

物聯網~空氣盒子

以Micropython實作
謝坤達

Jumbokh@gmail.com

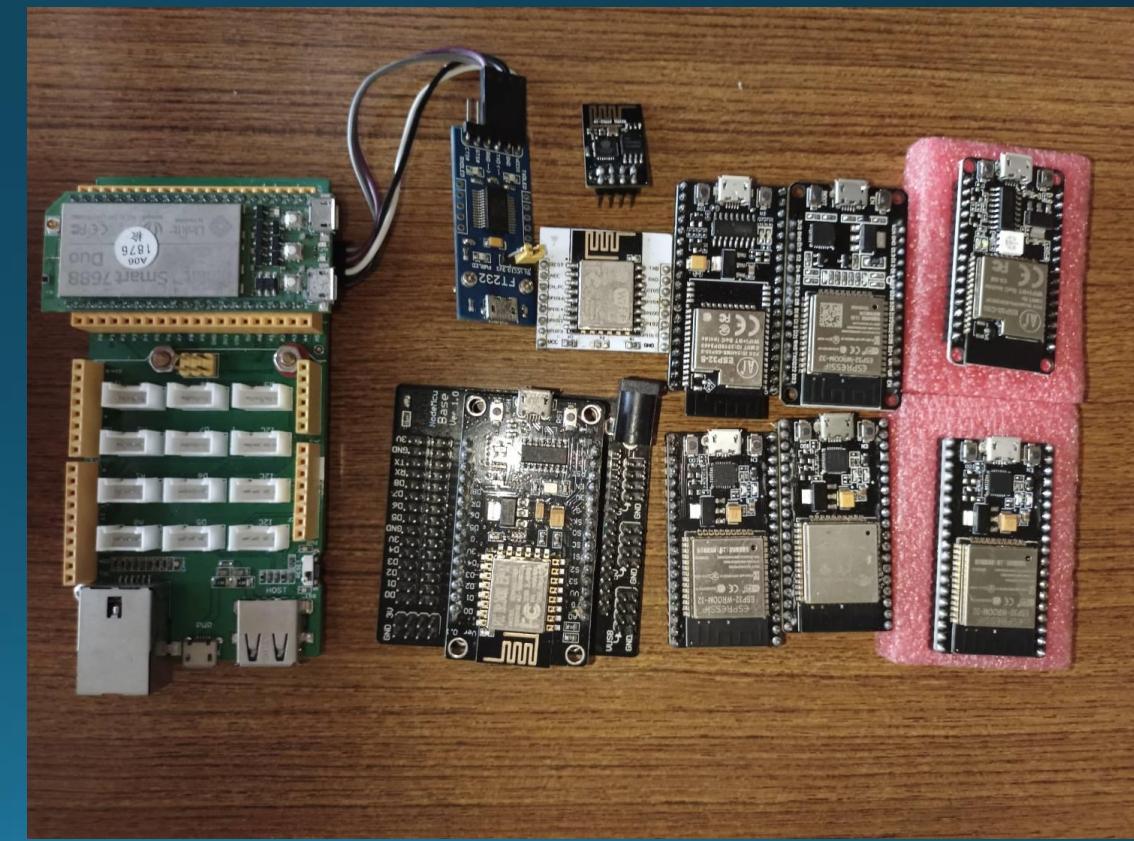
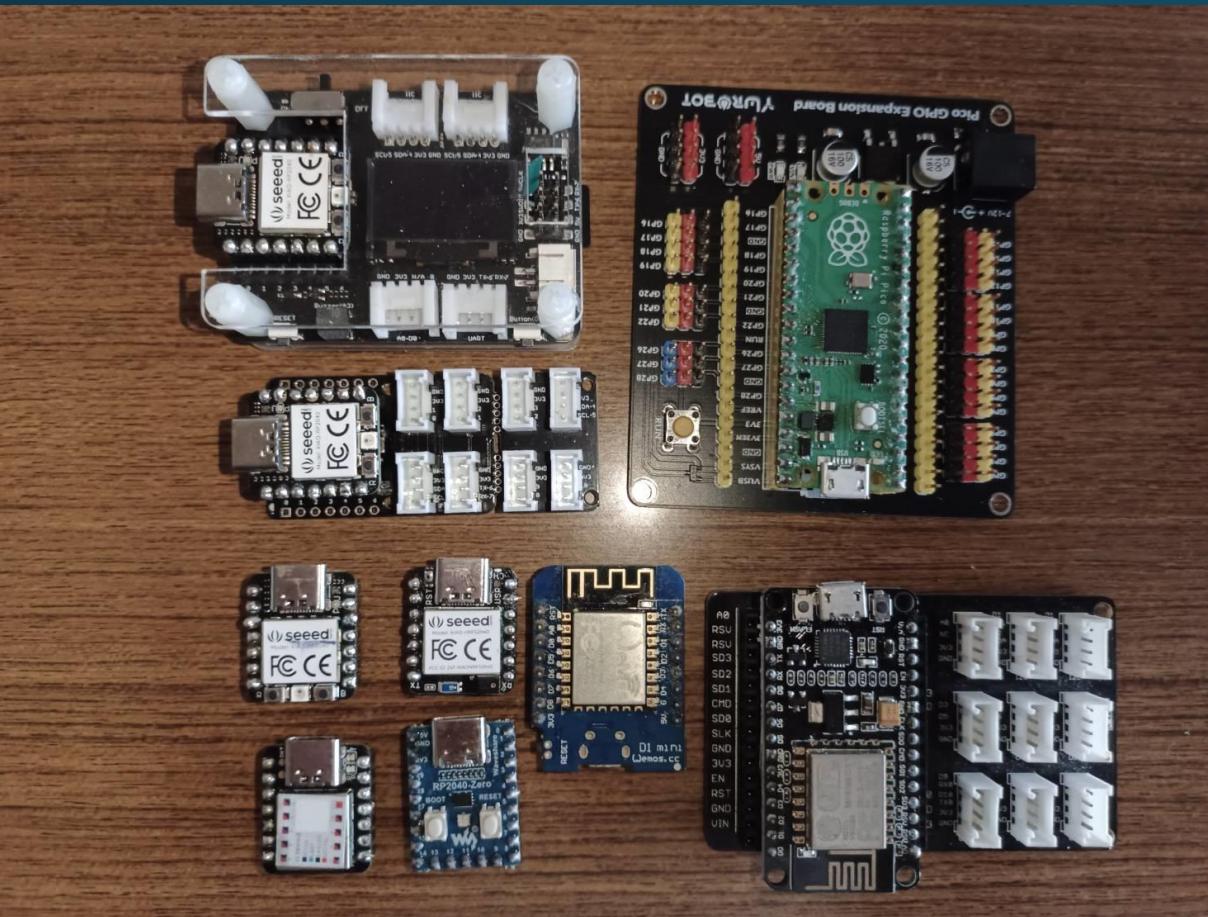
大綱

IOT Applications (shorturl.at/gBMW4)

Basic

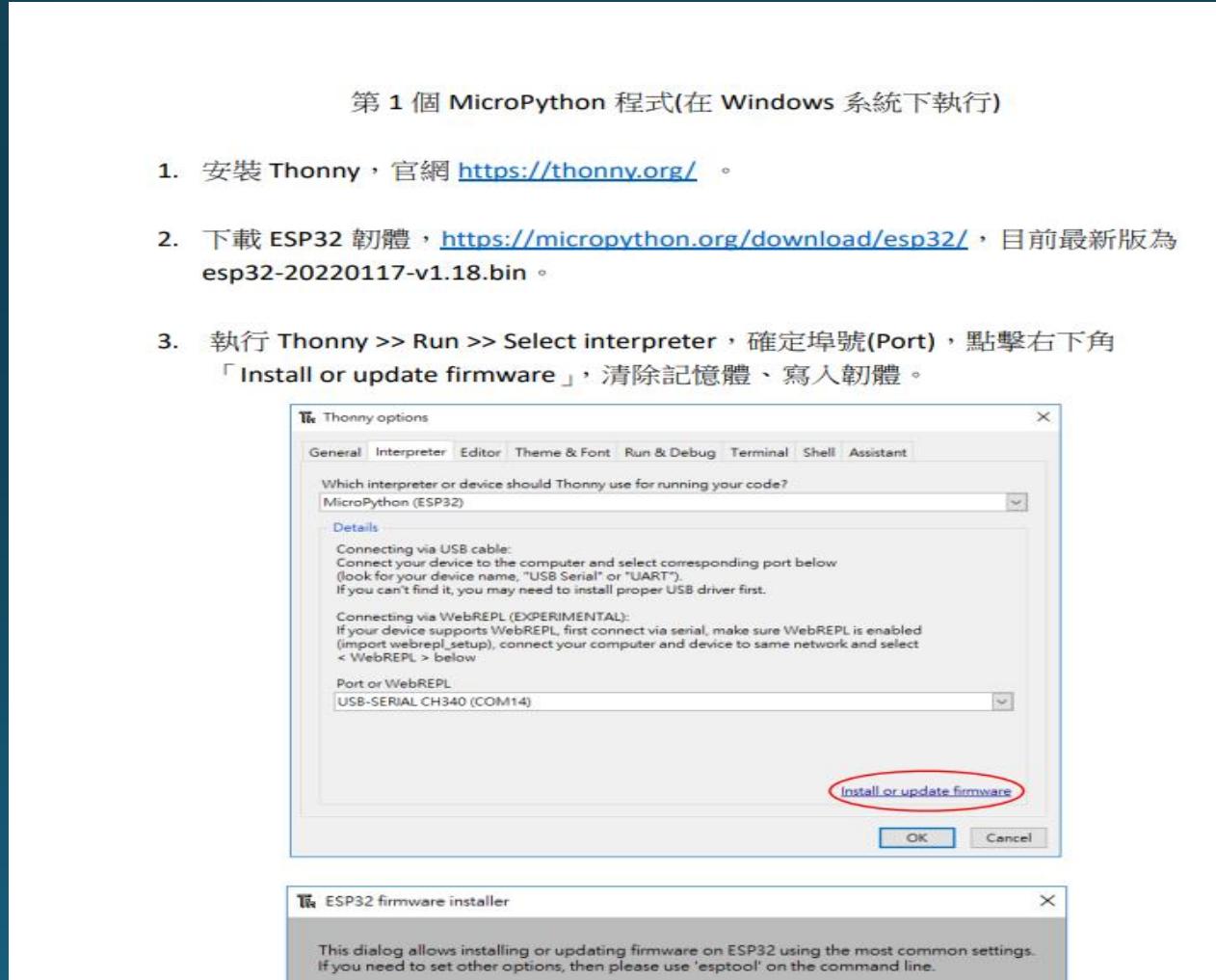
大綱

- ESP32 for Micropython and Thonny
- Basic Digital and analog Lab
 - DHT11 Temperature and Humidity sensor
 - I2C ssd1306 OLED display [ssd1306 lib](#)
 - Sharp dust sensor GP2Y1010AUOF
 - PM2.5 開放資料入口網站
 - [g0v](#)
- Internet of Things
 - url Request
 - MQTT
 - Air box

