NODEMCU FOR INTERNET OF THING

Basic IoT Training Book

Document Version 1.0 by Maximus

Worachet@MaxInnoTech.com

Contents

DO NOT COPY

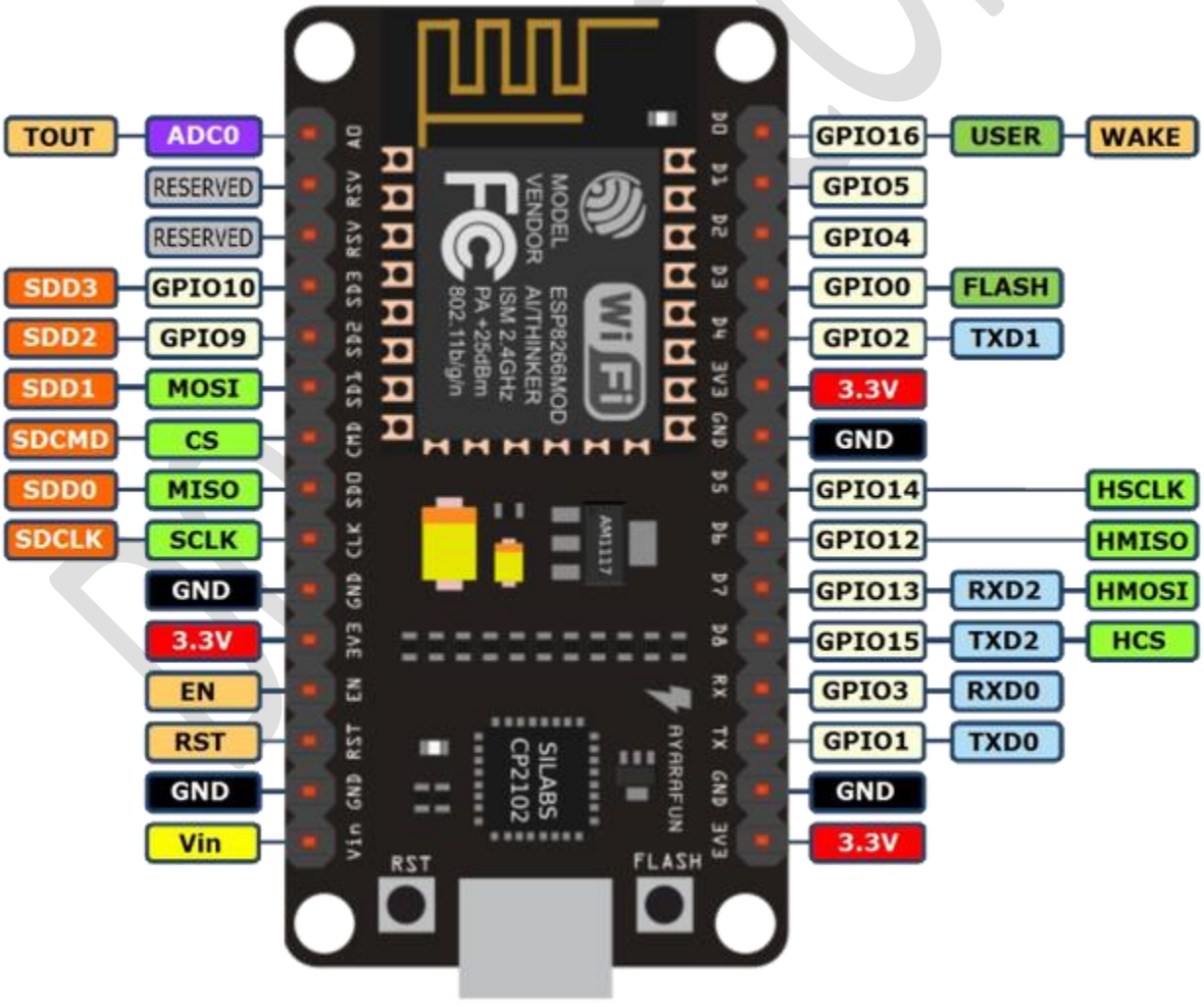
DO NOT COPY

การใช้งาน NodeMCU เบื้องต้น

มารู้จัก NodeMCU Dev Kit V1.0

NodeMCU นั้นเป็น open-source hardware and firmware ซึ่งเป็นบอร์ดที่ช่วยให้เราสามารถใช้งานเกี่ยวกับ IoT (Internet of Thing) ได้อย่างง่าย เพราะการเขียนโปรแกรมต่าง ๆ ควบคุม GPIO ของชิพ ที่ใช้ภาษา Lua ในการเขียน script เพียงไม่กี่บรรทัด หรือใช้ C ก็ได้เช่นกัน

โดยเป็นโมดูลที่ประกอบด้วย ESP8266-12E มีเสาอากาศแบบ PCB Antenna และรองรับรูปแบบการสื่อสารสำหรับขาสัญญาณต่าง ๆ หลาย ๆ ชนิดได้แก่ GPIO, PWM, I2C, 1-wire, ADC และ มี SPI เพิ่มขึ้นมาจาก Version เดิม ในการสื่อสารกับ Computer ส่วนของ USB-to-TTL และพอร์ต micro USB ใช้ชิพ USB to Serial ของ Silicon Lab CP2102 สิ่งที่จำเป็นคือเราต้องรู้รูปแบบการจัดวางขาสัญญาณต่าง ๆ ตามรูป



