# 第10章 訊息通知：IFTTT寄送電郵+LINE Notify

- 10-1 MicroPython檔案系統
- 10-2 上傳和使用本書提供的工具箱模組
- 10-3 申請與使用IFTTT寄送電子郵件
- 10-4 申請與使用LINE Notify
- 10-5 整合應用：使用LINE Notify送出天氣通知

# 10-1 MicroPython檔案系統

- 10-1-1 MicroPython檔案管理工具
- 10-1-2 uos模組的檔案管理

# 10-1-1 MicroPython檔案管理工具

- 當ESP8266開發板擁有超過1M或更多快閃記憶體的儲存空間時，在燒錄MicroPython韌體後，第一次啟動就會自動建立FAT格式的檔案系統（Filesystem）。
- 雖然我們可以使用MicroPython程式碼來管理開發板上的檔案系統，不過，使用MicroPython檔案管理工具是更好的選擇，例如：ampy命令列工具、AmpyGUI或Thonny內建的檔案管理工具等。

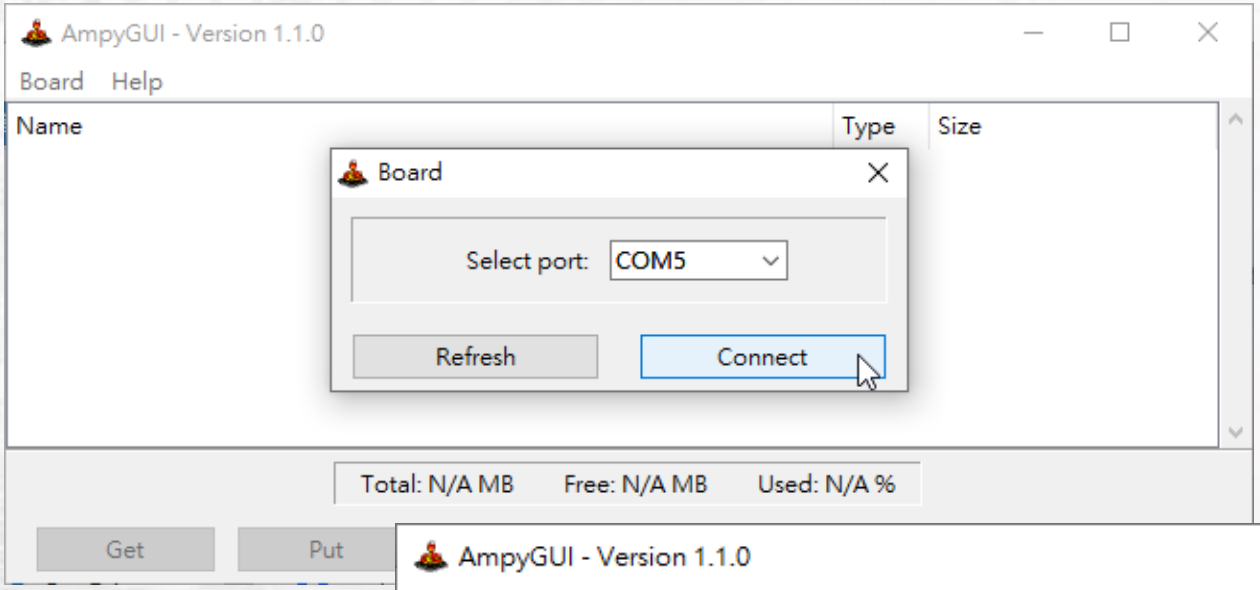
# 10-1-1 MicroPython檔案管理工具 AmpyGUI檔案管理工具

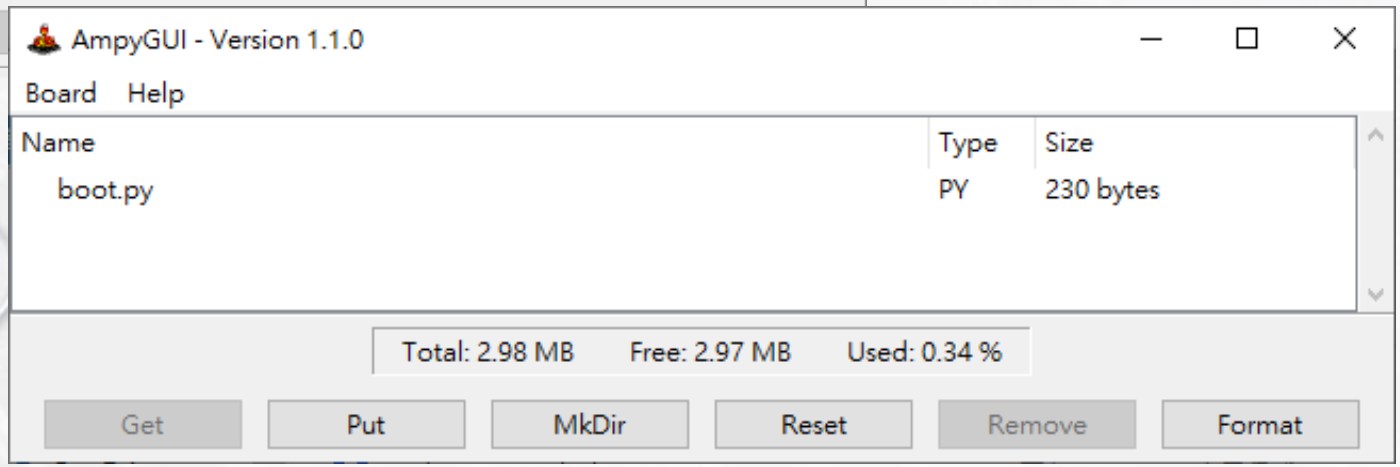
- AmpyGUI是ampy工具的圖形化使用介面，ampy是一個MicroPython命令列工具（https://github.com/scientifichackers/ampy），可以使用序列埠連接MicroPython開發板來管理檔案系統的檔案和執行MicroPython程式。AmpyGUI的Github官方網址如下所示：