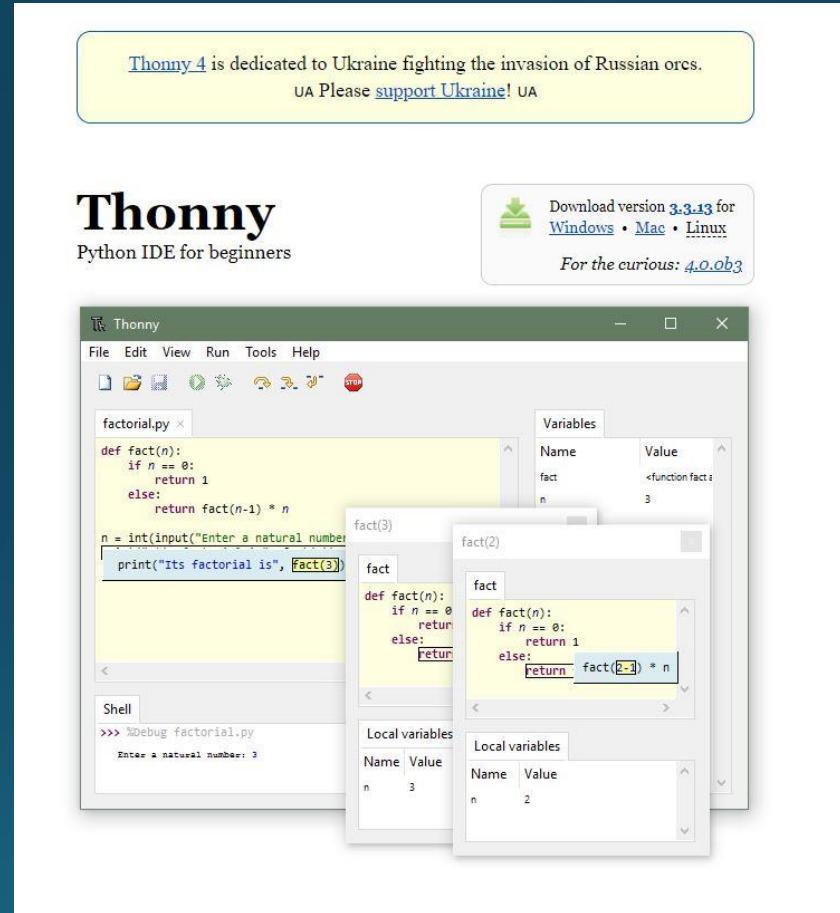


Thonny簡介

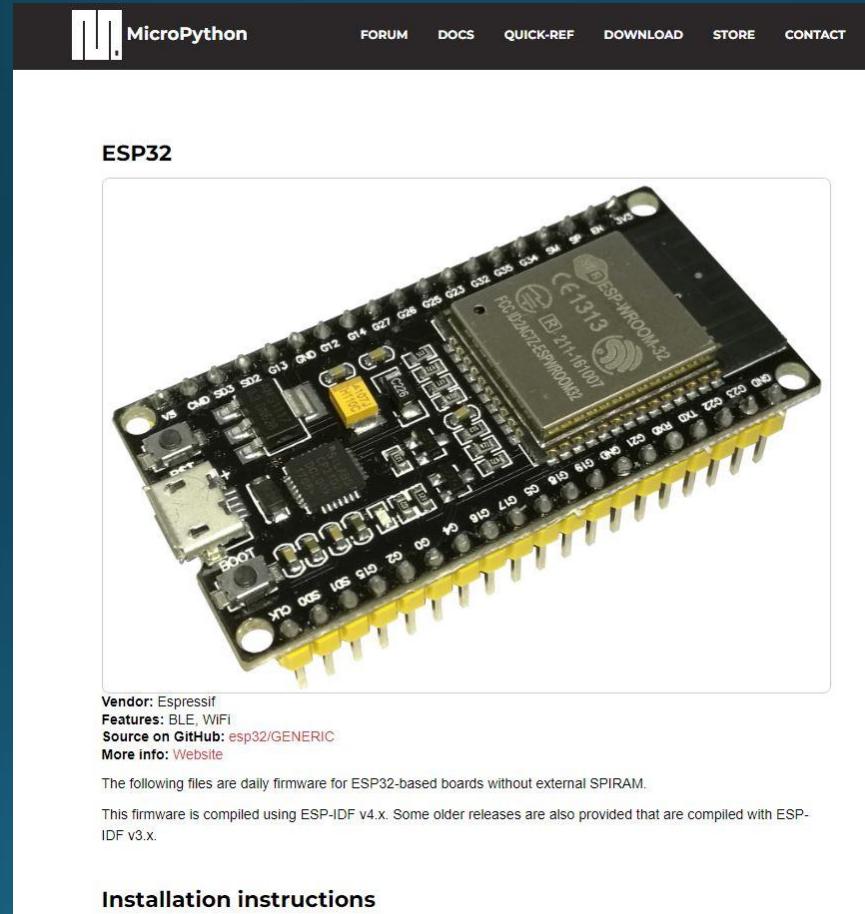
<https://reurl.cc/rDYoxE>



<https://thonny.org/>



<https://reurl.cc/rDYo8E>

A screenshot of the MicroPython website showing the ESP32 page. The page features the MicroPython logo at the top left, followed by a navigation bar with links to Forum, Docs, Quick-Ref, Download, Store, and Contact. The main content area is titled "ESP32" and displays a photograph of an ESP32 development board. Below the image, there is vendor information and a link to the GitHub repository. A note about daily firmware files and compilation details follows. At the bottom, a "Installation instructions" link is visible.

ESP32

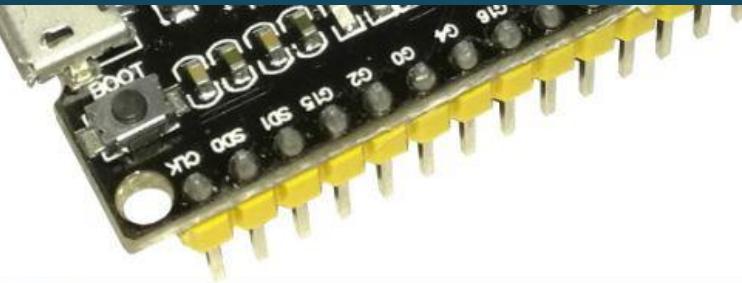
A photograph of an ESP32 development board, showing the central ESP32 chip and various pins and components.

Vendor: Espressif
Features: BLE, WIFI
Source on GitHub: [esp32/Generic](#)
More info: [Website](#)

The following files are daily firmware for ESP32-based boards without external SPIRAM.

This firmware is compiled using ESP-IDF v4.x. Some older releases are also provided that are compiled with ESP-IDF v3.x.

Installation instructions



Vendor: Espressif

Features: BLE, WiFi

Source on GitHub: [esp32/Generic](#)

More info: [Website](#)

The following files are daily firmware for ESP32-based boards without external SPIRAM.

This firmware is compiled using ESP-IDF v4.x. Some older releases are also provided that are compiled with ESP-IDF v3.x.

Installation instructions

Program your board using the esptool.py program, found [here](#).

If you are putting MicroPython on your board for the first time then you should first erase the entire flash using:

```
esptool.py --chip esp32 --port /dev/ttyUSB0 erase_flash
```

From then on program the firmware starting at address 0x1000:

```
esptool.py --chip esp32 --port /dev/ttyUSB0 --baud 460800 write_flash -z 0x1000 esp3  
2-20190125-v1.10.bin
```

Firmware

Releases

[v1.18 \(2022-01-17\) .bin](#) [.elf] [.map] [Release notes] (latest)

[v1.17 \(2021-09-02\) .bin](#) [.elf] [.map] [Release notes]

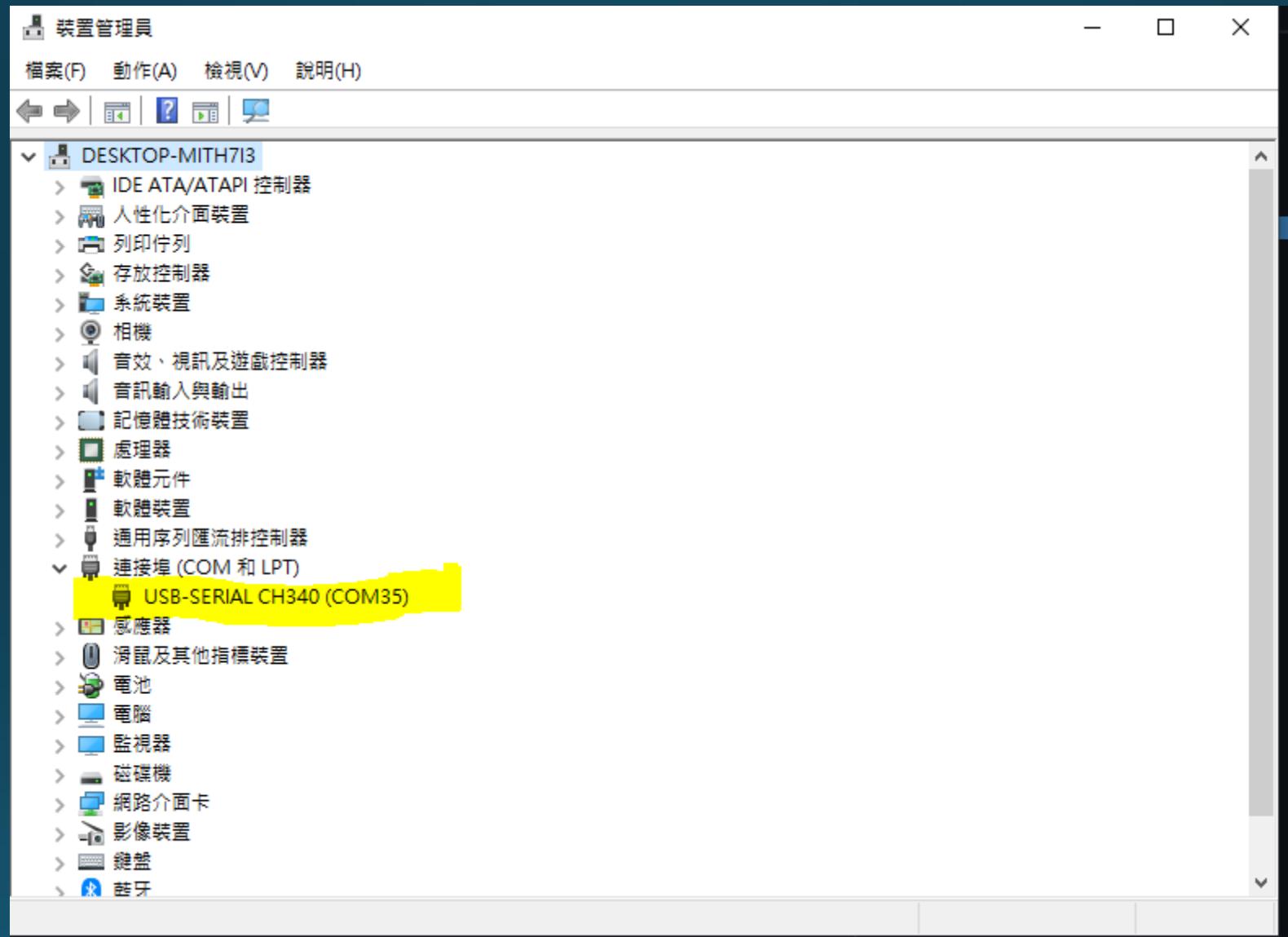
[v1.16 \(2021-06-23\) .bin](#) [.elf] [.map] [Release notes]

[v1.15 \(2021-04-18\) .bin](#) [.elf] [.map] [Release notes]

[v1.14 \(2021-02-02\) .bin](#) [.elf] [.map] [Release notes]

[v1.13 \(2020-09-02\) .bin](#) [.elf] [.map] [Release notes]

[v1.12 \(2019-12-20\) .bin](#) [.elf] [.map] [Release notes]



Thonny 選項

X

一般 直譯器 編輯器 主題和字型 執行 & 除錯 終端機 互動環境 (Shell) 協助功能

Thonny 應該使用哪一個直譯器或設備來執行你的程式？

MicroPython (ESP32)

詳細

Connecting via USB cable:

將設備連接到電腦，並選擇下方的連接埠

(尋找名稱中有 "USB Serial" 或是 "UART" 的設備).

