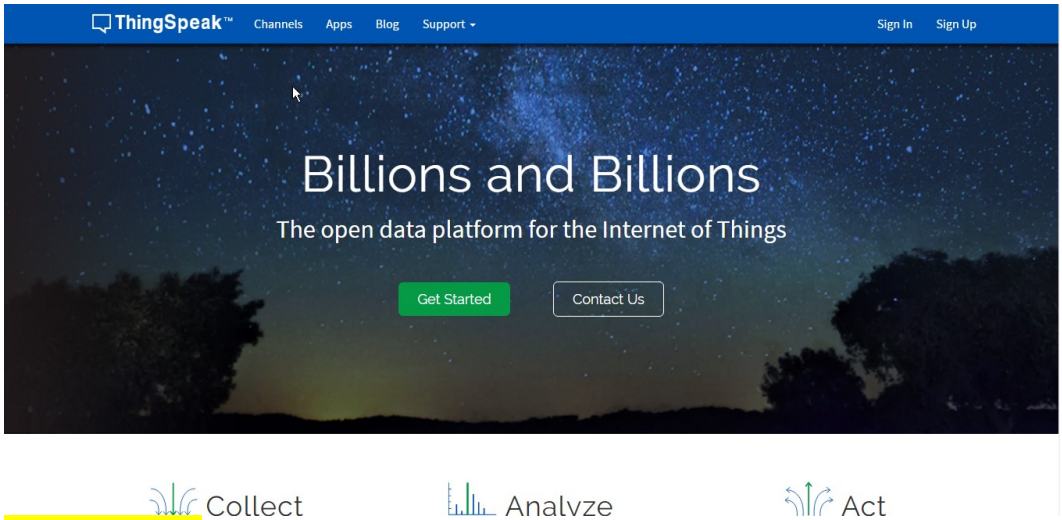
ThingSpeak with ESP 32

Week10 @3,4,5 ตุลาคม 2560

http://robotics-za.blogspot.com/2015/05/nodemcu-thingspeak.html



1. Introduction

Thingspeak เป็นเว็ปที่ให้การบริการในการเก็บข้อมูล และสามารถแสดงข้อมูลแบบ real-time ได้ ซึ้งเราสามารถ update ข้อมูล หรือจะเรียกดูข้อมูลได้ตลอดเวลา ที่ไหนก็ได้ เพราะทำงานบน cloud ซึ่ง thingspeak สร้างมาเพื่อต้องการให้ตอบโจทย์ของ IoT อยู่แล้ว ส่วนข้อมูลที่เก็บอยู่บน cloud นั้นก็ขึ้นอยู่กับเราว่าจะใช้ยังไง รูปแบบไหน ในการที่จะส่งข้อมูล data ไปไว้บน cloud นั้น ทาง thingspeak ได้มี api ในการติดต่อไว้เรียบร้อยแล้ว