  https://github.com/FlorianPoot/AmpyGUI
- Windows版AmpyGUI檔案管理工具的下載網址，如下所示：

  https://github.com/FlorianPoot/AmpyGUI/releases

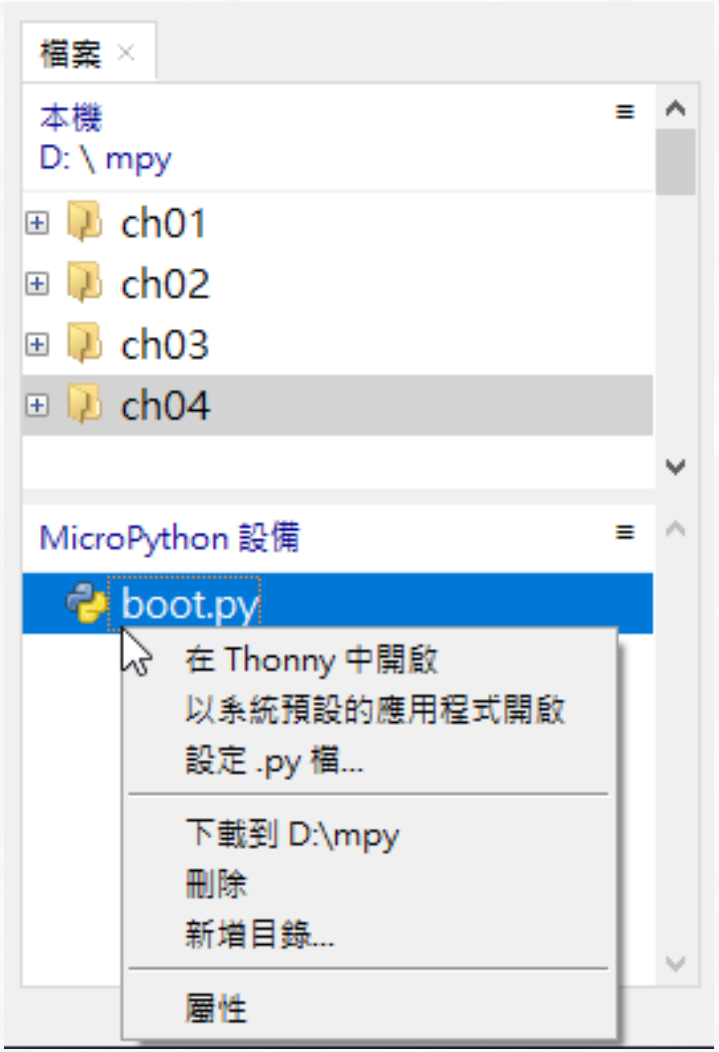
# 10-1-1 MicroPython檔案管理工具 AmpyGUI檔案管理工具



# 10-1-1 MicroPython檔案管理工具 Thonny檔案管理

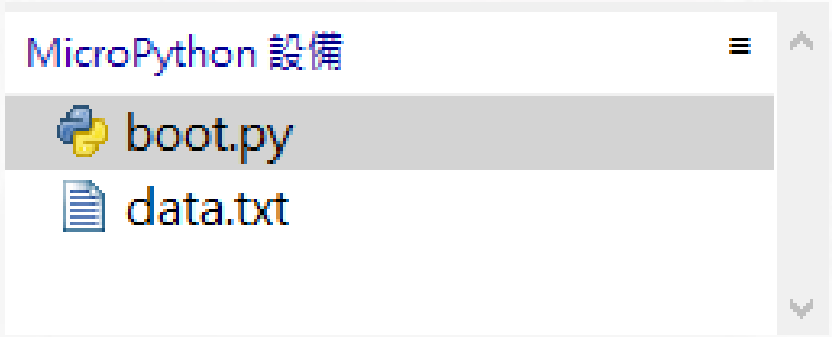
- Thonny內建檔案管理來管理Windows電腦和MicroPython設備的檔案與目錄，請啟動Thonny連接ESP8266開發板，執行「檢視>檔案」命令開啟Thonny檔案管理，可以在最左邊看到【檔案】標籤，如右圖所示：