如果找不到，你需要先安裝正確的 USB 驅動程式

Connecting via WebREPL (EXPERIMENTAL):

If your device supports WebREPL, first connect via serial, make sure WebREPL is enabled
(import webrepl_setup), connect your computer and device to same network and select
< WebREPL > below

選擇連接埠或是 WebREPL

USB-SERIAL CH340 (COM35)

[安裝或更新韌體](#)

確認

取消

Th Thonny

檔案 編輯 檢視 執行 工具 說明

檔案

本機 E:\MCU\micropython

- csuee
- web
- adcPWM.py
- breath.py
- btnBunting.py
- btndht.py
- btnLedsw.py
- ch10-2-2.py
- ch10-2-2a.py
- ch10-3-3.py
- ch10-3-3a.py

MicroPython 設備

- boot.py
- cnt.py
- config.py
- dht11try.py
- dhtoled.py
- i2cscan.py
- ssd1306.py

互動環境 (Shell)

```
today temp =31, humidity= 68
today temp =31, humidity= 68
today temp =31, humidity= 69
today temp =31, humidity= 68
today temp =31, humidity= 68
```

MicroPython v1.17 on 2021-09-02; ESP32 module with ESP32
Type "help()" for more information.

>>>

MicroPython (ESP32)

A screenshot of a GitHub repository page for "jumbokh / csuee1102-class".

The repository is public and contains a single file, "README.md".

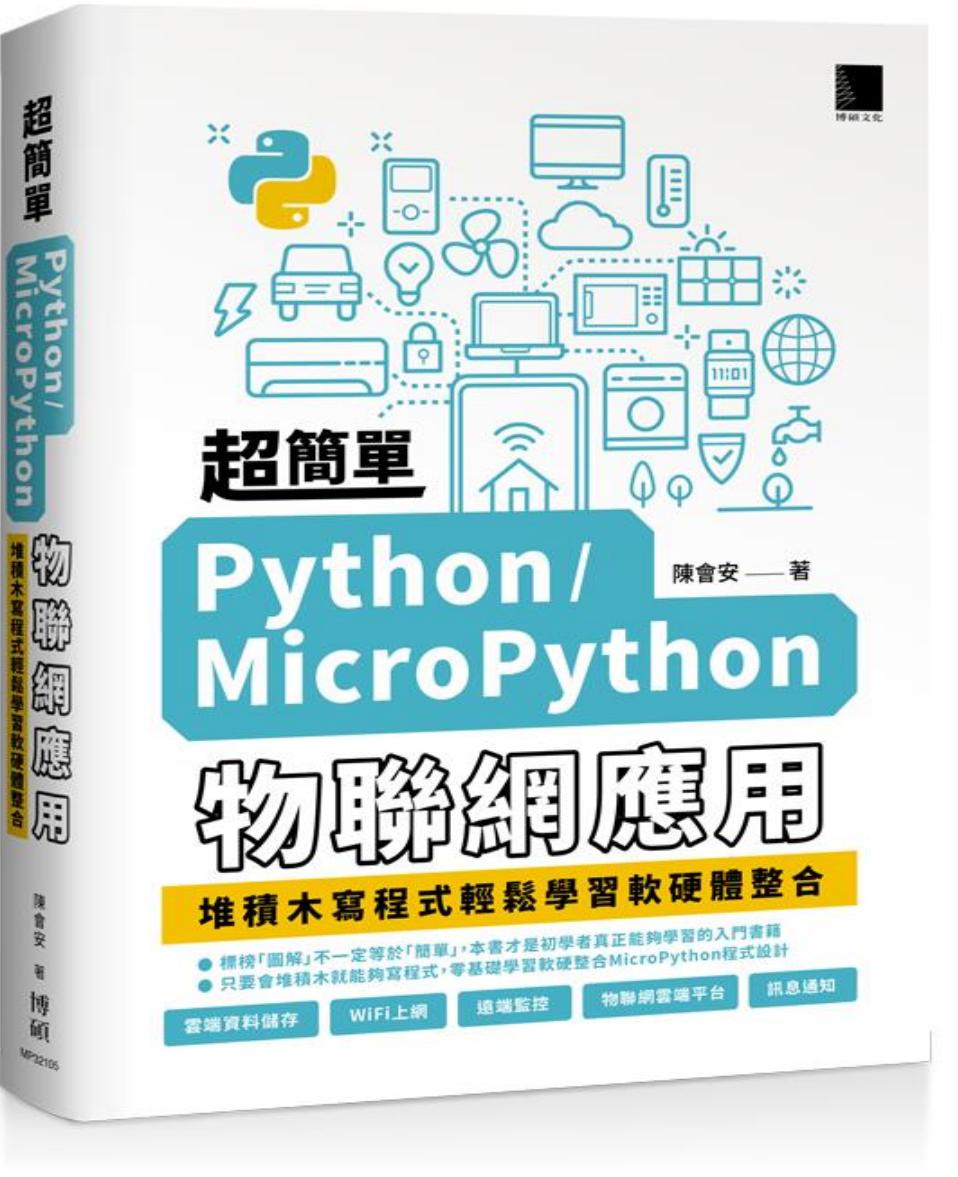
Key details from the page:

- Code tab is selected.**
- File: README.md**
- Contributor:** jumbokh (1 contributor)
- Last commit:** 85ce8a8 4 hours ago
- File statistics:** 38 lines (38 sloc) | 2.39 KB
- Actions:** Go to file, History, Raw, Blame, Copy, Download, Edit, Delete

The content of the README.md file is as follows:

```
# csuee1102-class

IOT Applications (shorturl.at/gBMW4)
```



書名：超簡單Python/MicroPython物聯網應用：堆積木寫程式輕鬆學習
軟硬體整合

書號：MP32105 作者：陳會安 ISBN：978-986-434-785-8

定價：NT\$650元 印刷：單色 頁數：448頁

書籍規格：17*23 上市日：2021/6/2 譯者：(無)

學習定位：初階 本書附件：官網下載

前往購買 >> [天璣](#) 類別：電腦技術

參與評論 成為朋友中第一個說這個讚的人：

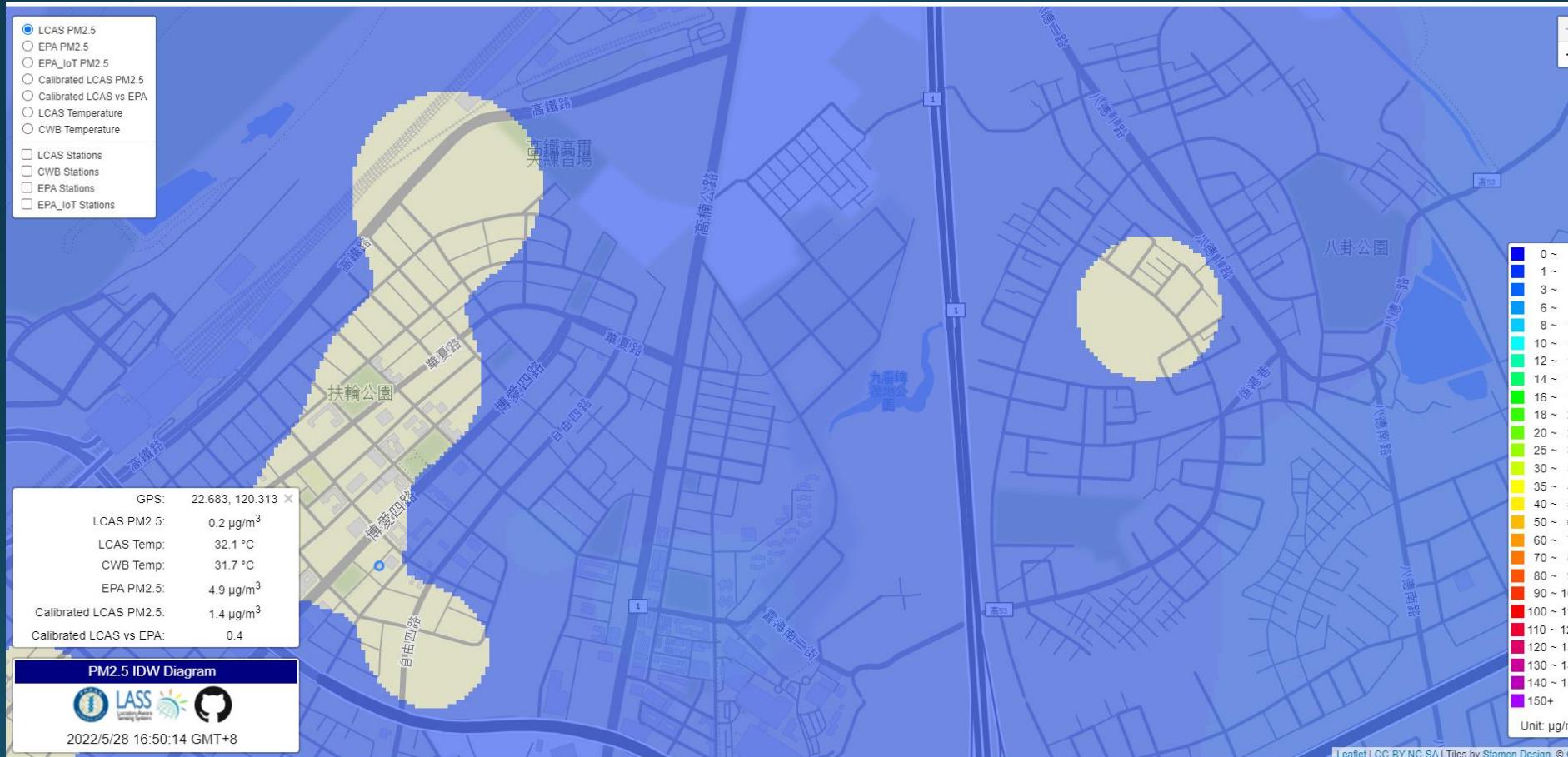
[Facebook](#) [Line](#) [Twitter](#)