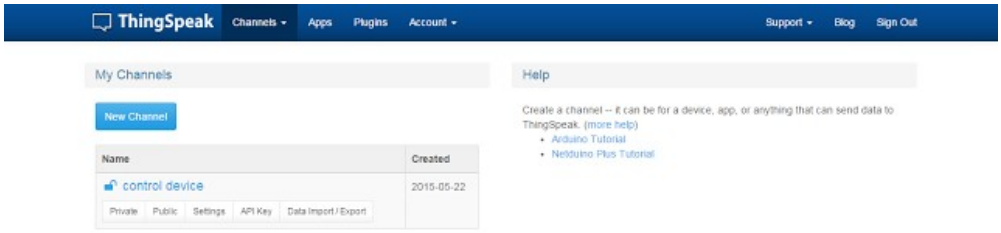
2. Read This

- http://robotics-za.blogspot.com/2015/05/nodemcu-thingspeak.html

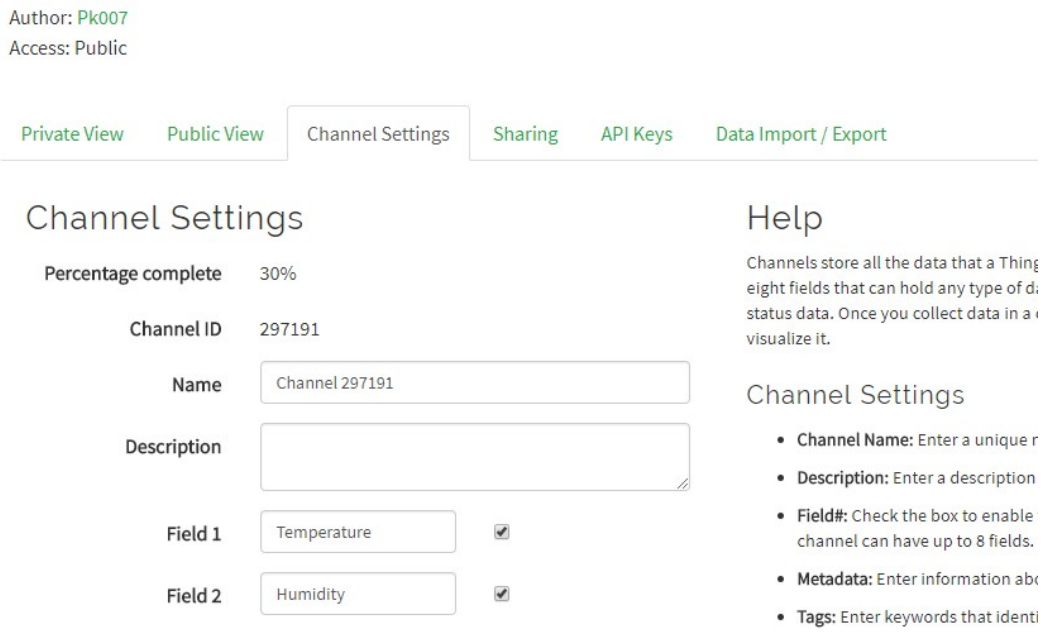
3. Experiment

1. การเปิดใช้งาน Thinkspeak

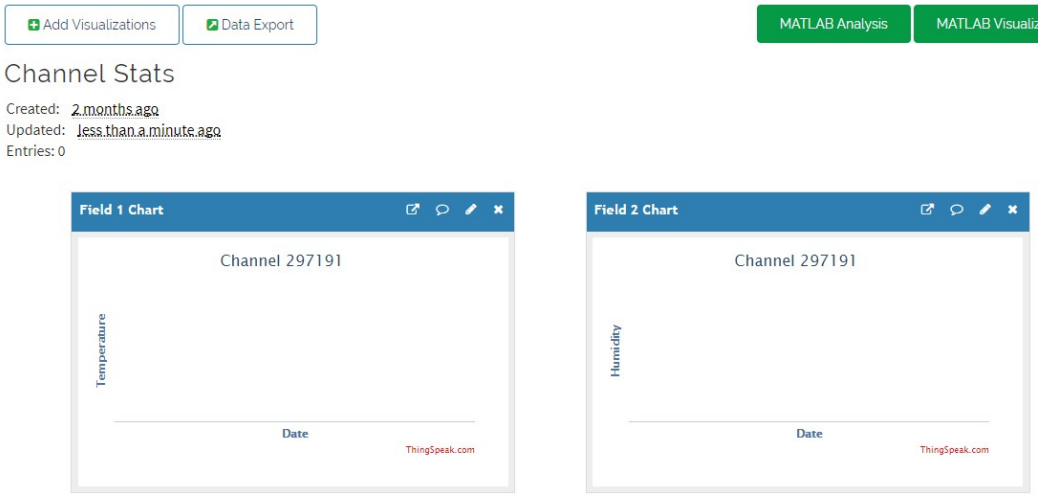
- อันดับแรกเลย ให้ทำการสมัครสมาชิกให้เรียบร้อย
- จากนั้นก็สร้าง channel ขึ้นมา โดยให้เรากดไปที่ My channel แล้วก็ new channel ขึ้นมา



- หลังจากที่ new channel ขึ้นมาแล้ว ไปที่ channel setting ก็ป้อนข้อมูลเข้าไป
- อย่าลืมเลือก ☑ Make Public
- เสร็จแล้วก็กด save channel

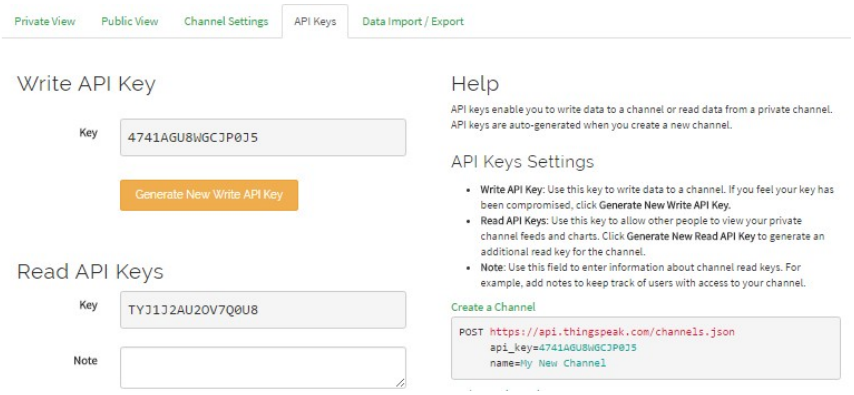


- เมื่อสร้างเสร็จแล้ว ก็จะแสดงหน้าต่าง ในหน้าต่างนี้จะแสดงข้อมูลเป็นแบบเส้นกราฟ ซึ่งตอนนี้ยังไม่มีข้อมูลใด ๆ ส่งมาจึงไม่เกิดอะไรขึ้น



2. การ update ข้อมูลไปยัง cloud ผ่าน api ของ thingspeak

- อันดับแรกให้ดูที่ API KEY ของเราว่ามันคืออะไร
  - Write API Key 4741AGU8WGCJP0J5
  - Read API Keys TYJ1J2AU2OV7Q0U8



- ในการ update ข้อมูลจะเป็นการส่ง HTTP Request ไปยัง server เพื่อ update ข้อมูลที่ต้องการตาม field ต่าง ๆ ที่เรากำหนด วิธีการ update โดยเราจะส่งข้อมูลแบบนี้

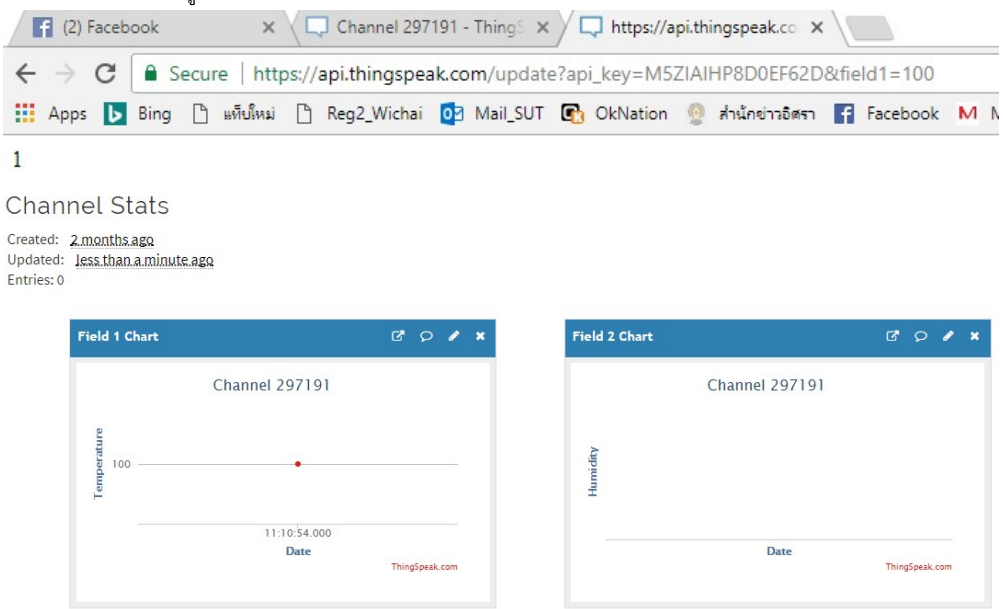
  https://api.thingspeak.com/update?key=4741AGU8WGCJP0J5&field1=100

  ที่ขีดเส้นไว้คือ

  4741AGU8WGCJP0J5 คือ API KEY

  field1=100 คือ field ที่เราต้องการ update ในตัวอย่างคือ field ที่ 1 และกำหนดให้มีค่าเท่ากับ 100

-
- มาดูผลลัพธ์กันที่แท็บ Private View



จากรูปตัวอย่างฝั่งขวาคือเจ้าของบล็อกได้ทำการ Update ข้อมูล โดยกำหนดให้ field ที่ 1 มีค่าเท่ากับ 0 และฝั่งซ้ายนั้นได้มีจุดเพิ่มมา 1 จุดก็คือค่าที่เราเพิ่มไปเมื่อซักครู่นั่นเอง

