

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
筆記型電腦 1.作業系統： Windows 10 (version 10.0.10240 以上) 2.Visual Studio 2015 update 1	1	請將環境設定為： 開發人員模式
樹莓派 2 model B+	1	With Windows IOT Core
LED燈	1	
4.7KΩ 電阻	1	色碼：黃紫紅 (詳見： 電阻色碼參考表)
220Ω 電阻	1	色碼：紅紅棕 (詳見： 電阻色碼參考表)
溫度感測器-DS18B20	1	
整流二極體1N4004	1	
無線網路卡 Tp-Link-TL_WN725N	1	

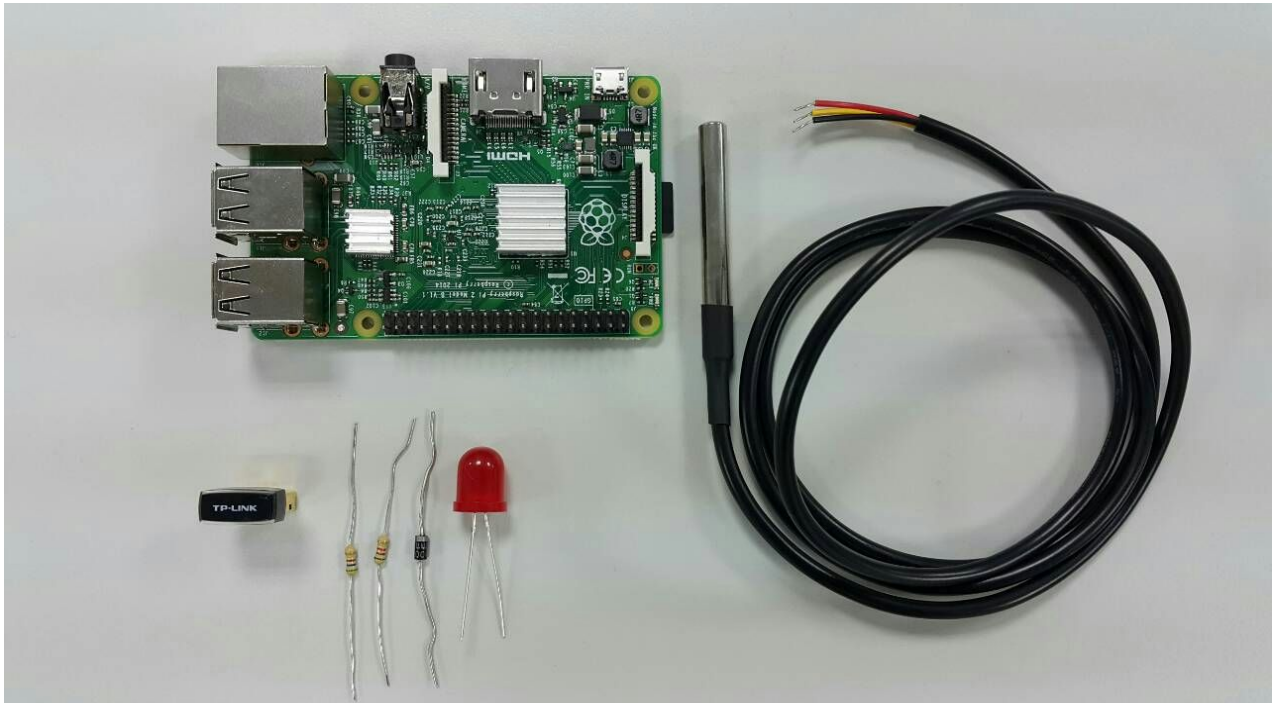


圖1：實驗所需材料



圖2：開發人員模式設定

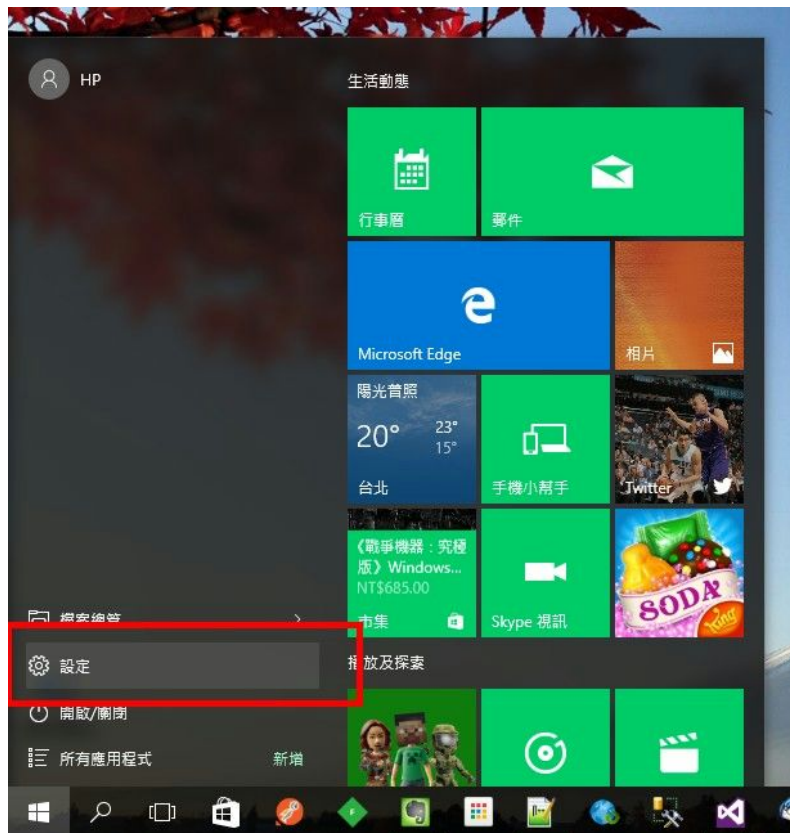


圖3：如何查詢筆電作業系統版本

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。本實驗中樹梅派作業系統為Windows Iot Core，開機後應可看到下列畫面：



