

# 玩轉IOT Workshop

以樹莓派結合Windows Azure 簡易實現物聯網之概念。

我們將會需要以下器材：

實驗器材	數量 (qts)	附註
樹莓派 2 model B+ 1. Python 3	1	(Linux Base) Raspbian
LED燈	1	
4.7K $\Omega$ 電阻	1	色碼：黃紫紅 (詳見： <a href="#">電阻色碼參考表</a> )
220 $\Omega$ 電阻	1	色碼：紅紅棕 (詳見： <a href="#">電阻色碼參考表</a> )
溫度感測器-DS18B20	1	
無線網路卡 Tp-Link-TL_WN725N	1	

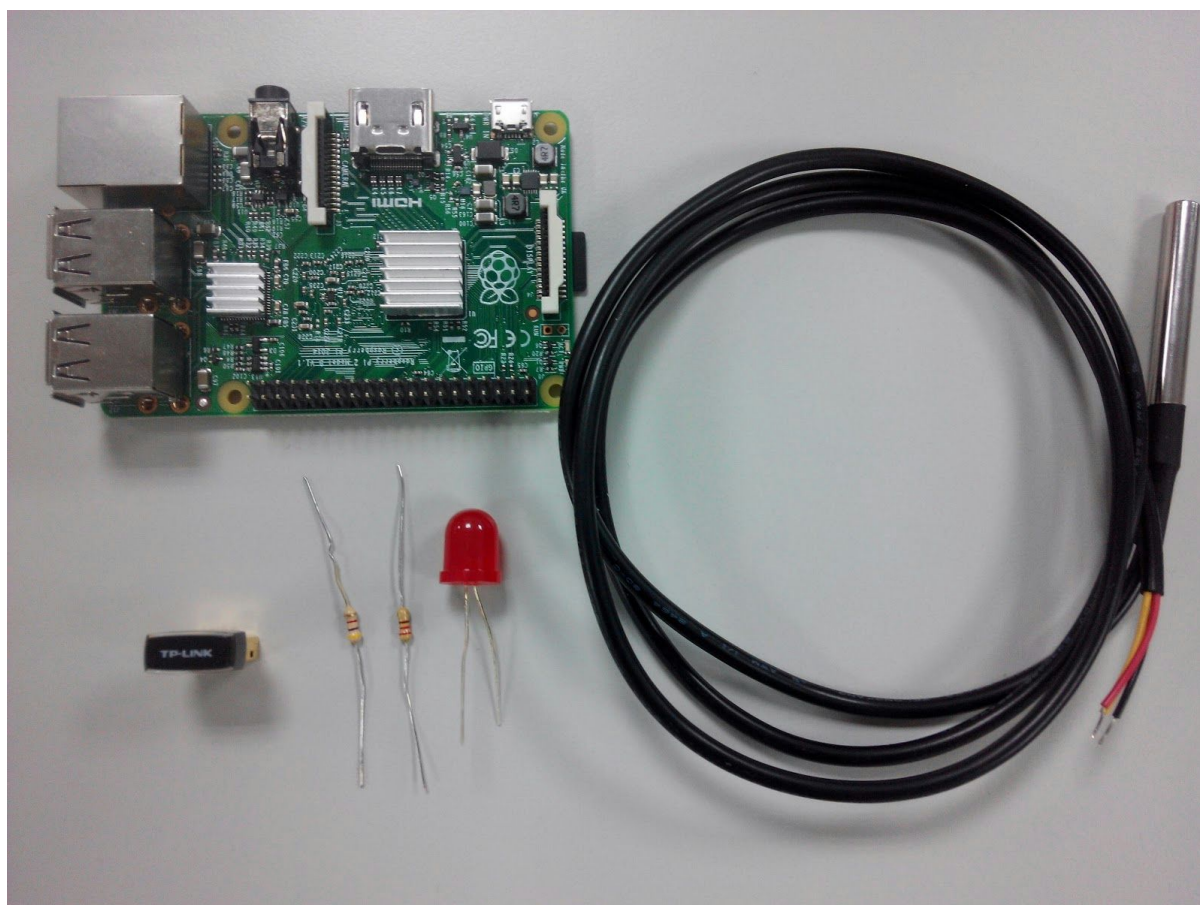


圖1：實驗所需材料

## 都準備好了，那我們就開始吧

當器材都準備好後，請各位將樹莓派的電源接上，並檢查畫面上有沒有正確顯示樹梅派的各項資訊。本實驗中樹梅派的作業系統為Linux Based -Raspbian，開機後可看到畫面：