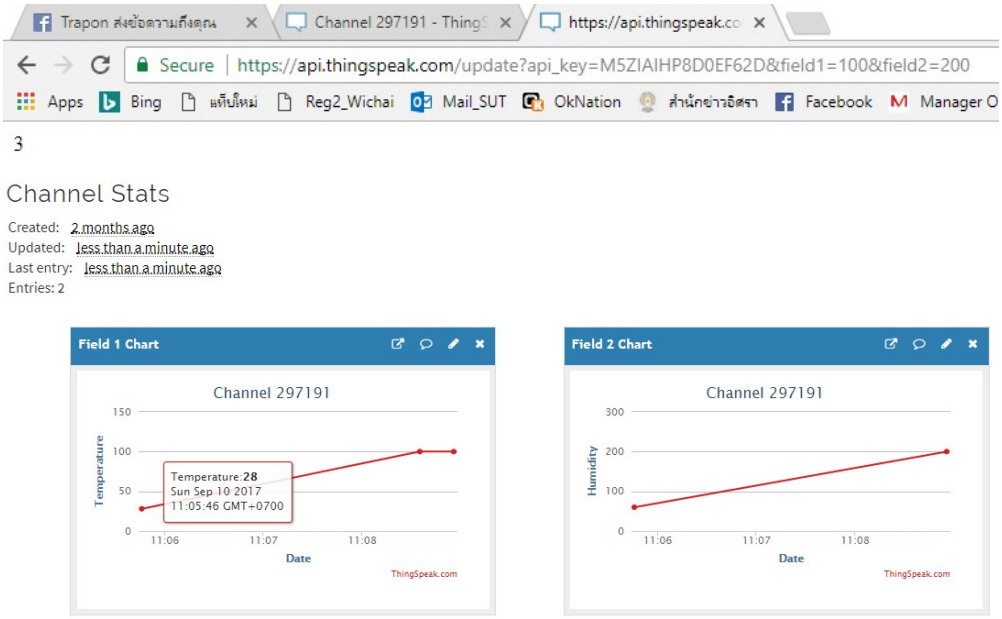
- หากต้องการ update ข้อมูลหลายๆ field พร้อมกันเราจะส่งข้อมูลแบบนี้

https://api.thingspeak.com/update?key=4741AGU8WGCJP0J5&field1=100&field2=200

4741AGU8WGCJP0J5 คือ API KEY

field1=100 คือ field ที่เราต้องการ update ในตัวอย่างคือ field ที่ 1 และกำหนดให้มีค่าเท่ากับ 100

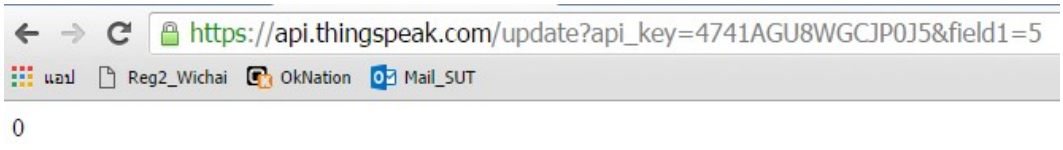
field2=200 คือ field ที่เราต้องการ update ในตัวอย่างคือ field ที่ 2 และกำหนดให้มีค่าเท่ากับ 200



3. การมอนิเตอร์ การสั่งงานผ่าน Web Browser

POST Data to ThinkSpeak

https://api.thingspeak.com/update?api_key=4741AGU8WGCJP0J5&field1=5



GET Data from ThinkSpeak

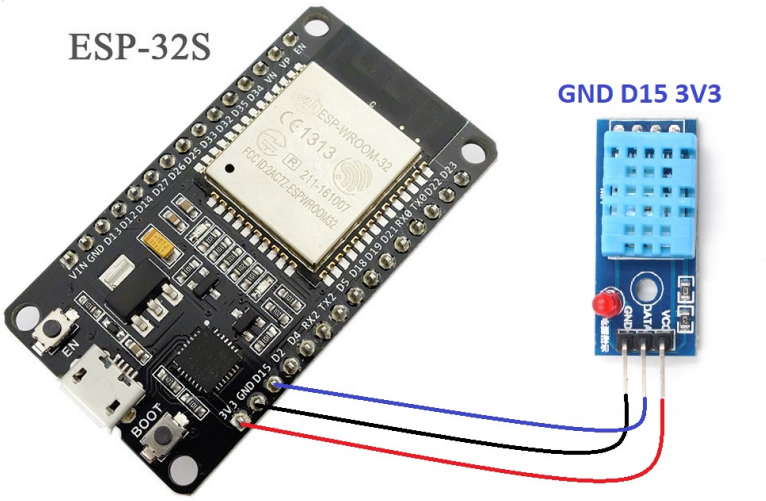
https://api.thingspeak.com/channels/110910/fields/1/last

4. เขียนโปรแกรมให้กับ ESP-32 ให้ส่งข้อมูลไป update

- Test DHT-11
  - Add Library http://www.dfrobot.com.cn/image/data/DFR0067/dht11.zip
  - Test Code File → Example → DHT-11 → Dht11_Test
  - แก้ไขขา DHT11_PIN 15

dht11_test §

```
//
//    FILE:  dht11_test1.pde
// PURPOSE: DHT11 library test sketch for Arduino
//
#include <dht11.h>
dht11 DHT;
#define DHT11_PIN 15
```



- Test Code on Arduino IDE
  - แก้ไข SSID, Password และ API Key

```
#include <WiFi.h>
#include <dht11.h>
#define DHT11_PIN 15

const char* ssid      = "IOT_Test";
const char* password  = "pk12345678";
String apiKey         = "M5ZIAIHP8D0EF62D";
const char* server    = "api.thingspeak.com";

dht11 DHT;
WiFiClient client;

void setup() {
  Serial.begin(115200);
  delay(1000);
  Serial.println();   Serial.println();
  Serial.print("Connecting to "); Serial.println(ssid);
  WiFi.begin(ssid, password);
  while (WiFi.status() != WL_CONNECTED)
  { delay(500);
    Serial.print(".");
  }
```

```
  Serial.println("");
  Serial.println("WiFi connected");
  Serial.println("IP address: ");
  Serial.println(WiFi.localIP());
}

void loop() {
  Serial.println("-------------------------------------");
  Serial.println("Requesting temperatures...");
  DHT.read(DHT11_PIN);
  float cTemp = DHT.temperature;
  float Humid = DHT.humidity;
  Serial.print("Temperature is: "); Serial.println(cTemp, 2);
  Serial.print("Humidity is: ");    Serial.println(Humid, 2);
  if (client.connect(server, 80))
  { String postStr = apiKey;
    postStr += "&field1=";    // Fields 1
    postStr += String(cTemp);
    postStr += "&field2=";    // Fields 2
    postStr += String(Humid);
    postStr += "\r\n\r\n";
    client.print("POST /update HTTP/1.1\n");
    client.print("Host: api.thingspeak.com\n");
    client.print("Connection: close\n");
    client.print("X-THINGSPEAKAPIKEY: " + apiKey + "\n");
    client.print("Content-Type: application/x-www-form-urlencoded\n");
    client.print("Content-Length: ");
    client.print(postStr.length());
    client.print("\n\n");
    client.print(postStr);
  }
  client.stop();
  Serial.println("Waiting...");
  delay(20000);       // thingspeak needs minimum 15 sec delay between updates
}
```