Features ของ NodeMCU

- Open-source
- สามารถเขียนโปรแกรมสั่งควบคุมได้
- ใช้ต้นทุนต่ำ
- ใช้งานง่าย
- เชื่อมต่อ Wi-Fi ได้
- Smart
- Interactive

สามารถควบคุม hardware I/O เหมือนกับ Arduino

มี API สำหรับการควบคุม I/O ของ Hardware ซึ่งจะช่วยลดการทำงานที่ซ้ำซ้อนสำหรับการกำหนดค่าและจัดการพวก Hardware สามารถเขียนโปรแกรมเช่นเดียวกับ Arduino ได้

เป็นอุปกรณ์ WiFi ที่ใช้ต้นทุนต่ำ

ต้นทุนน้อยกว่า 2 ดอลลาร์ NodeMCU WiFi MCU ESP8266 ตัวนี้ผลิตออกมาได้ครบวงจร สามารถพัฒนาต่อได้ง่ายในงาน IoT และต้นทุนต่ำ ตอนนี้ขายประมาณ 400 บาท

Specification ของบอร์ด

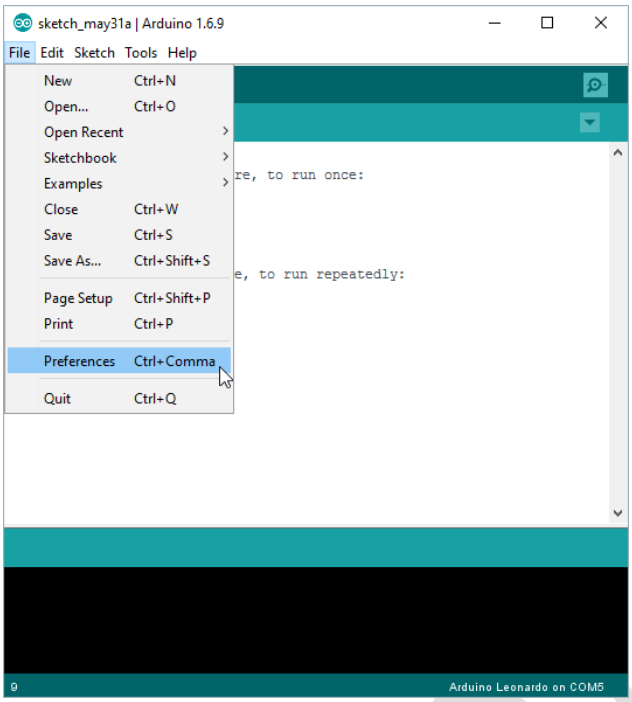
เป็นชุดพัฒนาที่เป็น Base on ESP8266 มี GPIO, PWM , I2C , 1-Wire และ ADC รวมทั้งหมดในบอร์ดเดียว เป็นช่องทางที่ช่วยในการพัฒนาที่รวดเร็วด้วย NodeMCU firmware!

- USB-TTL included, plug&play (CP2102)
- ESP-12E 32 Mbps
- 10 GPIO, ทุก ๆ GPIO สามารถเป็น PWM, I2C, 1-wire
- 1 ADC
- WI-FI module
- PCB antenna