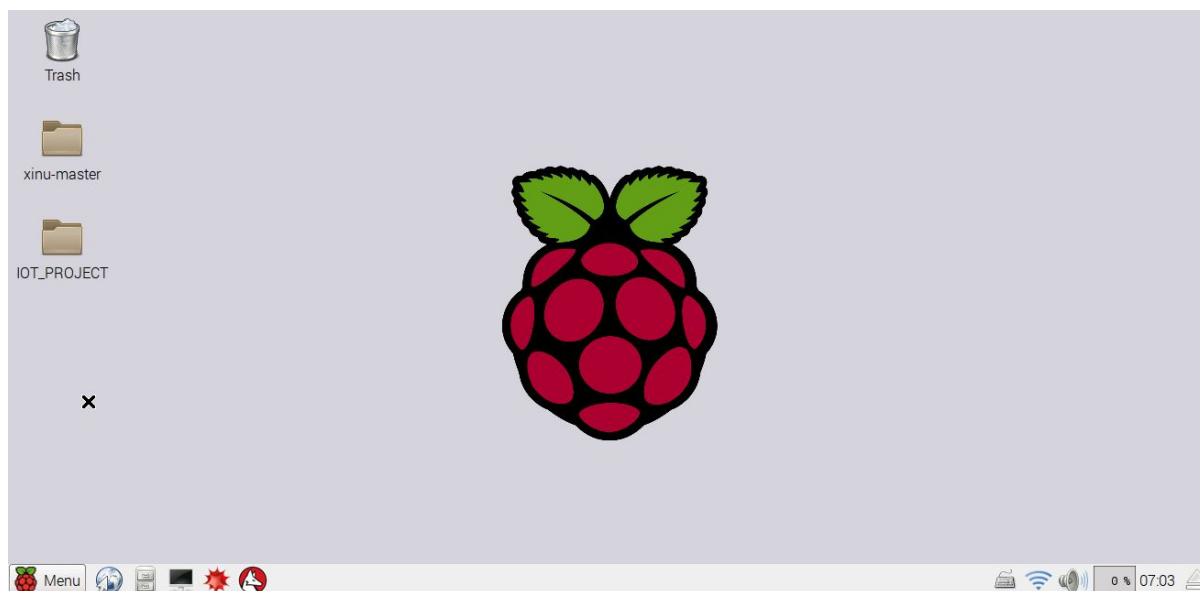


圖2：Raspbian 開機畫面

請連接上無線網路，如下圖所示：

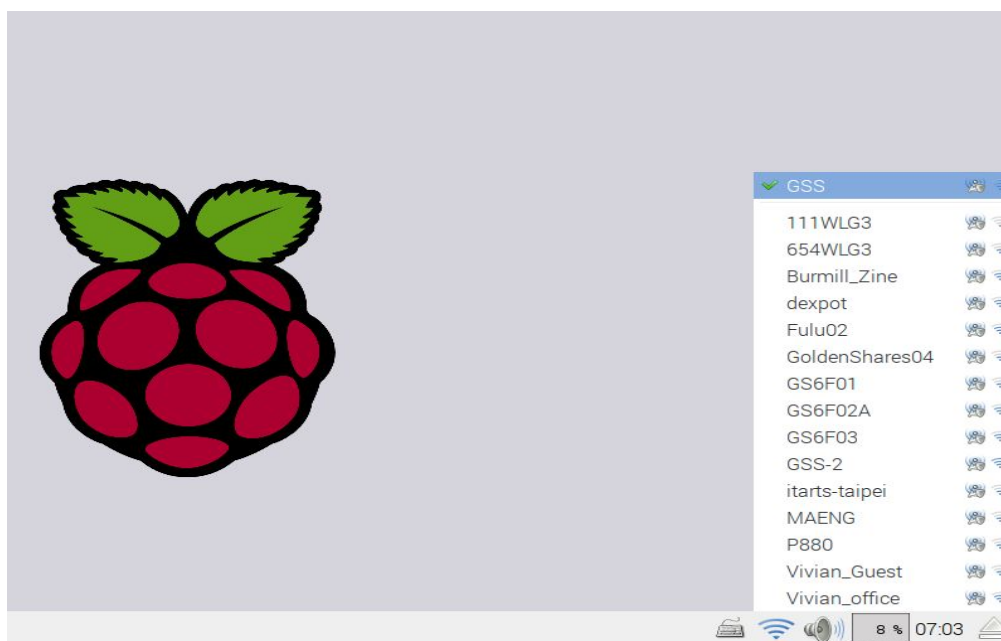


圖3：連結無線網路

# 準備動手接電路-控制LED燈閃爍！

為了讓各位熟悉基本的流程，我們要先實作一個簡易的專案。專案內容是藉由樹莓派來控制LED燈，並使其閃爍。請各位拿起手中的麵包板，將相關實驗器材依照下列圖示接起來。對於連結麵包板較不熟習的同學請大方舉手詢問，或參考[這篇文章](#)。

