

程式
課程

Python × 物聯網 × 人工智慧

Chapter 1：感測器與 Python 簡介

賴秉樑 debugger

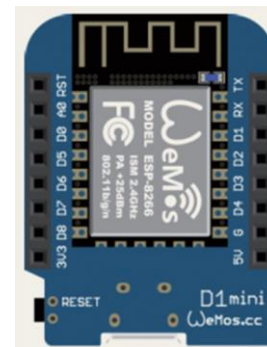
本章重點

感測器與 D1 mini 的基本介紹。

準備材料



可上網的電腦



D1 mini (esp8266)

學習目標

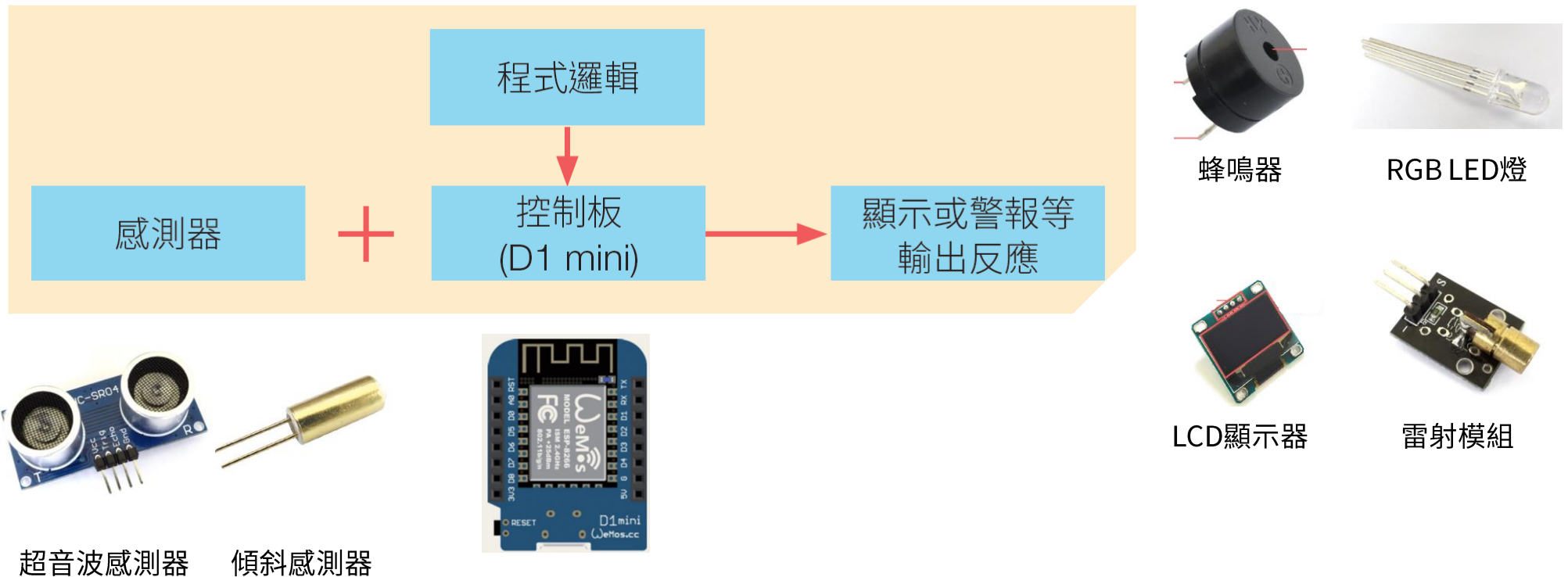
1. 介紹感測器與 D1 mini。
2. 控制 D1 mini 內建的 LED。

Outline

- 1-1. 感測器簡介：控制板 x 感測器 x 動作器
- 1-2. D1 mini 控制板簡介
- 1-3. 安裝 Python 開發環境
- 1-4. Python 物件、資料型別、變數、匯入模組
- 1-5. 安裝與設定 D1 mini
- 1-6. 認識硬體
- 1-7. D1 mini 的 IO 腳位以及數位訊號輸出
- 1-8. 流程控制 (while 迴圈) 與區塊縮排
- 補給站：安裝 MicroPython 到 D1 mini

1-1. 感測器簡介：控制板 x 感測器 x 動作器

- 感測器 (sensors) 可偵測外在環境變化，日常生活中隨處可見。例如：倒車雷達、計步器、防盜系統等。我們將使用以下架構實作出上述幾種應用：



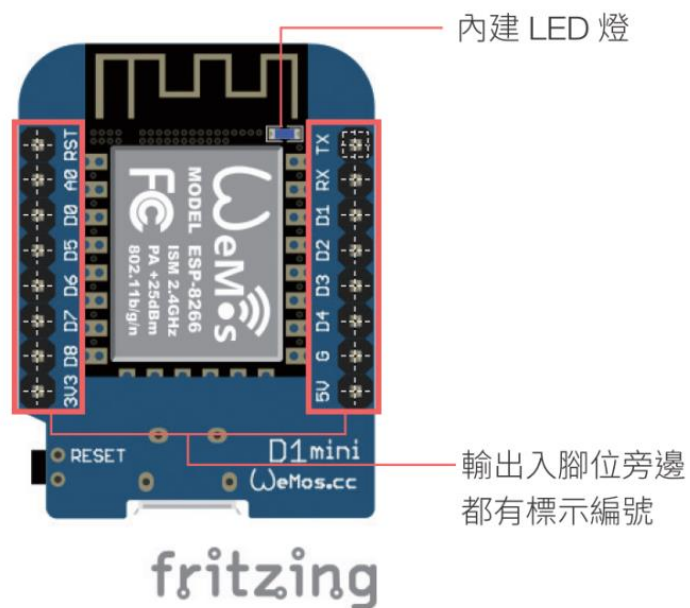
- 控制板可說是一個智慧中心，取得感測數值後送上網路。這個智慧中心一般使用單晶片開發板來達成。

Outline

- 1-1. 感測器簡介：控制板 x 感測器 x 動作器
- 1-2. D1 mini 控制板簡介
- 1-3. 安裝 Python 開發環境
- 1-4. Python 物件、資料型別、變數、匯入模組
- 1-5. 安裝與設定 D1 mini
- 1-6. 認識硬體
- 1-7. D1 mini 的 IO 腳位以及數位訊號輸出
- 1-8. 流程控制 (while 迴圈) 與區塊縮排
- 補給站：安裝 MicroPython 到 D1 mini

1-2. D1 mini 控制板簡介

- D1 mini 是一片單晶片開發板，能執行透過程式描述的運作流程，並且藉由兩側的輸出/輸入腳位控制外部的電子元件，或從外部電子元件獲取資訊。
- D1 mini 還具備 Wi-Fi 連網能力，可將電子元件資訊傳送出去，也可透過網路從遠端控制 D1 mini。
- D1 mini 可透過易學易用的 Python，這是目前當紅的程式語言！

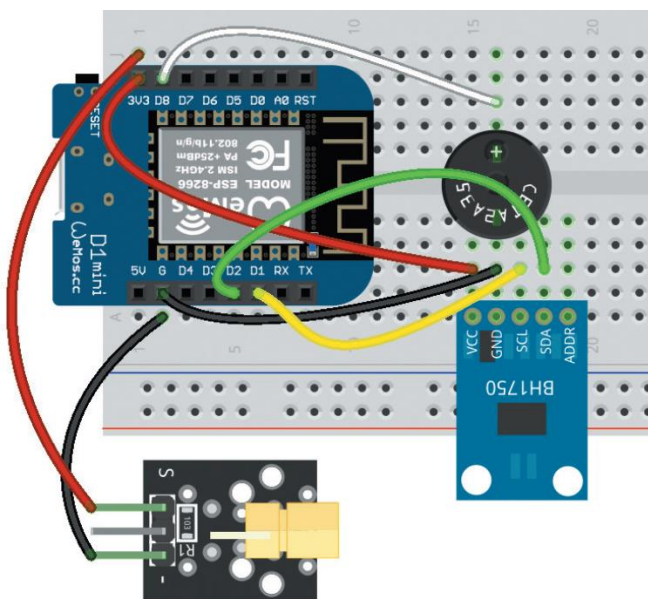


實驗流程

連接電子元件與電路

撰寫程式碼

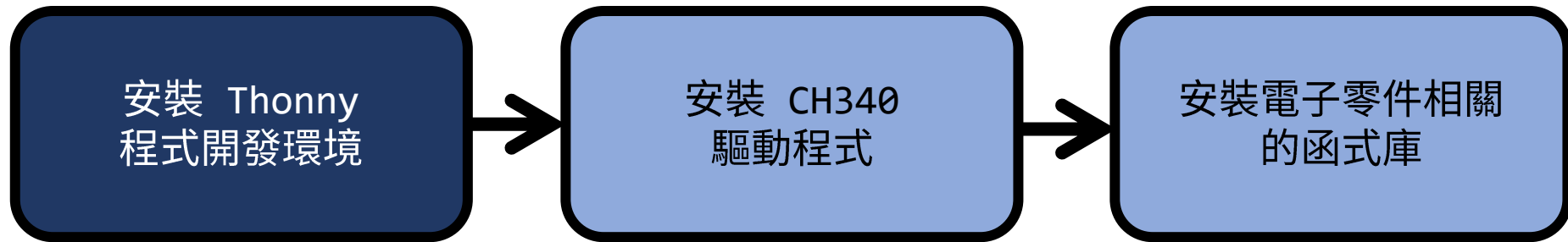
將D1mini連接至電腦
並執行程式碼



```
Lab 21.py
1 import network, urequests, utime
2 from machine import Pin, I2C
3 from hcsr04 import HCSR04
4
5 sonar = HCSR04(trigger_pin=14, echo_pin=12)
6
7 ssid = "你的WiFi名稱"
8 pw = "你的WiFi密碼"
9 key = "你的金鑰"
10
11 url = "https://maker.ifttt.com/trigger/sonar_detected/wit"
12
13 print("連接 WiFi: " + ssid + "...")
14 wifi = network.WLAN(network.STA_IF)
15 wifi.active(True)
16 wifi.connect(ssid, pw)
17 while not wifi.isconnected():
```



前置作業



Outline

- 1-1. 感測器簡介：控制板 x 感測器 x 動作器
- 1-2. D1 mini 控制板簡介
- 1-3. 安裝 Python 開發環境
- 1-4. Python 物件、資料型別、變數、匯入模組
- 1-5. 安裝與設定 D1 mini
- 1-6. 認識硬體
- 1-7. D1 mini 的 IO 腳位以及數位訊號輸出
- 1-8. 流程控制 (while 迴圈) 與區塊縮排
- 補給站：安裝 MicroPython 到 D1 mini