DO NOT COPY

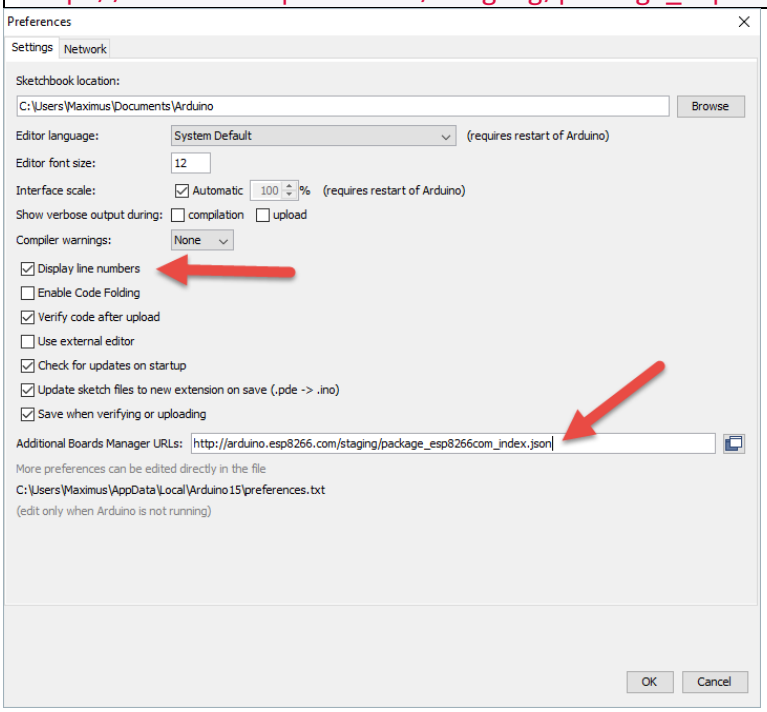
ขั้นตอนติดตั้ง NodeMCU ใน Arduino IDE

ไปที่ Arduino IDE เลือก Preference



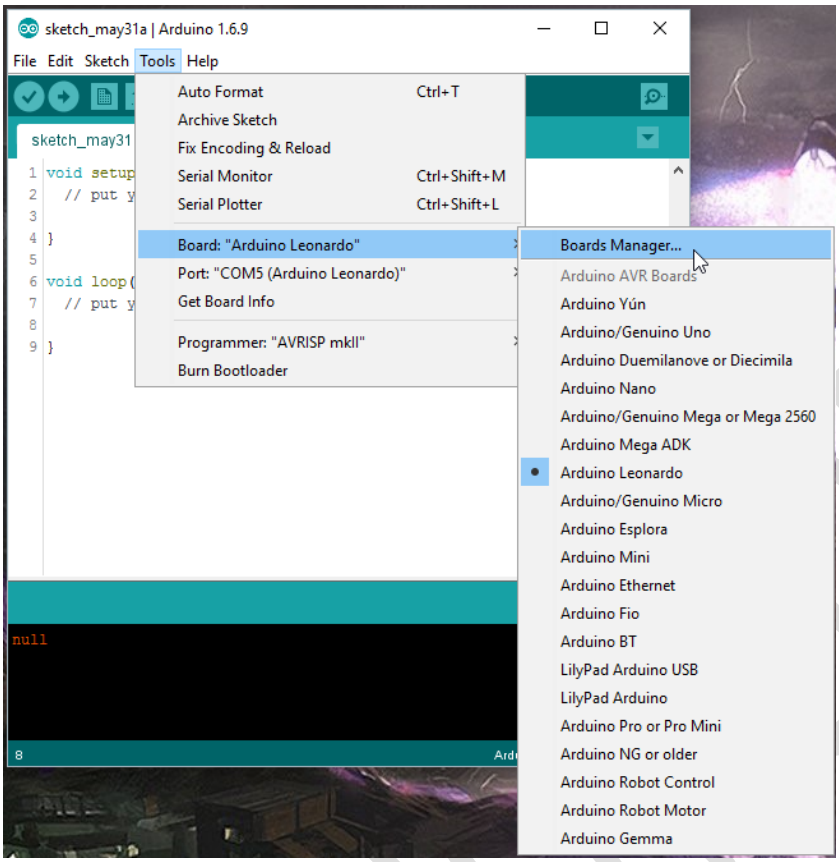
ทำการป้อน Additional Boards Manager URL

```
http://arduino.esp8266.com/staging/package_esp8266com_index.json
```

