目錄

[第1章 wifi連線與設定 3](#_Toc501965281)

[1.1 準備材料 3](#_Toc501965282)

[1.2 AT 指令測試 4](#_Toc501965283)

[1.3 燒錄 8](#_Toc501965284)

[1.4 Wifi初始設定 10](#_Toc501965285)

[1.5 ThingSeapk註冊與設定 11](#_Toc501965286)

[1.6 連線測試 14](#_Toc501965287)

[第2章 感測器連接與設定 16](#_Toc501965288)

[2.1 準備材料 16](#_Toc501965289)

[2.2 連接Arduino 18](#_Toc501965290)

[2.3 資料讀取 18](#_Toc501965291)

[第3章 WIFI與感測器結合 23](#_Toc501965292)

[3.1 事前準備 23](#_Toc501965293)

[3.2 字串撰寫與數據整理 23](#_Toc501965294)

[3.3 資料輸出 24](#_Toc501965295)

[3.4 程式碼整合 25](#_Toc501965296)

圖目錄

[圖 1‑1 3](#_Toc501965357)

[圖 1‑2 4](file:///C:\Users\User\Desktop\水族箱wifi感測器.docx#_Toc501965358)

[圖 1‑3 4](file:///C:\Users\User\Desktop\水族箱wifi感測器.docx#_Toc501965359)

[圖 1‑4 6](#_Toc501965360)

[圖 1‑5 6](#_Toc501965361)

[圖 1‑6 7](#_Toc501965362)

[圖 1‑7 7](#_Toc501965363)

[圖 1‑8 8](#_Toc501965364)

[圖 1‑9 8](#_Toc501965365)

[圖 1‑10 9](#_Toc501965366)

[圖 1‑11 11](#_Toc501965367)

[圖 1‑12 12](#_Toc501965368)

[圖 1‑13 12](#_Toc501965369)

[圖 1‑14 13](#_Toc501965370)

[圖 1‑15 15](#_Toc501965371)

[圖 2‑1 16](#_Toc501965372)

[圖 2‑2 17](#_Toc501965373)

[圖 2‑3 18](#_Toc501965374)

# wifi連線與設定

## 準備材料

Wifi晶片使用的是上海樂鑫科技所製作的

ESP 8266 - 01

燒路使用的是廠商自行開發的**USB轉ESP8266 轉接板**

ATcomend測試使用的軟體是Realterm



圖 1‑1

使用的開發板為Arduino UNO R3

上傳數據的網站為Tingspeak

## AT 指令測試

測試的方法可分為使用arduino通訊或是使用轉接板，且利用Realterm

通訊軟體檢測，這邊使用的是轉接板來做通訊測試，使用轉接板也為接下來的燒錄做準備，主要目的為測試Wifi晶片是否正常，且能收發訊號。

我們使用的轉接板為民間自製的ESP01轉USB(圖1.1)，減少腳位錯誤的機會，常見的轉接板還有CP2103(圖1.2)