圖4：Windows Iot Core 開機畫面

請將無線網路卡接上USB，並點選右上角齒輪設定連結無線網路。而回到主畫面後可看到樹莓派的IP Address：



圖5：連結無線網路

也請將筆電與樹莓派連結到同一網域之無線網路：



圖6：佈署專案前需與樹莓派於連接同網域之無線網路

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

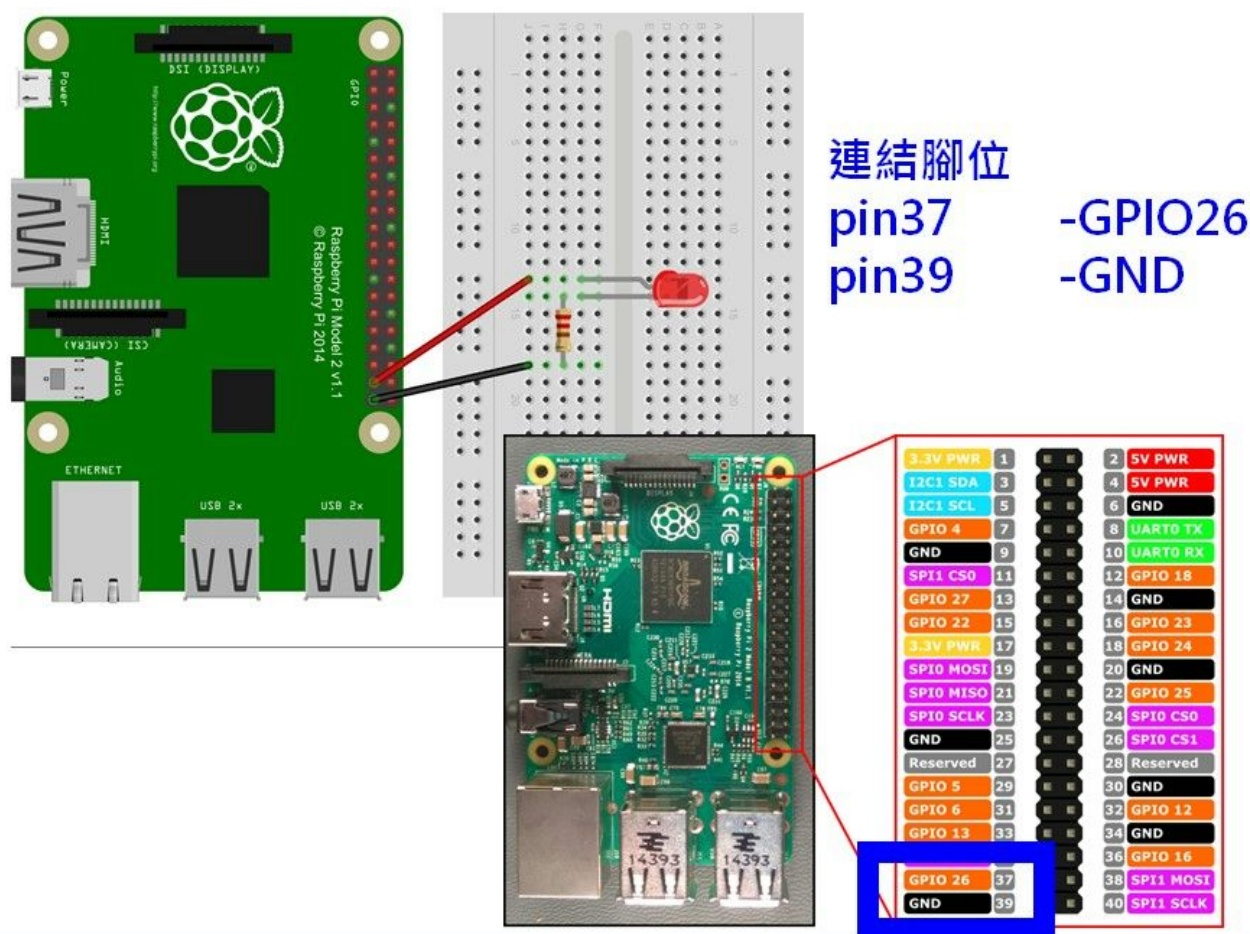


圖7：LED燈閃爍專案-電路圖

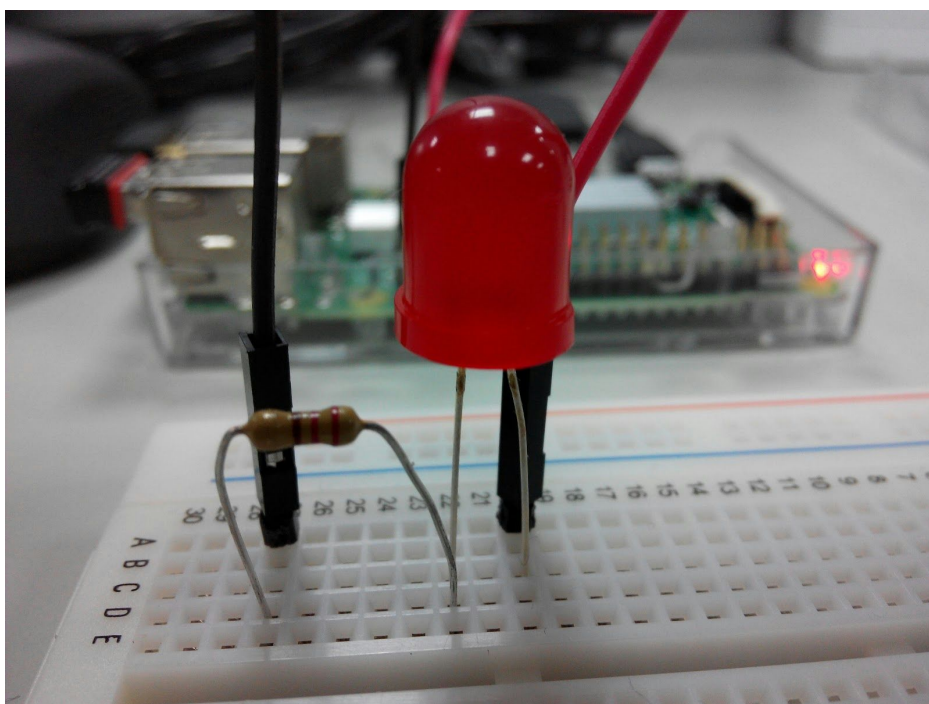


圖8：LED燈閃爍專案-實際接線情形

將樹莓派接上LED後，請由[GitLab](#)下載範例專案至筆電。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：『[..IOT_Workshops\Blinky\MainPage.xaml.cs](#)』