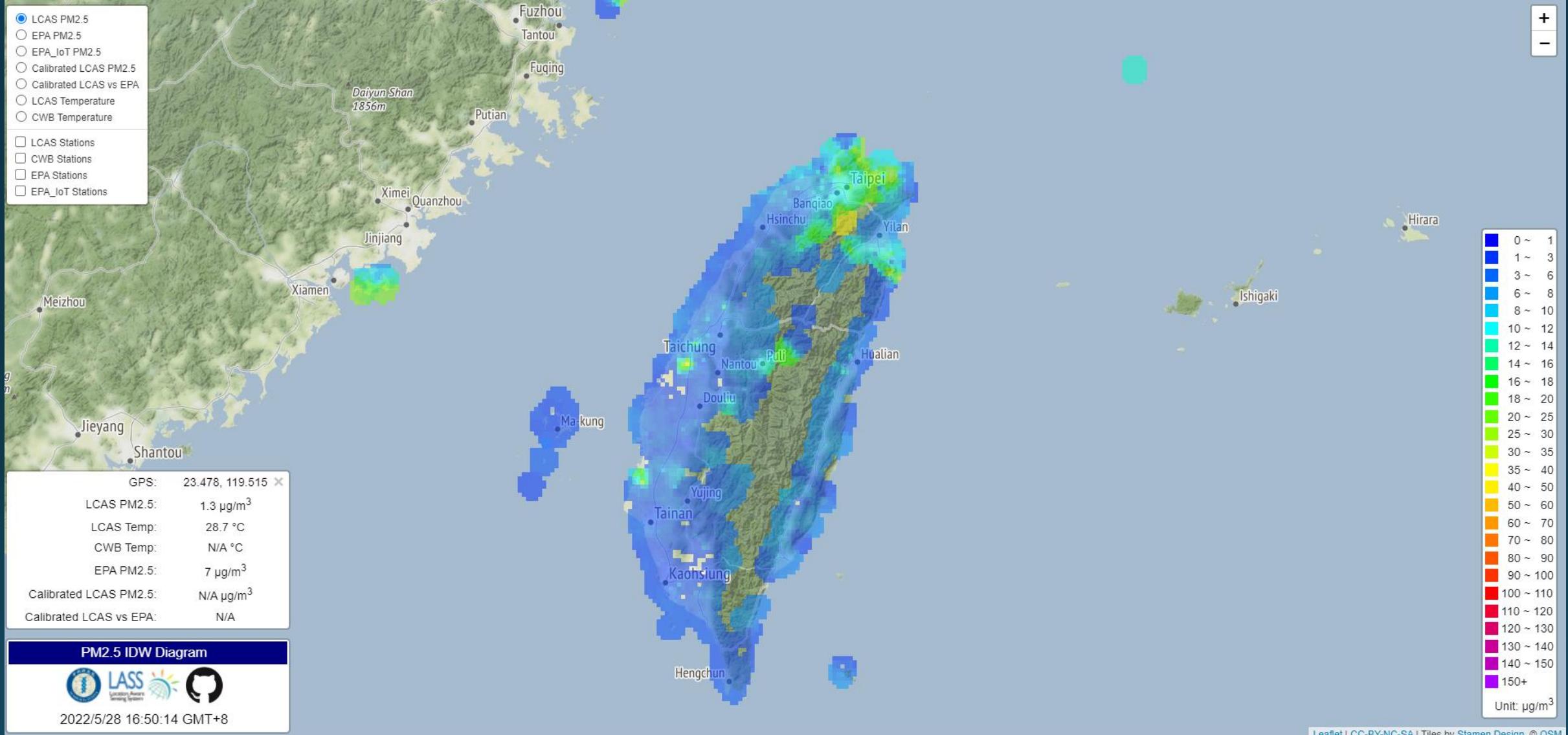
檔案下載

<https://pm25.lass-net.org/>

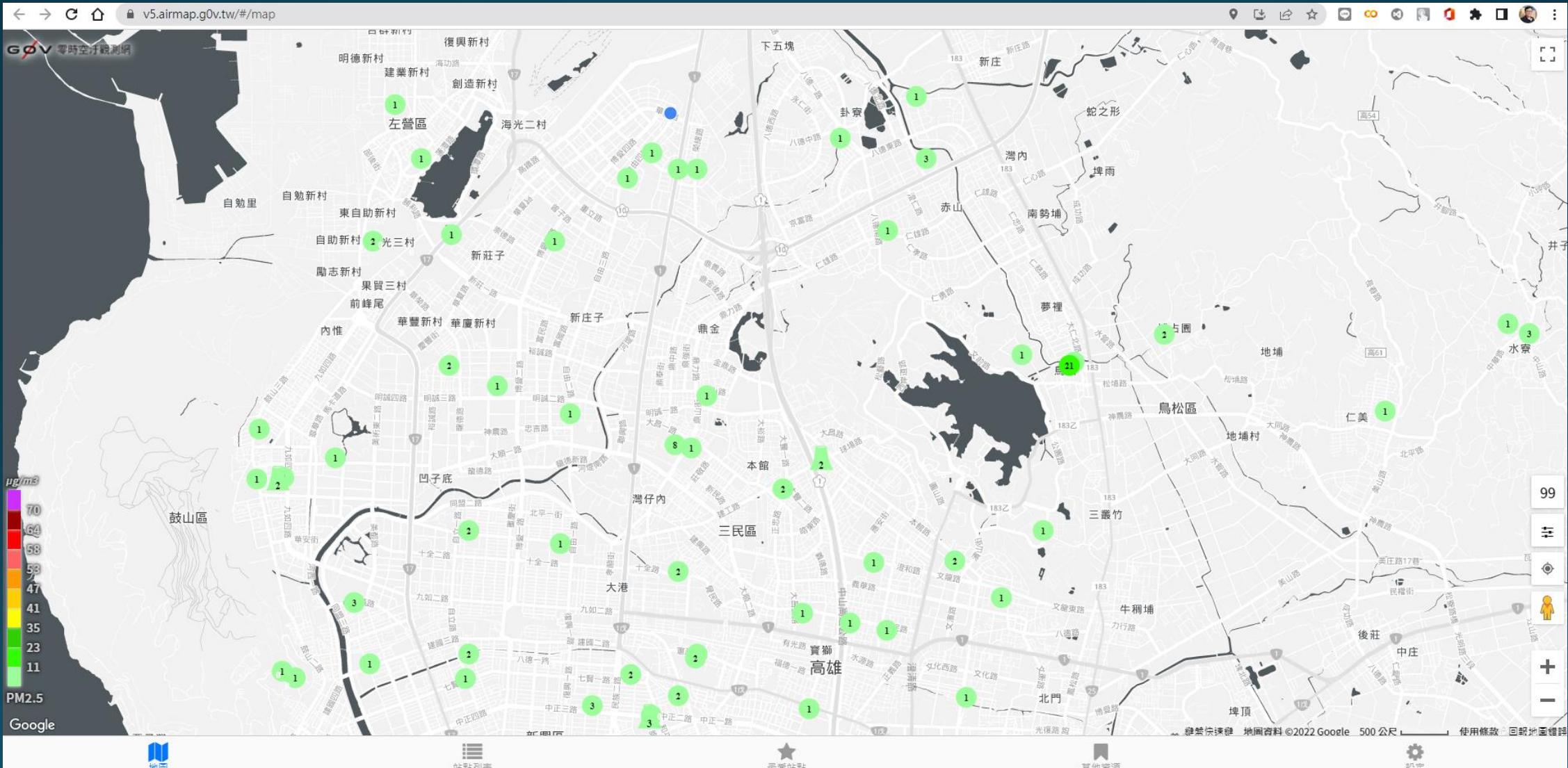
The screenshot shows the homepage of the PM2.5 OPEN DATA website. The header features a navigation bar with links: PM2.5, 儀表板 (Dashboard), 資料視覺化 (Data Visualization), 動畫 (Animation), APIS, APIS-DOC, 感測器狀態 (Sensor Status), 即時校正 (Real-time Correction), 社群參與 (Community Participation), and the Taiwan flag. The main content area has a blue background with a white central box. Inside the box, there's a graphic of a sun, mountains, lungs, and a factory emitting smoke. The LASS logo (Location Aware Sensing System) is prominently displayed. The text "PM 2.5 開放資料入口網站" (PM 2.5 Open Data Entry Website) is centered. Below it, a message says "歡迎使用！" (Welcome!) and "本站提供 PM_{2.5} 即時感測資料的 API、資料分析結果以及相關的資料視覺化工具服務。" (This site provides real-time sensing data of PM_{2.5} through API, analysis results, and related visualization tools.)

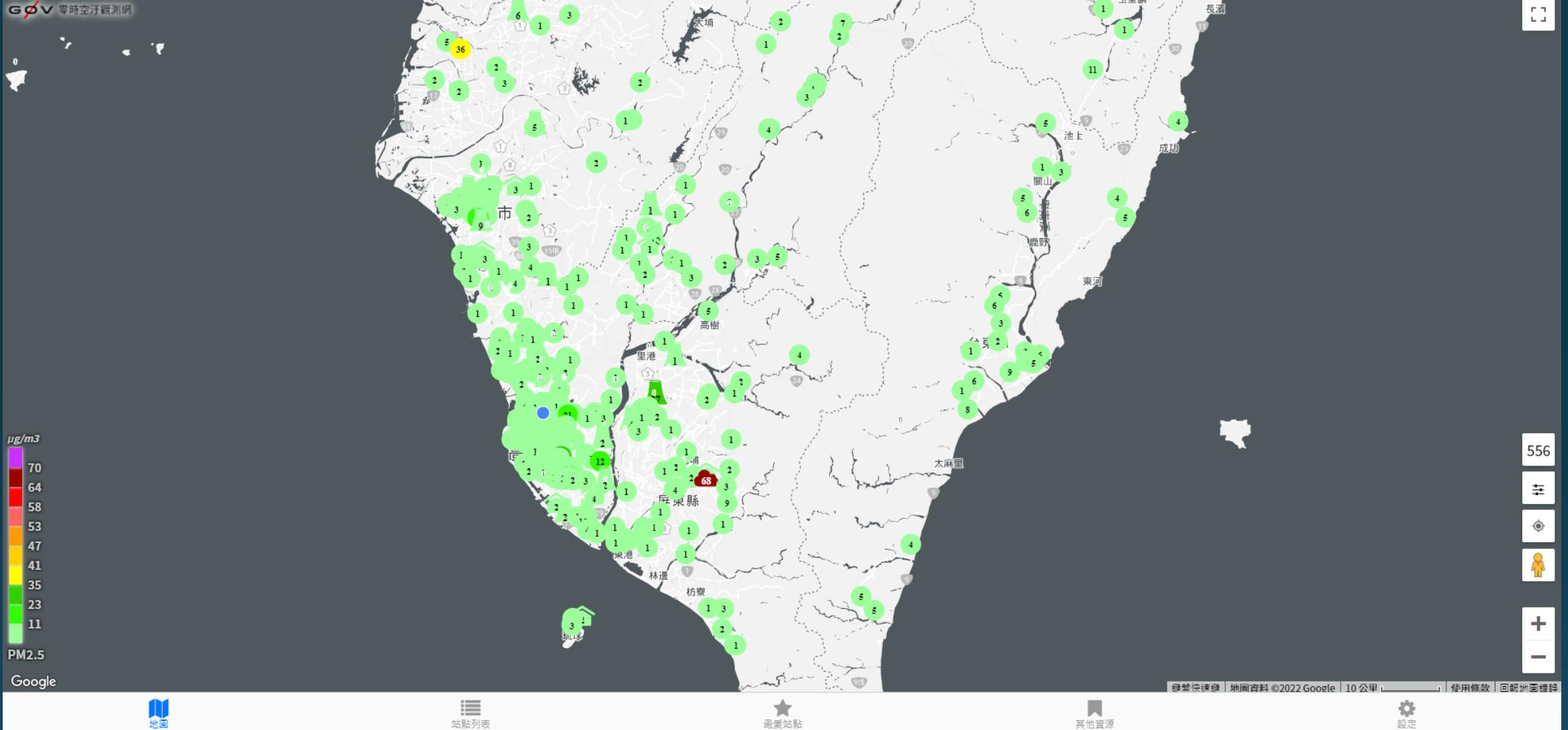
<https://pm25.lass-net.org/GIS/IDW/>





<https://v5.airmap.gov.tw/#/map>





📁	Lab3.DustSensor	add mqtt oled
📁	Lab4.WiFi	update config
📁	Lab5. MQTT	add mqtt oled
📁	Lab6. LineBot	commit all lab
📁	Lab7. LASS	commit all lab
📁	Lab8. URL-Request	update config
📁	Lab9. openWeatherMap	add mqtt oled
📁	refers	add mqtt Lass Oled
📄	README.md	Update README.md
📄	esp32-20210902-v1.17.bin	commit all lab