1-3. 安裝 Python 開發環境

- 開始學 Python 控制硬體前，先安裝好 Python 開發環境。
 - 下載與安裝 Thonny。

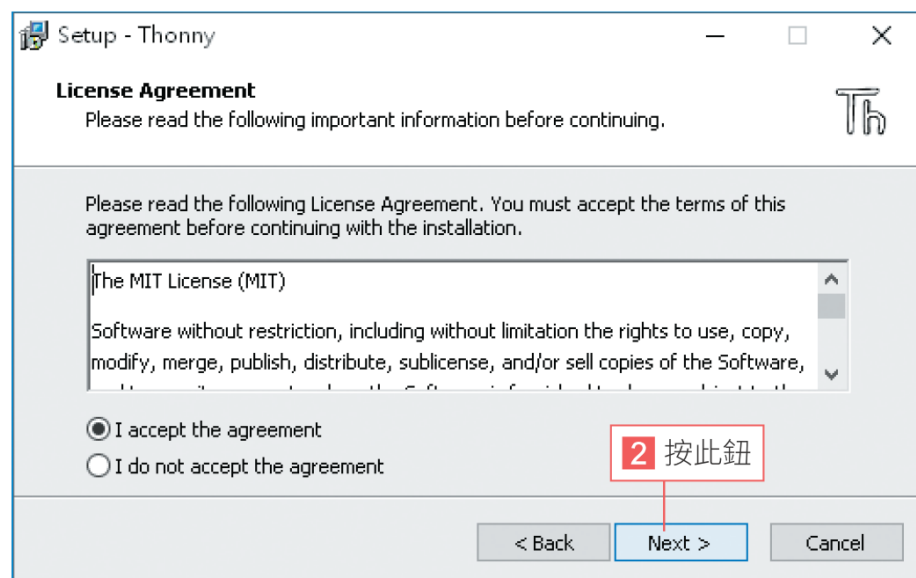


1-3. 安裝 Python 開發環境

- 下載後雙按執行該檔案：

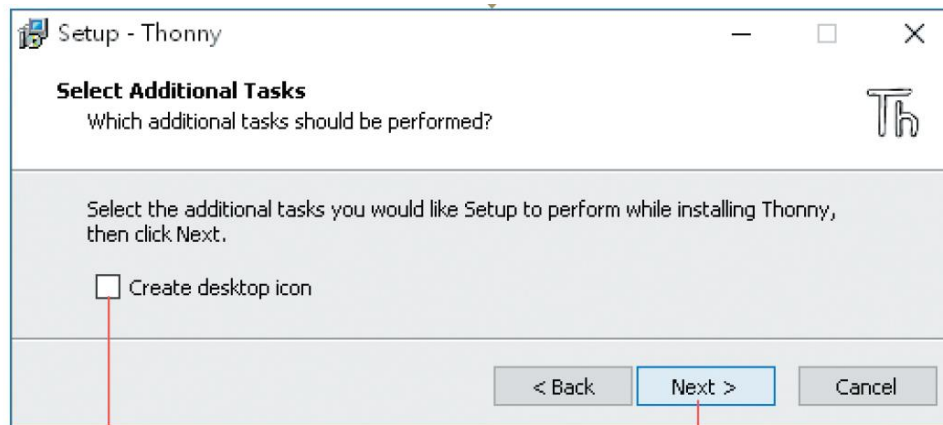
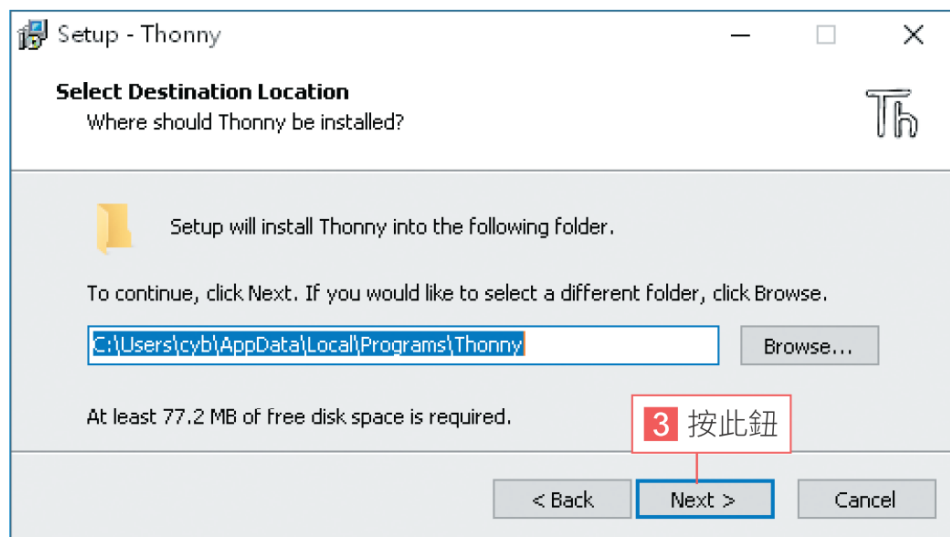