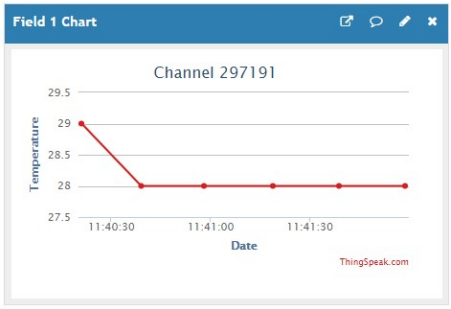
- ESP-32 ส่ง http request ไปที่ server เพื่อ update ข้อมูล ซึ่งฝั่ง thingspeak ก็ทำการ update แบบ real-time และพลอตออกมาในรูปแบบกราฟ ซึ่งทาง thingspeak สามารถบันทึกข้อมูลได้ทุก ๆ 15 วินาที ซึ่งหากว่าเราสามารถบันทึกค่าข้อมูลลง cloud ได้แล้ว ก็เหมือนกับว่าเรามีค่าอะไรบางอย่างที่เราพร้อมเรียกตลอดเวลาไม่ว่าจะอยู่ที่ไหนก็ตาม
- ส่วนของการส่งข้อมูล ก็จะเป็นการส่งแบบ http request ซึ่งเราส่งออกไปว่า

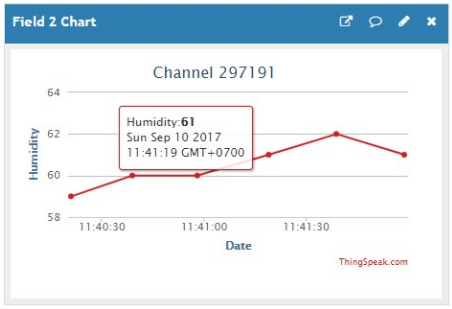
```
GET /update?key=4741AGU8WGCJP0J5&field1=0&field2=0 HTTP/1.1\r\n
        Host: api.thingspeak.com\r\n
        Connection: close\r\n\r\n
```

```
Connecting to IOT_Test
..
WiFi connected
IP address:
192.168.160.113
-------------------------------------
Requesting temperatures...
Temperature is: 28.00
Humidity is: 59.00
Waiting...
-------------------------------------
Requesting temperatures...
Temperature is: 31.00
Humidity is: 60.00
Waiting...
```

Channel Stats

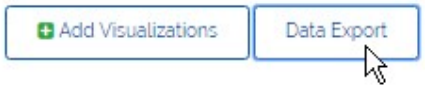
Created: 2 months ago
Updated: about a minute ago
Last entry: about a minute ago
Entries: 2





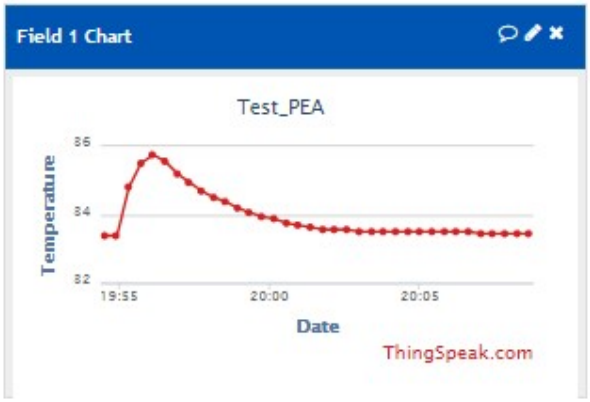
5. การนำไฟล์ออกมาเพื่อทำรายงาน

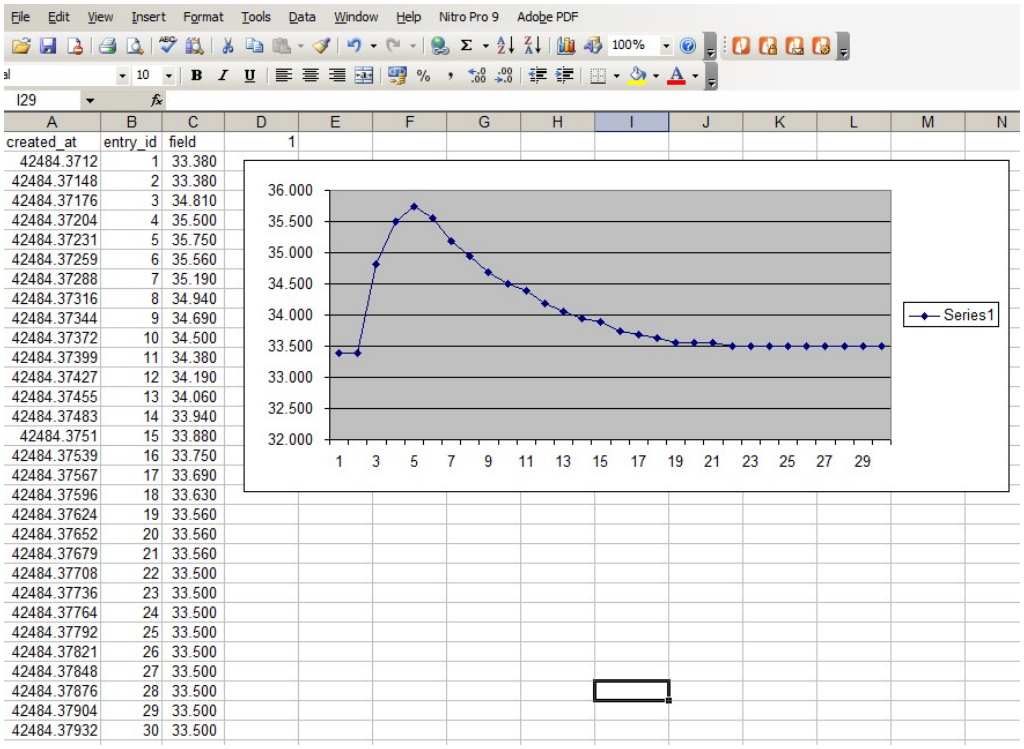
- จากข้อมูลสามารถ Export CSV ไฟล์เพื่อทำรายงานต่างๆได้