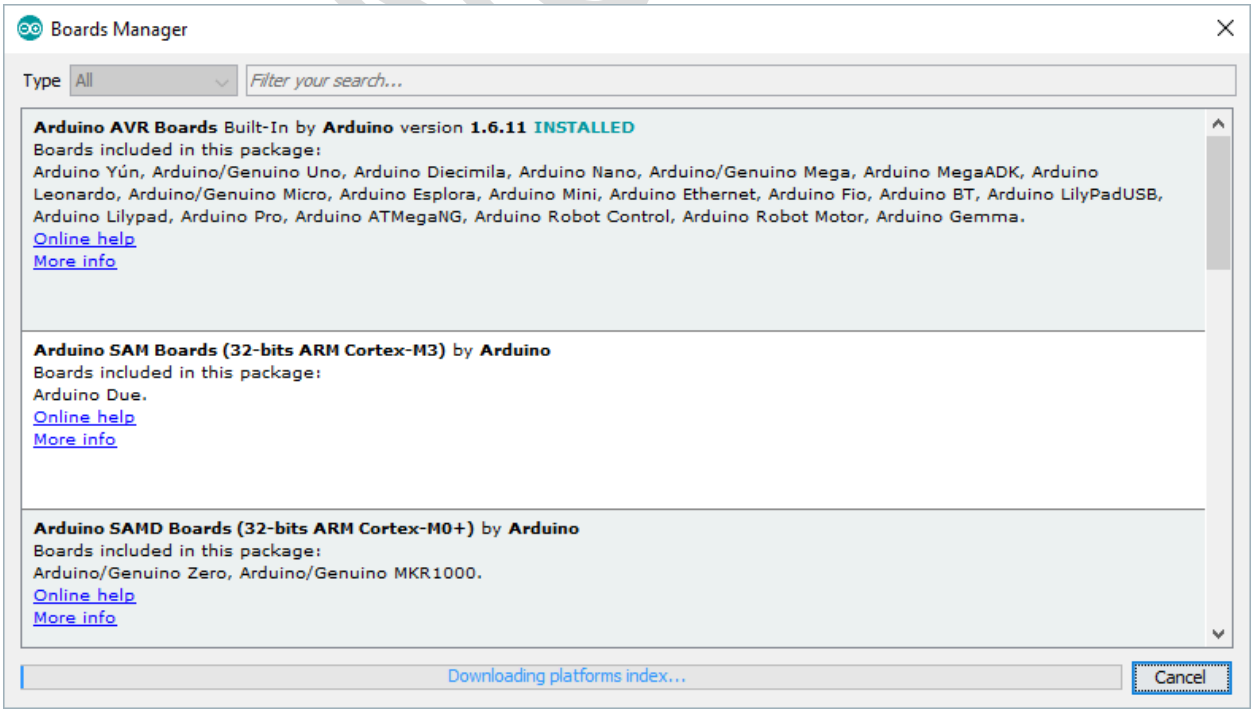


COPY

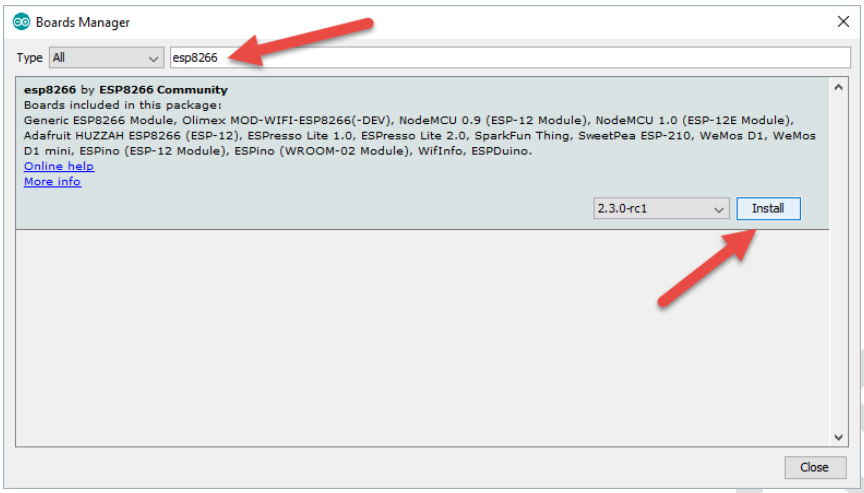
หลังจากนั้นไปที่ Menu Tools > Board Manager...



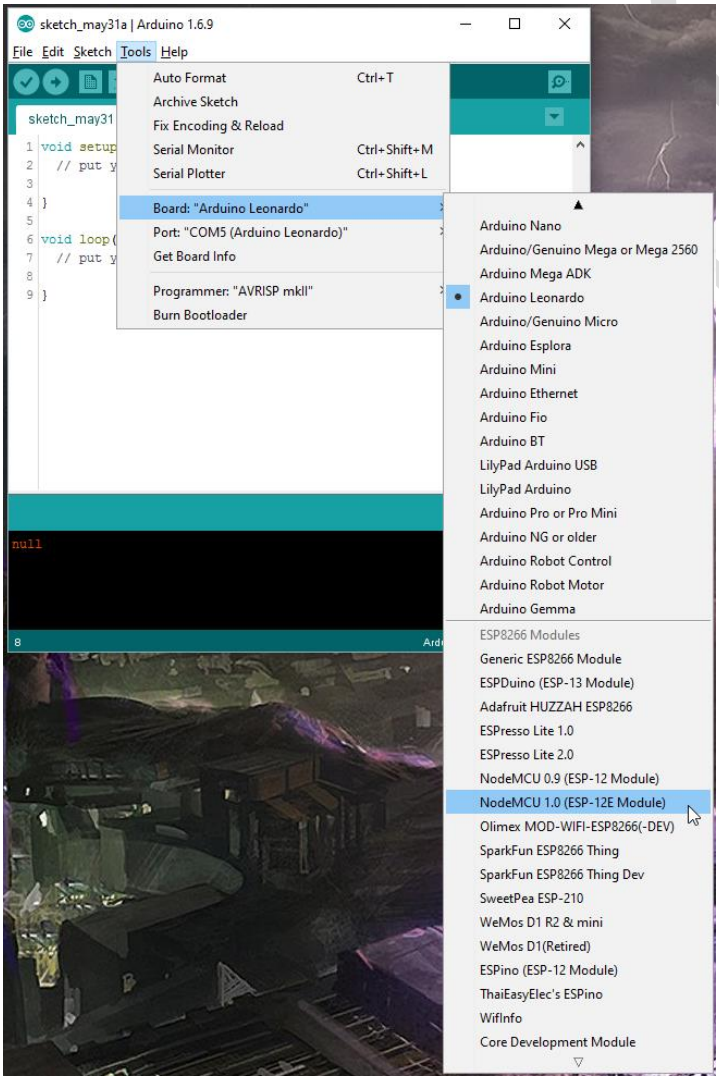
จะขึ้นหน้าจอ Board Manager



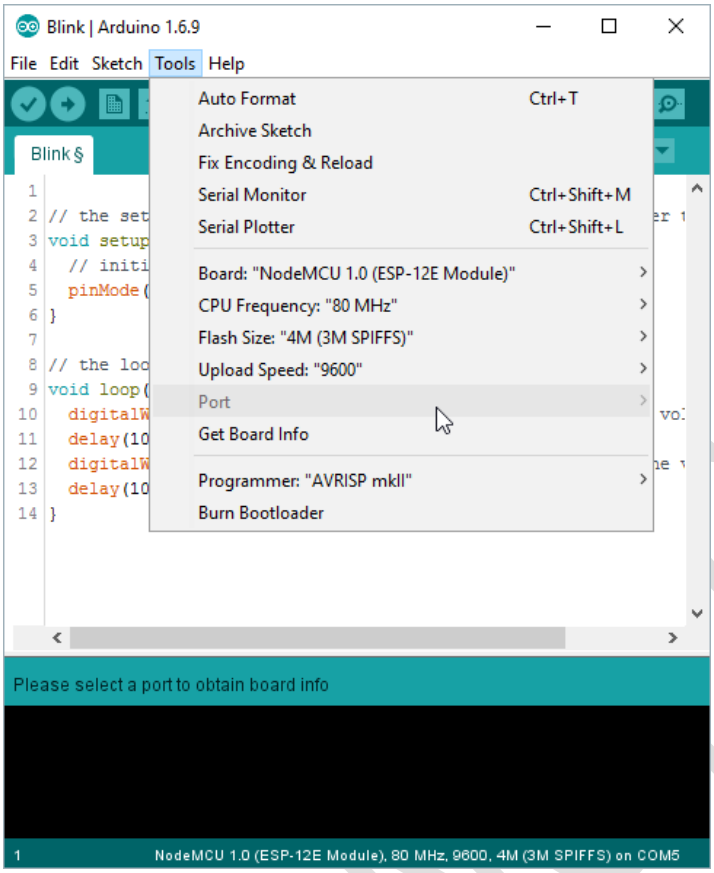
ใส่ ESP8266 เพื่อค้นหา package จากนั้น install version ล่าสุด