1 按此鈕



2 按此鈕

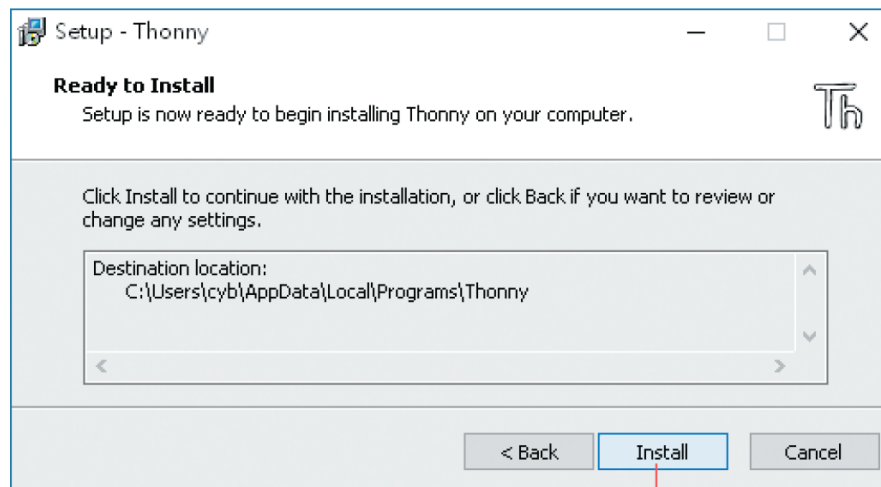
1-3. 安裝 Python 開發環境



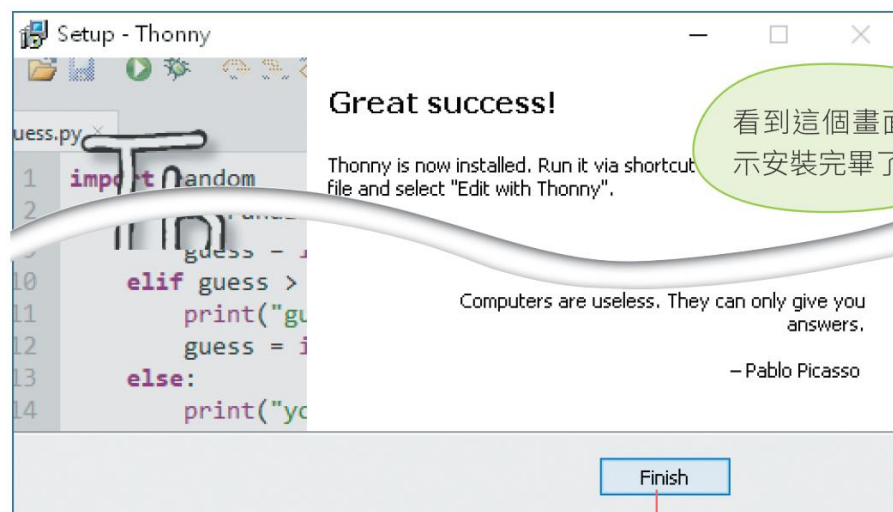
4 勾選這個項目在桌面建立捷徑

5 按此鈕

1-3. 安裝 Python 開發環境



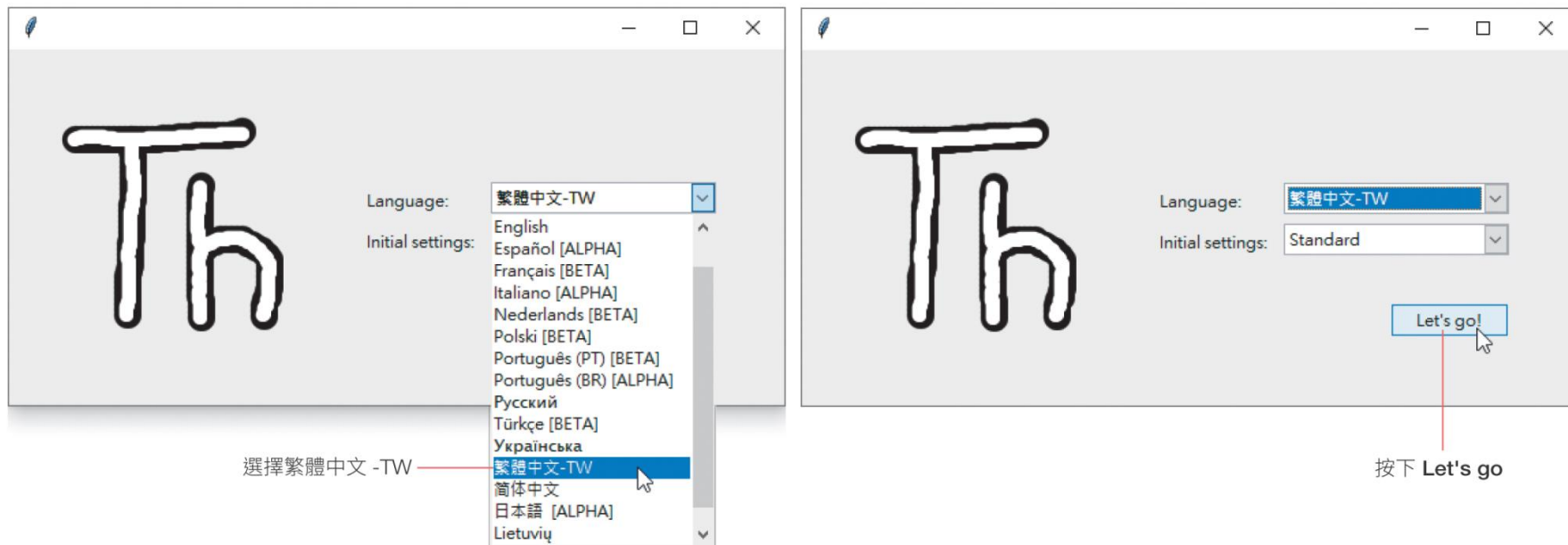
6 按此鈕開始安裝



7 按此鈕結束安裝程序

1-3. 安裝 Python 開發環境

- 開始寫第一行程式
 - 按 Windows 開始功能表中的 Thonny 項目或桌面上的捷徑，開啟 Thonny 開發環境：



1-3. 安裝 Python 開發環境

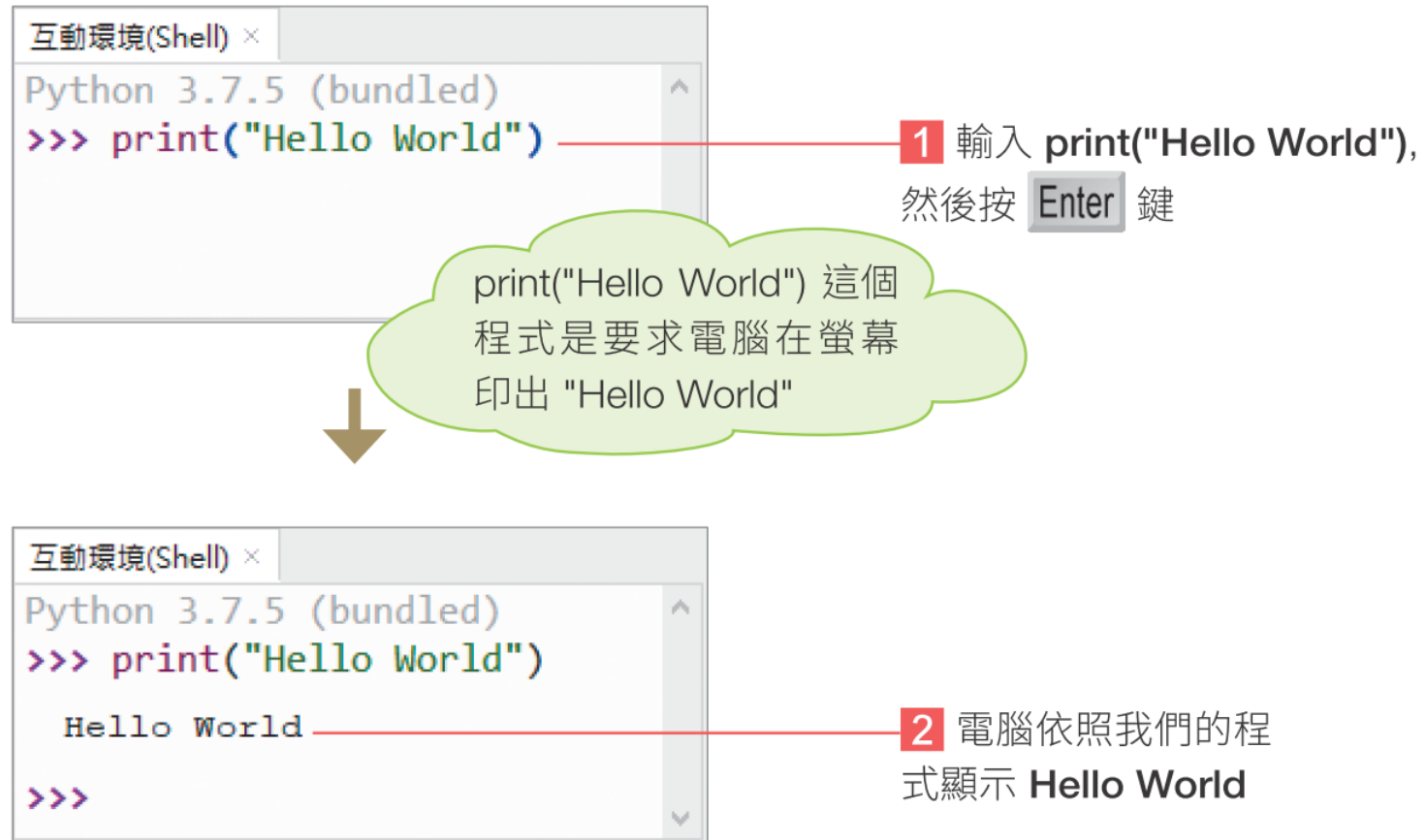


互動性程式執行區

程式編輯區

1-3. 安裝 Python 開發環境

- 請如下在 Shell 窗格寫下第一行程式：



互動環境(Shell) × Python 3.7.5 (bundled)

```
>>> print("Hello World")
```

1 輸入 `print("Hello World")`, 然後按 **Enter** 鍵

print("Hello World") 這個程式是要求電腦在螢幕印出 "Hello World"

互動環境(Shell) × Python 3.7.5 (bundled)

```
>>> print("Hello World")
Hello World
>>>
```

2 電腦依照我們的程式顯示 **Hello World**

1-3. 安裝 Python 開發環境

- 寫程式就像是寫劇本，用來控制電腦如何動作。
 - 但這個控制需要將每一個步驟都寫下來，這樣電腦才能依照程式指令來完成動作。



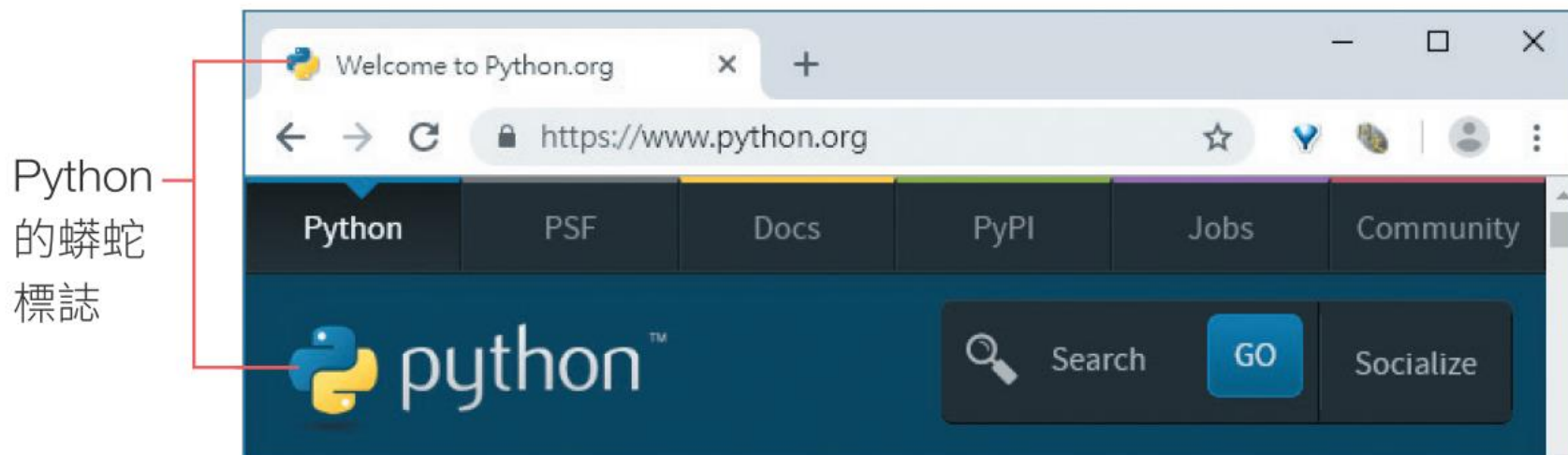
喂！電腦～
唱一首歌！

我... 我... 我
不知道怎麼唱



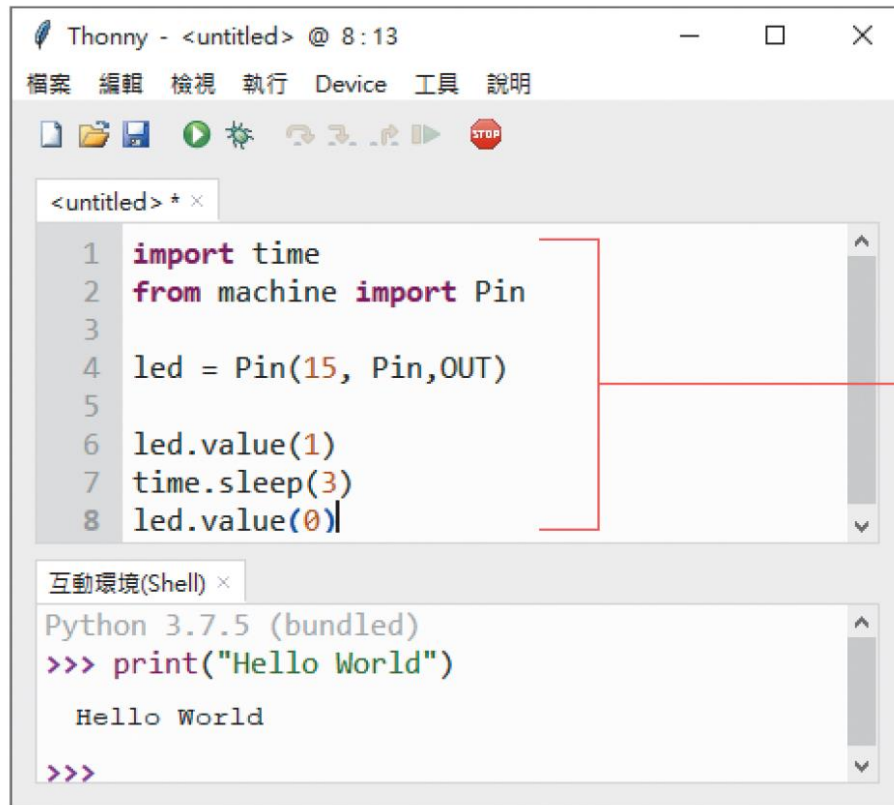
1-3. 安裝 Python 開發環境

- Python 程式語言
 - 在電腦的世界裡有不同的程式語言，每一種程式語言的語法與特性都不相同，各有其優缺點。



1-3. 安裝 Python 開發環境

- Thonny 開發環境基本操作



```
Thonny - <untitled> @ 8:13
檔案 編輯 檢視 執行 Device 工具 說明

<untitled> * x
1 import time
2 from machine import Pin
3
4 led = Pin(15, Pin,OUT)
5
6 led.value(1)
7 time.sleep(3)
8 led.value(0)

互動環境(Shell) x
Python 3.7.5 (bundled)
>>> print("Hello World")
Hello World
>>>
```

在此區域撰寫程式

Shell 窗格是一個交談介面，寫下一行指令後，電腦會立刻執行，適合用來作為程式測試。

1-3. 安裝 Python 開發環境

- 如果覺得 Thonny 開發環境的文字過小：

The image shows two screenshots from the Thonny IDE. The top screenshot shows the 'Tools' menu with 'Options...' selected. The bottom screenshot shows the 'Thonny Options' dialog box with the 'Theme and Fonts' tab active. The font size for the editor is set to 13, and the font for console output is set to Courier New. A preview of Python code is shown with the current font settings. Red arrows and numbers 1-4 point to specific actions: 1. Clicking 'Options...' in the Tools menu. 2. Switching to the 'Theme and Fonts' tab. 3. Selecting a font size (14) in the dropdown menu. 4. Clicking the 'Confirm' button.