完成連接後利用通訊軟體Realterm，配合AT指令來測試Wifi晶片



圖 1‑2

圖 1‑3

本次使用的為圖1.1.1-2 自製的ESP01轉USB，即可簡單進行連線

如果使用的是CP2103，可參考下列接線方式

VCC接3.3V

CH\_PD接3.3V

GND接地

UTXD接RXD

URXD接TXD

其他GPIO0、GPIO2、RST不接

我們可以透過AT指令來測定wifi晶片的設定

ESP-01分為以下三種模式

1.工作站(Station)：做為獨立模組，可連接到現有連線

2.AP模式(Access Point)：做為基地台，可讓現有裝置連接

3.複合模式(Station+AP)：同時使用以上兩種功能

本實驗所實用到的AT指令有：

測試類

AT 確認連線回應，如有「OK」即為正常

AT+NAME? 確認晶片命名

AT+NAME=X 命名晶片為X

AT+CIOBAUD? 胞率查詢

模式類

AT+CWMODE? 查詢目前模式

AT+CWMODE=X 變更為X模式

連線類

AT+CILAP 列出目前的Wifi清單

AT+CIFER 目前wifi晶片的IP位置

AT+CWJAP? 目前連接到的Wifi網路

AT+CWQAP 切斷目前連接網路

AT+CWJAP=<SSID>,<PASSWORD> 連線現有SSID網路，PASSWORD為 密碼

我們使用的是Realterm來作為通訊工具，在Realterm打開後，在下方子頁進行設定。

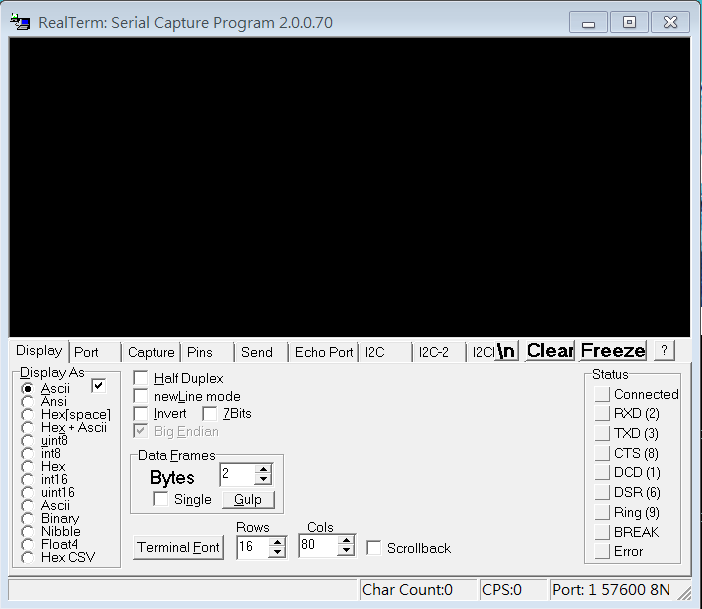


圖 1‑4

Display僅需注意編碼顯示方式為Ascii

切換到port

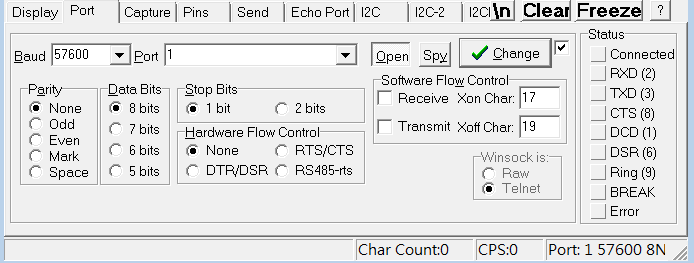


圖 1‑5

Baud請更換為115200或57600

Port 請更換為目前使用wifi晶片連線的USB連接埠

接著按下Change，沒有問題後，切換到Send

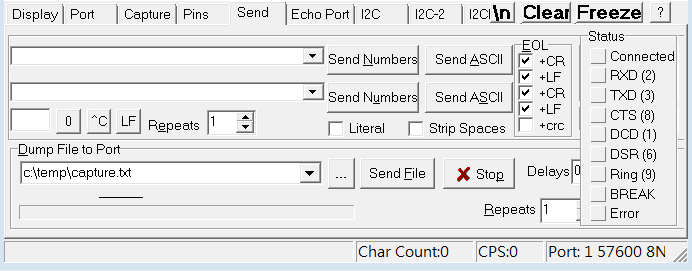


圖 1‑6

勾選+CR、+LF、+CR、+LF

確認以上步驟完成後，在空格處輸入AT指令，如圖1.1.1-6

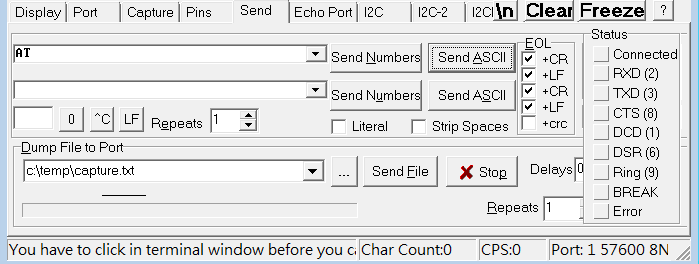


圖 1‑7

使用最簡單的AT來確認wifi連接，上方黑屏幕顯示OK即可進行下一步燒錄。

如果沒有顯示AT，可以參考以下方法

1.參考圖1.1.1-4，更換Bund為115200或57600