Channel Stats

Created about 20 hours ago
Updated 16 minutes ago
0 Entries

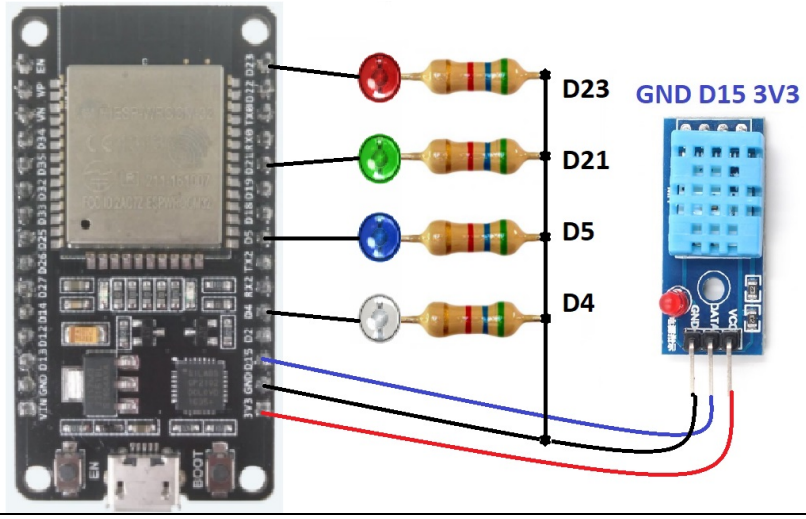




created_at	entry_id	field	D
			1
42484.3712	1	33.380	
42484.37148	2	33.380	
42484.37176	3	34.810	
42484.37204	4	35.500	
42484.37231	5	35.750	
42484.37259	6	35.560	
42484.37288	7	35.190	
42484.37316	8	34.940	
42484.37344	9	34.690	
42484.37372	10	34.500	
42484.37399	11	34.380	
42484.37427	12	34.190	
42484.37455	13	34.060	
42484.37483	14	33.940	
42484.3751	15	33.880	
42484.37539	16	33.750	
42484.37567	17	33.690	
42484.37596	18	33.630	
42484.37624	19	33.560	
42484.37652	20	33.560	
42484.37679	21	33.560	
42484.37708	22	33.500	
42484.37736	23	33.500	
42484.37764	24	33.500	
42484.37792	25	33.500	
42484.37821	26	33.500	
42484.37848	27	33.500	
42484.37876	28	33.500	
42484.37904	29	33.500	
42484.37932	30	33.500	

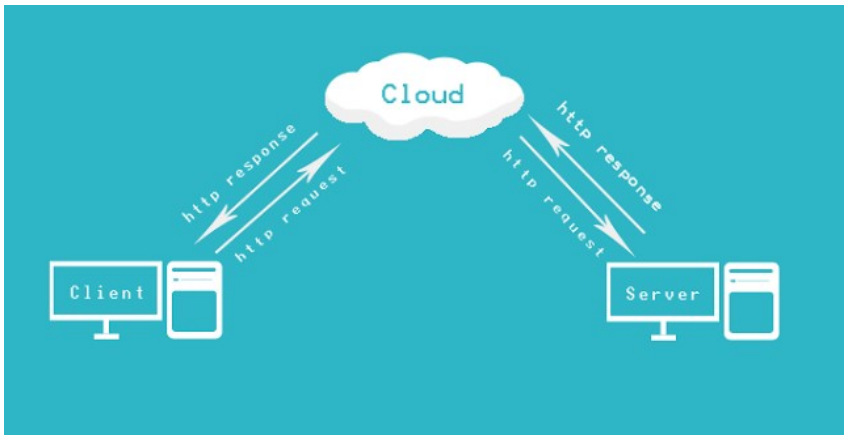
6. การควบคุมเปิดปิด LED ผ่าน Internet ด้วย ESP32

- ต่อวงจร LED ที่ D23, D21, D5, D4 และทดสอบการทำงาน



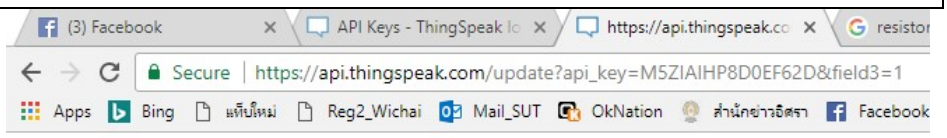

```
#define LED_1 4
#define LED_2 5
#define LED_3 21
#define LED_4 23

void setup()
{ pinMode(LED_1, OUTPUT);   digitalWrite(LED_1, LOW);
  pinMode(LED_2, OUTPUT);   digitalWrite(LED_2, LOW);
  pinMode(LED_3, OUTPUT);   digitalWrite(LED_3, LOW);
  pinMode(LED_4, OUTPUT);   digitalWrite(LED_4, LOW);
}

void loop()
{ digitalWrite(LED_1, HIGH);   delay(200);     digitalWrite(LED_1, LOW);   delay(100);
  digitalWrite(LED_2, HIGH);   delay(200);     digitalWrite(LED_2, LOW);   delay(100);
  digitalWrite(LED_3, HIGH);   delay(200);     digitalWrite(LED_3, LOW);   delay(100);
  digitalWrite(LED_4, HIGH);   delay(200);     digitalWrite(LED_4, LOW);   delay(100);
}
```

- วิธีที่เราจะควบคุม LED นั้นเราจำเป็นต้องมีข้อมูลเพื่อนำไปเช็คเป็นเงื่อนไขว่า ถ้าเกิดเป็นค่านี้ให้ LED เปิด ถ้าเป็นอีกค่านึงให้ LED ปิด ดังนั้นหากเราต้องการสั่งให้ LED เปิดหรือปิดผ่าน Internet นั้น เราจึงจำเป็นต้องมีข้อมูลบน cloud โดยเราจะทำการส่ง Http request ไปยัง server เพื่อร้องขอข้อมูลที่เราต้องการ ถ้าเราส่ง http request ได้อย่างถูกต้องก็จะมี http response ตอบกลับจาก server โดยส่งมาพร้อมกับข้อมูลที่เราต้องการ และเราก็ทำการเขียนโปรแกรมเพื่อใช้ค่าที่ server ตอบกลับมานั้น มาเป็นค่าในการควบคุม LED เพื่อเปิดหรือปิดนั่นเอง