1 執行選單的『工具 / 選項...』命令，開啟設定視窗

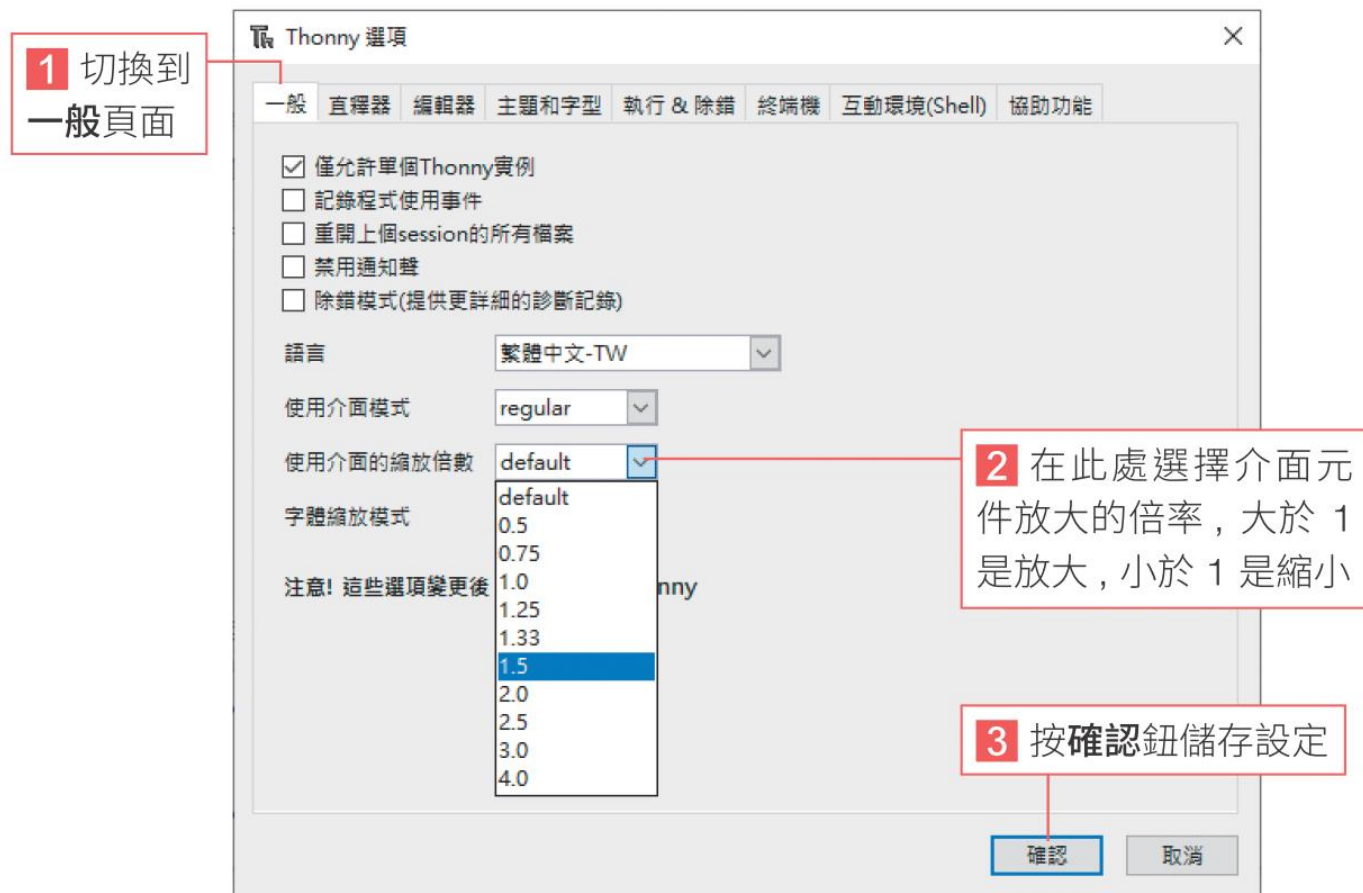
2 切換到主題和字型頁面

3 在此處選擇字型大小

4 按確認鈕儲存設定

1-3. 安裝 Python 開發環境

- 如果覺得介面上按鈕太小不好按：



1-3. 安裝 Python 開發環境

- 當撰寫好程式要儲存：



- 若要打開之前儲存的程式或範例程式檔：



1-3. 安裝 Python 開發環境

- 若要讓電腦執行，或停止程式：



Outline

- 1-1. 感測器簡介：控制板 x 感測器 x 動作器
- 1-2. D1 mini 控制板簡介
- 1-3. 安裝 Python 開發環境
- 1-4. Python 物件、資料型別、變數、匯入模組
- 1-5. 安裝與設定 D1 mini
- 1-6. 認識硬體
- 1-7. D1 mini 的 IO 腳位以及數位訊號輸出
- 1-8. 流程控制 (while 迴圈) 與區塊縮排
- 補給站：安裝 MicroPython 到 D1 mini

Python

```
# 印出 Hello World! 字串物件  
print("Hello World!")
```

C

```
/* 印出 Hello World! 字串物件*/  
include <stdio.h>  
  
int main()  
{  
    printf("Hello, World!\n");  
    return 0;  
}
```

C++

```
//印出 Hello World! 字串物件  
#include <iostream>  
  
using namespace std;  
  
int main()  
{  
    cout << "Hello World" << endl;  
    return 0;  
}
```

Java

```
//印出 Hello World! 字串物件  
public class HelloWorld{  
  
    public static void main(String []args){  
        System.out.println("Hello World");  
    }  
}
```

物件

- 英文一般寫句子時，會以**名詞 + 動詞**。Python 是以**物件.方法**來描述。

文章寫作	寫 Python 程式	
車子	<code>car</code>	car 物件
車子向前進	<code>car.start()</code>	car 物件的 start 方法

- 方法後面會加上括號()`()`，有些方法需要加入額外的參數。
 - 例如：`car.go(100)`，車子加速到 100。
- 若方法有多個參數，以逗點分隔。
 - 例如：`car.left(50, 30)`，以 50 的速度，向左轉 30 度。

練習：字串物件

- 在互動模式中，輸入下列敘述：(>>> 指的是在互動模式中，執行單行敘述)

```
>>> "Hello World!".upper()—— 使用字串物件 "Hello World!" 的  
upper() 方法，將字串轉成大寫  
'HELLO WORLD!'  
  
>>> "Hello World!".find('r')—— find() 方法尋找 'r' 的位置  
(從 0 開始)  
8  
  
>>> "Hello World!".replace('r', 'u')—— replace() 方法將所有  
的 'r' 取代成 'u'  
'Hello Would!'
```

- 不同的物件會有不同的方法。例如：字串物件與整數物件。

資料型別

- 除字串物件以雙引號或單引號來表示，寫程式常有整數與浮點數（小數）物件，例如：111 與 11.1。

```
>>> 111 + 111 ————— 整數物件相加
```

```
222
```

```
>>> "111" + "111" ————— 字串物件串聯
```

```
'111111'
```

- 上述 + 的運算，因物件的資料不同而產生不同的結果。物件的種類，程式語言稱之為『物件型態』或『資料型態』(Data Type)。

練習：要分清楚資料型別

- 兩個資料型別若不同，可能會導致程式錯誤。

```
>>> 111 + "111" —— 不同型別的資料相加發生錯誤
```

```
Traceback (most recent call last):
```

```
  File "<ipython-input-6-4832c22160be>", line 1, in  
<module>
```

```
    111 + "111"
```

```
TypeError: unsupported operand type(s) for +: 'int'  
and 'str'
```

變數

- 『變數』(variable) 就像是掛在物件的名牌，幫物件取名之後，讓我們方便識別物件與操作，其語法為：

變數名稱 = 物件

- 例如：

```
>>> n1 = 123456789 —— 將整數物件 123456789 指派給變數 n1
>>> n2 = 987654321 —— 將整數物件 987654321 指派給變數 n2
>>> n1 + n2 —— 實際上是 123456789 + 987654321
1111111110
```

內建函式

- 『**函式**』 (function) 是一段預先寫好的程式，方便重複使用。而程式先將經常使用到的功能以函式的形式先寫好，稱為『**內建函式**』。
- 例如：print() 是最常用的顯示函數：

```
>>> print("abc") —— 顯示字串物件
```

```
abc
```

```
>>> print("abc".upper()) —— 顯示字串物件.方法的執行結果
```

```
ABC
```

```
>>> print(111 + 111) —— 顯示整數物件運算的結果
```

```
222
```

匯入模組

- 內建函式不就越多越好？若內建函式無限制增加，會導致啟動速度越來越慢，執行時佔用的記憶體越來越多。
- 『**模組**』 (module)，就是將同一類的函式打包成模組，預設不會啟用。需要時，再用**匯入** (import) 的方式啟用。預先寫好的稱為『**內建模組**』。

```
>>> import time —— 匯入時間相關的 time 模組
>>> time.sleep(3) —— 執行 time 模組的 sleep() 函式，暫停 3 秒

>>> from time import sleep —— 從 time 模組裡匯入 sleep() 函式
>>> sleep(5) —— 執行 sleep() 函式，暫停 5 秒
```


練習：匯入模組

程式

暫停 5 秒後，印出 Hello World! 字串物件

```
# 暫停 5 秒後，印出 Hello World!  
from time import sleep  
sleep(5)  
print("Hello World!")
```

指的是在程式編輯窗格中編輯與執行。

Outline

- 1-1. 感測器簡介：控制板 x 感測器 x 動作器
 - 1-2. D1 mini 控制板簡介
 - 1-3. 安裝 Python 開發環境
 - 1-4. Python 物件、資料型別、變數、匯入模組
 - 1-5. 安裝與設定 D1 mini
 - 1-6. 認識硬體
 - 1-7. D1 mini 的 IO 腳位以及數位訊號輸出
 - 1-8. 流程控制 (while 迴圈) 與區塊縮排
- 補給站：安裝 MicroPython 到 D1 mini

1-5. 安裝與設定 D1 mini

- 前面的程式都是在個人電腦上執行，但缺少對外連接的腳位，所以將改用 D1 mini 來執行 Python 程式。
- 下載與安裝驅動程式
 - 連線下載 D1 mini 驅動程式：

1 連線 http://www.wch.cn/downloads/CH341SER_EXE.html

2 按此鈕下載

若您使用 Mac 或是 Linux 系統的話，請依照您的系統點這兩個連結

通用範圍	版本	上傳時間	資料大小	
CH340G、CH340T、CH340C、CH340E、CH340B、CH341A、CH341T、CH341B、CH341C、CH341U	3.4	2016-09-28	237KB	下載

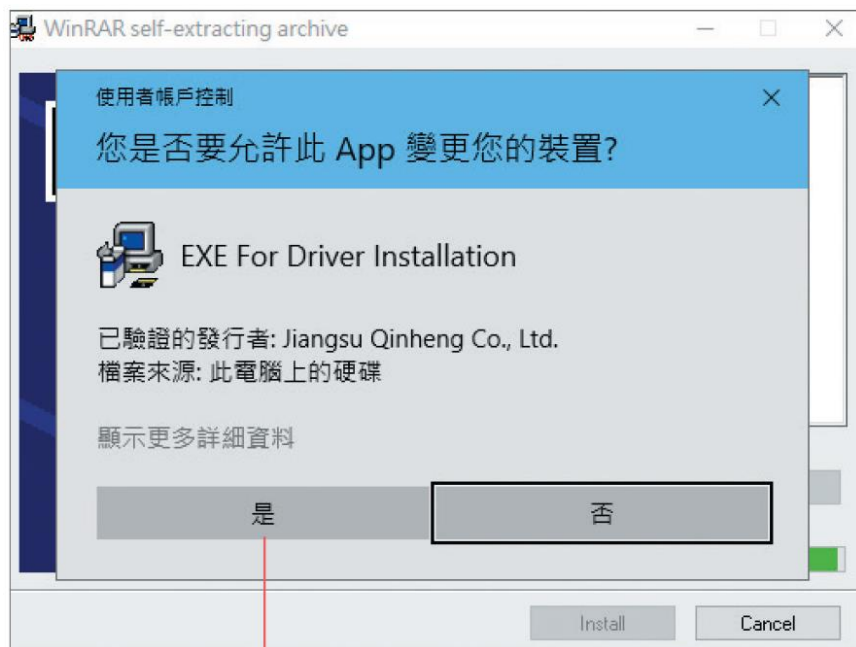
CH340/CH341USB轉串口WINDOWS驅動程序，支持32/64位 Windows 10/8.1/8/7/VISTA/XP、SERVER 2016/2012/2008/2003、2000/ME/98，通過微軟數字簽名認證，支持USB轉3綫和9綫串口等，用於隨產品發行到最終用戶。