函式名稱：[ClassProperty](#)、[MainPage\(\)](#)

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
3 private const int LED_PIN = 26;    // 輸出之GPIO腳位
4
5 //*****//
```

```
1 //*****WorkShop*****//
2
3 // 初始化GPIO 控制器
4 var gpio = GpioController.Default();
5
6 // 檢查有無正確連接感測器裝置
7 if (gpio == null)
8 {
9     pin = null;
10     GpioStatus.Text = "There is no GPIO controller on this
11 device.";
12 }
13
14 // 取得GPIO對應之實際pin腳位
15 pin = gpio.OpenPin(LED_PIN);
16
17 // 設置該GPIO腳位屬性
18 // 1.為輸出腳位
19 // 2.輸出電壓為High
20 pinValue = GpioPinValue.High;
21 pin.Write(pinValue);
22 pin.SetDriveMode(GpioPinDriveMode.Output);
23
24 GpioStatus.Text = "GPIO pin initialized correctly.";
25
//*****//
```

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，我們就可以藉由計時事件 [Time_Tick\(\)](#) 函式來實現閃爍的功能。

各位編輯好程式碼後，接著請按下列圖示設定佈署選項：

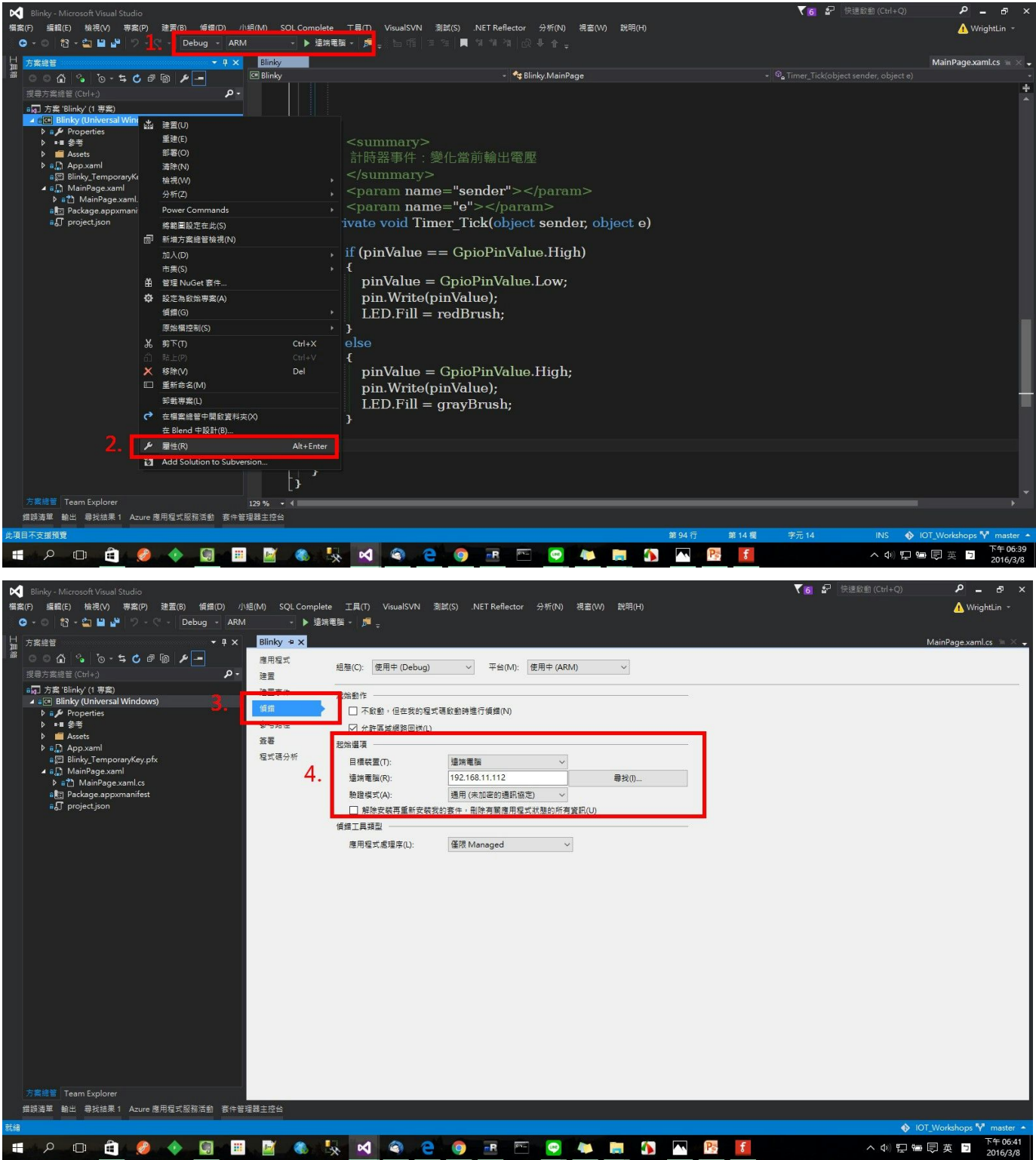


圖9：以VS2015遠端佈署設定說明

佈署選項設定好後，請直接按下啟動鍵。專案便會透過網路佈署至樹莓派中！成功後應該可以看到底下的情況：

※註一：步驟4中的『遠端電腦』欄位請輸入樹莓派的IP位置

※註二：第一次佈署會花個幾分鐘，請各位 Be Patient！

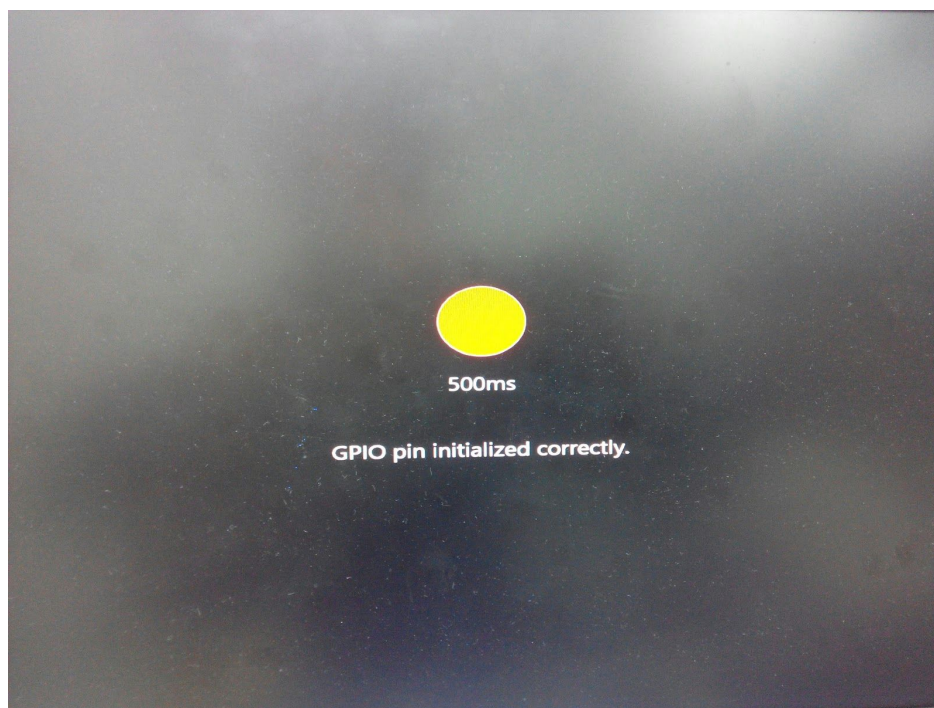


圖10：BLINKY專案佈署執行情形

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
整流二極體	1

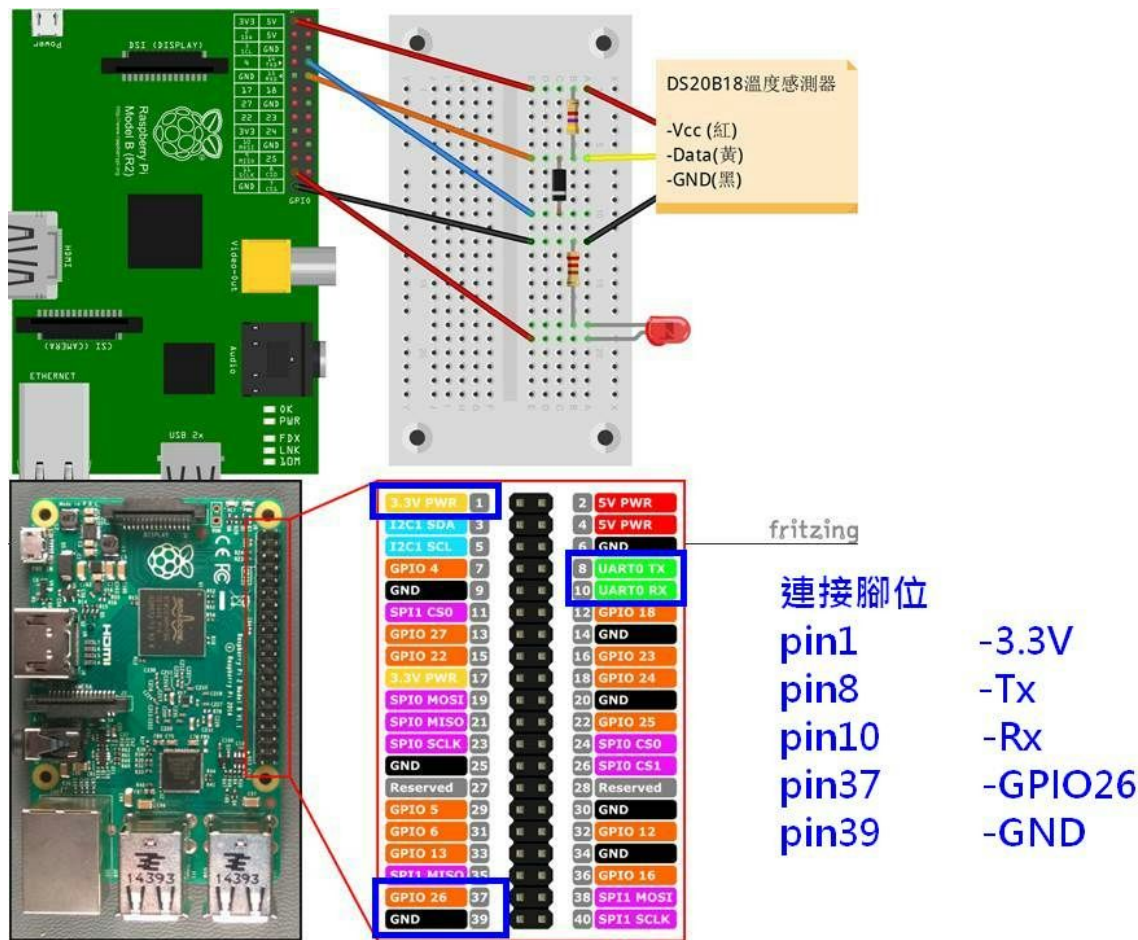