實驗器材	數量(qts)
樹莓派2	1
LED燈	1
220 Ω 電阻(色碼:紅紅棕)	1

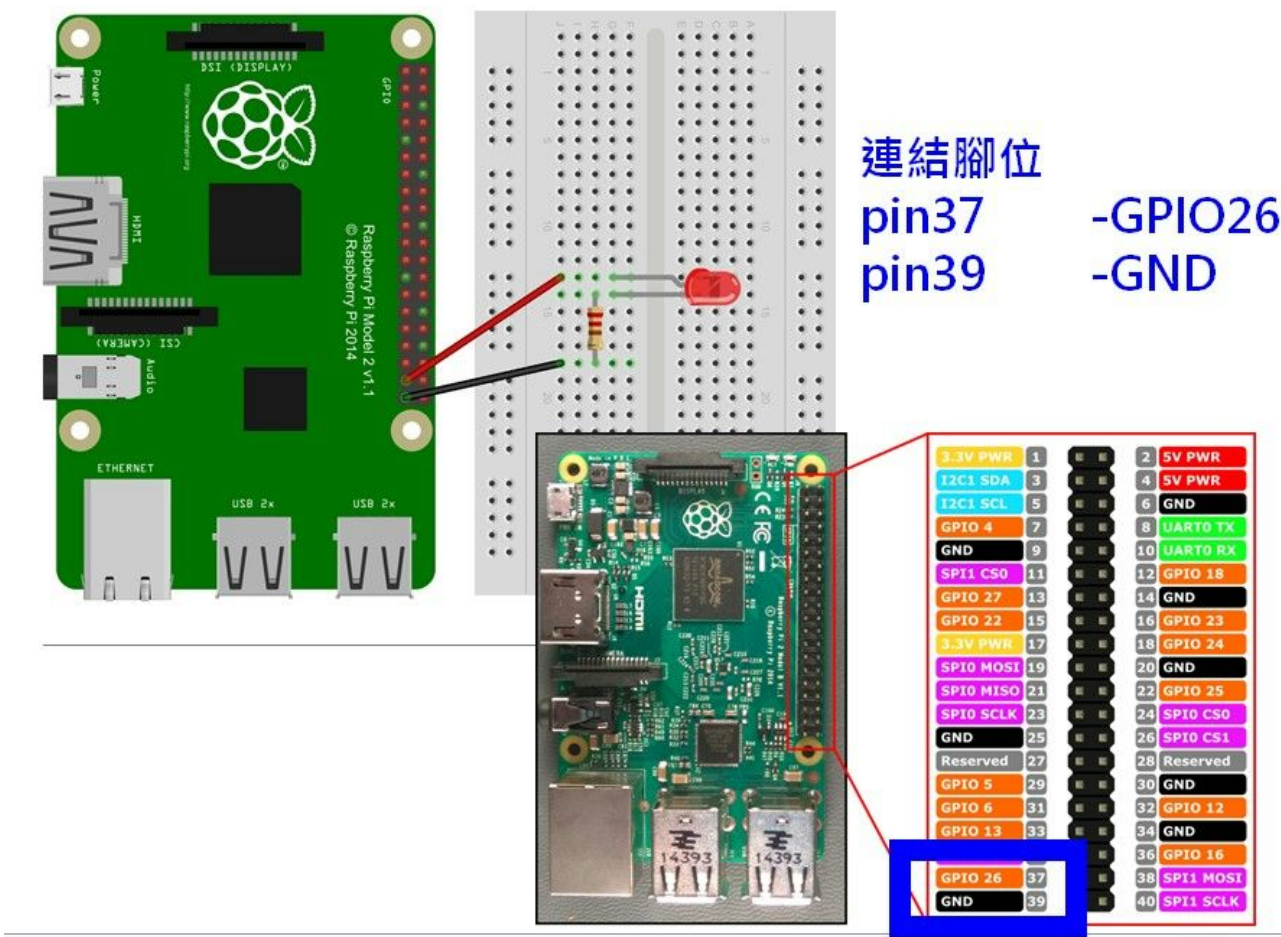


圖4：LED燈閃爍專案-電路圖

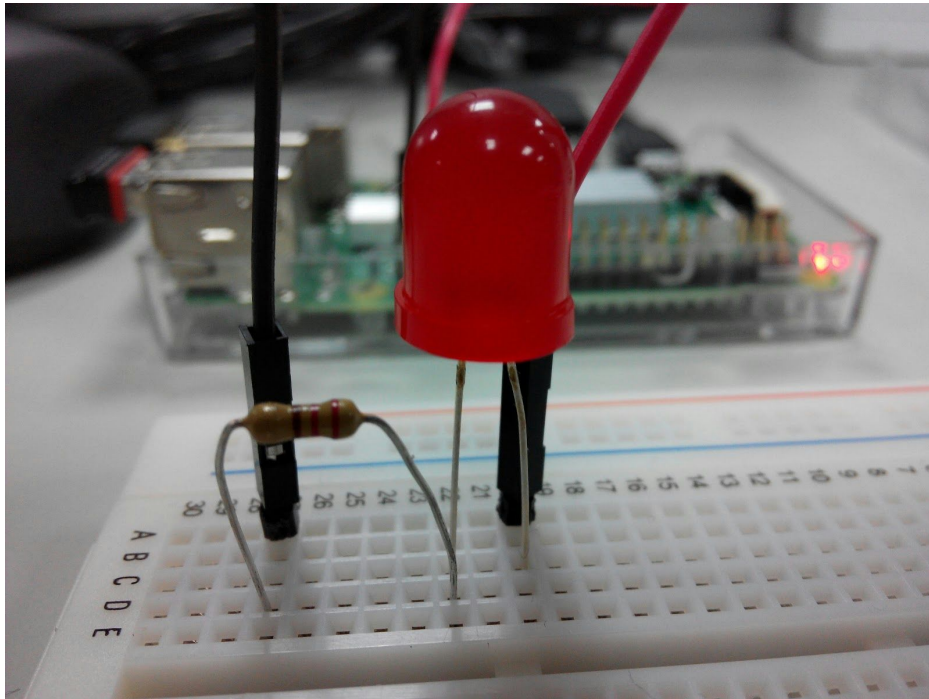


圖5：LED燈閃爍專案-實際接線情形



將樹莓派接上LED後，接下來打開『Blinky』專案。我們要編輯專案並加入一點功能：

檔案目錄：『[../home/pi/Document/IoT\\_Project/Linux-Raspbian/Blinky/Blinky.py](#)』

函式名稱：

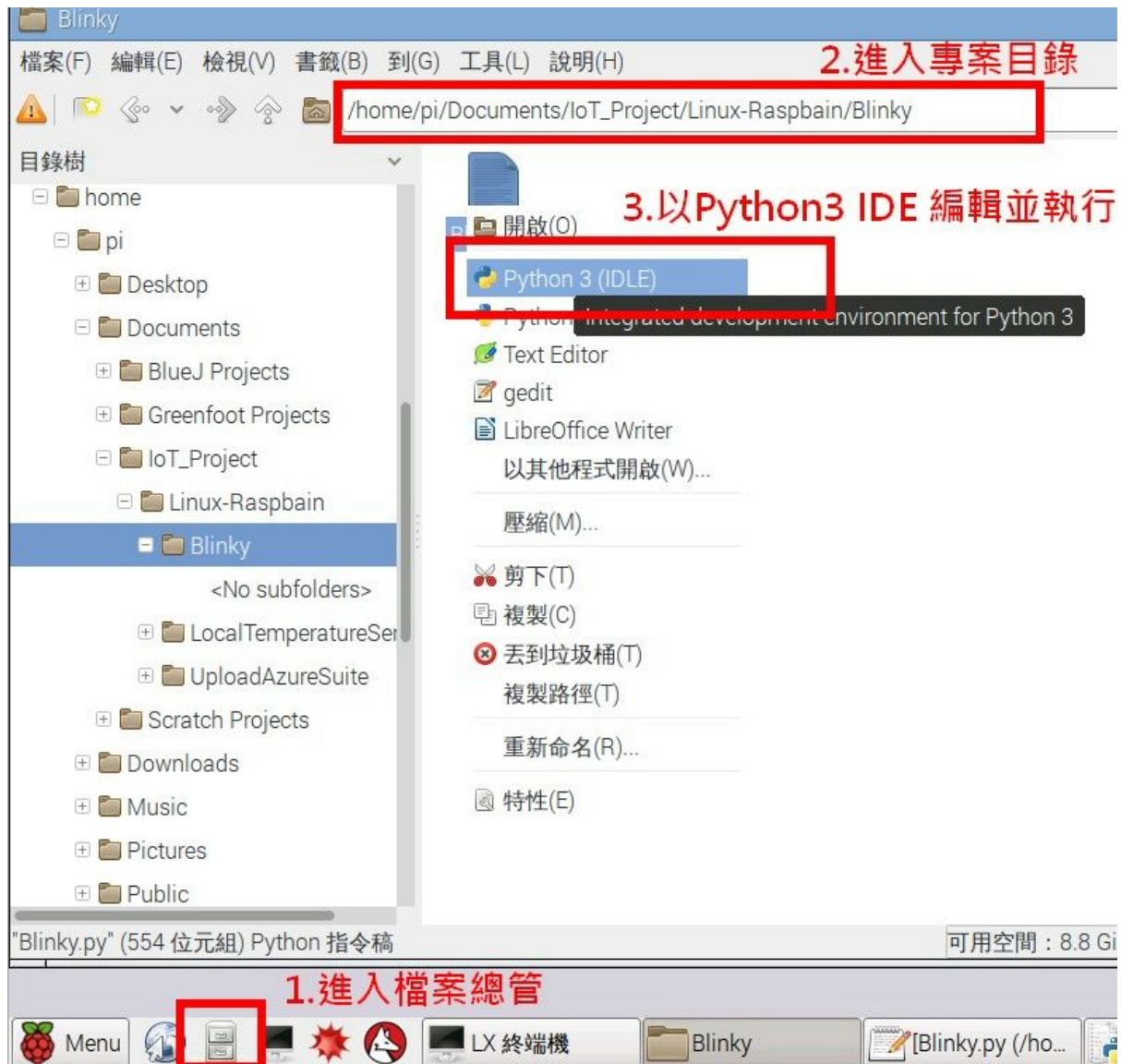


圖6：於樹莓派編輯程式碼說明

請於程式碼中將註解打開 (將#字去除) :

```
1 import time
2 import RPi.GPIO as GPIO    # GPIO
3
4 # GPIO Config
5 GPIO_PIN = 26    # GPIO Pin
6 GPIO.setmode(GPIO.BCM)
7 GPIO.setup(GPIO_PIN, GPIO.OUT)
8
9 # Blink LED
10 IS_LED_LIGHT = False
11
12 while True:
13     # Toggle LED
14     IS_LED_LIGHT = not IS_LED_LIGHT
15     if IS_LED_LIGHT:
16         print("Light")
17         GPIO.output(GPIO_PIN, GPIO.HIGH)
18     else:
19         print("Dark")
20         GPIO.output(GPIO_PIN, GPIO.LOW)
21
22     time.sleep(0.5)    # 0.5 seconds
```