相關資料

資料名稱	資料簡介
CH341SER.ZIP	CH340/CH341USB轉串口WINDOWS驅動程序，內含DLL動態庫及表標準波特率的設置等使用說明，支持32/64位 Windows 10/8.1/8/7/VISTA/XP、SERVER 2016/2012/2008/2003、2000/ME/98，通過微軟數字簽名認證，支持USB轉3綫和9綫串口。
CH341SER_LINUX.ZIP	CH340/CH341的USB轉串口LINUX驅動程序，支持32/64位系統。
CH341SER_MAC.ZIP	CH340/CH341的USB轉串口MAC OS驅動程序，支持32/64位系統，內有使用說明。

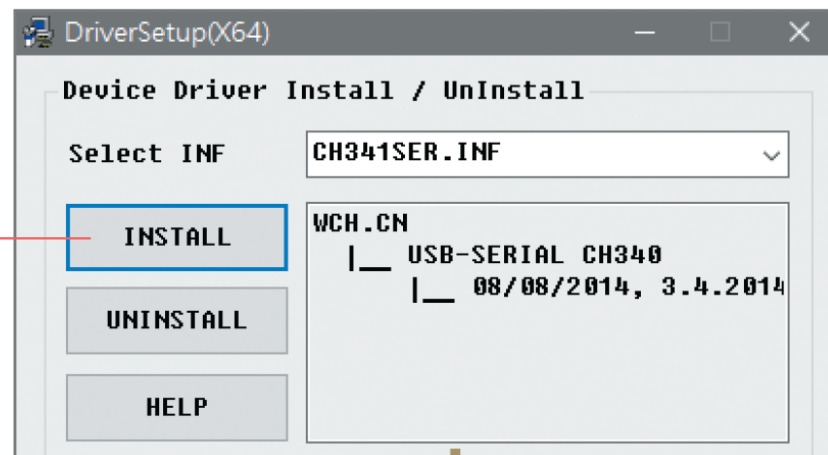
1-5. 安裝與設定 D1 mini

- 下載後雙按執行該檔案：

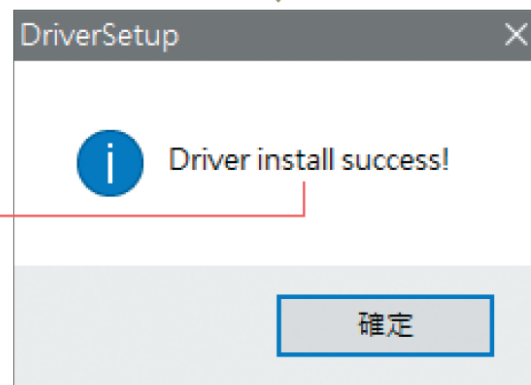


1 請選是允許安裝

2 按此鈕進行安裝



看到 success便表示安裝成功了！



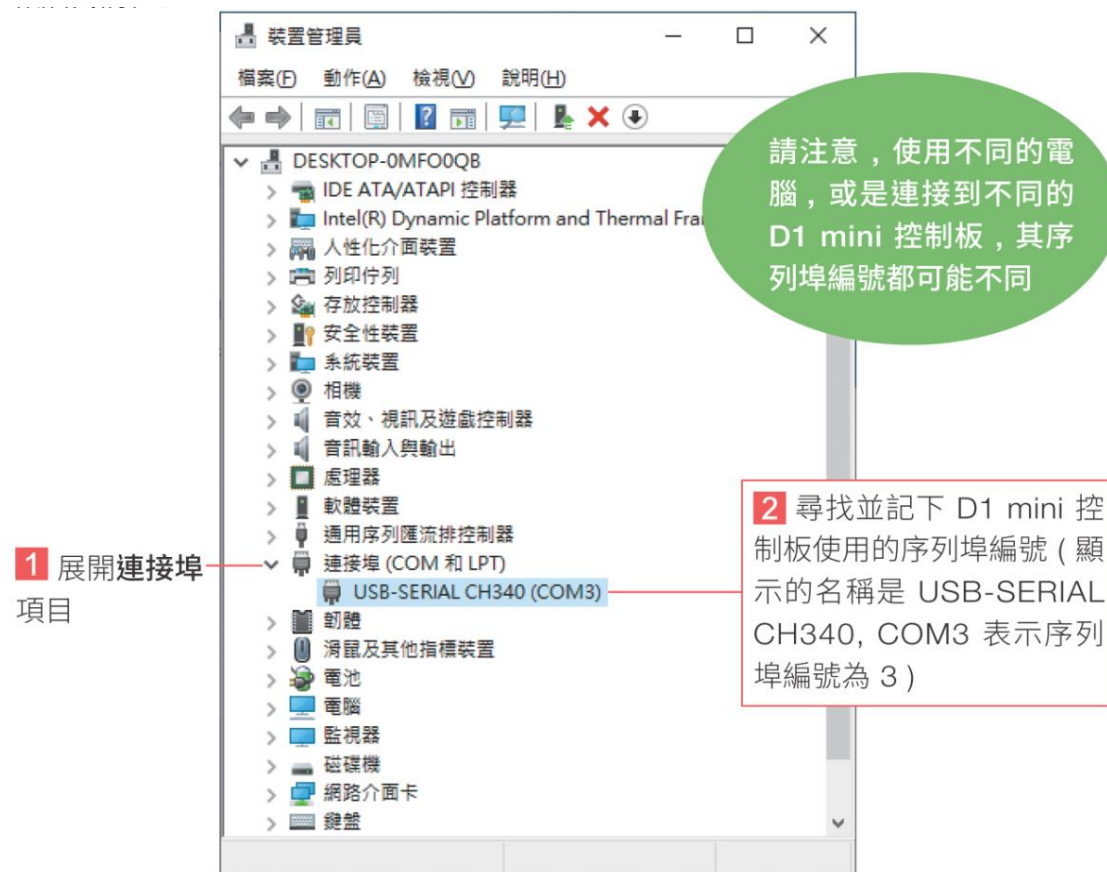
1-5. 安裝與設定 D1 mini

- 連接 D1 mini
- 將 USB 連接線接上 D1 mini 的 USB 孔，另一端接上電腦：



1-5. 安裝與設定 D1 mini

- 在電腦開始圖示上右鈕執行「裝置管理員」命令，或執行「開始 / 控制台 / 系統及安全性 / 系統 / 裝置管理員」命令，開啟裝置管理員，尋找 D1 mini 板使用的序列埠：



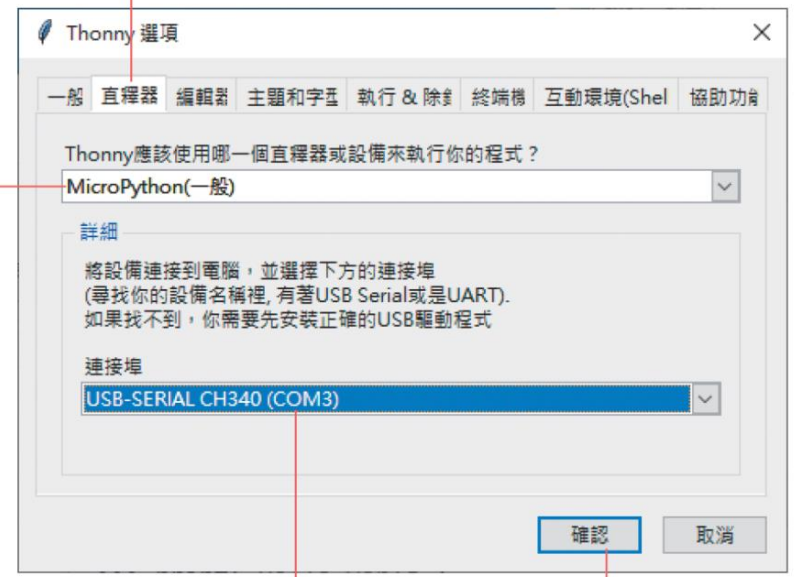
1-5. 安裝與設定 D1 mini

- 如下設定 Thonny 連線 D1 mini：



1 執行選單的『工具 / 選項...』命令，開啟設定視窗

2 切換到直釋器頁面

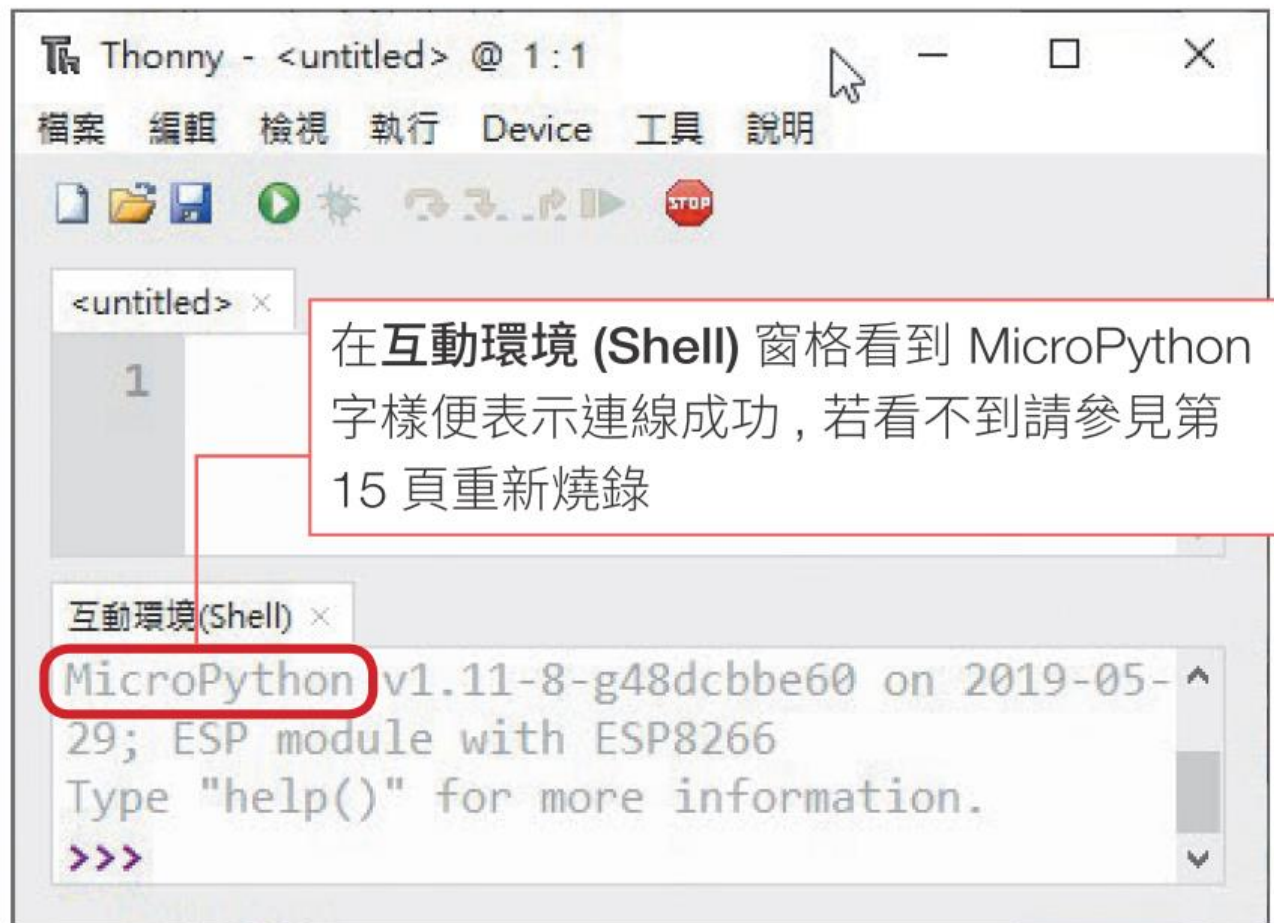


3 拉下選單選擇 MicroPython(一般)