圖11：感測器連結示意圖

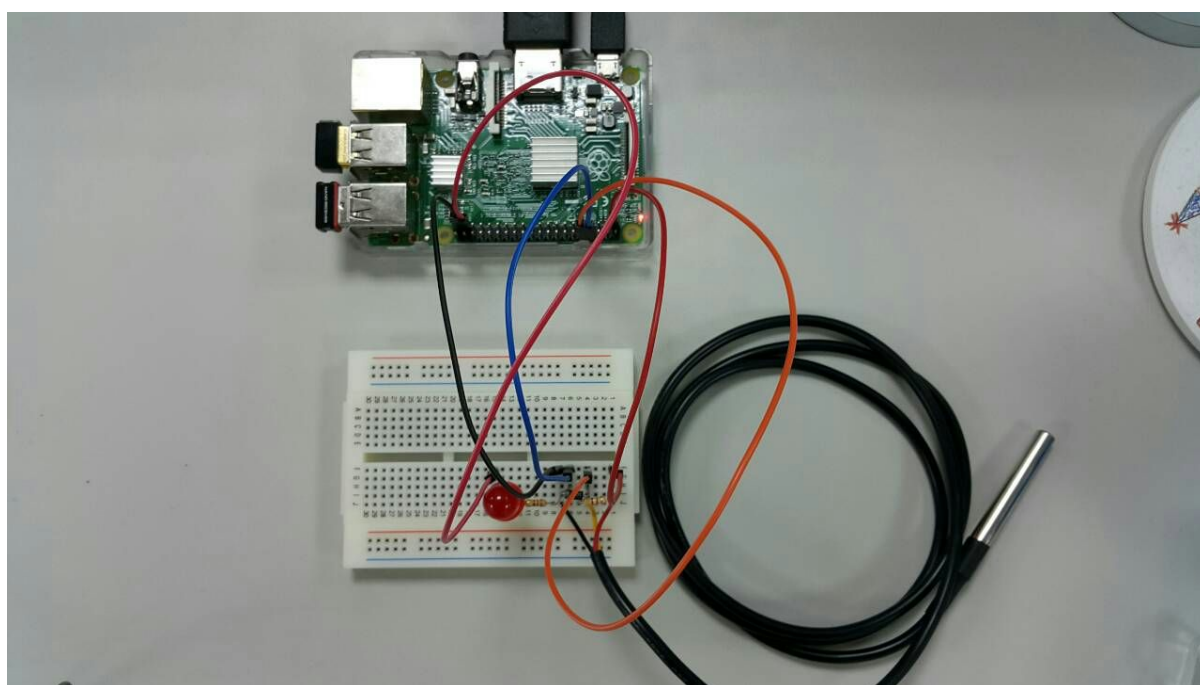
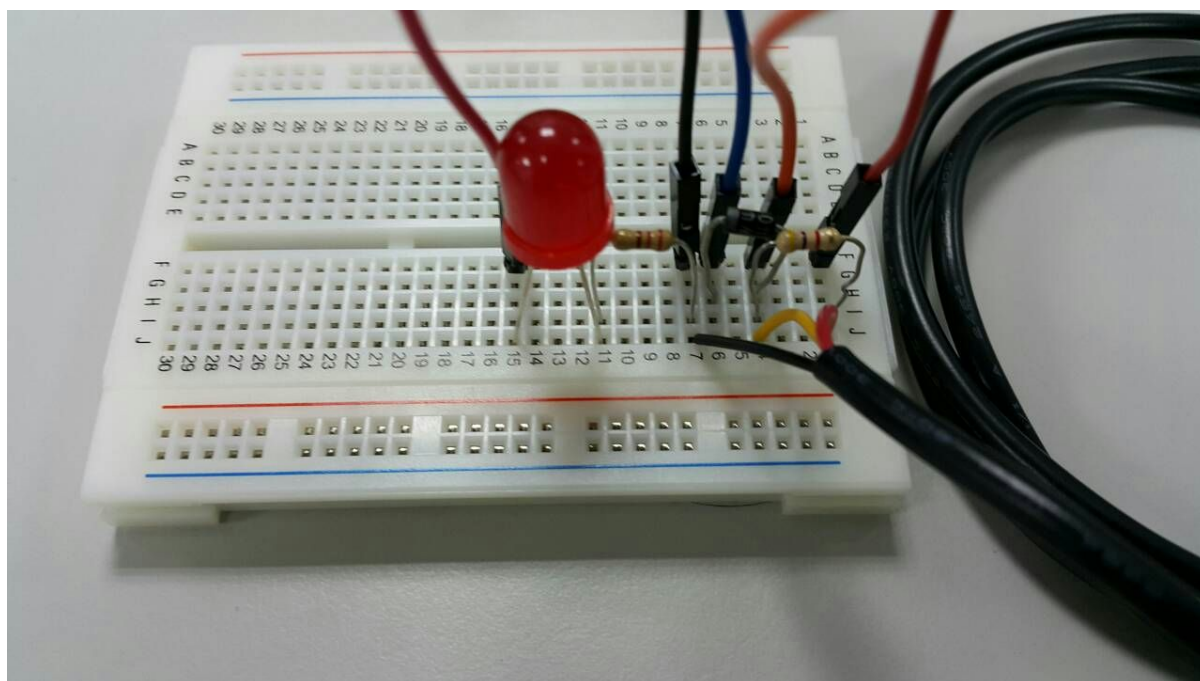


圖12：實際接線情形

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：

『.\\IoT_Workshops\\LocalTemperatureSensor\\DS18B20_1WireBus\\MainPage.xaml.cs』

函式名稱：*ClassProperty*、*Timer_Tick(object sender, object e)*

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 // 取溫度資料的時間間隔
4 timer.Interval = TimeSpan.FromMilliseconds(1200);
5
6 //*****//
```

```
1 //*****WorkShop-1*****//
2
3 // 取得溫度資料
4 tempData.Temperature = await onewire.getTemperature(deviceId);
5
6 //*****//
```

