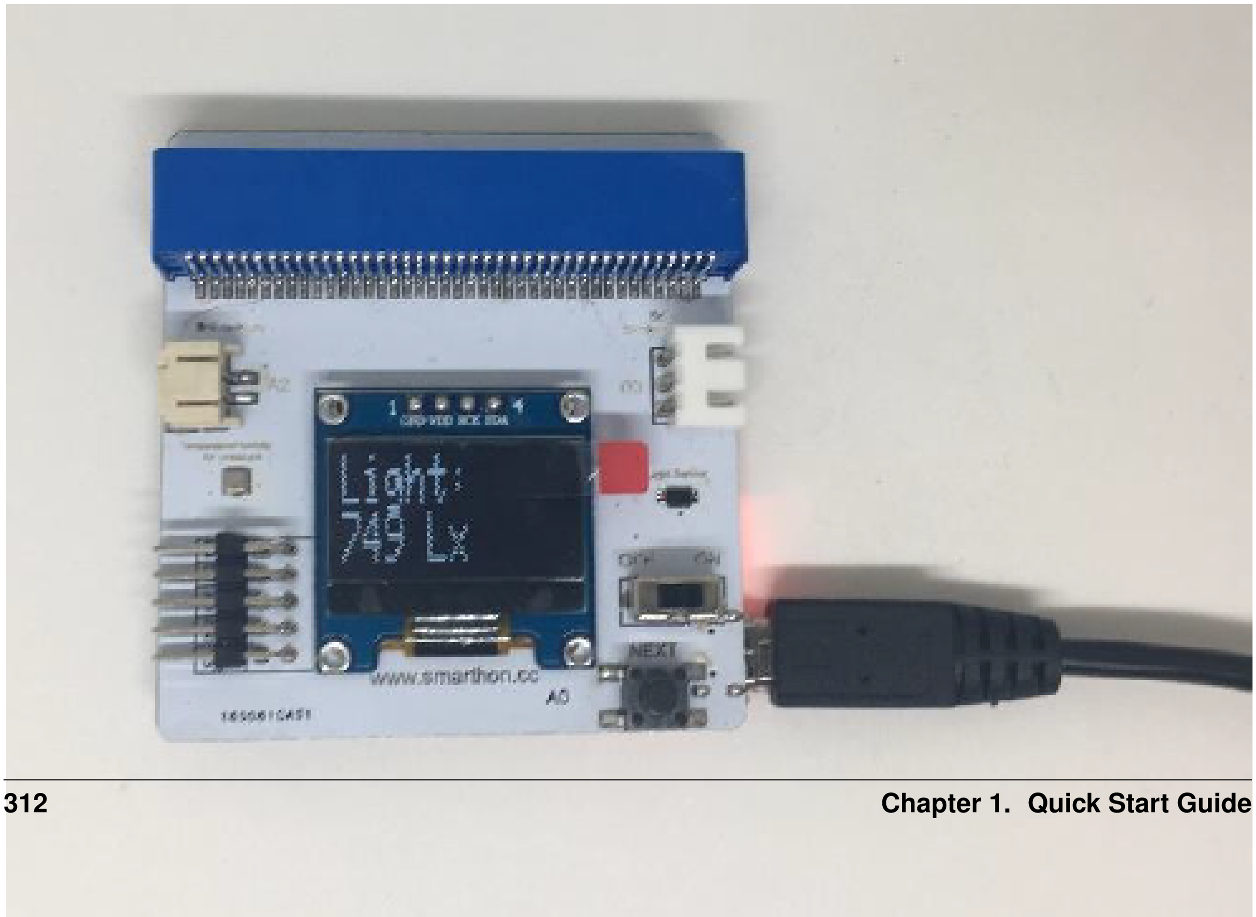
1.4.12 Appendix I (Answer of Reading Sensors)

Temperature & Humidity:

Please feel free to write down the value of the current condition.

Digital Light sensor:





When there is a sunlight, the digital light sensor value is around 900-1055 lx; while otherwise when you use a finger to cover the digital light sensor has decreased 17 lx. It reveals that when it is dark, the digital light sensor value becomes smaller.