2.確認連接正確，電壓穩定，電流足夠

3.確認右邊的Staus的RXD與TXD沒有紅色燈號閃爍，如有閃爍，可能為轉接板有問題。

## 燒錄

Esp8266有多種角位與型號，我們使用的為腳位最少，且低成本的Esp-01  
元件如圖1.1.2-1

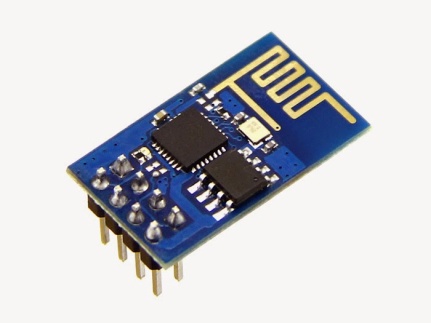


圖 1‑8

ESP01的腳位分布如圖1.1.2-2

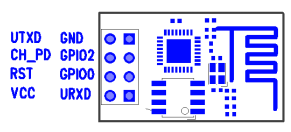


圖 1‑9

一般而言，舊版的ESP-01的Baud Rate為115200，此數值超過了arduino所支援的Baud Rate，我們透過軟體的方式更改ESP-01的Baud Rate到9600，

如果燒錄時使用的轉接器需要自己連接腳位的話，可以參考以下連接方法

VCC接3.3V

CH\_PD接3.3V

GND接地

GPIO0接地

UTXD接RXD

URXD接TXD

其他GPIO2、RST不接

請注意，GPIO0再燒錄時需接地，這點與單純通訊不同

進行燒錄，請注意ESP-01的電壓需穩定，可使用外接電壓的模組

使用的為原廠提供的ESP8266\_flasher以及v0.9.2.2 AT Firmware。

按Bin選擇燒錄程式，請確認v0.9.2.2 AT Firmware的檔案位置。

更改COM為連接ESP8266的USB埠，可以由裝置管理員查看目前使用的連接埠，接著按下Download，程式即會開始燒錄。

如果更新軟體時發生問題，提供以下幾個解決方式

1.重新連接，確認連接埠正確無異狀

2.電源供應是否正常，或電流供應不足，電流供應應在100mA-200mA

3.軟體使用系統管理員方式開啟

4.Wifi晶片受損

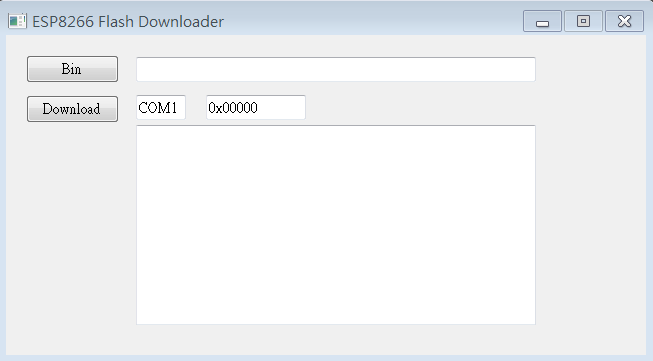


圖 1‑10

在燒錄完成後，再一次重新連接Realterm，使用AT指令**AT+CIOBAUD?**

確認目前Baud Rate為9600，進行下一步驟wifi初始連線設定。

## Wifi初始設定

我們希望在設備通電後，wifi將自動連上我們所設定的Wifi基地台。

在Realterm下使用AT指令切換到station模式

AT+CWMODE=1

OK

再來讓ESP01使用單一連線模式

AT+CIPMUX=0

OK

然後使用TCP使用的無線基地台

AT+CWJAP="SSID","PASSWORD"