上面這段程式碼主要是告知樹莓派，我們是以『何種協定』以及透過『哪個腳位』去和LED燈溝通。初始化這些訊息後，我們就可以藉由『**while**』 + 『**time.sleep**』函式來實現閃爍的功能。

程式碼編輯完成後，以Python3 IDE啟動專案，如下圖所示。便可看到專案的執行情況，同時也可看到各位的LED燈閃爍：



圖7：啟動專案

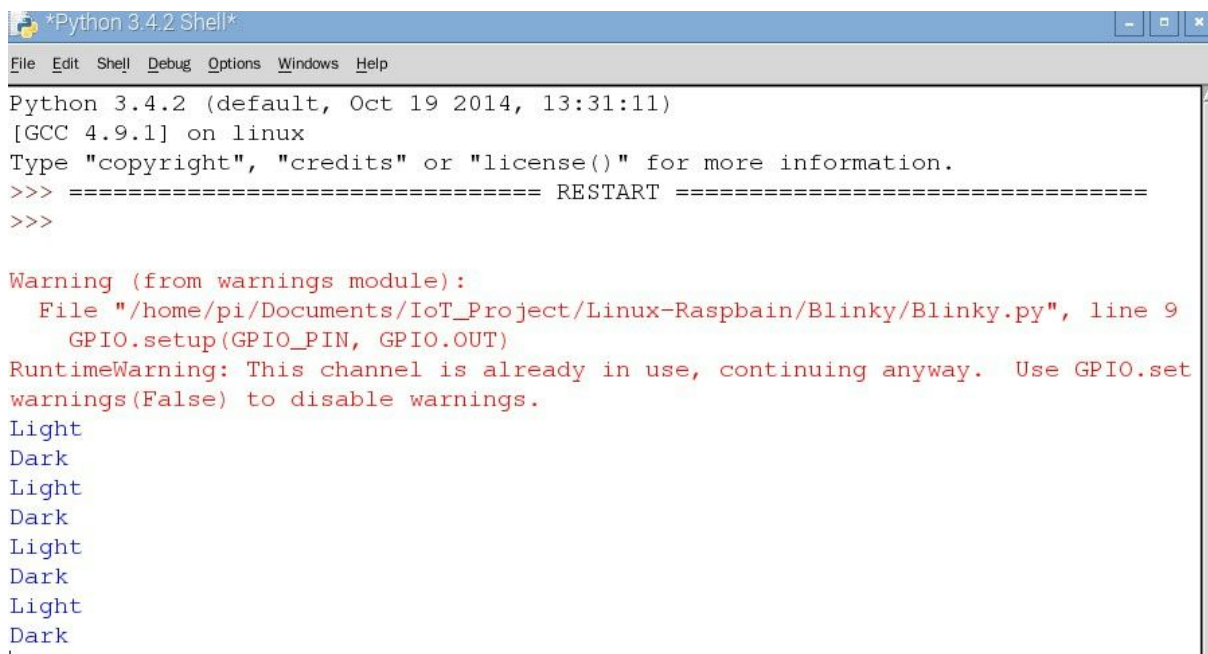


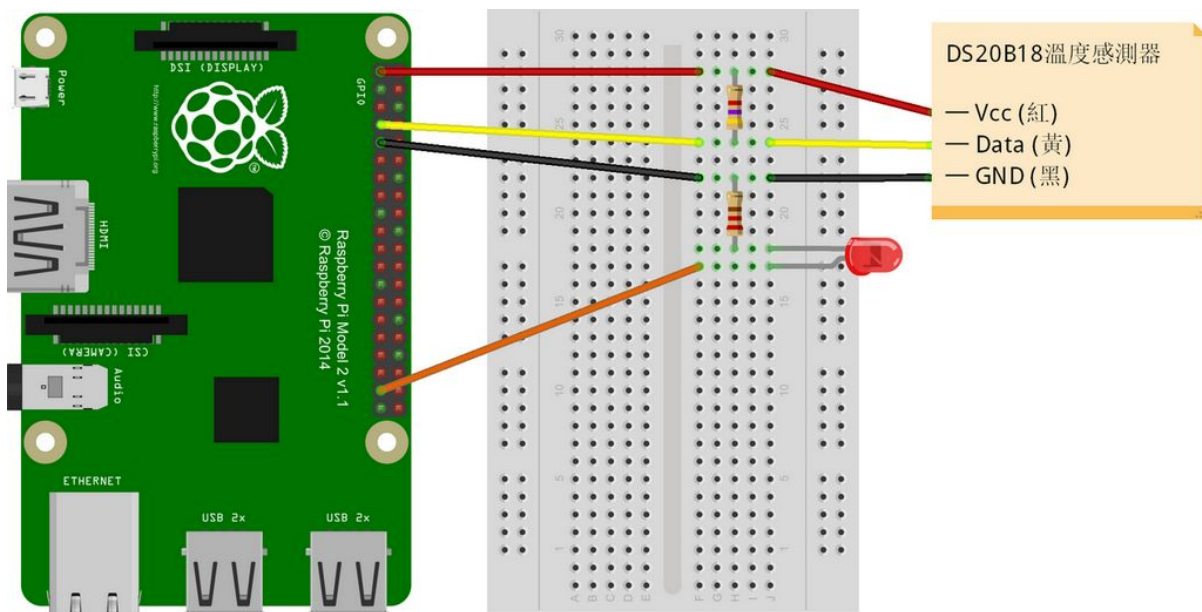
圖8：Blinky專案啟動情形



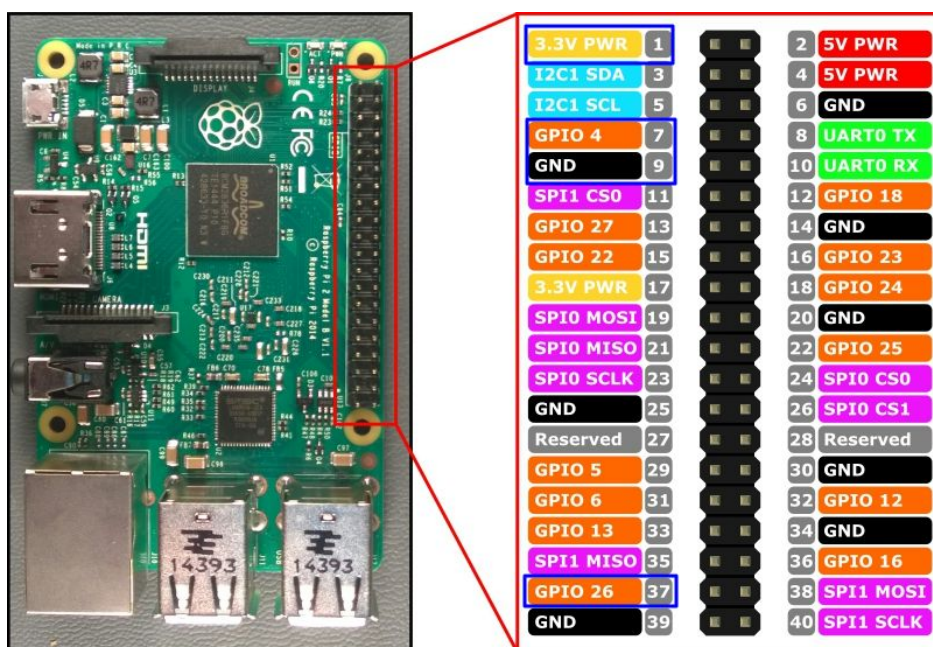