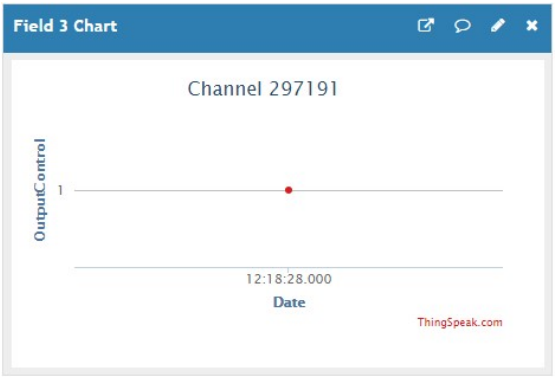


- การ get data หรือจะเป็นการ update data เราสามารถศึกษา หรือดู Example ได้จาก Documentation จาก Support ของ ThingSpeak
- ในส่วนของการ fill data จะใช้การเขียนทั่วไป

https://api.thingspeak.com/update?key=4741AGU8WGCJPOJ5&field3=9

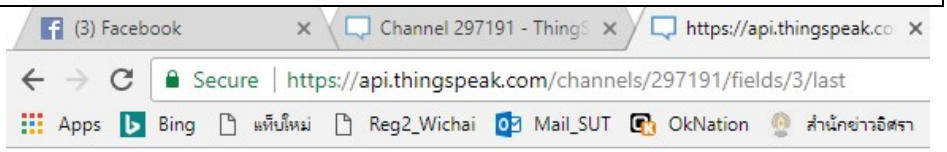


54



- ในการ get data จะใช้ url

https://api.thingspeak.com/channels/297191/fields/3/last



1

- ที่ขีดเส้นใต้ไว้นั้น 297191 คือช่อง channels ของเรา
- ส่วน 3 คือ ต้องการดูที่ field ไหน ส่วนเจ้าของบล็อกขอเลือกดูที่ field 3 แล้วกันเพราะเจ้าของบล็อกบันทึกค่าไว้ที่ field นี้
- ค่า response ตอบกลับมาเป็นเลข 1 นั่นก็คือค่าที่เราต้องการที่จะทราบว่า ค่าที่บันทึกล่าสุดของ field ที่ 3 คือค่าอะไร ก็มีการ response กลับมาเป็นค่าดังกล่าว

- ตัวอย่างนี้เป็นเพียงการทดสอบดูว่าสามารถ get ค่าจาก server ได้หรือไม่ ต่อมาเราก็มาดูกันว่าเราจะเขียนให้ ESP-32 ของเรานั้นไป get ค่ามาแล้วมาควบคุม LED ได้อย่างไร
  - แก้ไข SSID,Password
  - แก้ไข USER ID และค่า Fields

```
#include <WiFi.h>
#define LED_1 4
#define LED_2 5
#define LED_3 21
#define LED_4 23

const char* ssid = "IOT_Test";
const char* password = "pk12345678";
const char* url = "/channels/297191/fields/3/last";
const char* host = "api.thingspeak.com";

String line = "";
const int PORT = 80;
WiFiServer server(80);

void setup() {
  Serial.begin(115200);
  delay(10);
  pinMode(LED_1, OUTPUT);
  pinMode(LED_2, OUTPUT);
  pinMode(LED_3, OUTPUT);
  pinMode(LED_4, OUTPUT);
  Serial.println("\n\n");
  Serial.print("Connecting to ");
  Serial.println(ssid);
  WiFi.begin(ssid, password);

  while (WiFi.status() != WL_CONNECTED) {
    delay(500);
    Serial.print(".");
  }
  Serial.println();
  Serial.println("WiFi Connected!!");
  Serial.print("IP address : ");
  Serial.println(WiFi.localIP());
}

void loop()
{ delay(5000);
  WiFiClient client = server.available();
  Serial.print("Connecting to ");      Serial.println(host);
  Serial.print("Requesting URL : ");  Serial.println(url);

  if (!client.connect(host, PORT))
  { Serial.print(".");
    return;
  }
  else
  { client.print(String("GET ") + url + " HTTP/1.1\r\n" + "Host: " + host + "\r\n" + "Connection: close\r\n\r\n");
    delay(50);
    Serial.println("SEND !!");  // เมื่อส่งไปแล้วให้แสดงข้อความ SEND !! ออกมา
  }
  while (!client.available())  //ถ้าหากไม่มี response ตอบกลับมา ก็จะให้วนลูป จนกว่าจะมี responce ตอบกลับมา
  {}
  while (client.available())  // และเมื่อมี response ตอบกลับมาก็จะให้เข้า loop while แล้วทำการเก็บค่าทั้งหมดไว้ที่ line
  { line += (char)client.read();
  }

  Serial.println();
  Serial.println("GET SUCCESSFULLY !!");
  Serial.print("###### rawData  >> ");  Serial.println(line);
  int lengthData = line.length();
  char data = line[lengthData - 1];
```

```
  String s = (String)"Position " + lengthData + (String)", Data " + data;
  Serial.print("###### splitted >> ");  Serial.println(s);

  switch (data) {
    case '1':
      digitalWrite(LED_1, HIGH); // ถ้าเป็น 1 led1 ติด
      break;
    case '2':
      digitalWrite(LED_1, LOW); // ถ้าเป็น 2 led1 ดับ
      break;
    case '3':
      digitalWrite(LED_2, HIGH); // ถ้าเป็น 3 led2 ติด
      break;
    case '4':
      digitalWrite(LED_2, LOW); // ถ้าเป็น 4 led2 ดับ
      break;
    case '5':
      digitalWrite(LED_3, HIGH); // ถ้าเป็น 5 led3 ติด
      break;
    case '6':
      digitalWrite(LED_3, LOW); // ถ้าเป็น 6 led3 ดับ
      break;
    case '7':
      digitalWrite(LED_4, HIGH); // ถ้าเป็น 7 led4 ติด
      break;
    case '8':
      digitalWrite(LED_4, LOW); // ถ้าเป็น 8 led4 ดับ
      break;
    case '9':
      digitalWrite(LED_1, HIGH); // ถ้าเป็น 9 All LED On
      digitalWrite(LED_2, HIGH);
      digitalWrite(LED_3, HIGH);
      digitalWrite(LED_4, HIGH);
      break;
    case '0':
      digitalWrite(LED_1, LOW); // ถ้าเป็น 0 All LED Off
      digitalWrite(LED_2, LOW);
      digitalWrite(LED_3, LOW);
      digitalWrite(LED_4, LOW);
      break;
  }
  line = ""; // เคลียข้อมูลในตัวแปร line ให้เท่ากับ "" เพื่อรอรับค่าใหม่ใน loop หน้า
  Serial.println();
  Serial.println("Closing Connection");
}
```