# 10-1-2 uos模組的檔案管理
# 寫入文字檔案：ch10-1-2.py

- MicroPython可以使用open()函式開啟寫入的文字檔案，然後呼叫write()方法寫入字串，如下所示：.

```
f = open("data.txt", "w")
f.write("陳會安")
f.close()
```



# 10-1-2 uos模組的檔案管理
# 讀取文字檔案：ch10-1-2a.py

- MicroPython程式在開啟讀取的文字檔案後，就可以使用read()方法來讀取檔案內容，如下所示：

```
f = open("data.txt", "r")
contents = f.read()
print(contents)
f.close()
```

# 10-1-2 uos模組的檔案管理
# 顯示檔案清單：ch10-1-2b.py

- MicroPython實作os模組子集的uos模組來管理檔案系統，首先需要匯入uos模組，如下所示：

```
import uos

print(uos.listdir())
```

```
['boot.py', 'data.txt']
```

# 10-1-2 uos模組的檔案管理
# 新增目錄和切換路徑：ch10-1-2c.py

- MicroPython可以使用uos模組的mkdir()方法新增目錄；chdir()方法切換目錄，如下所示：

```
import uos

uos.mkdir("test")
print(uos.listdir())
uos.chdir("test")
print(uos.listdir())
uos.chdir("..")
print(uos.listdir())
```

```
['boot.py', 'data.txt', 'test']
[]
['boot.py', 'data.txt', 'test']
```

# 10-1-2 uos模組的檔案管理
# 刪除目錄或檔案：ch10-1-2d.py

- MicroPython可以使用uos模組的rmdir()方法刪除目錄；remove()方法刪除檔案，如下所示：

```
import uos

uos.rmdir("test")
print(uos.listdir())
uos.remove("data.txt")
print(uos.listdir())
```

```
['boot.py', 'data.txt']
['boot.py']
```

# 10-2 上傳和使用本書提供的工具箱模組

- 10-2-1 上傳本書提供的工具箱模組
- 10-2-2 使用本書提供的工具箱模組

# 10-2-1 上傳本書提供的工具箱模組

- 在開發MicroPython專案時，有一些函式我們需要常常使用，所以筆者已經收集常用函式成為xtools.py自訂模組，xrequests.py模組是改寫urequests模組新增params參數和執行data參數的URL編碼，urlencode.py是從micropython-lib抽出URL編碼的urlencode()函式。
- 在本書使用的工具箱模組共有4個MicroPython程式：xtools.py、config.py、xrequests.py和urlencode.py。

# 10-2-1 上傳本書提供的工具箱模組
# 本書工具箱模組提供的函式說明

| 函式 | 說明 |
| --- | --- |
| get_id() | 取得開發板MAC，MAC是網路卡獨一無二的地址碼，這是六組00~FF代碼，在生產時就燒入EEPROM，在本書是作為第12章MQTT客戶端的client id |
| get_num(x) | 取得參數字串之中的數字 |
| random_in_range(low,high) | 取得參數指定範圍的整數亂數 |
| map_range(x,in_min,in_max, out_min,out_max) | 將數值從一個範圍對應轉換至另一個範圍，類似Arduino的map()函式 |
| connect_wifi_led(ssid, passwd,timeout) | 提供預設超時15秒來連線WiFi，參數是SSID和密碼，沒有參數是使用config.py常數，成功連線內建LED燈會亮起，和回傳IP位址 |
| show_error() | 閃爍內建LED來顯示有錯誤發生 |
| webhook_post(url, value) | 使用xrequests.py送出HTTP POST請求，請求失敗呼叫show_error()函式閃爍內建LED |
| webhook_get(url) | 使用urequests送出HTTP GET請求，請求失敗呼叫show_error()函式閃爍內建LED |
| line_msg(token, message) | 使用xrequests.py的post()方法來送出LINE Notify訊息，詳見第10-4節的說明 |
| format_datetime(localtime) | 將參數localtime元組轉換成日期/時間字串 |