main ▾

csuee1102-class / refers / LabPractice.txt



jumbokh add mqtt Lass Oled

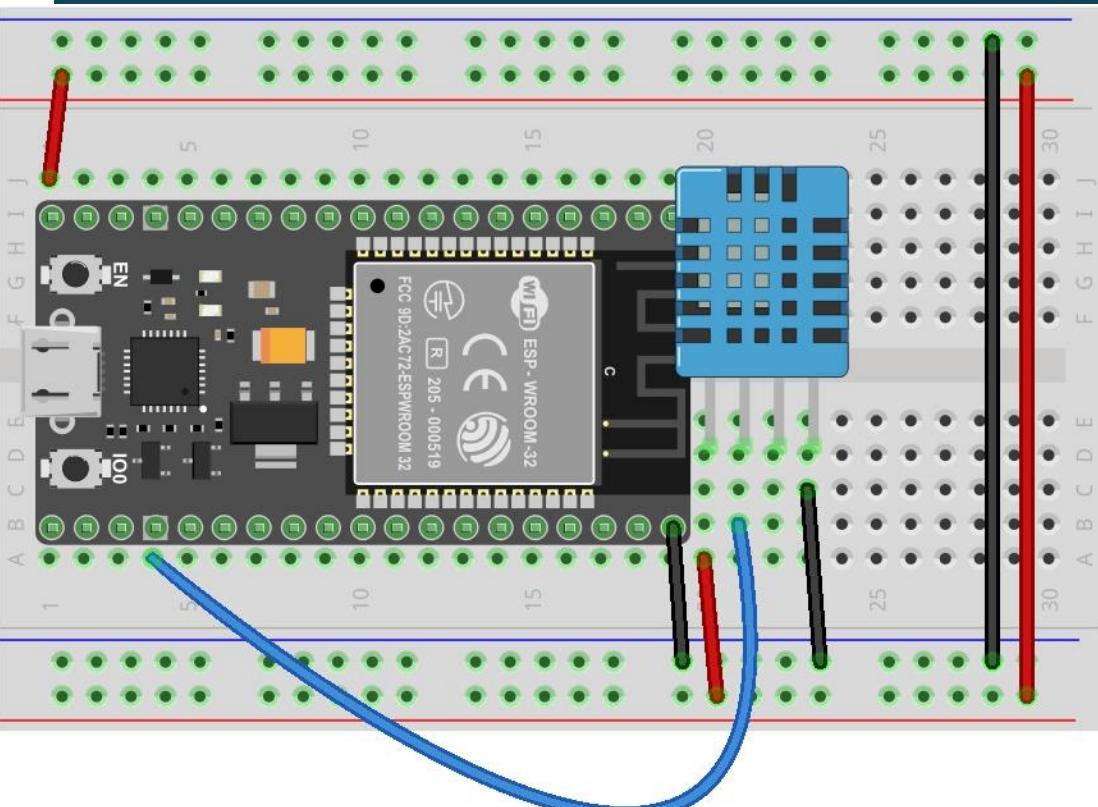
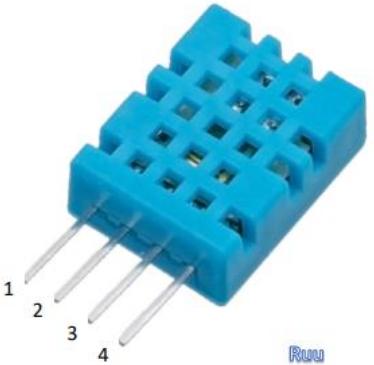
1 contributor

12 lines (12 sloc) | 206 Bytes

```
1  1. dht11try.py
2  2. i2cscan.py
3  3. oledTest.py
4  4. dhtoled.py
5  5. DustValue.py
6  6. airqoled.py
7  7. config.py, cnt.py
8  8. openweathermap.py
9  9. urllab.py
10 10. mqttTemp.py
11 11. mqttLass.py
12 12. mqttLassOled.py
```

DHT 11 Pin

1	VCC
2	Data
3	NC
4	GND



DHT11 PIN

- 1 VCC
2. Data (GPIO 15)
3. NC
4. GND

Thonny - MicroPython 設備 :: /dht11try.py @ 10:19

檔案 編輯 檢視 執行 工具 說明

檔案 x [dht11try.py]

```
1 import machine, dht
2 import utime
3
4 d11=dht.DHT11(machine.Pin(15))
5 while True:
6     d11.measure()                      # start to measure
7     temp=d11.temperature()            # return the temperature
8     humid=d11.humidity()              # return the humidity
9     print("%3d 度C, %3d %%"%(temp,humid))
10    utime.sleep(3)
```

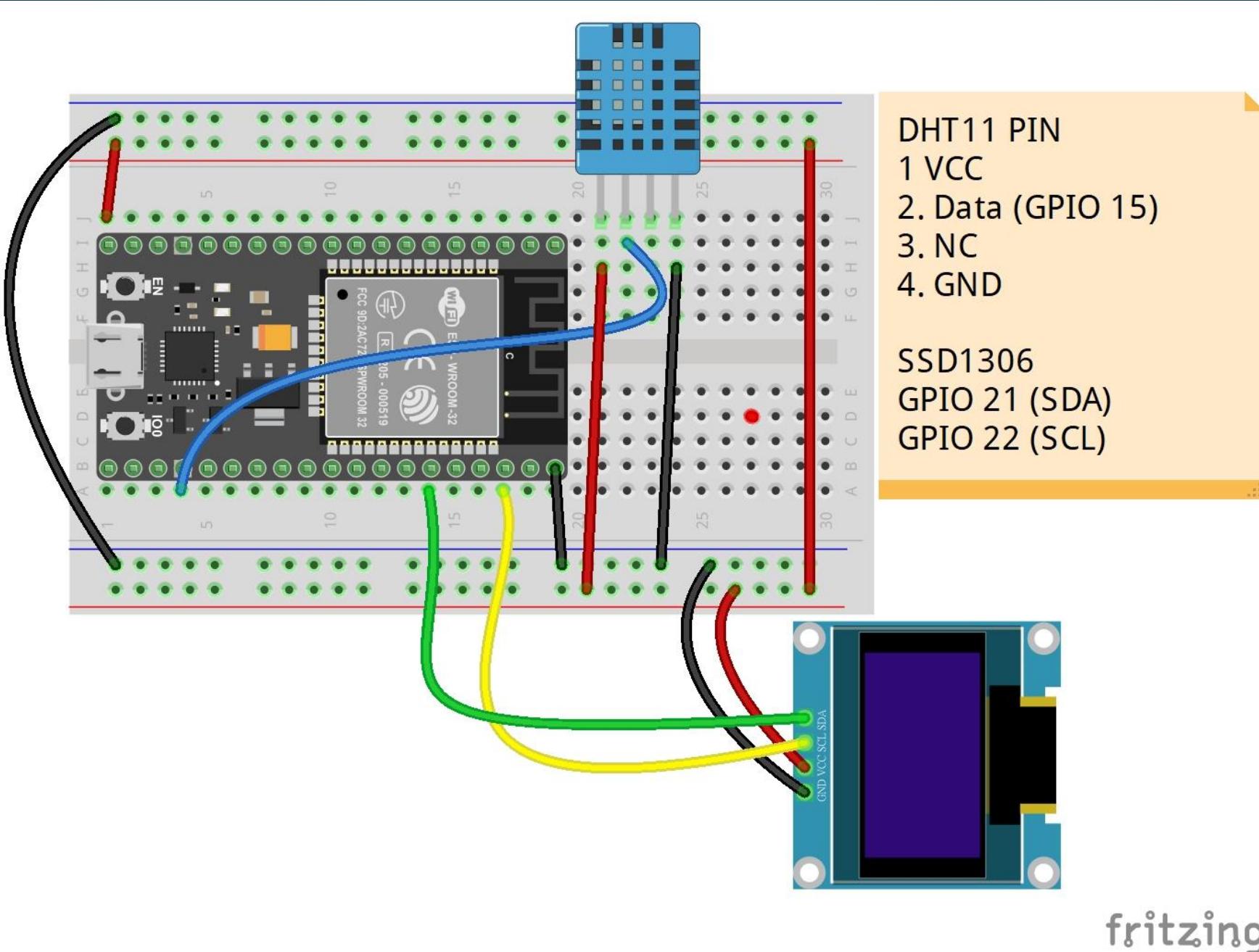
MicroPython 設備

- boot.py
- cnt.py
- config.py
- dht11try.py
- dhtoled.py
- i2cscan.py
- ssd1306.py

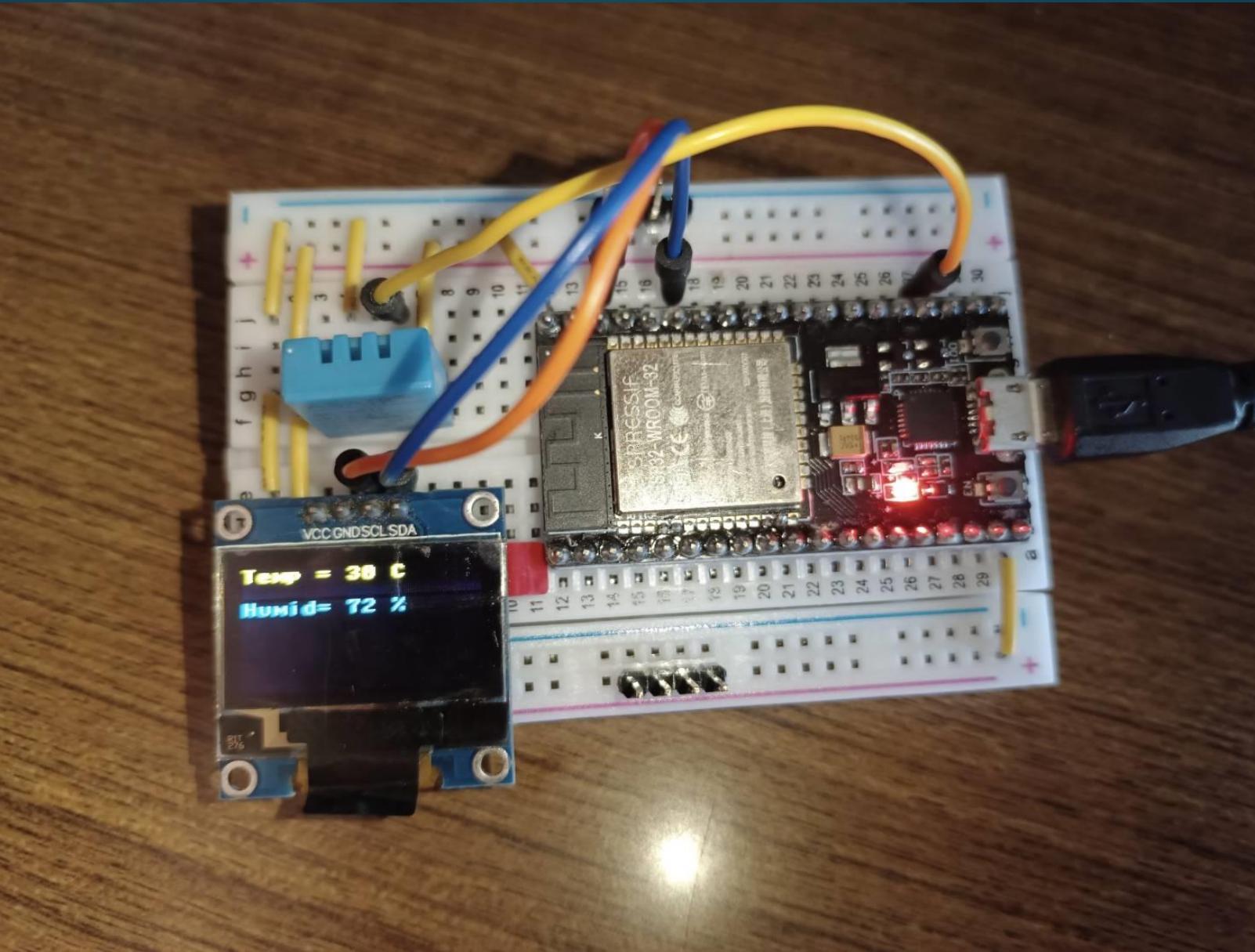
互動環境 (Shell) x

```
today temp =31, humidity= 68
```

```
MicroPython v1.17 on 2021-09-02; ESP32 module with ESP32
Type "help()" for more information.
>>> %Run -c $EDITOR_CONTENT
31 度C, 68 %
31 度C, 70 %
```



fritzing



Thonny - MicroPython 設備 ::/i2cscan.py @ 18:70

檔案 編輯 檢視 執行 工具 說明

檔案 x [dht11try.py] [i2cscan.py]

本機 E:\MCU\micropython

csuee
web
adcPWM.py
breath.py
btnBunting.py
btndht.py
btnLedsw.py
ch10-2-2.py
ch10-2-2a.py
ch10-3-3.py
ch10-3-3a.py