- เป็นการ get ค่าจาก Server โดยส่ง http request ขอดูในข้อมูลล่าสุดของ field 1 ถ้าส่ง request ถูกต้อง Server ก็จะส่ง http response กลับมาเป็นค่าที่เราต้องการ
- เมื่อได้ค่าที่ต้องการแล้วก็ทำการ substring ก็คือตัดเอาข้อความเอาเฉพาะที่จำเป็นมาใช้งาน
- การทดสอบ POST ค่า 9 = All On

https://api.thingspeak.com/update?api_key=M5ZIAIHP8DOEF62D&field3=9

- การทดสอบ POST ค่า 0 = All Off

https://api.thingspeak.com/update?api_key=M5ZIAIHP8DOEF62D&field3=0

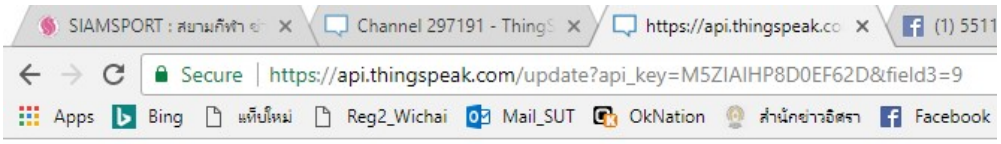
6. การควบคุมเปิดปิด LED และเก็บค่า Temperature, Humidity ผ่าน Internet ด้วย ESP32

- ใช้วงจรเดิม
- ทดสอบโปรแกรมตัวอย่างนี้
  - แก้ไข SSID, Password
  - แก้ไข API Key
  - แก้ไข USER ID และค่า Fields

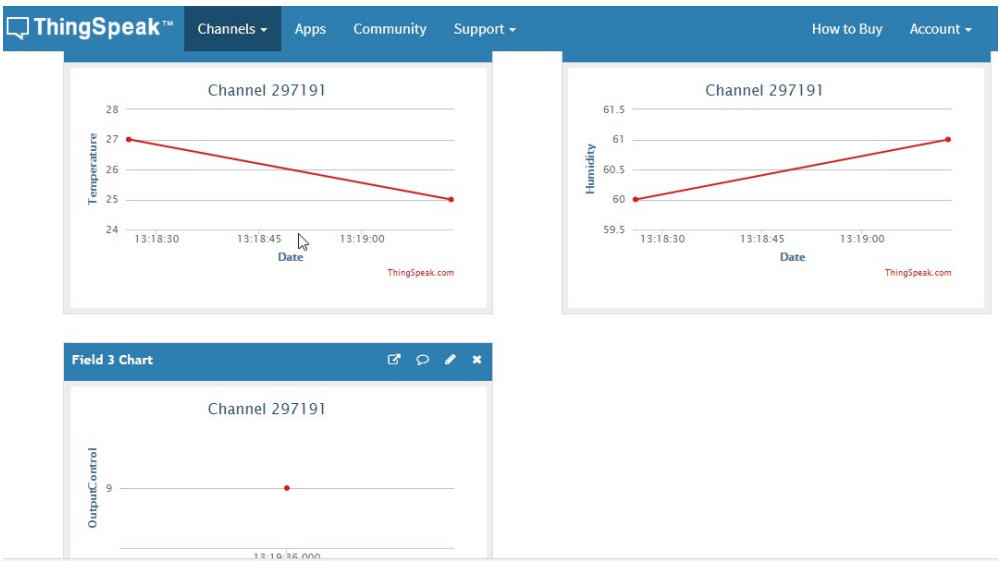
```
#include <WiFi.h>
#include <dht11.h>

#define DHT11_PIN 15
#define LED_1 4
#define LED_2 5
#define LED_3 21
#define LED_4 23

const char* ssid          = "Wifi-4837";
const char* password      = "817094837";
String apiKey             = "M5ZIAIHP8D0EF62D";
const char* url           = "/channels/297191/fields/3/last";
const char* host          = "api.thingspeak.com";

String line = "";
const int PORT = 80;

dht11 DHT;
WiFiServer server(80);
WiFiClient client;

void setup() {
  Serial.begin(115200);
  delay(10);
  pinMode(LED_1, OUTPUT);
  pinMode(LED_2, OUTPUT);
  pinMode(LED_3, OUTPUT);
  pinMode(LED_4, OUTPUT);
  Serial.println("\n\n");
  Serial.print("Connecting to ");
  Serial.println(ssid);
  WiFi.begin(ssid, password);
  while (WiFi.status() != WL_CONNECTED) {
    delay(500);
    Serial.print(".");
  }
  Serial.println();
  Serial.println("WiFi Connected!!");
  Serial.print("IP address : ");
  Serial.println(WiFi.localIP());
}

void loop()
{ Serial.println(" - POST ----------------------------------");
  PostData();
  DelayShow(30);
  Serial.println(" - GET ----------------------------------");
  GetData();
  DelayShow(30);
}

//=====================================================
void DelayShow(int nSec)
{ Serial.print("Delay >>");
  for (int i = nSec; i >= 0; i--)
  { Serial.print(" ");
    Serial.print(i);
    delay(1000);
  }
```

```
  Serial.println();
}

//================================================
void PostData(void)
{ Serial.println("Requesting temperatures...");
  DHT.read(DHT11_PIN);
  float cTemp = DHT.temperature;
  float Humid = DHT.humidity;
  Serial.print("Temperature is: "); Serial.println(cTemp, 2);
  Serial.print("Humidity is: ");     Serial.println(Humid, 2);
  if (client.connect(host, 80))
  { Serial.print("On POST..... ");
    String postStr = apiKey;
    postStr += "&field1=";    // Fields 1
    postStr += String(cTemp);
    postStr += "&field2=";    // Fields 2
    postStr += String(Humid);
    postStr += "\r\n\r\n";
    client.print("POST /update HTTP/1.1\n");
    client.print("Host: api.thingspeak.com\n");
    client.print("Connection: close\n");
    client.print("X-THINGSPEAKAPIKEY: " + apiKey + "\n");
    client.print("Content-Type: application/x-www-form-urlencoded\n");
    client.print("Content-Length: ");
    client.print(postStr.length());
    client.print("\n\n");
    client.print(postStr);
    Serial.println("POST Compleat");
  }
  client.stop();
}

//================================================
void GetData(void)
{ WiFiClient client = server.available();
  Serial.print("Connecting to ");      Serial.println(host);
  Serial.print("Requesting URL : ");  Serial.println(url);
  if (!client.connect(host, PORT))
  { Serial.print(".");
    return;
  }
  else
  { client.print(String("GET ") + url + " HTTP/1.1\r\n" + "Host: " + host + "\r\n" + "Connection: close\r\n\r\n");
    delay(50);
    Serial.println("SEND !!");  // เมื่อส่งไปแล้วให้แสดงข้อความ SEND !! ออกมา
  }

  int i = 400;
  while (!client.available())  //ถ้าหากไม่มี response ตอบกลับมา ก็จะให้วนลูป จนกว่าจะมี responce ตอบกลับมา
  { delay(10);
    i--;
    if (i == 0) return;
  }

  while (client.available())  // และเมื่อมี response ตอบกลับมาก็จะให้เข้า loop while แล้วทำการเก็บค่าทั้งหมดไว้ที่ line
  { line += (char)client.read();
  }

  Serial.println();
  Serial.println("GET SUCCESSFULLY !!");
  Serial.print("###### rawData  >> ");  Serial.println(line);
  int lengthData = line.length();
  char data = line[lengthData - 1];
  String s = (String)"Position " + lengthData + (String)", Data " + data;
  Serial.print("###### splitted >> ");  Serial.println(s);

  switch (data) {
    case '1':
      digitalWrite(LED_1, HIGH); // ถ้าเป็น 1 led1 ติด
      break;
    case '2':
      digitalWrite(LED_1, LOW); // ถ้าเป็น 2 led1 ดับ
      break;
```

```
    case '3':
      digitalWrite(LED_2, HIGH); // ถ้าเป็น 3 led2 ติด
      break;
    case '4':
      digitalWrite(LED_2, LOW); // ถ้าเป็น 4 led2 ดับ
      break;
    case '5':
      digitalWrite(LED_3, HIGH); // ถ้าเป็น 5 led3 ติด
      break;
    case '6':
      digitalWrite(LED_3, LOW); // ถ้าเป็น 6 led3 ดับ
      break;
    case '7':
      digitalWrite(LED_4, HIGH); // ถ้าเป็น 7 led4 ติด
      break;
    case '8':
      digitalWrite(LED_4, LOW); // ถ้าเป็น 8 led4 ดับ
      break;
    case '9':
      digitalWrite(LED_1, HIGH); // ถ้าเป็น 9 All LED On
      digitalWrite(LED_2, HIGH);
      digitalWrite(LED_3, HIGH);
      digitalWrite(LED_4, HIGH);
      break;
    case '0':
      digitalWrite(LED_1, LOW); // ถ้าเป็น 0 All LED Off
      digitalWrite(LED_2, LOW);
      digitalWrite(LED_3, LOW);
      digitalWrite(LED_4, LOW);
      break;
  }
  line = ""; // เคลียข้อมูลในตัวแปร line ให้เท่ากับ "" เพื่อรอรับค่าใหม่ใน loop หน้า
  Serial.println();
  Serial.println("Closing Connection");
}
```