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：[1-Wire DS18B20 Sensor on Windows 10 IoT Core/Raspberry Pi 2](#)

修改完成後，請再次執行專案。看到樹莓派顯示應用程式的畫面後請按下『start』鍵，樹莓派就會將由溫度感測器接受到的資訊顯示出來。

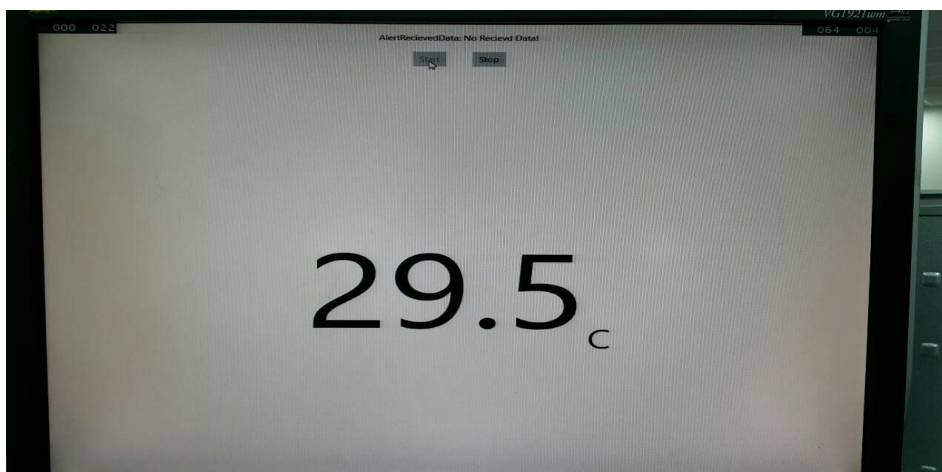


圖13：成功讀取溫度感測器資訊

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：

『[.\\IOT_Workshops\\LocalTemperatureSensor\\DS18B20_1WireBus\\MainPage.xaml.cs](#)』

函式名稱：*ClassProperty*、*Timer_Tick(object sender, object e)*

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 temperatureThreshold = 25.0;    // 溫度門檻值
4 ledPin = 26;                  // 輸出之GPIO腳位
5
6 //*****//
```