MicroPython 設備

boot.py
cnt.py
config.py
dht11try.py
dhtoled.py
i2cscan.py
ssd1306.py

```
1 # Scanner i2c en MicroPython | MicroPython i2c scanner
2 # Renvoi l'adresse en decimal et hexa de chaque device connecte sur le bus i2c
3 # Return decimal and hexa adress of each i2c device
4 # https://projetsdiy.fr - https://diyprojects.io (dec. 2017)
5
6 import machine
7 i2c = machine.I2C(scl=machine.Pin(22), sda=machine.Pin(21))
8
9 print('Scan i2c bus...')
10 devices = i2c.scan()
11
12 if len(devices) == 0:
13     print("No i2c device !")
14 else:
15     print('i2c devices found:',len(devices))
16
17 for device in devices:
18     print("Decimal address: ",device," | Hexa address: ",hex(device))|
```

互動環境 (Shell)

```
>>> %Run -c $EDITOR_CONTENT
Warning: I2C(-1, ...) is deprecated, use SoftI2C(...) instead
Scan i2c bus...
i2c devices found: 1
Decimal address: 60 | Hexa address: 0x3c
>>>
```

Thonny - MicroPython 設備 ::/dhtoled.py @ 20:32

檔案 編輯 檢視 執行 工具 說明

開啟舊檔 (Ctrl+O)

本機 E:\MCU\micropython

csuee
web
adcPWM.py
breath.py
btnBunting.py
btndht.py
btnLedsw.py
ch10-2-2.py
ch10-2-2a.py
ch10-3-3.py
ch10-3-3a.py

MicroPython 設備

boot.py
cnt.py
config.py
dht11try.py
dhtoled.py
i2cscan.py
ssd1306.py

[dht11try.py] [i2cscan.py] [dhtoled.py]

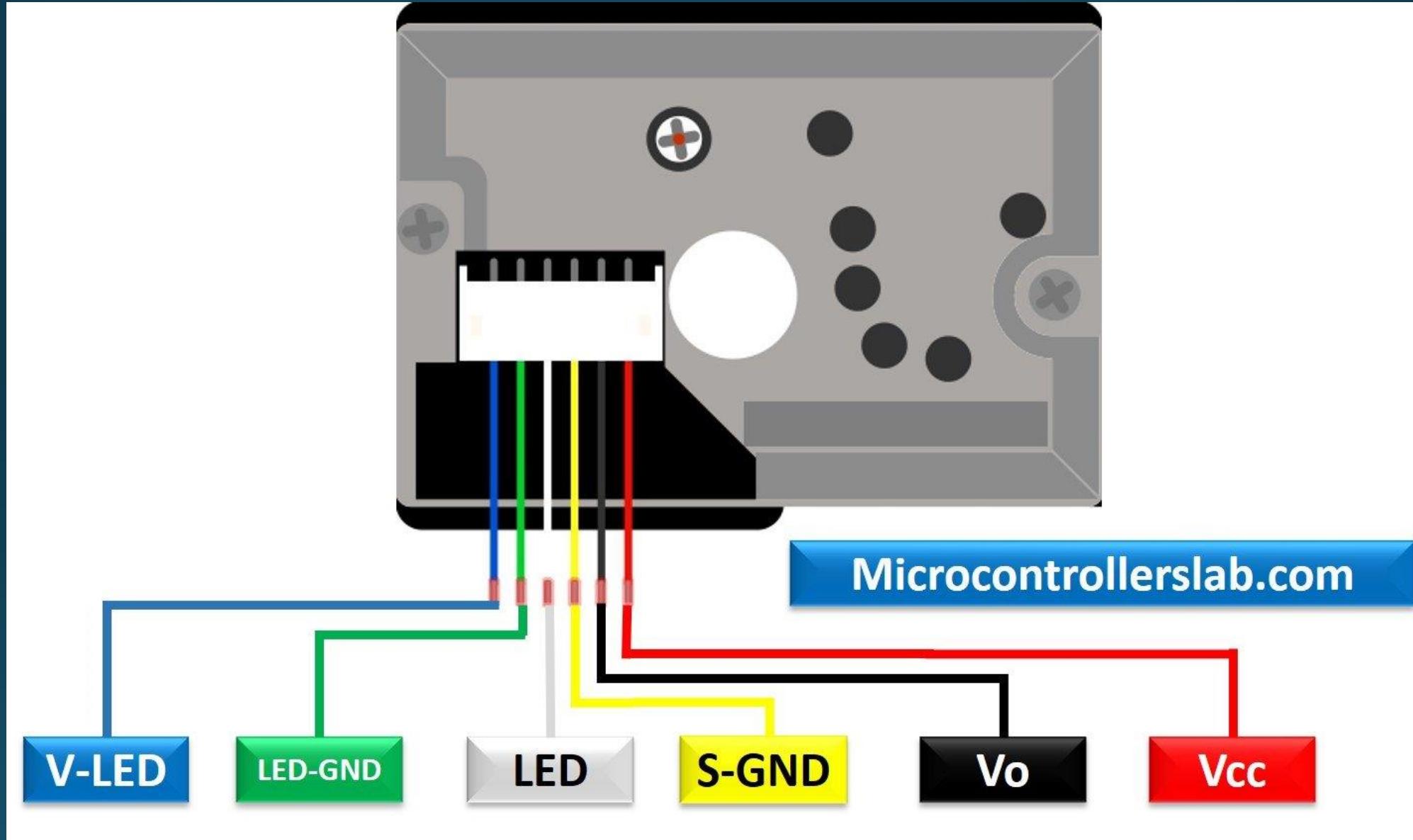
```
1 import machine, dht, utime, ssd1306
2 d11=dht.DHT11(machine.Pin(15))
3 hw_i2c1 = machine.I2C(scl=machine.Pin(22), sda=machine.Pin(21))# machine.I2C(0x3c, freq=2000)
4 oled096=ssd1306.SSD1306_I2C(128, 64, hw_i2c1) #128,64
5 oled096.fill(0)
6 #oled096.text('~ TEMP & HUMID ~',0,0)
7 oled096.text('Temp = {} C'.format(d11.temperature()),0,0) #0,16
8 oled096.text('Humid= {} %'.format(d11.humidity()),0,16) #0,32
9 oled096.show()
10 try:
11     while 1:
12         utime.sleep_ms(2000)
13         d11.measure()
14         oled096.fill(0)
15         #oled096.text('~ TEMP & HUMID ~',0,0)
16         oled096.text('Temp = {} C'.format(d11.temperature()),0,0) #0,16
17         oled096.text('Humid= {} %'.format(d11.humidity()),0,16) #0,32
18         oled096.show()
19         print('today temp ={}, humidity= {}'.format(d11.temperature(),d11.humidity()))
20 except Exception as e: print(e)
```

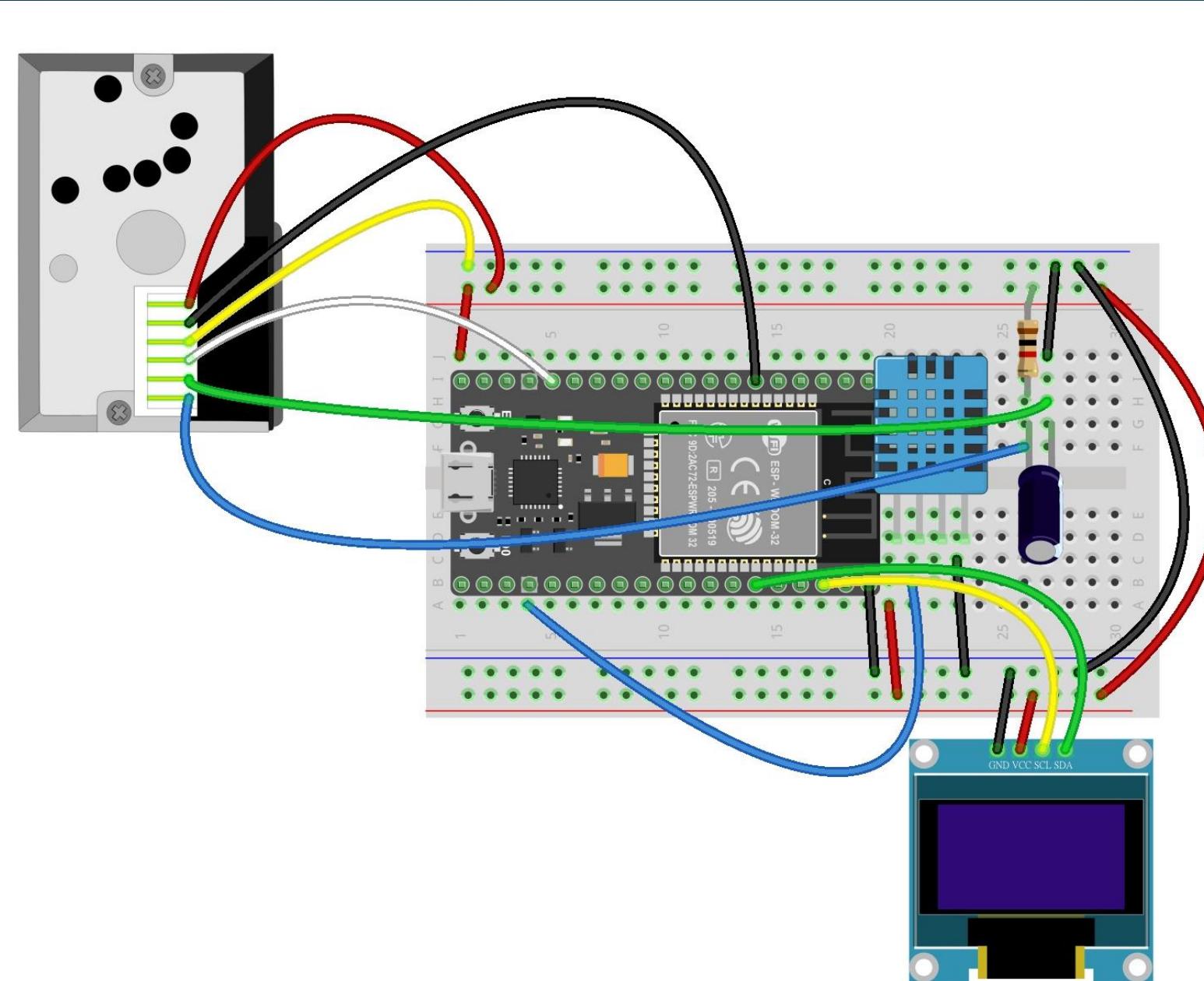
互動環境 (Shell)

```
Warning: I2C(-1, ...) is deprecated, use SoftI2C(...) instead
today temp =31, humidity= 71
today temp =31, humidity= 70
```

MicroPython (ESP32)







DHT11 PIN

- 1 VCC
2. Data (GPIO 15)
3. NC
4. GND

GP2Y1010AUOF (L--R)
(654321)

1. 5V (Red)
2. AO (GPIO35) (Black)
3. GND (yellow)
4. LED (GPIO13) (White)
5. GND (Green)
6. 5V (1K,10uF,5V) (Blue)

SSD1306

1. GND
2. 5V
3. SCL (GPIO22)
4. SDA (GPIO21)

```

# ADC : 34
# ILED : GPIO13
=====
from machine import Pin, ADC
import time
p0 = Pin(13, Pin.OUT)
adc = ADC(Pin(34))
def measure():
    p0.value(1)                      # d?but du cr?neau
    time.sleep_us(280)                # les 0.28 ms
    readvalue = adc.read()           # Lecture de l'adc
    time.sleep_us(40)                # compl?ment du cr?neau ? 0.32 ms
    p0.value(0)                      # fin du cr?neau
    time.sleep_us(9680)              # compl?ment du cycle ? 10 ms
    return readvalue
#-----
print('PM2.5 Result:')
t,tot,t1=0,0,0
ppm=0.0
t1=measure()
print('Test measure Value : ',t1,'---- this value always too low')
for i in range(10): # on fait 10 mesures
    t=measure()
    print(i,"---",t)
    tot=tot+t
ppm=tot/10
tot2='PM2.5 Value = '+str(ppm)[:5]
print(tot2)
print('Finish.....')