เมื่อ Install เสร็จสิ้นแล้วจะมี Board NodeMCU ขึ้นมา



ลองเสียบสาย USB ที่ NodeMCU หากไม่พบ Port ที่เชื่อมต่อ ให้เช็คในส่วนของ Device Manager ว่า Windows ตรวจเจอ Driver ของ Chip CP2102 หรือไม่ ปัญหาส่วนใหญ่คือ Windows ไม่เจอ Driver ของ CP2102

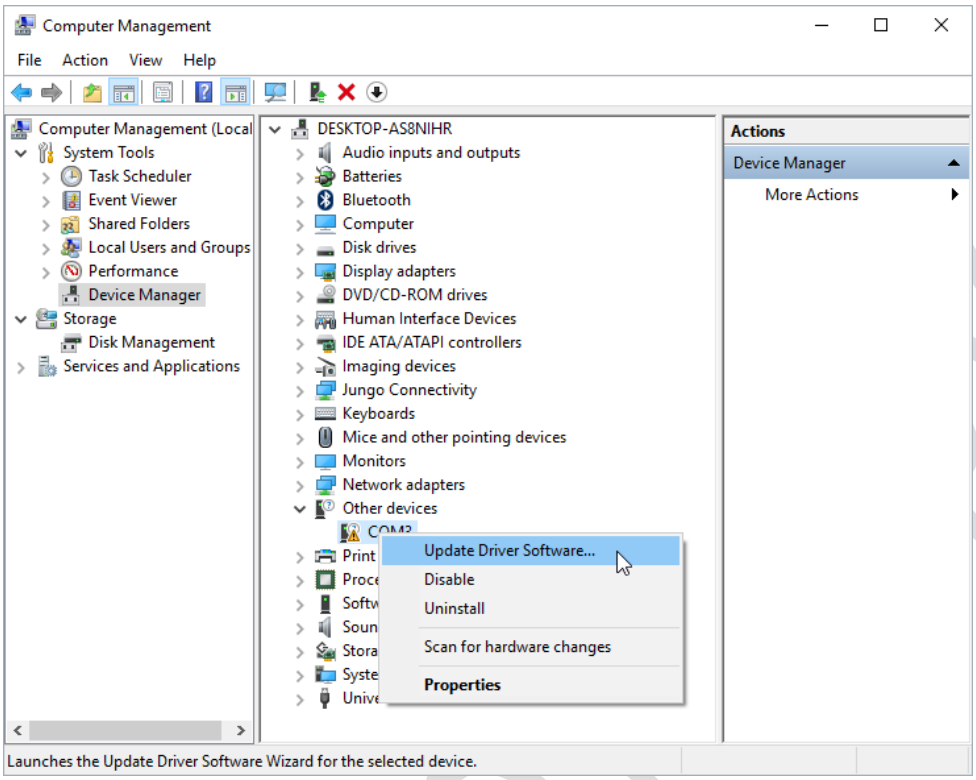


ให้ทำการ Download Driver CP2102 มาติดตั้ง

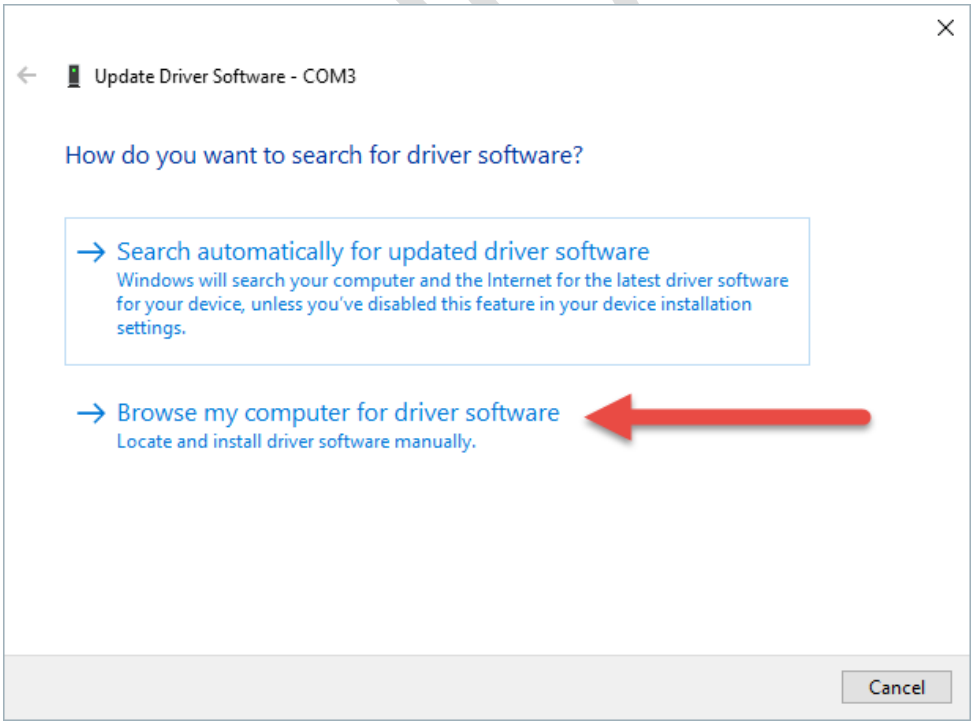
Name	Date modified	Type	Size
x64	4/11/2014 4:56 PM	File folder	
x86	4/11/2014 4:56 PM	File folder	
CP210xVCPInstaller_x64	5/31/2016 10:58 PM	Application	1,026 KB
CP210xVCPInstaller_x86	5/31/2016 10:58 PM	Application	901 KB
dpinst	5/31/2016 10:58 PM	XML Document	12 KB
ReleaseNotes	5/31/2016 10:58 PM	Text Document	11 KB
SLAB_License_Agreement_VCP_Windows	5/31/2016 10:58 PM	Text Document	9 KB
slabvcp	5/31/2016 10:58 PM	Security Catalog	12 KB
slabvcp	5/31/2016 10:58 PM	Setup Information	5 KB