# 透過樹莓派看看環境溫度吧！

為了要讀取環境中的溫度資訊，我們需要將溫度感測器以及警示用的LED燈連結到樹莓派上。請各位將實驗器材依照下列圖示接起來。

實驗器材	數量(qts)
樹莓派2	1
LED燈	1
220 Ω 電阻 (色碼:紅紅棕)	1
DS18B20溫度感測器	1
4.7K Ω 電阻 (色碼:黃紫紅)	1



fritzing



連接腳位

Pin1 -3.3V  
Pin7 -GPIO4  
Pin9 -GND  
Pin37 -GPIO26

圖9：感測器連結示意圖

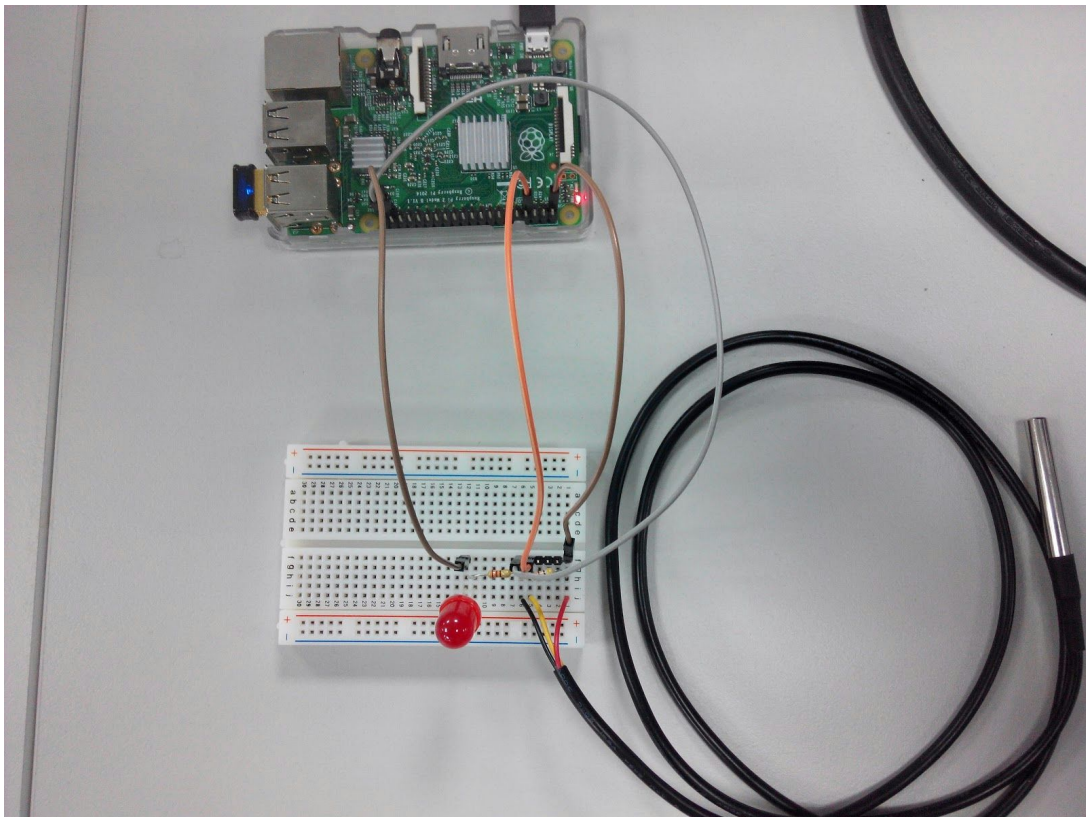
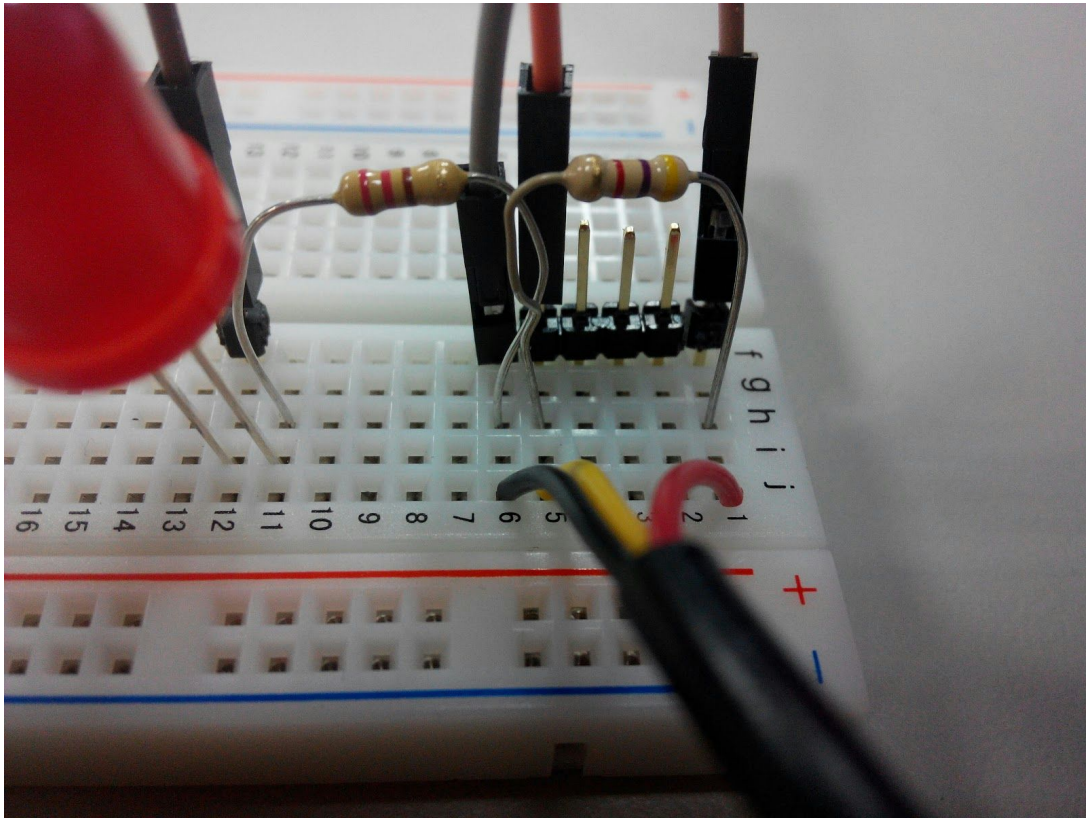