OK

這裡的SSID與PASSWORD我們帶入要使用的wifi基地台

範例：

AT+CWJAP="class318","loveMust520"

設定完之後，設定將會紀錄在ESP01的flash中，每次重啟電源都會記住設定，自動連線到所設定的基地台。

接著確認與ThingSpeak的連結

在AT模式下輸入

AT+CIPSTART="TCP","184.106.153.149",80

184.106.153.149為ThingSpeak的TCP連接位置，在80的連接埠上連接。

連接成功會顯示

OK

Linked，代表連線成功。

## ThingSeapk註冊與設定

以ThingSpeak網站做為IoT的資料上傳TCP

註冊後在Channels選擇My Cheannels

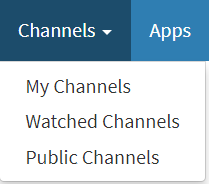
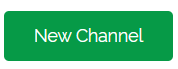


圖 1‑11

選擇New Channel



Name就是Cheannel名稱

Description為Cheannel的描述，可自行添加

勾選field為接受數據並做圖，可自定義名稱

每一個field可接受一組訊號做圖

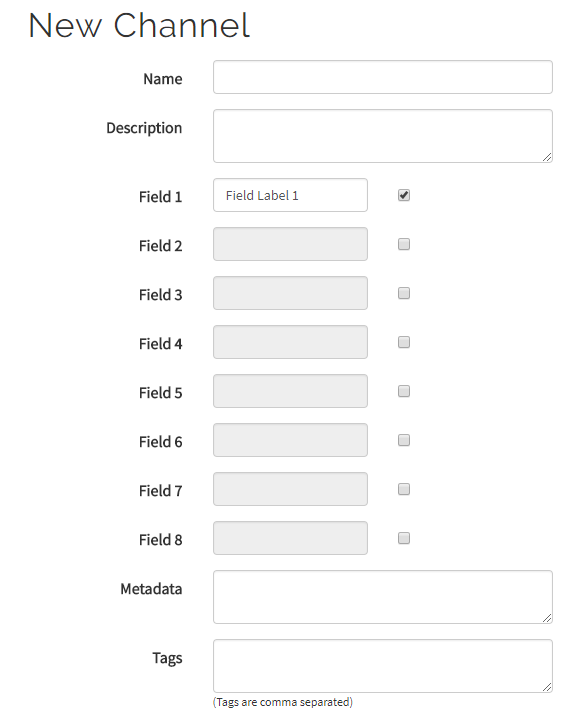
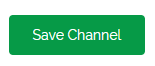