4 拉下選單選擇剛剛記下的序列埠編號 (Mac 上請選有 "/dev/cu.wchusbserial." 字樣的項目)

5 按確認鈕儲存設定

1-5. 安裝與設定 D1 mini

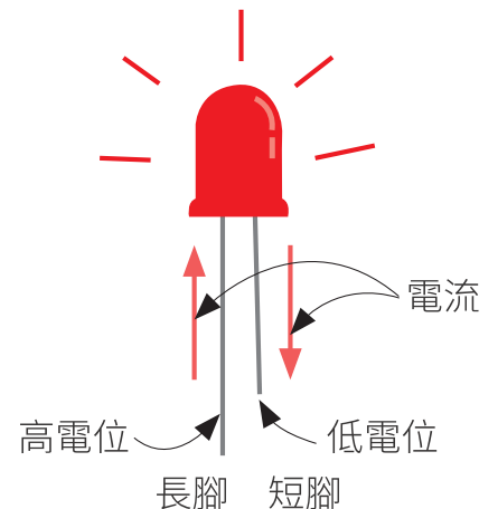


Outline

- 1-1. 感測器簡介：控制板 x 感測器 x 動作器
- 1-2. D1 mini 控制板簡介
- 1-3. 安裝 Python 開發環境
- 1-4. Python 物件、資料型別、變數、匯入模組
- 1-5. 安裝與設定 D1 mini
- 1-6. 認識硬體**
- 1-7. D1 mini 的 IO 腳位以及數位訊號輸出
- 1-8. 流程控制 (while 迴圈) 與區塊縮排
- 補給站：安裝 MicroPython 到 D1 mini

1-6. 認識硬體

- LED
 - LED又稱為發光二極體，有長短兩隻接腳。
 - 長腳接高電位，短腳接低電位，產生高低電位差讓電流流過 LED 即可發光。



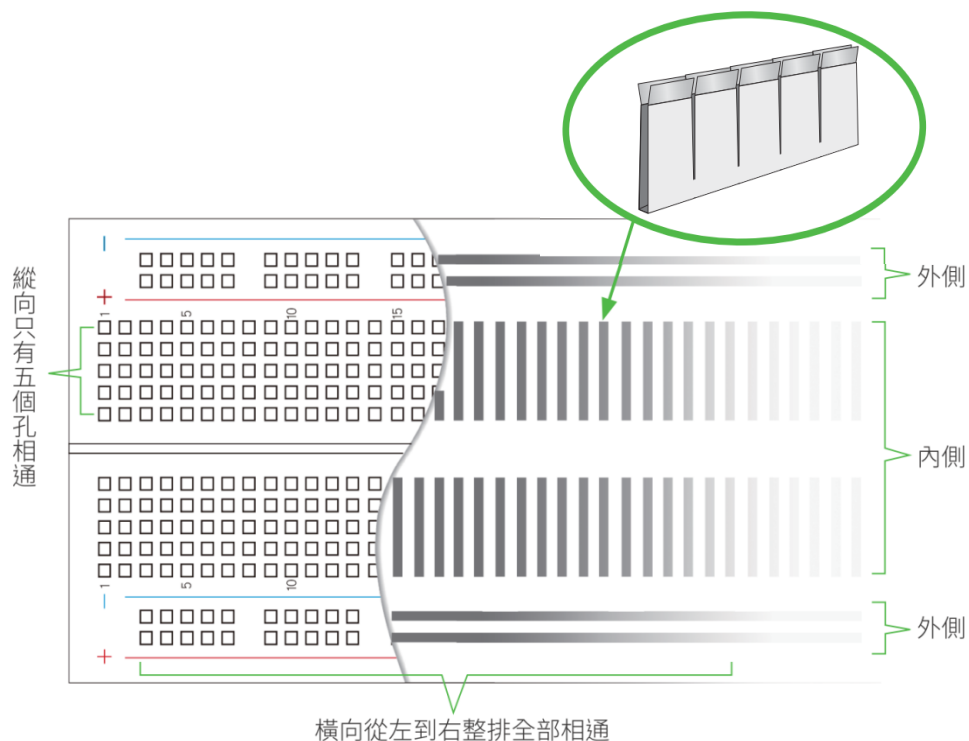
- 電阻
 - 用電阻來限制電路中的電流，避免因電流過大而燒壞元件。



1-6. 認識硬體

- 麵包板

- 麵包板的表面有很多的插孔。當零件接腳插入插孔時，實際上是插入下方相連的金屬夾，進而和同一條金屬夾上的其他插孔上的零件接通。



1-6. 認識硬體

- 杜邦線與排針

- 杜邦線是二端已經做好接頭的單心線，可用來連接電子元件。使用排針可將杜邦線或裝置上的母頭變成公頭：

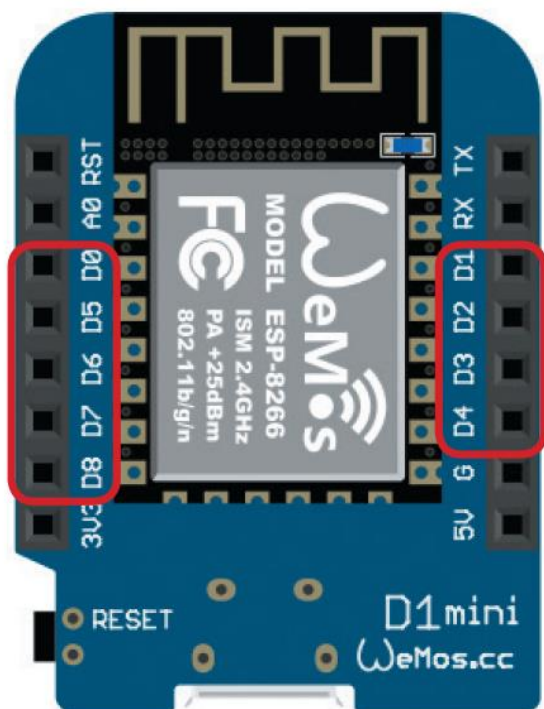


Outline

- 1-1. 感測器簡介：控制板 x 感測器 x 動作器
 - 1-2. D1 mini 控制板簡介
 - 1-3. 安裝 Python 開發環境
 - 1-4. Python 物件、資料型別、變數、匯入模組
 - 1-5. 安裝與設定 D1 mini
 - 1-6. 認識硬體
 - 1-7. D1 mini 的 IO 腳位以及數位訊號輸出
 - 1-8. 流程控制 (while 迴圈) 與區塊縮排
- 補給站：安裝 MicroPython 到 D1 mini

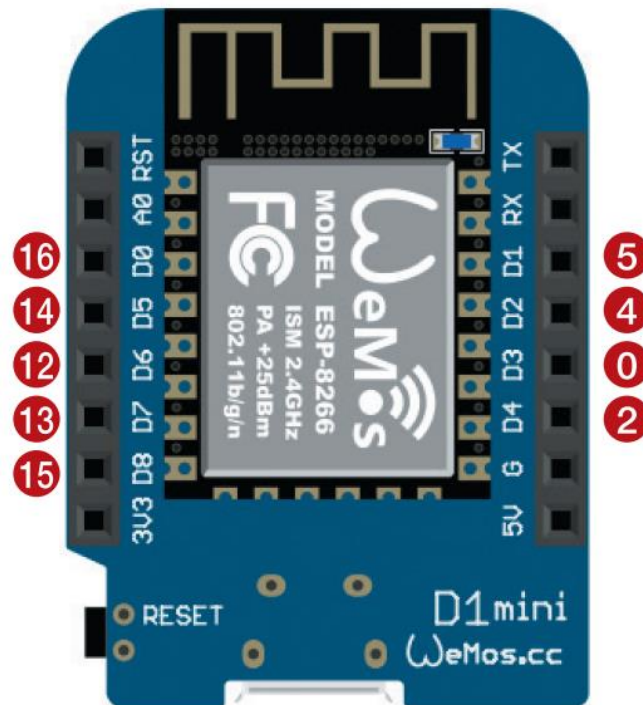
1-7. D1 mini 的 IO 腳位以及數位訊號輸出

- 電子世界中訊號只分高電位跟低電位兩個值，稱為數位訊號。
 - D1 mini 兩側 D0~D8 的 9 個腳位，可用程式控制高、低電位，所以被稱為數位 IO (Input/Output) 腳位。



1-7. D1 mini 的 IO 腳位以及數位訊號輸出

- 在程式中會以 1 代表高電位，0 代表低電位。
 - D1 mini 兩側數位 IO 腳位內側標示是 D0 ~ D8，但實際上真正編號在 D1 mini 晶片內部：



1-7. D1 mini 的 IO 腳位以及數位訊號輸出

- Lab01

點亮/熄滅 LED

實驗目的

用 Python 程式控制 D1 mini 腳位，藉此點亮或熄滅該腳位連接的 LED 燈。

材料

- D1 mini

- 線路圖
- 無需接線

- 設計原理

- D1 mini 板上已內建一個藍色 LED 燈，短腳接到腳位 D4，長腳接到高電位處。在程式中將 D1 mini 2 號腳位設為低電位，即可點亮 LED 燈。
- 為了在程式中控制 D1 mini 的腳位，必須先從 machine 模組匯入 Pin 物件：

```
>>> from machine import Pin
```