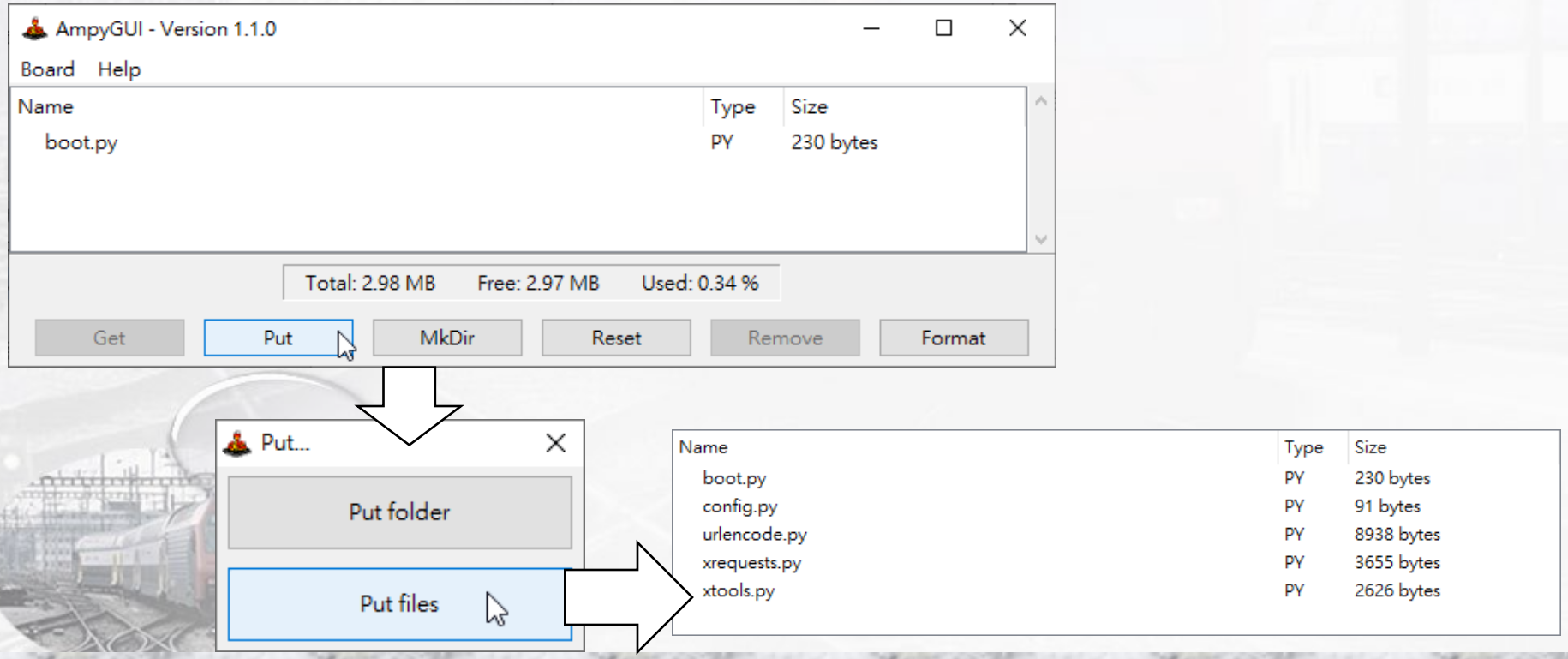
# 10-2-1 上傳本書提供的工具箱模組
# 本書工具箱模組提供的函式說明

- 在urlencode.py自訂模組提供的函式說明，如下表所示：

| 函式 | 說明 |
| --- | --- |
| urlencode() | 將參數的字典URL編碼成URL參數 |

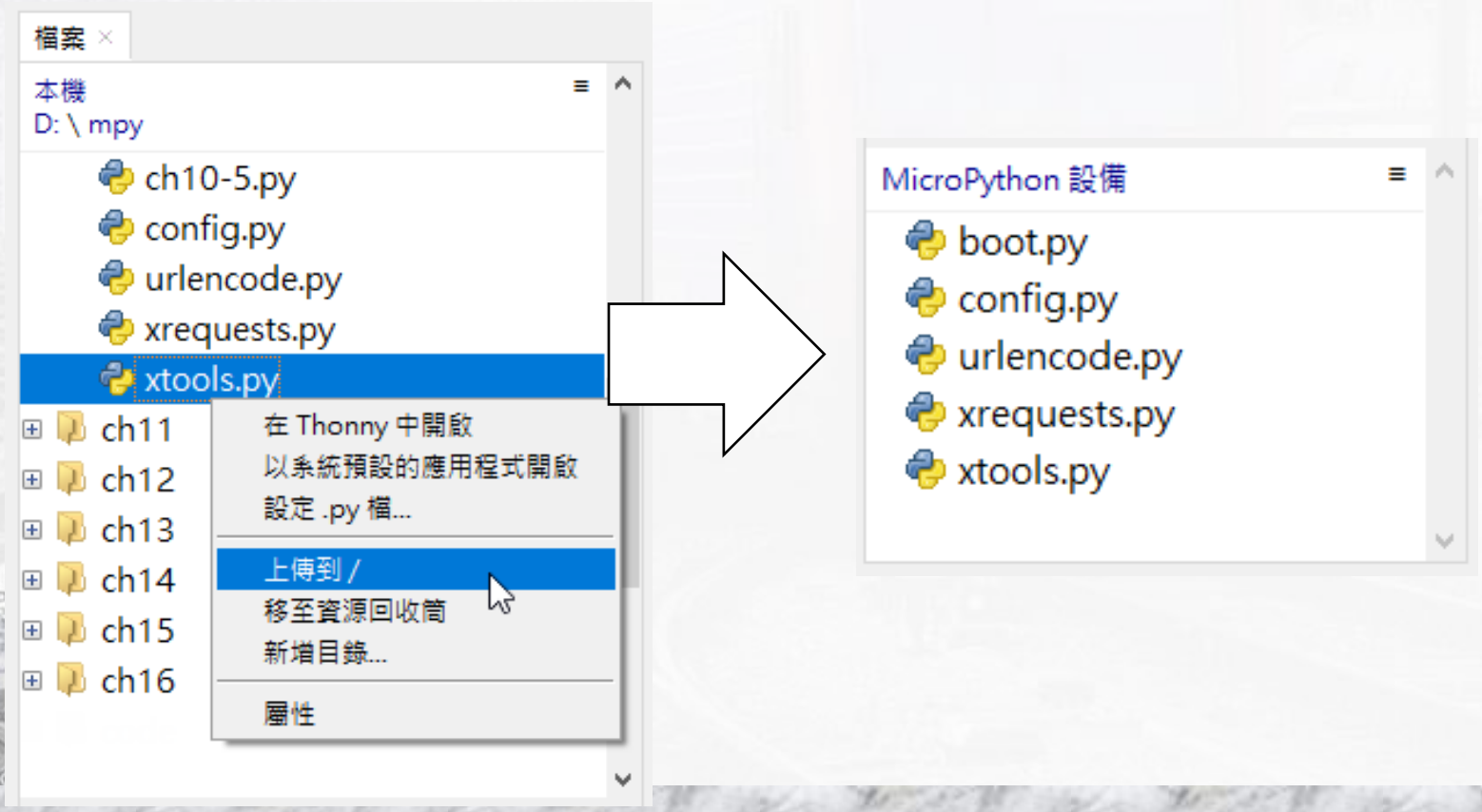
# 10-2-1 上傳本書提供的工具箱模組 使用AmpyGUI檔案管理工具上傳模組檔案