```

```

# ADC : 34
# ILED : GPIO13
=====
from machine import Pin, ADC
import time
p0 = Pin(13, Pin.OUT)          ✓✓
adc = ADC(Pin(34))            ✓✓
def measure():
    p0.value(1)                      # d?but du cr?neau
    time.sleep_us(280)                # les 0.28 ms
    readvalue = adc.read()           # Lecture de l'adc
    time.sleep_us(40)                # compl?ment du cr?neau ? 0.32 ms
    p0.value(0)                      # fin du cr?neau
    time.sleep_us(9680)              # compl?ment du cycle ? 10 ms
    return readvalue
#-----
print('PM2.5 Result:')
t,tot,t1=0,0,0
ppm=0.0
t1=measure()
print('Test measure Value : ',t1,'---- this value always too low')
for i in range(10): # on fait 10 mesures
    t=measure()          ✓
    print(i,"---",t)
    tot=tot+t
ppm=tot/10          ✓
tot2='PM2.5 Value = '+str(ppm)[:5]
print(tot2)
print('Finish.....')

```

Thonny - E:\MCU\micropython\csuee\airqoled.py @ 48:32

檔案 編輯 檢視 執行 工具 說明

airqoled.py

```
28 d11=dht.DHT11(machine.Pin(15))
29 hw_i2c1 = machine.I2C(scl=machine.Pin(22), sda=machine.Pin(21))# machine.I2C(0x3c, freq=2000)
30 oled096=ssd1306.SSD1306_I2C(128, 64, hw_i2c1)
31 oled096.fill(0)
32 oled096.text('~ TEMP & HUMID ~',0,0)
33 oled096.text('Temp =      C',0,16)
34 oled096.text('Humid=     %',0,32)
35 oled096.text('PM2.5=     um',0,48)
36 oled096.show()
37 try:
38     while 1:
39         utime.sleep_ms(2000)
40         d11.measure()
41         oled096.fill(0)
42         oled096.text('~ TEMP & HUMID ~',0,0)
43         oled096.text('Temp = {} C'.format(d11.temperature()),0,16)
44         oled096.text('Humid= {} %'.format(d11.humidity()),0,32)
45         oled096.text('PM2.5= {}um'.format(getDust()),0,48)
46         oled096.show()
47         print('today temp ={}, humidity= {}, PM2.5= {}'.format(d11.temperature(),d11.humidity(),getDust()))
48 except Exception as e: print(e)
```

互動環境 (Shell)

```
MicroPython v1.17 on 2021-09-02; ESP32 module with ESP32
Type "help()" for more information.
>>>
```

```
# config.py
SSID = "Your-SSID"          # WiFi名稱
PASSWORD = "Your-Password"    # WiFi密碼
KEY="API Key for openweathermap" #openweathermap
APIKEY = "IFTTT API Key" # iFTTT Key
```



檔案

本機

E:\MCU\micropython\csuee

airqoled.py

cnt.py

config.py

dht11try.py

dhtoled.py

DustValue.py

i2cScan.py

LabPractice.txt

mqttLass.py

mqttLassOled.py

mqttTemp.py

MicroPython 設備

boot.py

cnt.py

config.py

dht11try.py

dhtoled.py

i2cscan.py

ssd1306.py

```

1 import network
2 import config
3
4 def connect_wifi(ssid, passwd):
5     sta = network.WLAN(network.STA_IF)
6     sta.active(True)
7     if not sta.isconnected():
8         print("Connecting to network...")
9         sta.connect(ssid, passwd)
10    while not sta.isconnected():
11        pass
12    print("network config:", sta.ifconfig())
13
14 SSID = config.SSID      # WiFi名稱
15 PASSWORD = config.KEY   # WiFi密碼
16 connect_wifi(SSID, PASSWORD)

```

config.py

```

SSID = "Your-SSID"          # WiFi名稱
PASSWORD = "Your-Password"   # WiFi密碼
KEY="API Key for openweathermap" #openweathermap
APIKEY = "IFTTT API Key" # iFTTT Key

```

>>>

Thonny - MicroPython 設備 ::/cnt.py @ 11:15

檔案 編輯 檢視 執行 工具 說明

檔案 × [airoled.py] [cnt.py] [urllab.py] [mqttTemp.py] [mqttLass.py] [mqttLassOled.py]

本機 E:\MCU\micropython\csuee

- mqttTemp.py
- oledTest.py
- openWeatherMap.py
- photoresistor.py
- readADC.py
- ssd1306.py
- umqttsimple.py
- urlencode.py
- urllab.py**
- xrequests.py
- xtools.py

MicroPython 設備

- HCSR04Oled.py
- hx711.py
- hxtest.py
- i2cScan.py
- Ledryg.py
- mqttLass.py
- mqttLassOled.py**
- mqttpub.py
- mqttTemp.py
- oledTest.py
- openWeatherMap.py

```
1 import network
2 import config
3
4 def connect_wifi(ssid, passwd):
5     sta = network.WLAN(network.STA_IF)
6     sta.active(True)
7     if not sta.isconnected():
8         print("Connecting to network...")
9         sta.connect(ssid, passwd)
10        while not sta.isconnected():
11            pass
12        print("network config:", sta.ifconfig())
13
14 SSID = config.SSID          # WiFi名稱
15 PASSWORD = config.KEY      # WiFi密碼
16 connect_wifi(SSID, PASSWORD)
```

互動環境 (Shell) >>> %Run -c \$EDITOR_CONTENT

```
network config: ('192.168.1.117', '255.255.255.0', '192.168.1.1', '192.168.1.1')
```

>>>

OpenWeather

Weather forecasts, nowcasts and history in fast and elegant way

Leaving everything behind, people are fleeing conflict in Ukraine. They need shelter, food, and water. When you subscribe to our service, you can join us to help with donation of just of £20. Openweather will add £40 to each donation and send it to Disastrous Emergency Committee's (DEC) Ukrainian Humanitarian Appeal.

Search city

Search

Different Weather?

Metric: °C, m/s Imperial: °F, mph

May 28, 03:20pm

Chishan, TW 31°C

Feels like 30°C. Broken clouds. Gentle Breeze

4.1m/s WSW 1006hPa
Humidity: 72% UV: 6
Dew point: 25°C Visibility: 10.0km

Hourly forecast

now 4pm 5pm 6pm 7pm 8pm 9pm 10pm 11pm Ma

**8-day forecast**

Sat, May 28



31 / 26°C

light rain

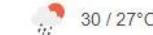
Sun, May 29



31 / 27°C

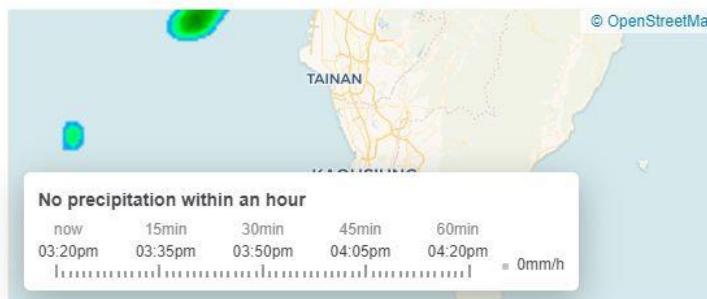
overcast clouds

Mon, May 30



30 / 27°C

light rain



Weather API

[Home](#) / Weather API

Please, [sign up](#) to use our fast and easy-to-work weather APIs. As a start to use OpenWeather products, we recommend our [One Call API 3.0](#). For more functionality, please consider our products, which are included in [professional collections](#).

One Call API 3.0 NEW

[API doc](#)[Subscribe](#)

Make one API call and receive all essential weather data in one response:

- Minute forecast for 1 hour
- Hourly forecast for 48 hours
- Daily forecast for 8 days
- Historical data for 40+ years back by timestamp
- National weather alerts

Read more about this API and subscription plan in the [FAQ](#).

Pay as you call

1,000 API calls per day for free
0.0012 GBP per API call over the daily limit

[Subscribe to One Call by Call](#)

This is a separate subscription plan, which include only One Call API.

Professional collections

For professionals and specialists with middle sized project, we recommend our Professional collections, which included [Current & Forecasts collection](#), [Historical weather data collection](#), [Weather Maps collection](#) and other APIs.

For Enterprise level projects we provide Enterprise license, which is included all forecast products and current state, along with alerts, maps, and other products. [Learn more](#)

You can read the [How to Start](#) guide and enjoy using our powerful weather APIs right now.

Current & Forecast weather data collection

Current Weather Data

[API doc](#)[Subscribe](#)

Hourly Forecast 4 days

[API doc](#)[Subscribe](#)

One Call API 1.0

[API doc](#)[Subscribe](#)

One Call API 3.0

[Home](#) / [API](#) / One Call API 3.0

Make just one API call and get all your essential weather data for a specific location with our new OpenWeather **One Call API 3.0**.

The One Call API provides the following weather data for any geographical coordinates:

- Current weather
- Minute forecast for 1 hour
- Hourly forecast for 48 hours
- Daily forecast for 8 days
- National weather alerts
- Historical weather data for 40+ years back (since January 1, 1979)

Current and forecast weather data

To get access to current weather, minute forecast for 1 hour, hourly forecast for 48 hours, daily forecast for 8 days and government weather alerts, please use this section of the documentation.

If you are interested in **historical weather data**, please read the "[Historical weather data](#)" section.

How to make an API call

API call

```
https://api.openweathermap.org/data/3.0/onecall?lat={lat}&lon={lon}&exclude={part}&appid={API key}
```



Parameters

`lat, lon` required Geographical coordinates (latitude, longitude)

`appid` required Your unique API key (you can always find it on your account page under the "[API key](#)" tab)

Current and forecasts weather data

[How to make an API call](#)

[Example of API response](#)

[Fields in API response](#)

Historical weather data

[How to make an API call](#)

[Example of API response](#)

[Fields in API response](#)

[List of weather condition codes](#)

Other features

[Units of measurement](#)

[Multilingual support](#)

[List of national weather alerts sources](#)

[Call back function for JavaScript code](#)



Weather in your city

Guide API Dashboard Marketplace Pricing Maps Our Initiatives Partners Blog For Business jumb... ▾ Support ▾

Notice

X

Signed in successfully.

New Products Services API keys Billing plans Payments Block logs My orders My profile Ask a question

You can generate as many API keys as needed for your subscription. We accumulate the total load from all of them.