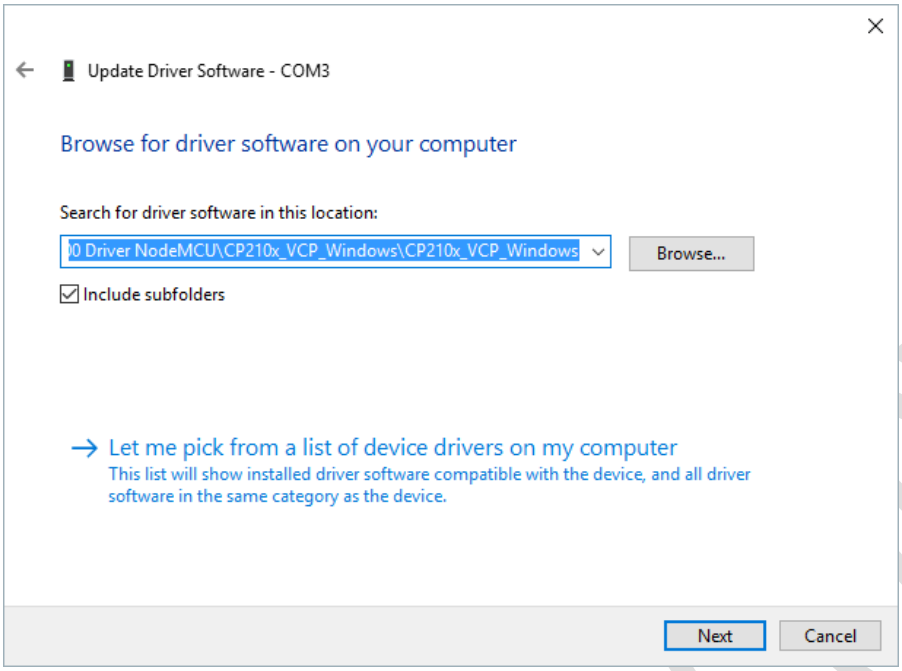
COPY

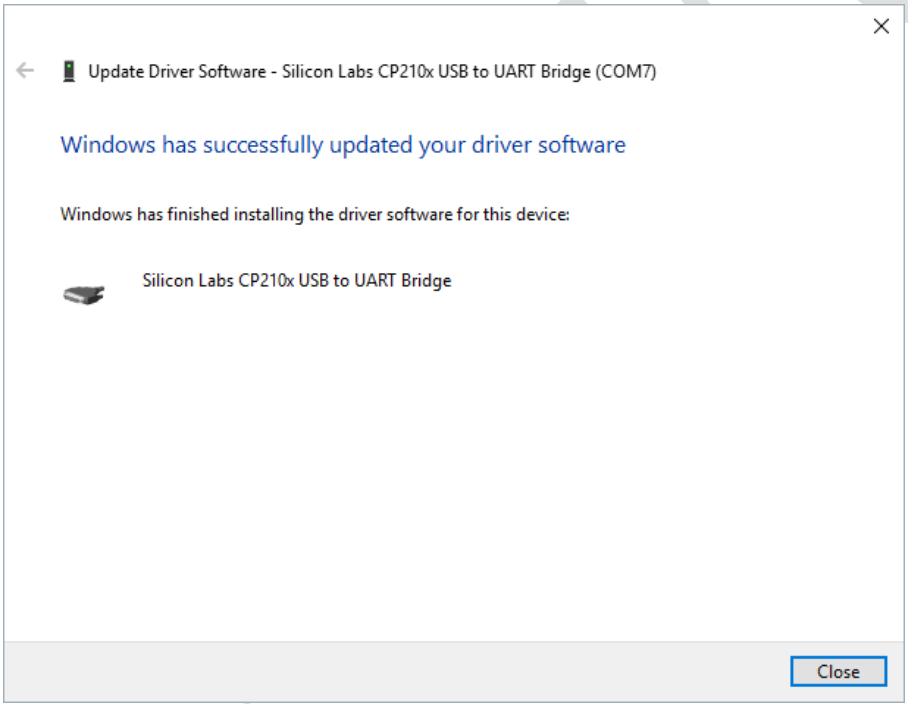
ในกรณีที่ไม่เจอ Com port ต้องทำการ Update USB Driver CP2102