圖10：實際接線情形



將樹莓派接上溫度感測器後，下一步當然是看看樹莓派能不能讀到溫度資訊！請打開『LocalTemperatureSensor』專案。我們要編輯專案並加入一點功能：

檔案目錄：

『  
[/home/pi/Documents/loT\\_Project/Linux-Raspbain/LocalTemperatureSensor/Temperature.py](/home/pi/Documents/loT_Project/Linux-Raspbain/LocalTemperatureSensor/Temperature.py)』

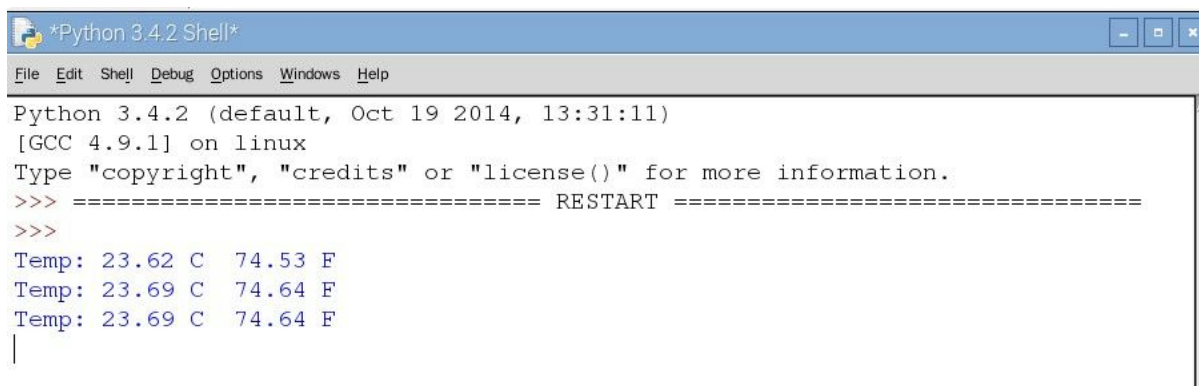
函式名稱：

請各位於函式對應的註解處，將下列的程式碼之註解打開。

```
1  #*****WorkShop-1*****#
2
3  # Start
4  while True:
5      read_temperature()
6      time.sleep(1)    # 1 seconds
7
8  #*****#
```

這段程式碼分別代表了我們要以何種頻率抓取感測器資料，以及實際抓取溫度資料的邏輯。各位若對抓取溫度資料的方式有興趣，可參考：[Adafruit's Raspberry Pi Lesson 11. DS18B20 Temperature Sensing](#)

修改完成後請執行專案，樹莓派就會將由溫度感測器接受到的資訊顯示出來。



The screenshot shows a terminal window titled '\*Python 3.4.2 Shell\*'. The window has a menu bar with 'File', 'Edit', 'Shell', 'Debug', 'Options', 'Windows', and 'Help'. The terminal output shows the Python version (3.4.2), GCC version (4.9.1), and a prompt to type 'copyright', 'credits', or 'license()' for more information. After pressing Enter, a separator line '==== RESTART ====' appears, followed by three lines of temperature readings: 'Temp: 23.62 C 74.53 F', 'Temp: 23.69 C 74.64 F', and 'Temp: 23.69 C 74.64 F'. The cursor is at the end of the last line.

圖11：成功讀取溫度感測器資訊

接下來我們想要設定溫度門檻，當讀取的本地溫度超過此門檻時便點亮警示燈：

檔案目錄：

『  
[/home/pi/Documents/loT\\_Project/Linux-Raspbain/LocalTemperatureSensor/Temperature.py](#)』

函式名稱：*is\_over\_heat(temp\_c)*

請各位於函式對應的註解處，將下列的程式碼之註解打開。

```
1  #####WorkShop-2#####
2
3  # Condition
4  condition = temp_c > 25
5
6  # Toggle LED
7  if condition:
8      GPIO.output(GPIO_PIN, GPIO.HIGH)
9  else:
10     GPIO.output(GPIO_PIN, GPIO.LOW)
11
12 #####
```

如註解所示，這段程式碼代表若感測器所接收的溫度超過門檻，樹莓派便會點亮LED燈！



編輯完成後請各位重新執行專案一次，並觀察感測器溫度在高於你所設定的門檻值時，LED燈是不是亮起來了？



圖12：高於溫度門檻值點亮警示燈

若一切順利，那麼到了這裡，我們就完成本地所有的裝置設定了！

## 下一步，將資料送上雲端，由Azure Web檢視

若要對讀出的資料做更廣泛的應用，或者說要實現『物聯網』的概念，那麼下一步就是將這些資料傳輸到雲端上。微軟提供了一套便捷的系統--『Azure IoT Suite - Remote Monitoring System』。讓我們可以很方便的將本地資料上傳，並藉由預設的看板做即時監控。

請各位先連到我們架設的[Azure IoT Suite Website](#)，你們將看到許多的控制選項，但別慌張，我們會一步一步來！首先要在Server上新增一個對應您樹莓派的Device，讓各位做裝置的資料上傳，檢視...等動作。請依下列指示操作：

Microsoft Azure IoT Suite ① Remote Monitoring Solution

DASHBOARD  
DEVICES  
RULES  
ACTIONS  
ADVANCED  
+  
ADD A DEVICE

1.

World map showing locations in North America, Europe, Asia, Africa, South America, and Australia. Scale: 2000 miles, 2500 km. © 2016 HERE Earthstar Geographics SIO © 2016 Microsoft Corporation

Alarm History

LOCAL TIME	DEVICE ID	RULE OUTPUT	VALUE
2016年3月2日 下午3點20分34秒	tdd-1	AlarmTemp	25.25
2016年3月2日 下午3點20分32秒	tdd-1	AlarmTemp	25.69

Microsoft Azure IoT Suite

Remote Monitoring Solution

HP-PC\HP  
IMPLICIT READ-C

DASHBOARD

DEVICES

RULES

ACTIONS

ADVANCED

2.