- Witty Cloud機智雲和Wemos D1 Min開發板可以使用AmpyGUI檔案管理工具一次就上傳4個MicroPython檔案（NodeMCU請使用Thonny檔案管理），如下圖所示：



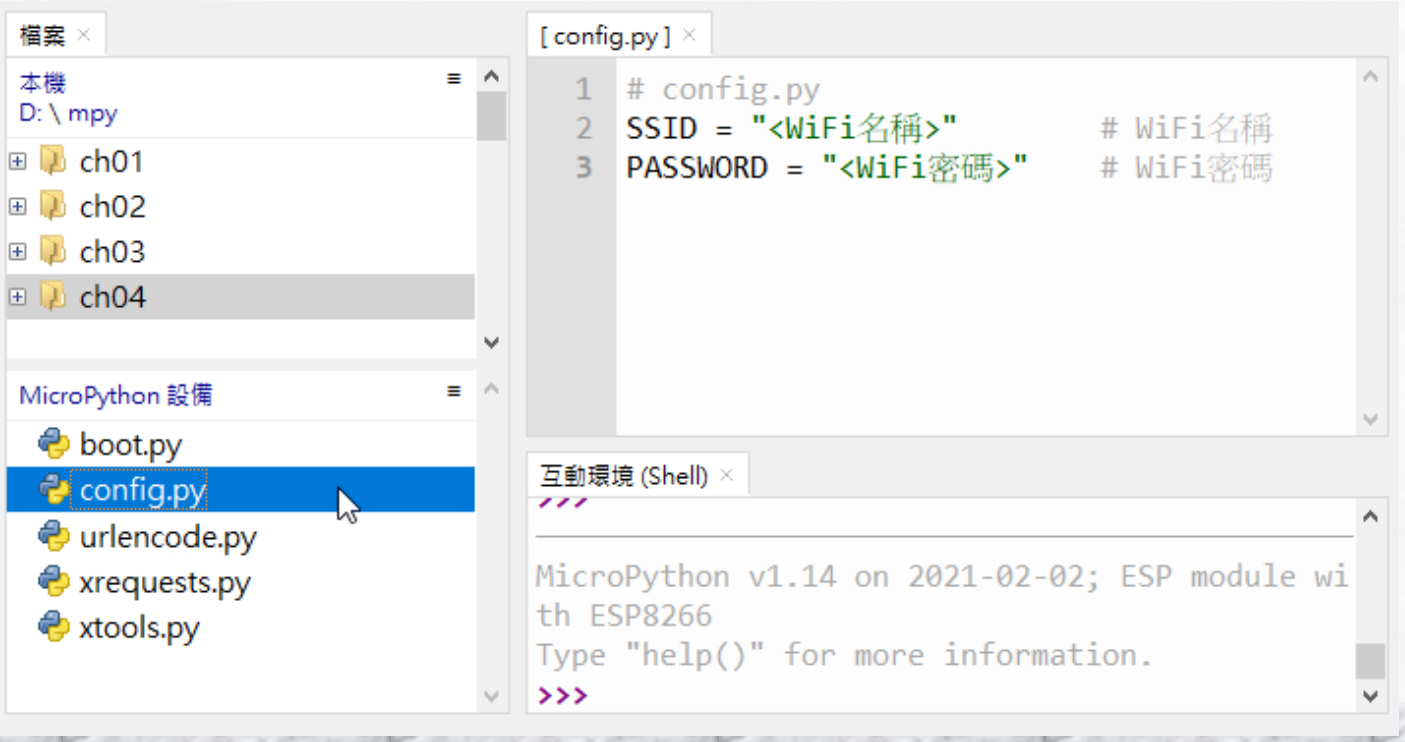
# 10-2-1 上傳本書提供的工具箱模組
# 使用Thonny檔案管理上傳模組檔案