圖 1‑12

按下Save Channel存檔



選取API Keys

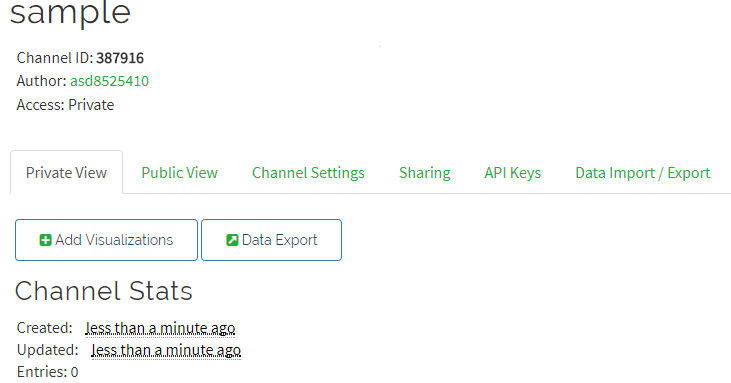


圖 1‑13

API KEY即為連接用密碼

API全名為Application Programming Interface

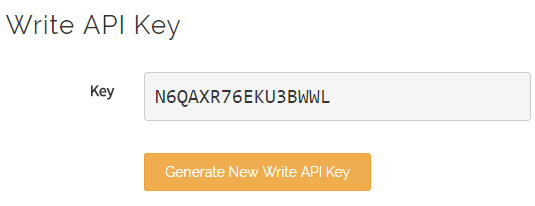


圖 1‑14

接著，就可以開始撰寫Arduino上的程式碼。

## 連線測試

將ESP01接上Arduino

VCC接3.3V

CH\_PD接3.3V

GND接地

UTXD接RXD(2)

URXD接TXD(3)

其他GPIO0、GPIO2、RST不接

RXD與TXD可用Arduino上的0與1，但這樣不易燒錄程式

這邊我們用Pin2,3來做為RXD與TXD

連接後，我們先測試Arduino與wifi晶片的連線

#include <SoftwareSerial.h>

SoftwareSerial sSerial(2,3); //(RX,TX) 與 ESP8266 介接的軟體串列埠

void setup() {

sSerial.begin(9600); //設定軟體序列埠速率 (ESP8266)

Serial.begin(9600); //設定軟體序列埠速率 (PC)

Serial.print("Set working mode as station ... ");

sSerial.println("AT+CWMODE=0"); //設定為 Station 模式

delay(1000);

if (sSerial.find("OK") || sSerial.find("no change")) {

Serial.println("OK");

}

else {

Serial.println("No response");

}

}

void loop() {

if (sSerial.available()) { //若軟體序列埠 Rx 收到資料 (來自 ESP8266)

Serial.write(sSerial.read()); //讀取後寫入硬體序列埠 Tx (PC)

}

if (Serial.available()) { //若硬體序列埠 Rx 收到資料 (來自 PC)

sSerial.write(Serial.read()); //讀取後寫入軟體序列埠 Tx (ESP8266)