+

ADD A DEVICE

Device List (7)

STATUS	DEVICE ID	MANUFACTURER	MODEL NUMBER	SERIAL NUMBER	FIRMWARE	PLATFORM	PROCESSOR	INSTALL
Running	SampleDevice001_25	Raspberry Pi Org	Raspberry Pi 2 B+	A9090	10.0586	Windows10 IoT Core	ARM	
Running	SampleDevice002_25	Raspberry Pi Org	Raspberry Pi 2 B+	A9090	10.0586	Windows10 IoT Core	ARM	
Running	SampleDevice003_25	Raspberry Pi Org	Raspberry Pi 2 B+	A9090	10.0586	Windows10 IoT Core	ARM	
Running	SampleDevice004_25	Raspberry Pi Org	Raspberry Pi 2 B+	A9090	10.0586	Windows10 IoT Core	ARM	
Running	dht_tw_235	655資訊	Raspberry Pi 2 B+	0521	10.0586	Windows10 IoT Core	ARM	
Running	tdd-1	Raspberry Pi Org	Raspberry Pi 2 B+	A9090	10.0586	Windows10 IoT Core	ARM	
Running	tdd-2	Raspberry Pi Org	Raspberry Pi 2 B+	A9090	10.0586	Windows10 IoT Core	ARM	

Microsoft Azure IoT Suite

Remote Monitoring Solution

HP-PC\HP  
IMPLICIT READ-C

DASHBOARD

DEVICES

RULES

ACTIONS

ADVANCED

ADD A DEVICE

STEP 1 of 3

3.

Simulated Device

Software to simulate a device. Easily extensible for arbitrary events and commands; can run in a Windows Azure worker role. To create a simulated device, please follow the cooler sample instructions.

Add New

Custom Device

A physical hardware device.

Add New

Microsoft Azure IoT Suite

Remote Monitoring Solution

DASHBOARD

DEVICES

RULES

ACTIONS

ADVANCED

+

ADD A DEVICE

←

ADD A CUSTOM DEVICE

STEP 2 of 3

How would you like to define the Device ID?  
(DeviceID is case-sensitive)

4.

☐ Generate a Device ID for me

☒ Let me define my own Device ID

MyDevice-1

Check ID

✓ Device ID is available

請輸入各組預設的裝置名稱

☐ Attach a SIM ICCID to the device

Create

Cancel

Microsoft Azure IoT Suite

Remote Monitoring Solution

HP-PC\HP

IMPLICIT READ-ONLY

Sign Out

DASHBOARD

DEVICES

RULES

ACTIONS

ADVANCED

+

ADD A DEVICE

5.

ADD A CUSTOM DEVICE

STEP 3 of 3

Copy credentials into the configuration file on the device

Device ID: MyDevice-1

IoT Hub Hostname: gsstdhub.azure-devices.net

Device Key: C2K9IQE0ajiESPe/M9u9RA==

Control+C to Copy

請將這三組資料複製下來，  
後續將用於驗證

Done

Instructions for your Custom Device (opens in new tab)

Microsoft Azure IoT Suite

Remote Monitoring Solution

DASHBOARD

DEVICES

RULES

ACTIONS

ADVANCED

Device List (8)

Device 尚未啟動，狀態仍為pending

STATUS	DEVICE ID	MANUFACTURER	MODEL NUMBER	SERIAL NUMBER	FIRMWARE	PLATFORM
<input type="radio"/> Pending	MyDevice-1					
<input checked="" type="radio"/> Running	SampleDevice001_25	Raspberry Pi Org	Raspberry Pi 2 B+	A9090	10.0586	Windows10 IoT Core
<input checked="" type="radio"/> Running	SampleDevice002_25	Raspberry Pi Org	Raspberry Pi 2 B+	A9090	10.0586	Windows10 IoT Core
<input checked="" type="radio"/> Running	SampleDevice003_25	Raspberry Pi Org	Raspberry Pi 2 B+	A9090	10.0586	Windows10 IoT Core
<input checked="" type="radio"/> Running	SampleDevice004_25	Raspberry Pi Org	Raspberry Pi 2 B+	A9090	10.0586	Windows10 IoT Core
<input checked="" type="radio"/> Running	dht_tw_235	655資訊	Raspberry Pi 2 B+	0521	10.0586	Windows10 IoT Core

圖13-18：於Azure IoT Suite新增裝置

新增裝置的過程並不複雜，各位依照網站上的指示應該可以順利完成。提醒各位的是要將步驟5中新增的裝置資料複製起來，稍後需把這些資料帶入程式碼中。另外由於還未對新增的裝置進行任何操作，所以各位點選『Devices』檢視新增的裝置時，狀態顯示為『pending』是正常的。

接著要提供程式一些資訊：包含裝置的驗證碼，ID，以及裝置的一些基本屬性..等。請於下列目錄設定驗證資料：

檔案目錄：

『  
[/home/pi/Documents/loT\\_Project/Linux-Raspbain/UploadAzureSuite/PhysicalDevice.py](#)  
 』

函式名稱：

1	#*****WorkShop1*****#
2	# Azure IoT Hub Setting
3	IOT_HUB = "IOT Hub Name"; # Note : Delete ".azure-devices.net"
4	DEVICE_ID = "Your Device ID";
5	DEVICE_KEY = "Your Device Key";

再加入裝置的經緯度資訊：

檔案目錄：

『  
[/home/pi/Documents/loT\\_Project/Linux-Raspbain/UploadAzureSuite/PhysicalDevice.py](#)  
 』

函式名稱：*update\_device\_info()*



```

1  #####WorkShop2#####
2  # Create Message
3  json_object = {
4      'DeviceProperties': {
5          'DeviceID': DEVICE_ID,
6          'HubEnabledState': 1,
7          'DeviceState': "normal",
8          'UpdatedTime': datetime.datetime.now().strftime("%Y-%m-%d
%H:%M:%S"),
9          'Manufacturer': "Raspberry Pi Org",
10         'ModelNumber': "Raspberry Pi 2 B+",
11         'SerialNumber': "A9090",
12         'FirmwareVersion': "8.0",
13         'Platform': "Raspbian",
14         'Processor': "ARM",
15         'Latitude': 23.583234,    # Latitude
16         'Longitude': 120.5825975 # Longitude
17     },
18     'Commands': [],
19     'IsSimulatedDevice': 1,
20     'ObjectType': "DeviceInfo",
21     'Version': "1.0"
22 }

```

裝置的經緯度(Longitude、Latitude)屬性，請各位依據您裝置的代號填入。

經緯度資訊如下表：

裝置代號(台北)	經度(Longitude)	緯度(Latitude)
漢堡	9.6	53.3
羅馬	12.3	41.5
台北	121.3	25.0
曼谷	100.3	13.5
開普敦	18.3	-35.6

裝置代號(高雄)	經度(Longitude)	緯度(Latitude)
哥本哈根	12.4	55.4
柏林	13.3	52.3
日內瓦	6.1	46.1
北京	116.2	39.6
平壤	125.5	39.0
雅典	23.4	37.6
東京	139.5	35.4
香港	115.1	21.2