- ESP8266開發板都可以使用Thonny檔案管理來一一上傳4個MicroPython檔案，如下圖所示：



# 10-2-2 使用本書提供的工具箱模組

- 在MicroPython程式使用xtools.py自訂模組前需要先設定config.py檔案的SSID和PASSWORD密碼，請啟動Thonny開啟【檔案】標籤，雙擊MicroPython設備的config.py檔案開啟此檔案，如下圖所示：



# 10-2-2 使用本書提供的工具箱模組
# 使用xtools模組來連線WiFi：ch10-2-2.py

- 現在，MicroPython程式可以使xtools模組的connect_wifi_led()函式來連線WiFi，如下所示：

```
import xtools

ip = xtools.connect_wifi_led()
print(ip)
```

```
#15 ets_task(4020ee68, 28, 3fff9e98, 10)
Connecting to network...
network config: ('192.168.1.108', '255.255.255.0',
'192.168.1.1', '192.168.1.1')
192.168.1.108
```

# 10-2-2 使用本書提供的工具箱模組 Google圖書查詢的Web API：ch10-2-2a.py

- 我們準備使用xtools模組修改第9-5節的Google圖書查詢的Web API，改用xtools模組的xtools.connect_wifi_led()函式來連線WiFi，如下所示：

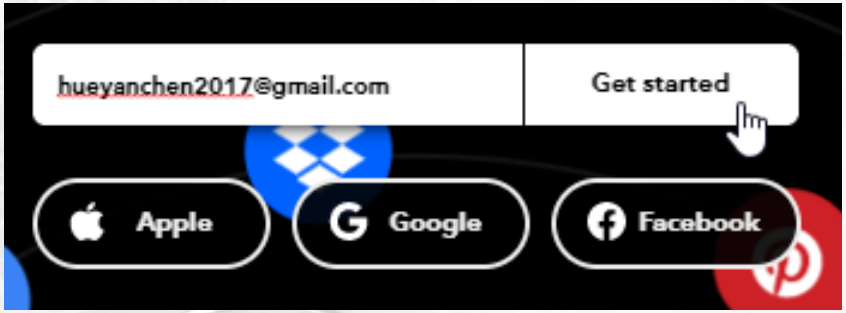
```
import urequests, ujson
import xtools

ip = xtools.connect_wifi_led()
print(ip)
…
```

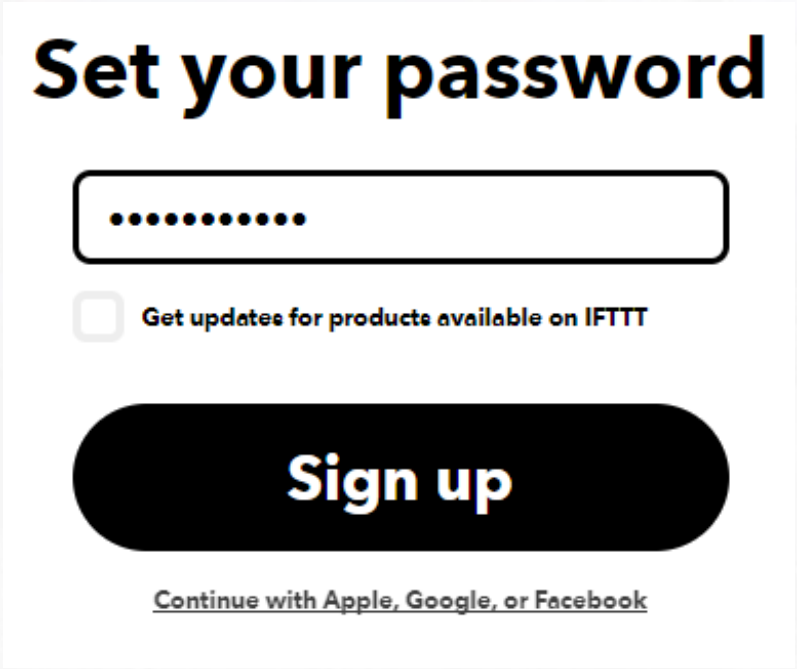
# 10-3 申請與使用IFTTT寄送電子郵件

- 10-3-1 註冊IFTTT服務
- 10-3-2 在IFTTT建立Applets
- 10-3-3 建立MicroPython程式使用IFTTT服務

# 10-3-1 註冊IFTTT服務

- IFTTT個人使用的標準版可以免費建立最多3個Applets，其官方網址是：https://ifttt.com/。我們可以使用Google或Facebook帳號登入IFTTT，也可以使用電子郵件地址註冊IFFFF服務，如下圖所示：