Key	Name	Status	Actions	Create key
773d70[REDACTED]	jumbokh	Active	<input type="button"/> <input type="button"/> <input type="button"/>	<input type="text" value="API key name"/> <input type="button" value="Generate"/>
a9751c79[REDACTED]	csu	Active	<input type="button"/> <input type="button"/> <input type="button"/>	

Thonny - MicroPython 設備 ::/openWeatherMap.py @ 26:18

檔案 編輯 檢視 執行 工具 說明

檔案

本機 E:\MCU\micropython\csuee

- mqttTemp.py
- oledTest.py
- openWeatherMap.py
- photoresistor.py
- readADC.py
- ssd1306.py
- umqttsimple.py
- urlencode.py
- urllib.py
- xrequests.py
- xtools.py

MicroPython 設備

- 2.timeMVDetect.py
- 3.IrdaLed.py
- airoled.py
- am2320.py
- boot.py
- cnt.py
- config.py
- dht11try.py
- dhtoled.py
- DustValue.py
- escale.py

airoled.py [cnt.py] [openWeatherMap.py] [config.py]

```
5    ledR = Pin(17, Pin.OUT)
6    ledG = Pin(12, Pin.OUT)
7    ledB = Pin(13, Pin.OUT)
8
9    def connect_wifi(ssid, passwd):
10        sta = network.WLAN(network.STA_IF)
11        sta.active(True)
12        if not sta.isconnected():
13            print("Connecting to network...")
14            sta.connect(ssid, passwd)
15        while not sta.isconnected():
16            pass
17        print("network config:", sta.ifconfig())
18
```

互動環境 (Shell)

```
>>> %Run -c $EDITOR_CONTENT
network config: ('192.168.1.117', '255.255.255.0', '192.168.1.1', '192.168.1.1')
-----
天氣描述: 多雲
-----
目前溫度: 31.06
大氣壓力: 1006
目前溼度: 71
最低溫度: 30.97
最高溫度: 31.13
-----
```

RED

Thonny - MicroPython 設備 ::/urllib.py @ 14:35

檔案 編輯 檢視 執行 工具 說明

檔案 × airoled.py [cnt.py] [urllib.py]

本機 E:\MCU\micropython\csuee

- mqttTemp.py
- oledTest.py
- openWeatherMap.py
- photoresistor.py
- readADC.py
- ssd1306.py
- umqttsimple.py
- urlencode.py
- urllib.py
- xrequests.py
- xtools.py

MicroPython 設備

- readDust.py
- scale.py
- ssd1306.py
- tryAm2320.py
- trydht.py
- TryHCSR04.py
- umqttsimple.py
- urlencode.py
- urllib.py
- weatherrpt.py
- xrequests.py

```
1 import network
2 import config
3
4 def connect_wifi(ssid, passwd):
5     sta = network.WLAN(network.STA_IF)
6     sta.active(True)
7     if not sta.isconnected():
8         print("Connecting to network...")
9         sta.connect(ssid, passwd)
10        while not sta.isconnected():
11            pass
```

互動環境 (Shell) : RED

```
>>> %Run -c $EDITOR_CONTENT
network config: ('192.168.1.117', '255.255.255.0', '192.168.1.1', '192.168.1.1')
ASP.NET 網頁程式設計
陳會安
W101
PHP 網頁程式設計
陳會安
W102
Java 程式設計
陳會安
P102
Android 程式設計
陳會安
M102
```

>>>



檔案 x

本機 E:\MCU\micropython\csuee

- mqttTemp.py
- oledTest.py
- openWeatherMap.py
- photoresistor.py
- readADC.py
- ssd1306.py
- umqttSimple.py
- urlencode.py
- urllib.py
- xrequests.py
- xtools.py

MicroPython 設備

- i2cScan.py
- Ledryg.py
- mqttLass.py
- mqttLassOled.py
- mqttpub.py
- mqttTemp.py
- oledTest.py
- openWeatherMap.py
- photoresistor.py
- RCtest.py
- readADC.py

airqoled.py [cnt.py] [urllib.py] [mqttTemp.py]

```
1 from umqtt.simple import MQTTClient
2 import machine,dht,utime,time,network
3 import config
4
5 mq_server = 'broker.emqx.io'
6 mq_id = 'esp00001'
7 mq_topicT = b'csuclass/lass/T'
8 mq_topicH = b'csuclass/lass/H'
9 mq_topicPM25 = b'csuclass/lass/PM25'
10 mq_user=''
11 mq_pass=''
12 SSID = config.SSID          # WiFi名稱
13 PASSWORD = config.PASSWORD  # WiFi密碼
14 d11=dht.DHT11(machine.Pin(15))      # GPIO15 as the DHT11 dataline
15 wifi= network.WLAN(network.STA_IF)
16 wifi.active(True)
17
18 try:
19     wifi.connect(SSID, PASSWORD)
```

互動環境 (Shell)

>>>

```
34 while True:  
35     d11.measure()  
36     #mq_message='T={},H={}'.format(d11.temperature(),d11.humidity())  
37     m1 = d11.temperature()  
38     m2 = d11.humidity()  
39     mqClient0.publish(mq_topicT, str(m1))  
40     mqClient0.publish(mq_topicH, str(m2))  
41     time.sleep(5)  
42     i=i+1  
43     print("T={}, H={} --> {}".format(m1,m2,i))  
44     if i>20: break  
45 except Exception as e:  
46     print('mqtt exception error: ',e)
```

Thonny - MicroPython 設備 ::/mqttTemp.py @ 5 : 29

檔案 編輯 檢視 執行 工具 說明

檔案 本機 E:\MCU\micropython\csuee

- mqttTemp.py
- oledTest.py
- openWeatherMap.py
- photoresistor.py
- readADC.py
- ssd1306.py
- umqttsimple.py
- urlencode.py
- urllib.py
- xrequests.py
- xtools.py

MicroPython 設備

- i2cScan.py
- Ledryg.py
- mqttLass.py
- mqttLassOled.py
- mqttpub.py
- mqttTemp.py
- oledTest.py
- openWeatherMap.py
- photoresistor.py
- RCtest.py
- readADC.py

airoled.py [cnt.py] [urllib.py] [mqttTemp.py]

```
28     print('Network Config=', wifi.ifconfig())
29 else:
30     print('WiFi connection Error')
31 mqClient0 = MQTTClient(mq_id, mq_server, user=mq_user, password=mq_pass)
32 mqClient0.connect()
33 i=0
34 while True:
35     d11.measure()
36     #mq_message='T={},H={}'.format(d11.temperature(),d11.humidity())
37     m1 = d11.temperature()
38     m2 = d11.humidity()
39     mqClient0.publish(mq_topicT, str(m1))
40     mqClient0.publish(mq_topicH, str(m2))
41     time.sleep(5)
```

互動環境 (Shell)

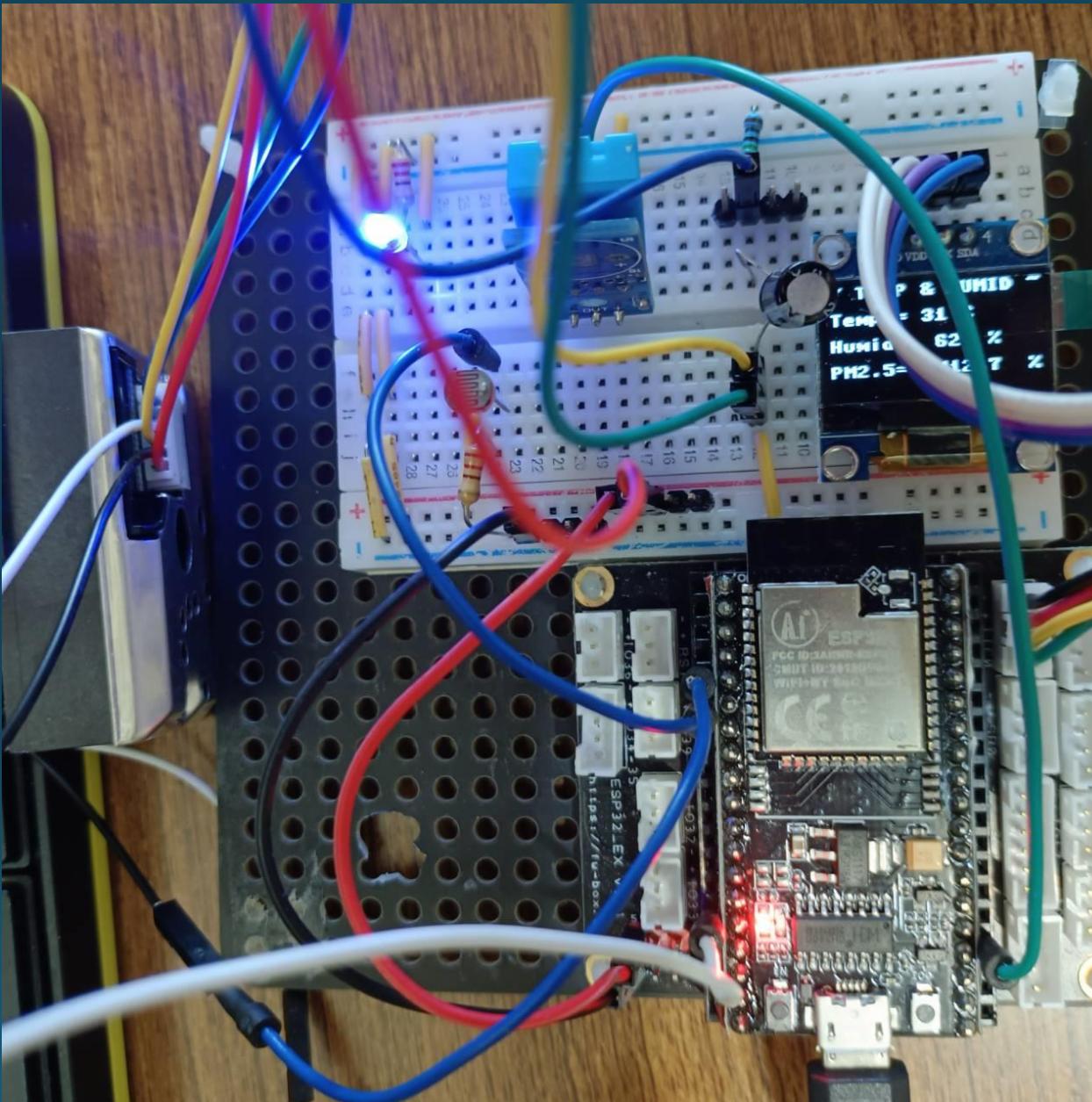
```
>>> %Run -c $EDITOR_CONTENT
start to connect wifi
try to connect wifi in 0s
WiFi connection OK!
Network Config= ('192.168.1.117', '255.255.255.0', '192.168.1.1', '192.168.1.1')
T=32, H=61 --> 1
T=31, H=61 --> 2
T=31, H=61 --> 3
```

The screenshot shows the Thonny IDE interface with the following details:

- Top Bar:** Thonny - MicroPython 設備 ::/mqttLass.py @ 72:1
- File Menu:** 檔案, 編輯, 檢視, 執行, 工具, 說明
- Toolbar:** Standard file operations (New, Open, Save, etc.)
- Left Sidebar:** Shows the project structure:
 - 本機: E:\MCU\micropython\csuee
 - MicroPython 設備: Ledryg.py, mqttLass.py, mqttLassOled.py, mqttpub.py, mqttTemp.py, oledTest.py, openWeatherMap.py, photoresistor.py, RCtest.py, readADC.py, readcsv.py
- Central Editor:** The code editor displays the `mqttLass.py` script:

```
1 from umqtt.simple import MQTTClient
2 from machine import Pin, ADC
3 import machine,dht,utime,time,network
4 import config
5
6 mq_server = 'broker.emqx.io'
7 #mq_server = '192.168.1.103'
8 mq_id = 'esp0001'
9 mq_topicT = b'csuclass/lass/T'
10 mq_topicH = b'csuclass/lass/H'
11 mq_topicPM25 = b'csuclass/lass/PM25'
12 mq_user=''
13 mq_pass=''
14 SSID = config.SSID          # WiFi名稱
15 PASSWORD = config.PASSWORD  # WiFi密碼
16 d11=dht.DHT11(machine.Pin(15))      # GPIO14 as the DHT11 dataline
17 p0 = Pin(13, Pin.OUT)
18 adc = ADC(Pin(35))
19 def measure():
20     return d11.read()           # Read the DHT11 sensor
```
- Bottom Shell:** Shows the output of the script execution:

```
WiFi connection OK!
Network Config= ('192.168.1.117', '255.255.255.0', '192.168.1.1', '192.168.1.1')
T=31, H=60, PM25=1066.7--> 1
T=31, H=61, PM25=1072.7--> 2
T=31, H=61, PM25=1029.4--> 3
```



敬請指導



THANKYOUFORYOURADVISE!