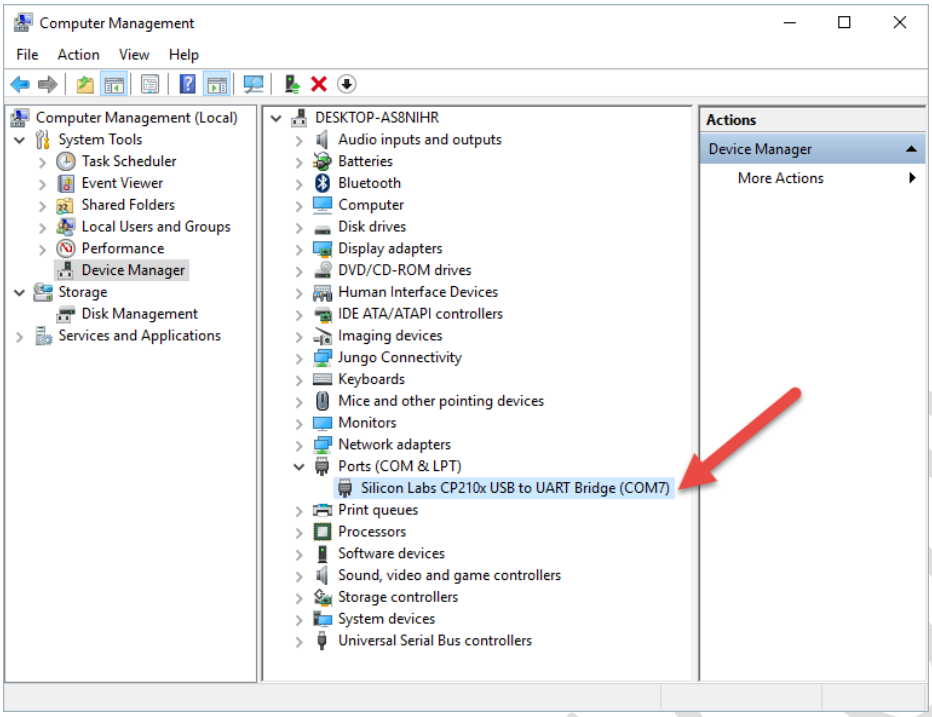


เลือกค้นหาจาก Driver ที่ Copy มาให้

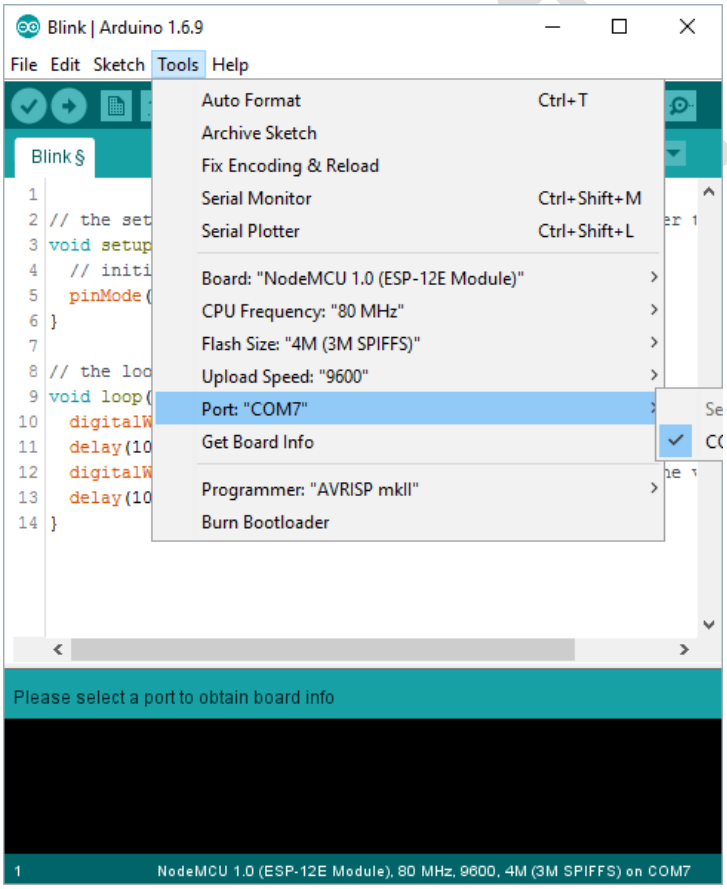



Update Driver Software - COM3
Browse for driver software on your computer
Search for driver software in this location:
0 Driver NodeMCU\CP210x_VCP_Windows\CP210x_VCP_Windows
Browse...
Include subfolders
Let me pick from a list of device drivers on my computer
This list will show installed driver software compatible with the device, and all driver software in the same category as the device.
Next
Cancel


Update Driver Software - Silicon Labs CP210x USB to UART Bridge (COM7)
Windows has successfully updated your driver software
Windows has finished installing the driver software for this device:
Silicon Labs CP210x USB to UART Bridge
Close