}

}

上傳後即可在序列埠監控視窗查看

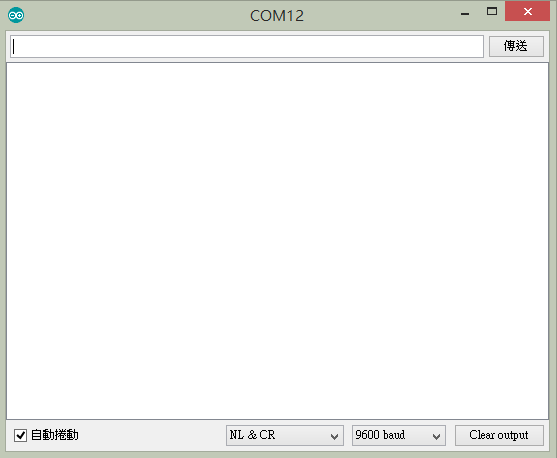


圖 1‑15

紫色邊框為指令輸入處

藍色邊框為指令行結尾，如想和ESP01連線，且改為NL&CR

紅色框處為Baud Rate

在此輸入AT指令，將會獲得與Realterm上通訊相同效果，藉此可確認Wifi與Arduino的通訊。

# 感測器連接與設定

## 準備材料

溫溼度感測器使用的是奧松的DHT22

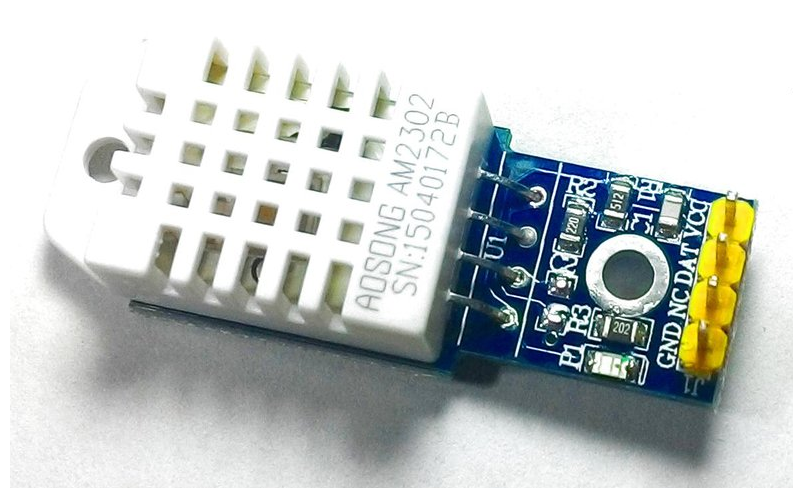


圖 2‑1

參數規格：

工作電壓:3V–5.5V

傳感器型號: AM2302溫濕度傳感器

信號輸出形式:數字信號

溫度測量範圍:-40℃–80℃

測量精度:0.5℃

濕度測量範圍:0–100%RH

測量精度:2%RH

分辨率:16位

帶固定螺絲孔,方便安裝及固定。孔徑2.6mm

pH計為上海儀電科學儀器股份有限公司



圖 2‑2

參數規格：

1、加熱電壓：5±0.2V（AC·DC）

2、工作電流：5-10mA

3、檢測濃度範圍：PH0-14

4、檢測溫度範圍：0-100℃

5、回應時間：≤5S

6、穩定時間：≤60S

7、元件功耗：≤0.5W

8、工作溫度：-10~50℃（標稱溫度20℃）

9、工作濕度：95&#xRH;（標稱濕度65&#xRH;）

10、使用壽命：3年

11、尺寸大小：42mm×32mm×20mm

12、重量大小：25g

13、輸出方式：模擬電壓信號輸出

## 連接Arduino

溫溼度模組

|  |  |
| --- | --- |
| DHT22模組 | Arduino |
| VCC | VCC(3V) |
| GND | GND |
| DATA | PIN2 |

※請依照手邊上的模組連接電阻

pH計模組

|  |  |
| --- | --- |
| pH模組 | Arduino |
| VCC | VCC(5V) |
| GND | GND |
| P0 | A2 |

## 資料讀取

連接完成後安裝程式所需要的函數庫在Arduino

<https://github.com/adafruit/DHT-sensor-library>

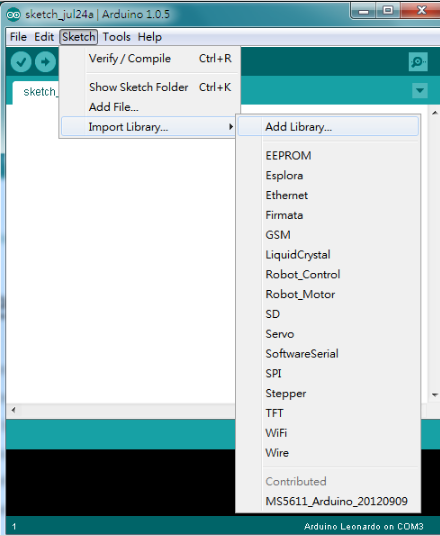


圖 2‑3

在add library中點選目的資料夾即會自動新增

程式碼如下

#include <DHT.h>

#define PIN 2

#define TYPE DHT22

DHT dht(PIN, TYPE);

void setup() {

Serial.begin(9600);

}

void loop() {

delay(1000);

float h = dht.readHumidity();

// 溫度讀取為攝氏

float t = dht.readTemperature();

// 讀取為華氏 = Ture

float f = dht.readTemperature(true);

檢查是否偵測成功

if (isnan(h) || isnan(t) || isnan(f)) {

Serial.println("Failed to read from DHT sensor!");

return;