```
1 //*****WorkShop-2*****//
2
3 // 溫度高於門檻值，閃爍LED指示燈
4 if (tempData.Temperature > temperatureThreshold)
5 {
6     Blink_LED();
7 }
8
9 //*****//
```

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，樹莓派便會交替的給予高低電壓，使LED燈閃爍！

編輯完成後請各位重新佈署一次，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？

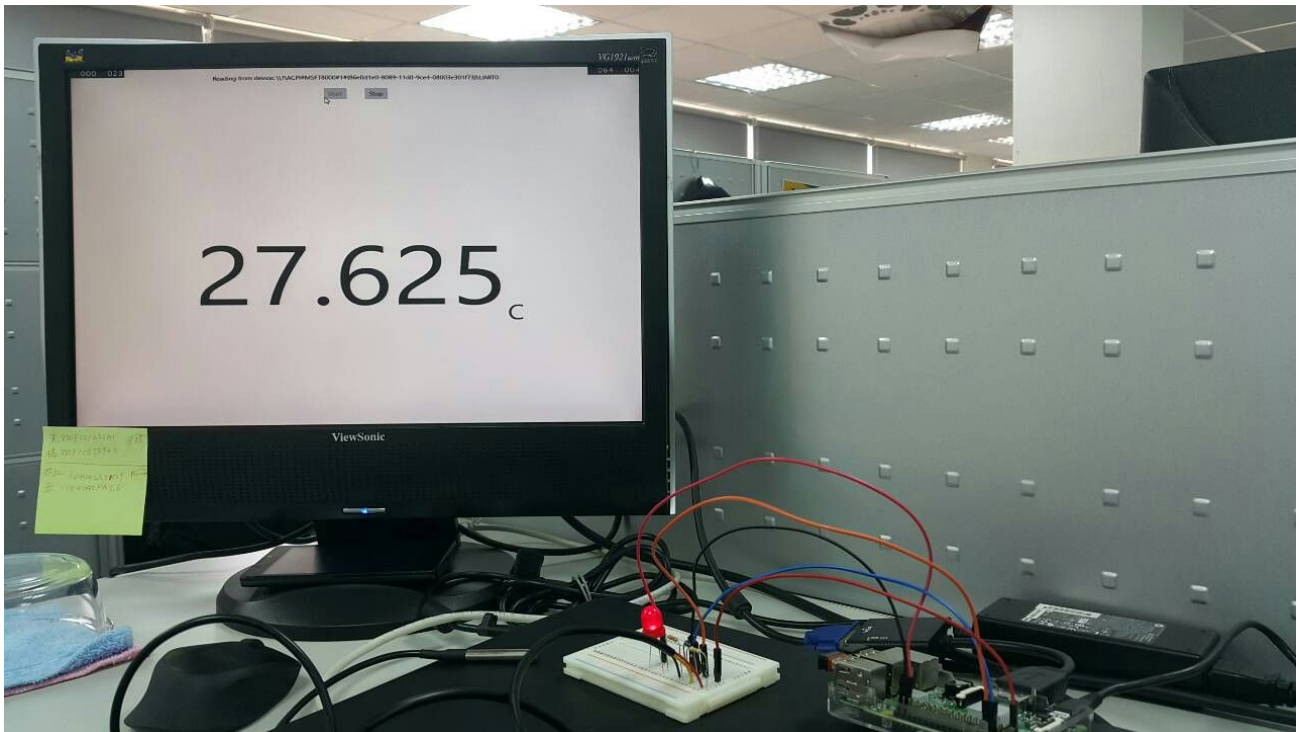


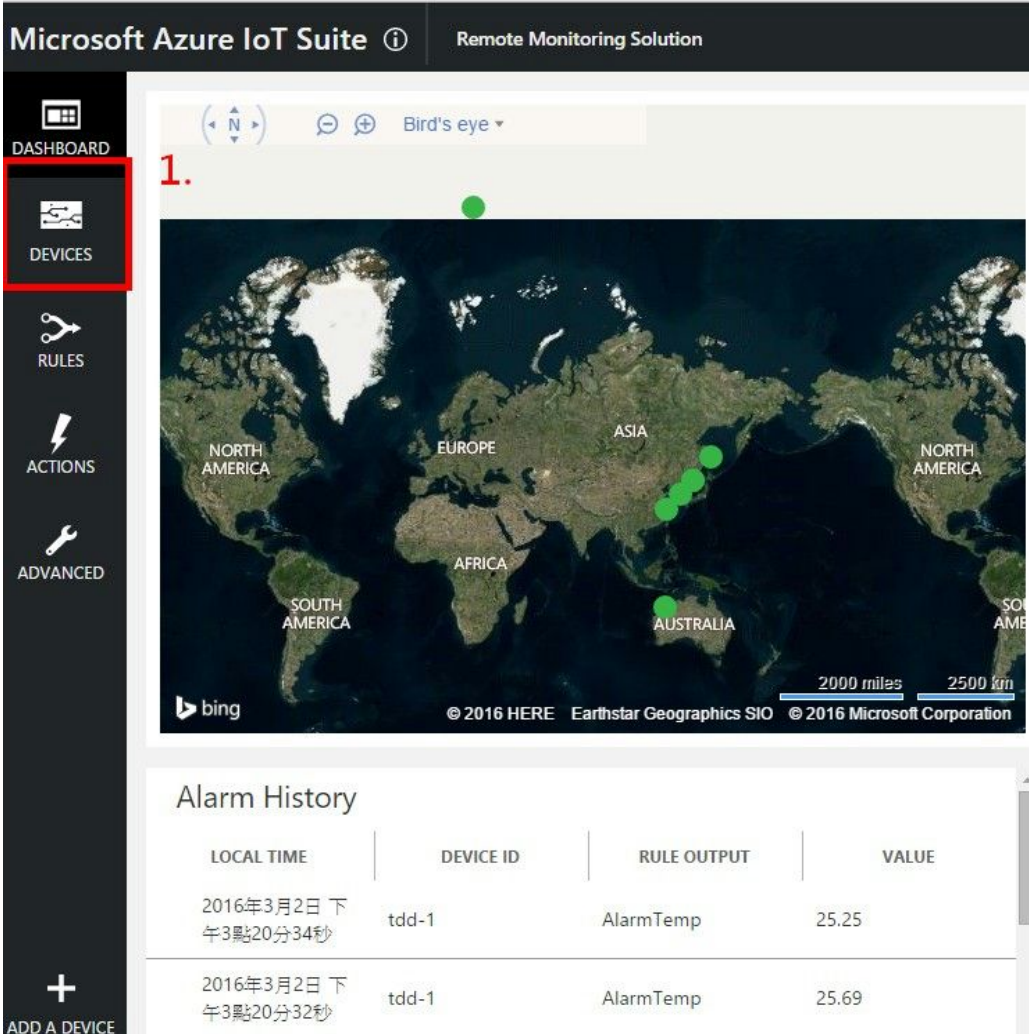
圖14：高於溫度門檻值點亮警示燈

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：



1.

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

+

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	

2.

Microsoft Azure IoT Suite

Remote Monitoring Solution

HP-PC\HP
IMPLICIT READ-C

DASHBOARD

DEVICES

RULES

ACTIONS

ADVANCED

+

ADD A DEVICE

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

← 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

ADD A CUSTOM DEVICE
STEP 3 of 3

5.

Copy credentials into the configuration file on the device

Device ID: MyDevice-1 Copy

IoT Hub Hostname: gsstdhub.azure-devices.net Control+C to Copy

Device Key: C2K9IQE0ajiESPe/M9u9RA== 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)



圖15-20：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中新增加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。

新增好裝置後，我們就要透過樹莓派將本地資料上傳至雲端。請先檢查專案中是否包含『Microsoft.Azure.Devices.Client』、『Newtonsoft.Json』兩個 package。如遺失請以套件管理員下載：

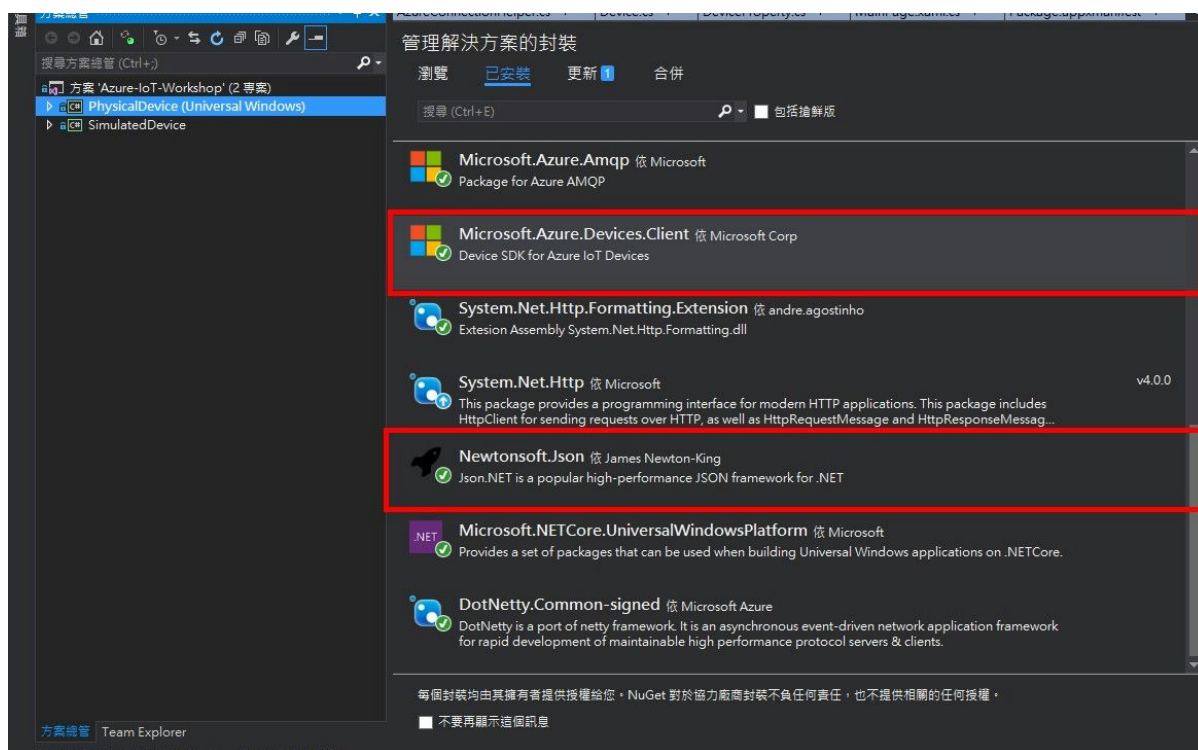


圖21：上傳資料過程所需Nuget套件。

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：『[..IOT_Workshops\UploadAzureSuite\PhysicalDevice\Device.cs](#) 』

函式名稱：*Device()*