接著加入將本地溫度資料上傳至雲端的程式碼：

專案目錄：

『  
[/home/pi/Documents/IoT\\_Project/Linux-Raspbain/UploadAzureSuite/PhysicalDevice.py](#)  
 』

函式名稱：*send\_temperature\_data(temp\_c, temp\_f)*

```

1  #*****WorkShop3*****#
2  # Create Azure Device Client
3  device = DeviceClient.DeviceClient(IOT_HUB, DEVICE_ID, DEVICE_KEY)
4  device.create_sas(600)
5
6  # Create Message
7  json_object = {
8      'DeviceId': DEVICE_ID,
9      'Temperature': temp_c,
10     'Humidity': 0,
11     'ExternalTemperature': temp_f
12 }
13
14 # Send Temperature Data To Azure IoT Hub
15 message = json.dumps(json_object)
16 message_bytes = message.encode(encoding='UTF-8')
17 response = device.send(message_bytes)
18 return

```

執行專案後，我們再回到[Azure IoT Suite Website](#)。Dashboard中可以看到裝置已被新增至地圖中，而右側的折線圖代表的是即時上傳的溫度資訊。點擊『Devices』也可看到有關裝置的各項資訊已經被更新。這代表我們已經成功將本地資訊上傳至雲端，同時也能利用看板做即時監控了！

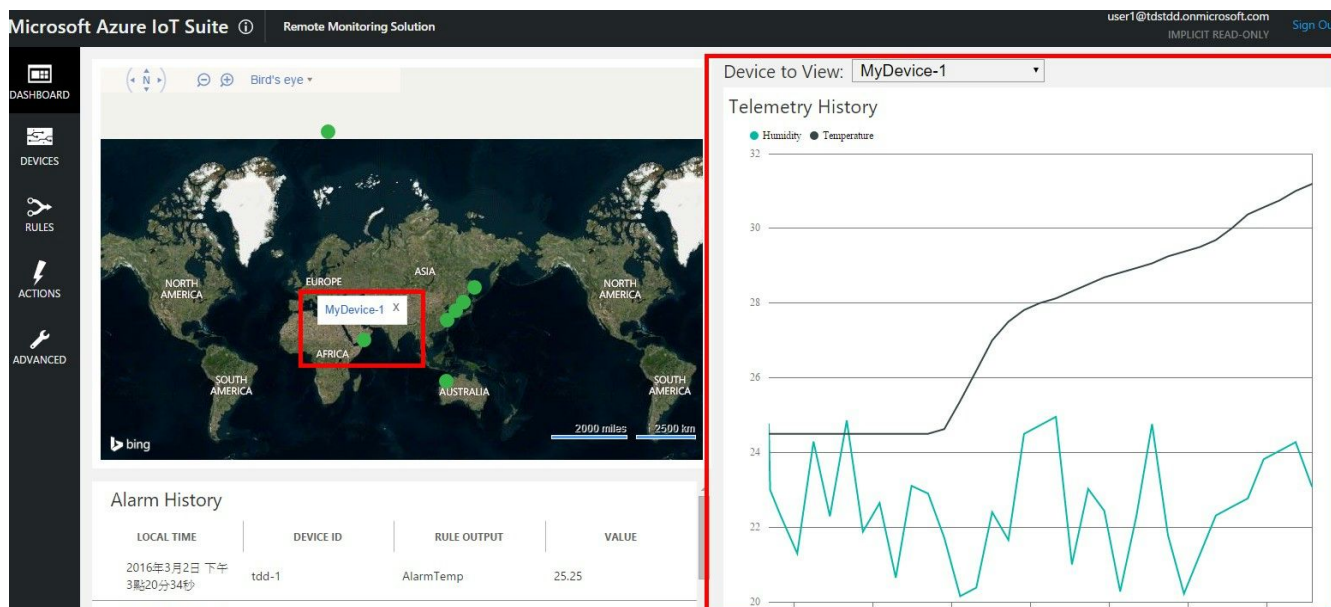


圖19：Dashboard顯示即時溫度數據

The screenshot shows the 'Device List' page in the Microsoft Azure IoT Suite. It displays a table with 8 devices. The first device, 'MyDevice-1', is highlighted with a red box. To the right of the table, the 'Device Properties' panel for 'MyDevice-1' is also highlighted with a red box, showing various attributes like Hostname, HubEnabledState, DeviceState, and Manufacturer.

STATUS	DEVICE ID	MANUFACTURER	MODEL NUMBER	SERIAL NUMBER	FIRMWARE	PLATFORM	PROCESSOR	INSTALLED RAM
Running	MyDevice-1	Raspberry Pi Org	Raspberry Pi 2 B+	A9090	10.0586	Windows10 IoT Core	ARM	
Running	SampleDevice001_25	Raspberry Pi Org	Raspberry Pi 2 B+	A9090	10.0586	Windows10 IoT Core	ARM	
Running	SampleDevice002_25	Raspberry Pi Org	Raspberry Pi 2 B+	A9090	10.0586	Windows10 IoT Core	ARM	
Running	SampleDevice003_25	Raspberry Pi Org	Raspberry Pi 2 B+	A9090	10.0586	Windows10 IoT Core	ARM	
Running	SampleDevice004_25	Raspberry Pi Org	Raspberry Pi 2 B+	A9090	10.0586	Windows10 IoT Core	ARM	
Running	dht_tw_235	655資訊	Raspberry Pi 2 B+	0521	10.0586	Windows10 IoT Core	ARM	
Running	tdd-1	Raspberry Pi Org	Raspberry Pi 2 B+	A9090	10.0586	Windows10 IoT Core	ARM	
Running	tdd-2	Raspberry Pi Org	Raspberry Pi 2 B+	A9090	10.0586	Windows10 IoT Core	ARM	

**Device Properties for MyDevice-1:**

- DEVICEID: MyDevice-1
- HOSTNAME: gssdtdhub.azure-devices.net
- HUBENABLEDSTATE: True
- DEVICESTATE: normal
- UPDATETIME (UTC): 03/03/2016 06:11:48
- CREATETIME (UTC): 03/03/2016 02:09:31
- MANUFACTURER: Raspberry Pi Org
- MODELNUMBER: Raspberry Pi 2 B+
- SERIALNUMBER: A9090
- FIRMWAREVERSION: 10.0586
- PLATFORM: Windows10 IoT Core

圖20：裝置資訊已更新

## 參考資料

1. [Adafruit's Raspberry Pi Lesson 11. DS18B20 Temperature Sensing](#)
2. [AzureIoTDeviceClientPY](#)
3. [\[IOT\]Raspberry Pi 2 /Windows IOT Core 將讀取到的溫、濕度資訊送到 Azure IoT Suite - Remote Monitoring System](#)