}

Serial.print(h, 2);

Serial.print(",");

Serial.println(t, 2);

}

  Serial.print(" \*C ");

  Serial.print(hif);

  Serial.println(" \*F");

}

PH meter程式如下：

#define SensorPin A2

#define Offset 0.00

#define LED 13

#define samplingInterval 20

#define printInterval 800

#define ArrayLenth 40

int pHArray[ArrayLenth];

int pHArrayIndex=0;

void setup(void)

{

pinMode(LED,OUTPUT);

Serial.begin(9600);

Serial.println("pH meter experiment!");

}

void loop(void)

{

static unsigned long samplingTime = millis();

static unsigned long printTime = millis();

static float pHValue,voltage;

if(millis()-samplingTime > samplingInterval)

{

pHArray[pHArrayIndex++]=analogRead(SensorPin);

if(pHArrayIndex==ArrayLenth)pHArrayIndex=0;

voltage = avergearray(pHArray, ArrayLenth)\*5.0/1024;

pHValue = 3.5\*voltage+Offset;

samplingTime=millis();

}

if(millis() - printTime > printInterval)

{

Serial.print("Voltage:");

Serial.print(voltage,2);

Serial.print(" pH value: ");

Serial.println(pHValue,2);

digitalWrite(LED,digitalRead(LED)^1);

printTime=millis();

}

}

double avergearray(int\* arr, int number){

int i;

int max,min;

double avg;

long amount=0;

if(number<=0){

Serial.println("Error number for the array to avraging!/n");

return 0;

}

if(number<5){

for(i=0;i<number;i++){

amount+=arr[i];

}

avg = amount/number;

return avg;

}else{

if(arr[0]<arr[1]){

min = arr[0];max=arr[1];

}

else{

min=arr[1];max=arr[0];

}

for(i=2;i<number;i++){

if(arr[i]<min){

amount+=min;

min=arr[i];

}else {

if(arr[i]>max){

amount+=max;

max=arr[i];

}else{

amount+=arr[i]; }

}

}

avg = (double)amount/(number-2);

}

return avg;

}

首次檢測使用pH7的溶液，將之與7相減，將此數填入Offset

將著使用其他的溶液，測量後旋轉校正紐來校正

需重複多次，且著實清理感應頭元件。

# WIFI與感測器結合

## 事前準備

將三塊模組組合在麵包版上，請參考1-6與2-2連接方法

## 字串撰寫與數據整理

字串的傳輸我參考的是HyperText Transfer Protocol（HTTP）HTTP是一個用戶端終端（用戶）和伺服器端（網站）請求和應答的標準（TCP）。通過使用網頁瀏覽器或者其它的工具，用戶端發起一個HTTP請求到伺服器上指定埠（預設埠為80）。我們稱這個用戶端為用戶代理程式（user agent）。應答的伺服器上儲存著一些資源，比如HTML檔案和圖像。我們稱這個應答伺服器為源伺服器（origin server）。在用戶代理和源伺服器中間可能存在多個「中間層」，比如代理伺服器、閘道器或者隧道（tunnel）。

而我們使用的請求方法為GET向指定的資源發出「顯示」請求。使用GET方法應該只用在讀取資料。目的網頁就是 Thingspeak 的更新通道資料的網頁程式 update :

update?api\_key=API KEY&欄位1=數值1&欄位2=數值2&欄位3=數值3....  
  