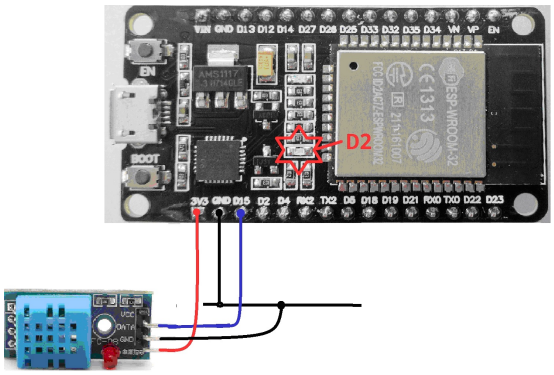
All On → https://api.thingspeak.com/update?api_key=M5ZIAIHP8D0EF62D&field3=9



PCเลขที่ _____รหัส_____ ชื่อ-สกุล _____

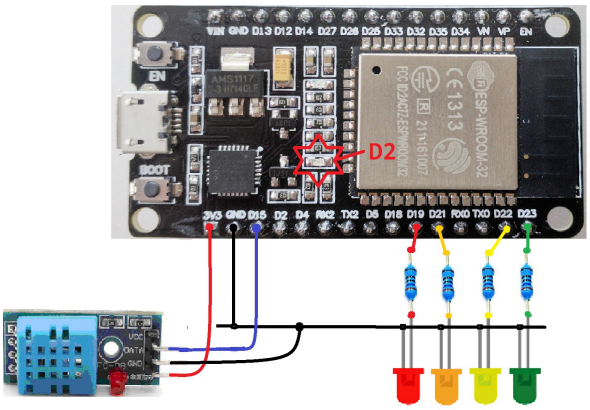
4. Exercise

1. ปรับแก้โปรแกรมให้ส่งค่าอุณหภูมิความชื้นขึ้นไปยัง ThingSpeak



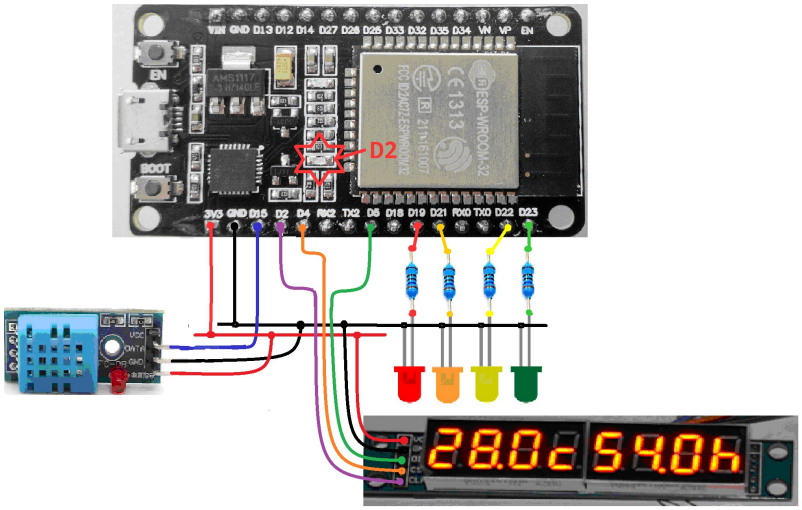
- GPIO D15 → DHT11

2. จากข้อ 1 เพิ่มเติมให้อ่านให้ส่งค่าอุณหภูมิความชื้นจาก DHT-11 และควบคุม 4 LED



- GPIO D15 → DHT11
- GPIO D19 → R
- GPIO D21 → O
- GPIO D22 → Y
- GPIO D23 → G

3. จากข้อ 2 เพิ่มเติมให้แสดงอุณหภูมิ ความชื้นด้วย



- D15 → DHT11
- GPIO D19 → R
- GPIO D21 → O
- GPIO D22 → Y
- GPIO D23 → G
- GPIO D6 → DIN
- GPIO D4 → CS
- GPIO D2 → CLK