1-7. D1 mini 的 IO 腳位以及數位訊號輸出

lab1-12

- 如下建立 2 號腳位的 Pin 物件：

```
>>> led = Pin(2, Pin.OUT)
```

- 接著即可使用 `value()` 方法，來指定腳位電位高低：

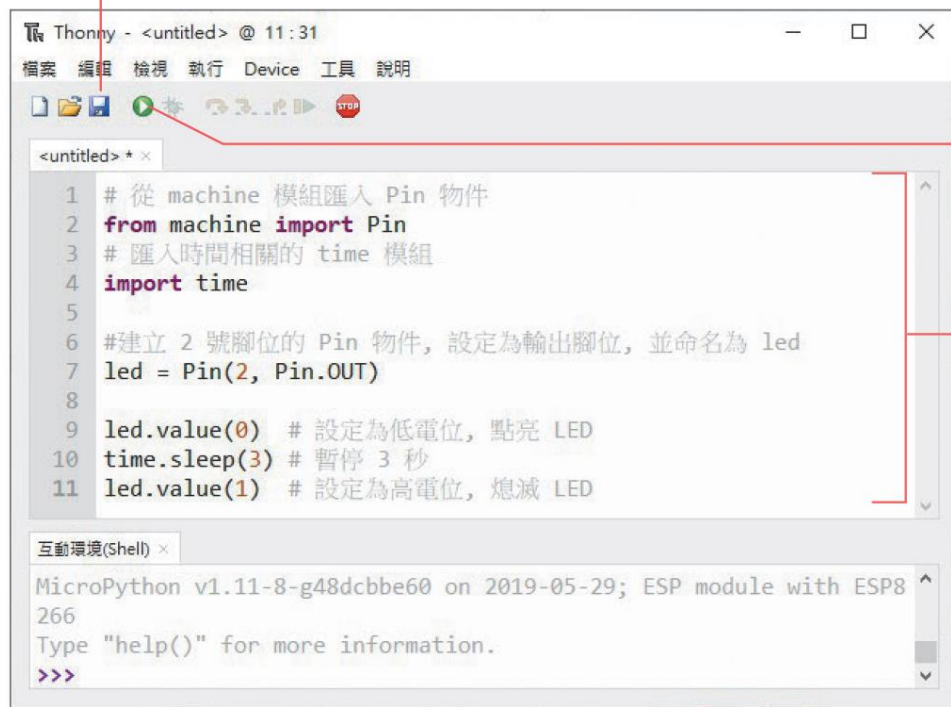
```
>>> led.value(1) ← 高電位  
>>> led.value(0) ← 低電位
```

1-7. D1 mini 的 IO 腳位以及數位訊號輸出

lab1-13

• 程式設計

2 按此鈕或按 **Ctrl** + **S** 儲存檔案

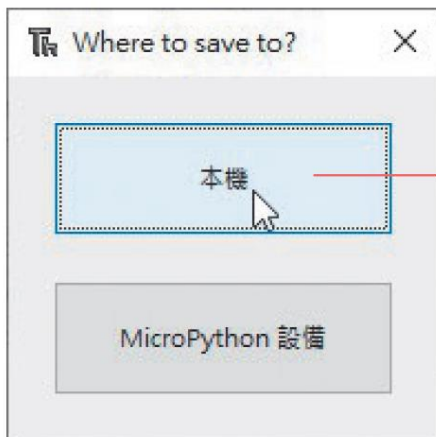


```
<untitled> * x
1 # 從 machine 模組匯入 Pin 物件
2 from machine import Pin
3 # 匯入時間相關的 time 模組
4 import time
5
6 #建立 2 號腳位的 Pin 物件，設定為輸出腳位，並命名為 led
7 led = Pin(2, Pin.OUT)
8
9 led.value(0) # 設定為低電位，點亮 LED
10 time.sleep(3) # 暫停 3 秒
11 led.value(1) # 設定為高電位，熄滅 LED

互動環境(Shell) x
MicroPython v1.11-8-g48dcbbe60 on 2019-05-29; ESP module with ESP8
266
Type "help()" for more information.
>>>
```

3 按此鈕或按 **F5** 執行程式

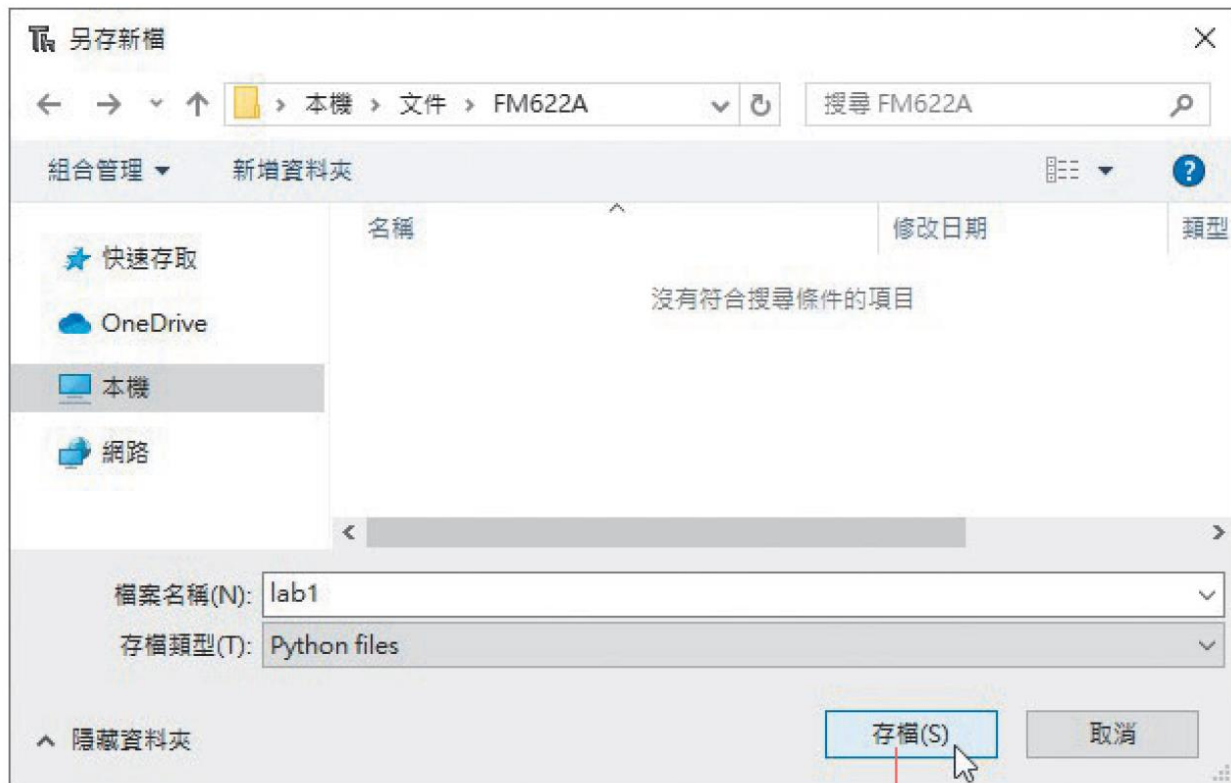
1 程式編輯區
輸入程式碼



選擇本機

⚠ 若看不到**本機**的字樣，
可以直接點選兩個方框
中位於上方的方框。

1-7. D1 mini 的 IO 腳位以及數位訊號輸出



輸入檔名後按存檔鈕儲存

- 實測

- 請按 F5 執行程式，即可看到 LED 點亮 3 秒後熄滅。

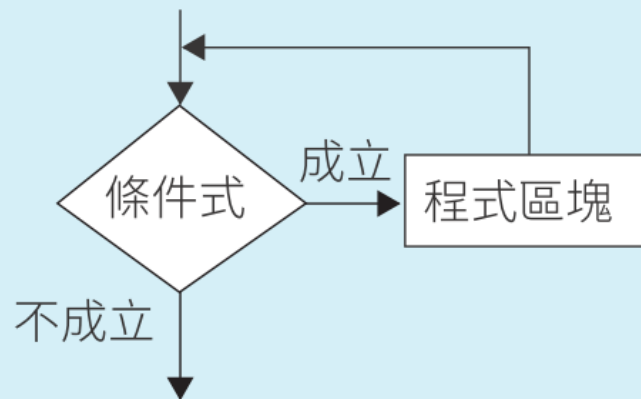
Outline

- 1-1. 感測器簡介：控制板 x 感測器 x 動作器
 - 1-2. D1 mini 控制板簡介
 - 1-3. 安裝 Python 開發環境
 - 1-4. Python 物件、資料型別、變數、匯入模組
 - 1-5. 安裝與設定 D1 mini
 - 1-6. 認識硬體
 - 1-7. D1 mini 的 IO 腳位以及數位訊號輸出
 - 1-8. 流程控制 (while 迴圈) 與區塊縮排
- 補給站：安裝 MicroPython 到 D1 mini

1-8. 流程控制 (while 迴圈) 與區塊縮排

- 如果要做出一直閃爍的效果，可利用 while 迴圈來依照條件重複執行。其語法如下：

while 條件式：
 程式區塊



1-8. 流程控制 (while 迴圈) 與區塊縮排

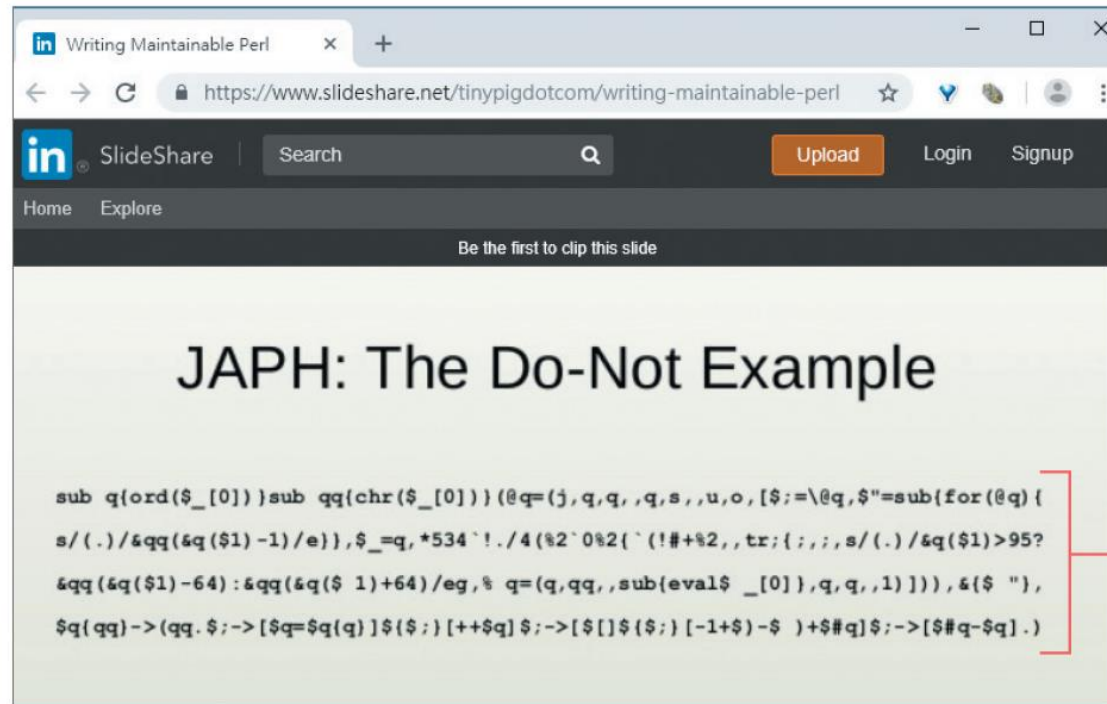
- 大多數狀況下會希望程式永遠重複執行，此時條件式就可用 `True` 來代替，例如：內建 LED 一直閃爍的效果，便可用以下程式碼：

```
while True:           # 一直重複執行
    led.value(0)      # 點亮 LED
    time.sleep(0.5)   # 暫停 0.5 秒
    led.value(1)      # 熄滅 LED
    time.sleep(0.5)   # 暫停 0.5 秒
```