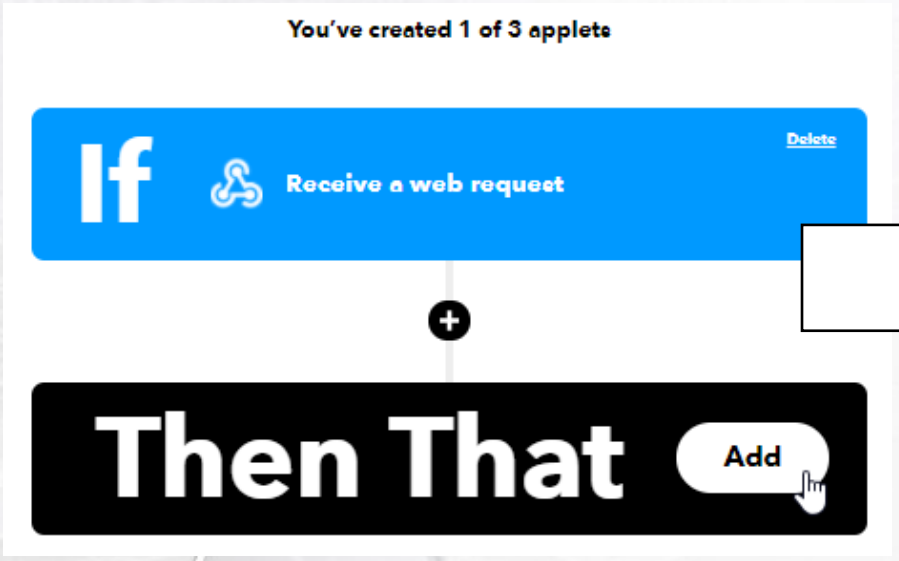


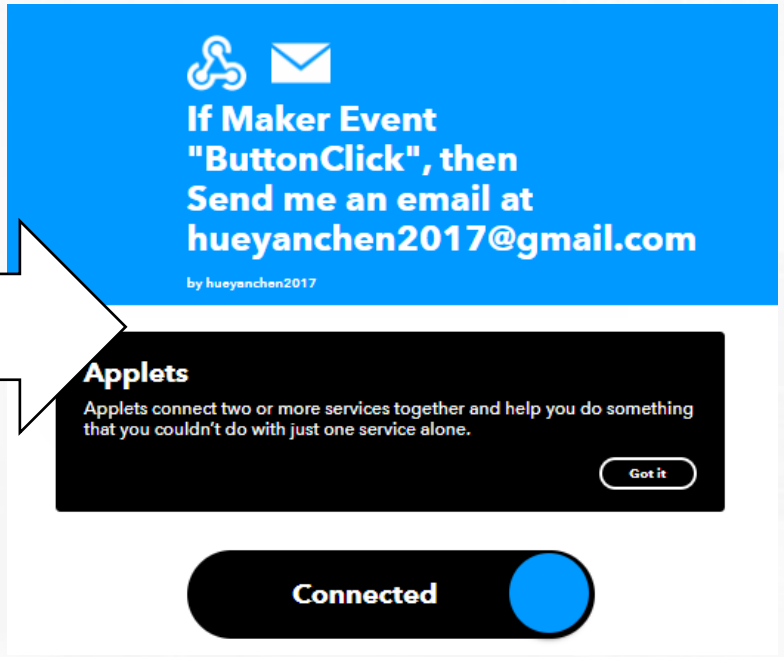



# 10-3-2 在IFTTT建立Applets

- IFTTT的Applets是一個小程序，可以建立條件來連接不同App、Web服務和裝置，以便當指定事件觸發時（條件成立），可以自動執行其他App、服務或操作裝置（動作）。例如：當HTTP請求的事件觸發（條件成立），就自動寄送電子郵件到自己的信箱（動作）。
  - 新增HTTP請求觸發寄送郵件的Applet
  - 在Webhooks服務取得API金鑰字串和測試服務
  - Webhooks服務觸發事件的URL網址

# 10-3-2 在IFTTT建立Applets
# 新增HTTP請求觸發寄送郵件的Applet



# 10-3-2 在IFTTT建立Applets
## 在Webhooks服務取得API金鑰字串和測試服務

Your key is: [redacted]

Back to service

### To trigger an Event

Make a POST or GET web request to:

`https://maker.ifttt.com/trigger/{event}/with/key/[redacted]`

With an optional JSON body of:

```
{ "value1" : "", "value2" : "", "value3" : "" }
```

The data is completely optional, and you can also pass `value1`, `value2`, and `value3` as query parameters or form variables. This content will be passed on to the action in your Applet.