我們要向 ESP8266 傳送的資料格式如下 :  
  
GET update?api\_key=NO5N8C7T2KINFCQE&field1=29.00&field2=84.20&field3=73.00

## 資料輸出

將此步驟依序可以整理為：

1. 確認模組傳出的訊號
2. 建立 TCP 連線 :

String cmd="AT+CIPSTART=\"TCP\",\"184.106.153.149\",80";

1. 傳送GET字串長度給 ESP8266 :

AT+CIPSEND=GET 字串長度

1. 傳送 GET 字串給 ESP8266 :

sSerial.println(GET字串)

1. 關閉 TCP 連線 :

AT+CIPCLOSE

## 程式碼整合

#include <SoftwareSerial.h>

#include <DHT.h>

#define DHTPIN 2

#define DHTTYPE DHT11

#define SensorPin A2

#define Offset 0.00

#define samplingInterval 20

#define printInterval 800

#define ArrayLenth 40

int pHArray[ArrayLenth];

int pHArrayIndex=0;

void setup(void)

{

Serial.begin(9600);

sSerial.begin(9600);

dht.begin();  
sSerial.println("AT+RST");

}

void loop(void)

DHT dht(DHTPIN, DHTTYPE);

String api\_key="NO5N8C7T2KINFCQE";

SoftwareSerial sSerial(10,11);

void setup() {                 
  Serial.begin(9600);

  sSerial.begin(9600);

  dht.begin();  
  sSerial.println("AT+RST");

  }  
  
void loop() {

{

static unsigned long samplingTime = millis();

static unsigned long printTime = millis();

static float pHValue,voltage;

if(millis()-samplingTime > samplingInterval)

{

pHArray[pHArrayIndex++]=analogRead(SensorPin);

if(pHArrayIndex==ArrayLenth)pHArrayIndex=0;

voltage = avergearray(pHArray, ArrayLenth)\*5.0/1024;

pHValue = 3.5\*voltage+Offset;

samplingTime=millis();

}

if(millis() - printTime > printInterval)

{

Serial.print("Voltage:");

Serial.print(voltage,2);

Serial.print(" pH value: ");

Serial.println(pHValue,2);

printTime=millis();

}

}

double avergearray(int\* arr, int number){

int i;

int max,min;

double avg;

long amount=0;

if(number<=0){

Serial.println("Error number for the array to avraging!/n");

return 0;

}

if(number<5){

for(i=0;i<number;i++){

amount+=arr[i];

}

avg = amount/number;

return avg;

}else{

if(arr[0]<arr[1]){

min = arr[0];max=arr[1];

}

else{

min=arr[1];max=arr[0];

}

for(i=2;i<number;i++){

if(arr[i]<min){

amount+=min;

min=arr[i];

}else {

if(arr[i]>max){

amount+=max;

max=arr[i];

}else{

amount+=arr[i]; }

}

}

avg = (double)amount/(number-2);

}

return avg;

}

  float h=dht.readHumidity();

  float c=dht.readTemperature();

  float f=dht.readTemperature(true);

float ph= pHValue

  String param="&field1=" + (String)c + "&field2=" + (String)f +  
               "&field3=" + (String)h + "field4=" + (String)ph;

  String cmd="AT+CIPSTART=\"TCP\",\"184.106.153.149\",80";  
  sSerial.println(cmd);  
  //偵測 TCP 連線是否成功  
  if (sSerial.find("Error")) {             
    Serial.println("AT+CIPSTART error!");  
    return;   
    }  
  Serial.println(cmd);

  //製作字串  
  String GET="GET /update?api\_key=" + api\_key + param + "\r\n\r\n";  
  Serial.println(GET);

  cmd="AT+CIPSEND=" + String(GET.length());

  sSerial.println(cmd);

  Serial.println(cmd);

  //檢查 ESP8266 是否回應  
  if (sSerial.find(">")) {  //若收到 ESP8266 的回應標頭結束字元  
    sSerial.print(GET);  //向 ESP8266 傳送 GET 字串內容  
    }  
  else {  //沒有收到 ESP8266 回應  
    sSerial.println("AT+CIPCLOSE");  //關閉 TCP 連線  
    Serial.println("AT+CIPCLOSE");    //顯示連線關閉訊息於監控視窗  
    }  
  delay(16000);  //延遲 16 秒 (Thingspeak 更新間隔16秒)  
  }