- 屬於 `while` 的程式區塊要「以 4 個空格向右縮排」

1-8. 流程控制 (while 迴圈) 與區塊縮排

- 區塊縮排是 Python 的特色，可讓程式碼更加簡潔易讀。
- 其他程式語言大多是用括號或是關鍵字來決定區塊：



沒有縮排全都擠在一起的程式碼

1-8. 流程控制 (while 迴圈) 與區塊縮排

- Lab02

閃爍 LED	
實驗目的	用 Python 的 while 迴圈重複執行 LED 的控制程式，使其每 0.5 秒閃爍一次。
材料	<ul style="list-style-type: none">• D1 mini

- 線路圖
- 無需接線。

1-8. 流程控制 (while 迴圈) 與區塊縮排

- 程式設計

- 在 Thonny 程式編輯區輸入程式碼，輸入完畢後 `ctrl+s` 存檔。

```
01 # 從 machine 模組匯入 Pin 物件
02 from machine import Pin
03 # 匯入時間相關的 time 模組
04 import time
05
06 # 建立 2 號腳位的 Pin 物件，設定為輸出腳位，並命名為 led
07 led = Pin(2, Pin.OUT)
08
09 while True:          # 一直重複執行
10     led.value(0)     # 點亮 LED
11     time.sleep(0.5)  # 暫停 0.5 秒
12     led.value(1)     # 熄滅 LED
13     time.sleep(0.5)  # 暫停 0.5 秒
```

- 實測

- 請按 `F5` 執程式，即可看到 LED 每 0.5 秒閃爍一次。

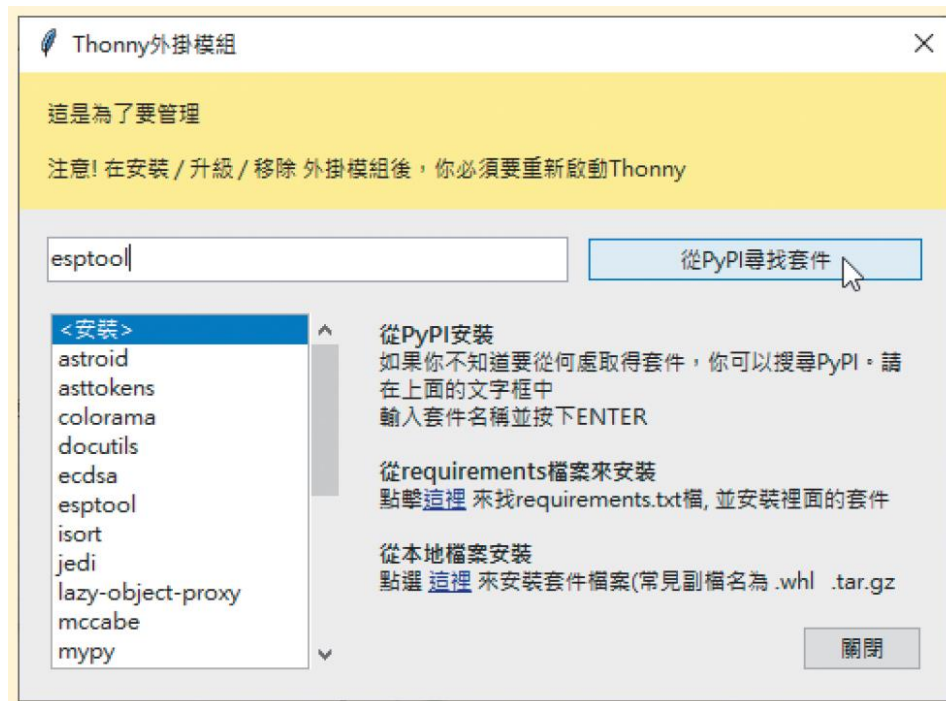
Outline

- 1-1. 感測器簡介：控制板 x 感測器 x 動作器
- 1-2. D1 mini 控制板簡介
- 1-3. 安裝 Python 開發環境
- 1-4. Python 物件、資料型別、變數、匯入模組
- 1-5. 安裝與設定 D1 mini
- 1-6. 認識硬體
- 1-7. D1 mini 的 IO 腳位以及數位訊號輸出
- 1-8. 流程控制 (while 迴圈) 與區塊縮排

補給站：安裝 MicroPython 到 D1 mini

補給站：安裝 MicroPython 到 D1 mini

1. 下載安裝 D1 mini 控制板驅動程式，並檢查連接埠編號。
2. 連線下載 MicroPython 韌體：
<https://micropython.org/download/esp8266/>。
3. 至 Thonny 功能表點選工具 / 管理外掛模組...，輸入 esptool 後，按下從 PyPI 尋找套件。



補給站：安裝 MicroPython 到 D1 mini

4. 按下安裝，完成後按下關閉。



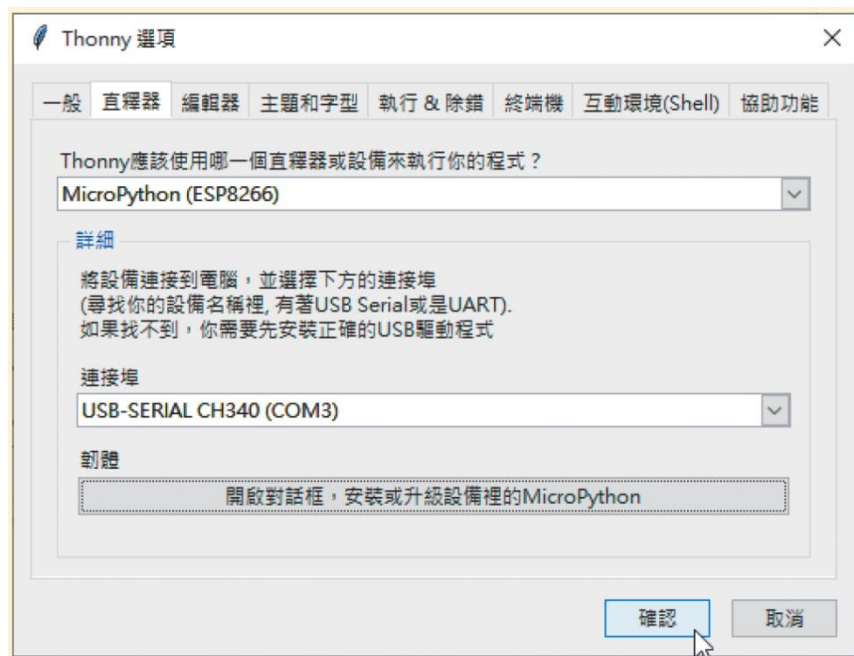
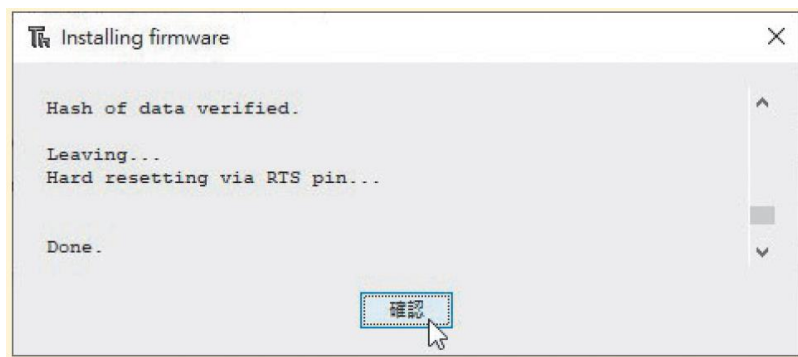
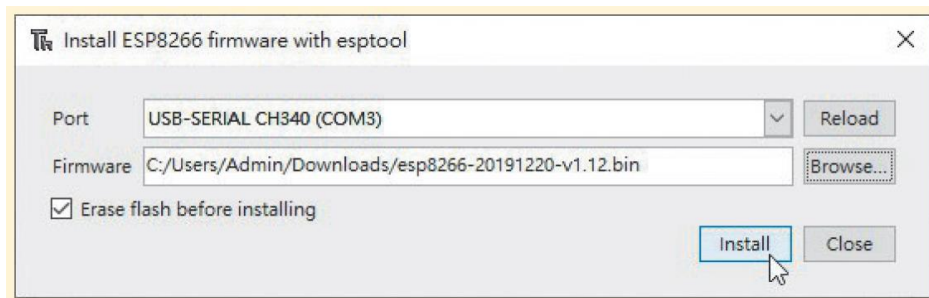
補給站：安裝 MicroPython 到 D1 mini

5. 安裝完 esptool 後回到 Thonny 功能表點選工具 / 選項 / 直譯器，選擇 MicroPython(ESP8266) 選項，連接埠選擇裝置管理員中顯示的埠號，之後按下開啟對話框，安裝或升級設備...按鈕。



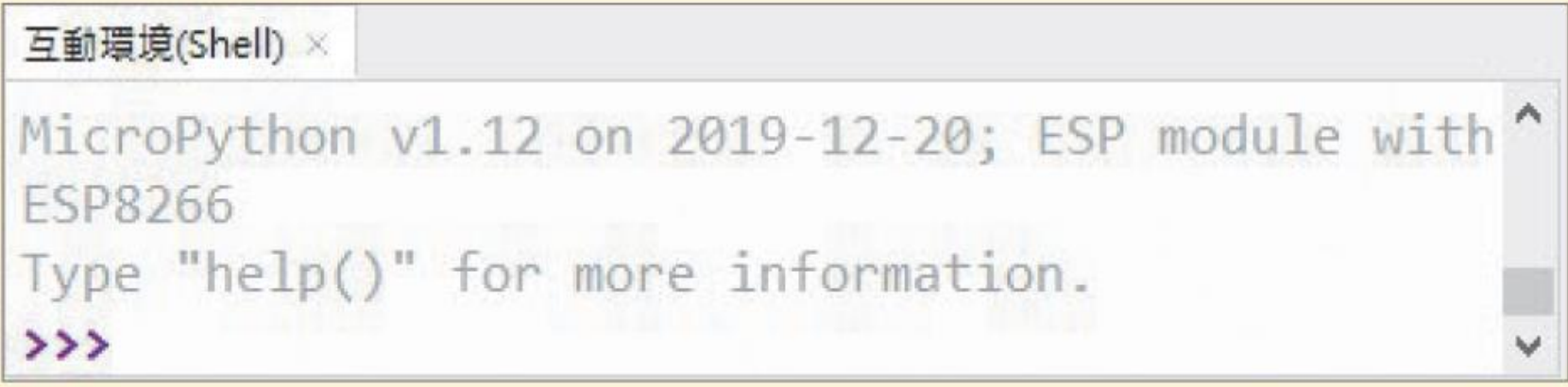
補給站：安裝 MicroPython 到 D1 mini

6. 選擇 Port，以及方才下載的 MicroPython 韌體的路徑後按下 Install，燒錄完畢按下確認。



補給站：安裝 MicroPython 到 D1 mini

7. 若 Shell 窗格中出現 MicroPython 字樣，代表燒錄成功。



```
互動環境(Shell) ×  
MicroPython v1.12 on 2019-12-20; ESP module with  
ESP8266  
Type "help()" for more information.  
>>>
```