```
1 //*****WorkShop1：填入裝置連線資訊*****//
2 this.DeviceId = "Your Device ID";
3 this.DeviceKey = "Your DeviceKey";
4 this.HostName = "Your HostName";
```

再加入裝置的經緯度資訊：

檔案目錄：『[..IOT_Workshops\UploadAzureSuite\PhysicalDevice\Device.cs](#) 』

函式名稱：*UpdateDeviceInfo()*

```
1 //*****WorkShop2：裝置屬性資訊*****//
2 deviceProp.Latitude = 9.6;           // 緯度
3 deviceProp.Longitude = 53.3;        // 經度
```

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：『[..IOT_Workshops\UploadAzureSuite\PhysicalDevice\Device.cs](#)』

函式名稱：*ProcessDevice()*

```

1  //*****WorkShop3：上傳裝置資訊至雲端*****//
2  // 建立連線
3  string IoTHubConnString =
   string.Format($"HostName={HostName};DeviceId={DeviceId};SharedAccessKey={DeviceKey}");
4  Client = DeviceClient.CreateFromConnectionString(IoTHubConnString,
   TransportType.Http1);
5
6  // 設定資料
7  JObject data = new JObject();
8  data.Add("DeviceId", DeviceId);
9  data.Add("Temperature", temperature);
10
11 // 傳送訊息
12 var messageString = JsonConvert.SerializeObject(data);
13 var message = new Message(Encoding.UTF8.GetBytes(messageString));
14 await Client.SendEventAsync(message);

```

『佈署』並『重新啟動應用程式』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！

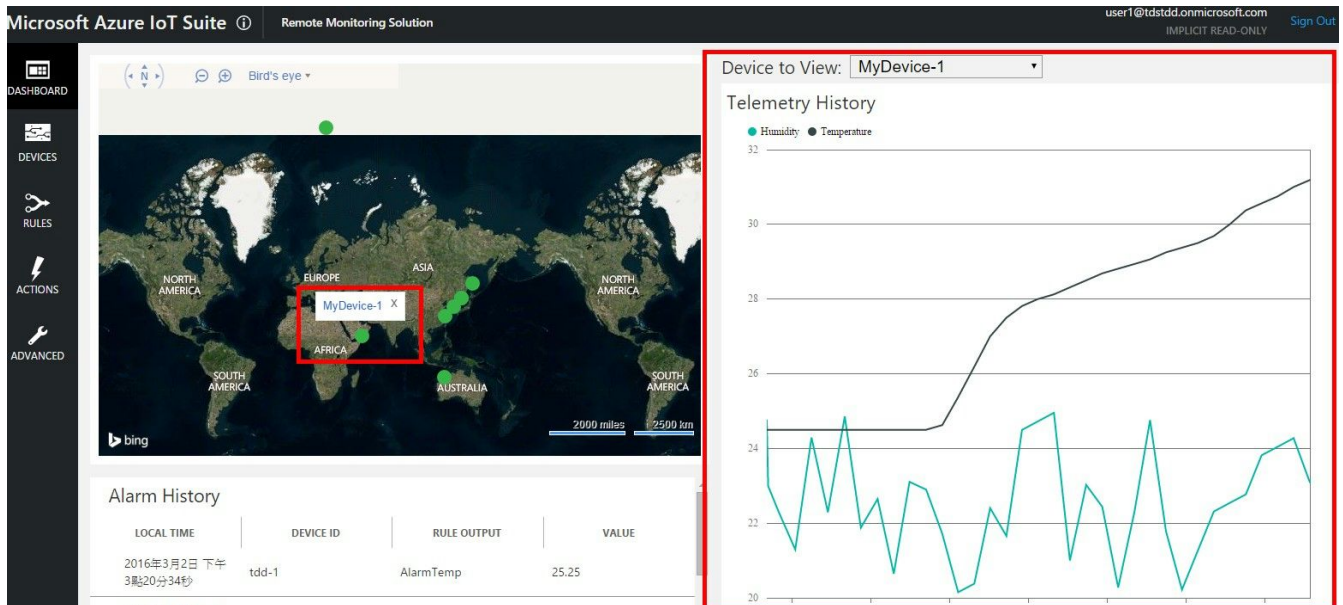


圖22：Dashboard顯示即時溫度數據

The screenshot shows the 'Device List' table in the Microsoft Azure IoT Suite. The table has columns for STATUS, DEVICE ID, MANUFACTURER, MODEL NUMBER, SERIAL NUMBER, FIRMWARE, PLATFORM, PROCESSOR, and INSTALLED RAM. The first row, 'MyDevice-1', is highlighted with a red box. To the right of the table is the 'Device Properties' panel, also highlighted with a red box, showing details for 'MyDevice-1'.

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23：裝置資訊已更新

參考資料

1. [Blinky Sample](#)
2. [DS18B20_Win10 IOT \(讀取溫度資訊\)](#)
3. [亂馬客 - \[IOT\]Raspberry Pi 2 /Windows IOT Core 讀取 DS18B20 的溫度資訊](#)
4. [Azure IOT使用教學 \(C#\)](#)
5. [Azure IOT Remote Monitoring專案](#)