You can also try it with `curl` from a command line.

```
curl -X POST https://maker.ifttt.com/trigger/{event}/with/key/[redacted]
```

# 10-3-2 在IFTTT建立Applets Webhooks服務觸發事件的URL網址

- IFTTT的Webhooks服務觸發事件的URL網址語法，如下所示：

  https://maker.ifttt.com/trigger/<Event_Name>/with/key/<API_KEY>/?value1=<V1>&value2=<V2>&value3=<V3>

- 上述<Event_Name>是事件名稱，<API_KEY>是API金鑰字串，<V1~3>是在電子郵件內容顯示的value1~3值。

# 10-3-3 建立MicroPython程式使用IFTTT服務中文參數值的URL編碼處理：ch10-3-3.py

- 在IFTTT服務的Webhook可以有3個參數value1~3，如果參數值是中文，就需先執行URL編碼處理，如下所示：

  https://maker.ifttt.com/trigger/buttonclick/with/key/<API_KEY>/?value1=100&value2=陳會安

- 上述value2參數值是中文，所以需要URL編碼處理，如下所示：

  https://maker.ifttt.com/trigger/buttonclick/with/key/<API_KEY>/?value1=100&value2=%E9%99%B3%E6%9C%83%E5%AE%89

# 10-3-3 建立MicroPython程式使用IFTTT服務中文參數值的URL編碼處理：ch10-3-3.py

- 在urlencode.py提供urlencode()函式，可以將字典資料URL編碼成參數字串，首先匯入urlencode()函式，如下所示：

```
from urlencode import urlencode

value1 = 100
value2 = "陳會安"
params = { "value1": value1,
           "value2": value2 }
print(urlencode(params))
```

```
value2=%E9%99%B3%E6%9C%83%E5%AE%89&value1=100
```

# 10-3-3 建立MicroPython程式使用IFTTT服務

使用urequests.get()方法觸發IFTTT服務：ch10-3-3a.py

- IFTTT的Webhooks服務可以使用urequests模組的get()方法送出HTTP請求來觸發事件，如下所示：

```
from machine import Pin
import utime, urequests
from urlencode import urlencode
import xtools

xtools.connect_wifi_led()
button = Pin(4, Pin.IN, Pin.PULL_UP)
```

# 10-3-3 建立MicroPython程式使用IFTTT服務

使用urequests.get()方法觸發IFTTT服務：ch10-3-3a.py

```
APIKEY = "<API金鑰>"
value1 = 100
value2 = "陳會安"
params = { "value1": value1,
           "value2": value2 }
WEBHOOK_URL="https://maker.ifttt.com/trigger/ButtonClick/with/key/"
WEBHOOK_URL+=APIKEY + "/?" + urlencode(params)
print("請按下按鍵開關來送出Email…")
```

# 10-3-3 建立MicroPython程式使用IFTTT服務

使用urequests.get()方法觸發IFTTT服務：ch10-3-3a.py

```
while True:
    if button.value() == 0:   # 值 0 是按下
        print("送出Email!")
        urequests.get(WEBHOOK_URL)
        utime.sleep(10)
```

# 10-4 申請與使用LINE Notify

- 10-4-1 申請LINE Notify存取權杖
- 10-4-2 建立MicroPython程式發送LINE Notify通知

# 10-4-1 申請LINE Notify存取權杖

- LINE提供LINE Notify官方帳號，我們只需申請存取權杖，就可以透過LINE Notify傳送通知訊息至目標使用者的LINE App。
- 在建立MicroPython程式使用LINE Notify傳送通知訊息給使用者前，我們需要申請LINE Notify存取權杖，其步驟如下所示：
  - 步驟一：登入LINE Notify服務
  - 步驟二：取得LINE Notify存取權杖（Token）
  - 步驟三：在LINE App加入LINE Notify成為好友

# 10-4-2 建立MicroPython程式發送LINE Notify通知：ch10-4-2.py

- 在成功取得LINE Notify存取權杖後，我們準備建立MicroPython程式來發送LINE Notify通知訊息。
- 因為urequests模組的post()方法沒有實作params參數，並無法使用此方法發送LINE Notify通知訊息，所以筆者改寫urequests模組成xrequests.py，新增params參數和加上URL編碼，可以直接使用post()方法來發送LINE Notify通知訊息。

# 10-4-2 建立MicroPython程式發送LINE Notify通知：ch10-4-2.py

- 在xtools.py自訂模組提供line_msg()函式寄送LINE Notify（這是使用xrequests模組的post()方法），如下所示：

```
import xtools

xtools.connect_wifi_led()

token = "<存取權杖>"
message = "使用MicroPython送出Notify通知訊息"
xtools.line_msg(token, message)
```

# 10-5 整合應用：使用LINE Notify送出天氣通知

- 我們只需整合第9-6節的OpenWeatherMap天氣資訊、第10-4節的LINE Notify通知訊息和第10-3-3節的按鍵開關功能，只需按下按鍵開關，即可發送天氣描述的LINE Notify通知。
- MicroPython程式：ch10-5.py的執行結果可以在LINE App的LINE Notify看到發送的天氣描述通知訊息，如下圖所示：