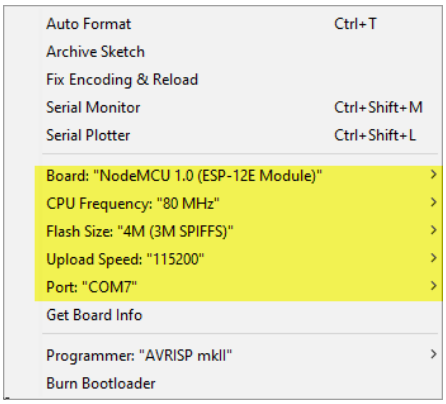


เมื่อติดตั้งเสร็จจะสามารถเลือก Com port ที่เชื่อมต่ออยู่ได้

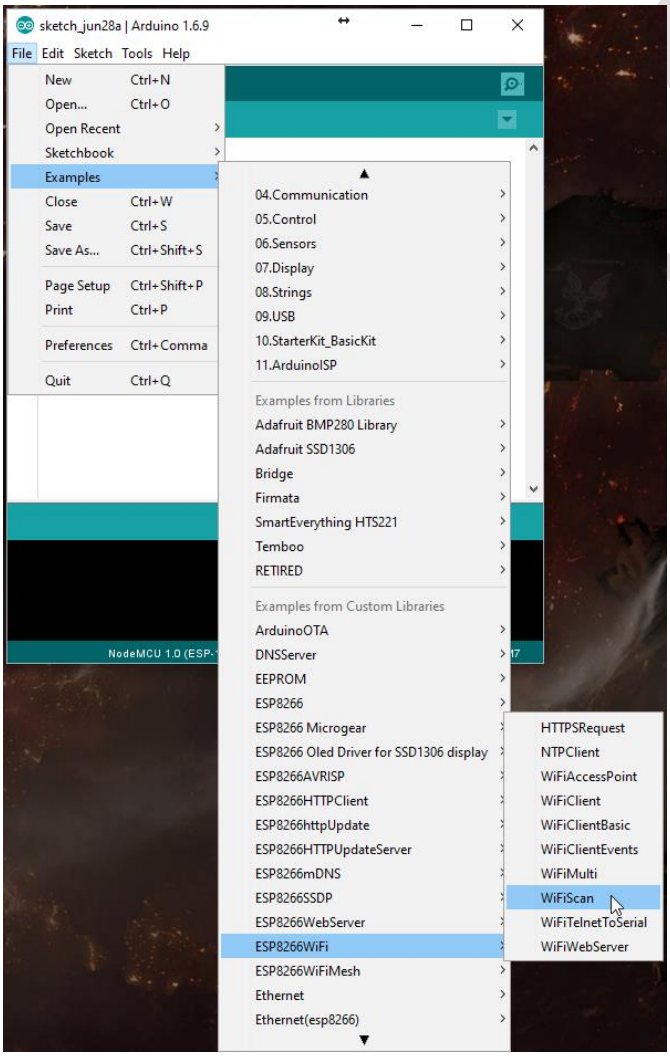


ทดสอบ NodeMCU ด้วยการ Scan Wifi

เลือกบอร์ด NodeMCU โดยเลือก version 1.0 และ Board late อยู่ที่ 115200

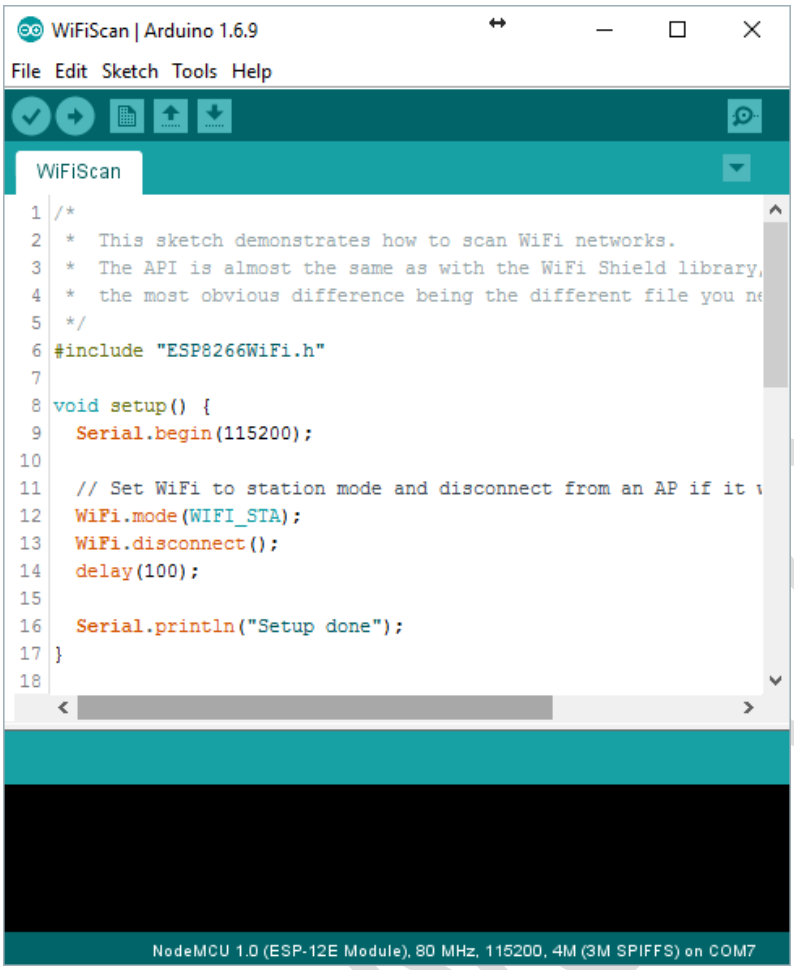


เปิดตัวอย่างในส่วนของ File > Examples > ESP8266WiFi > WiFiScan

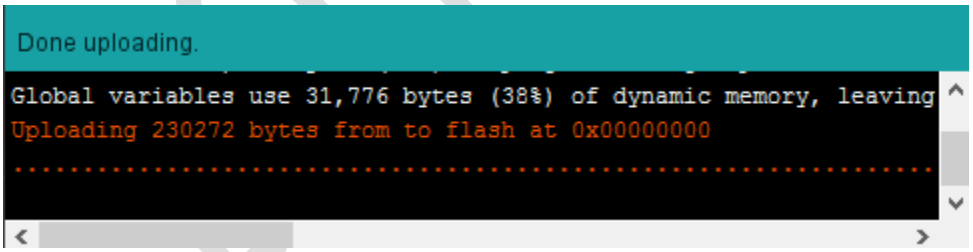


COPY

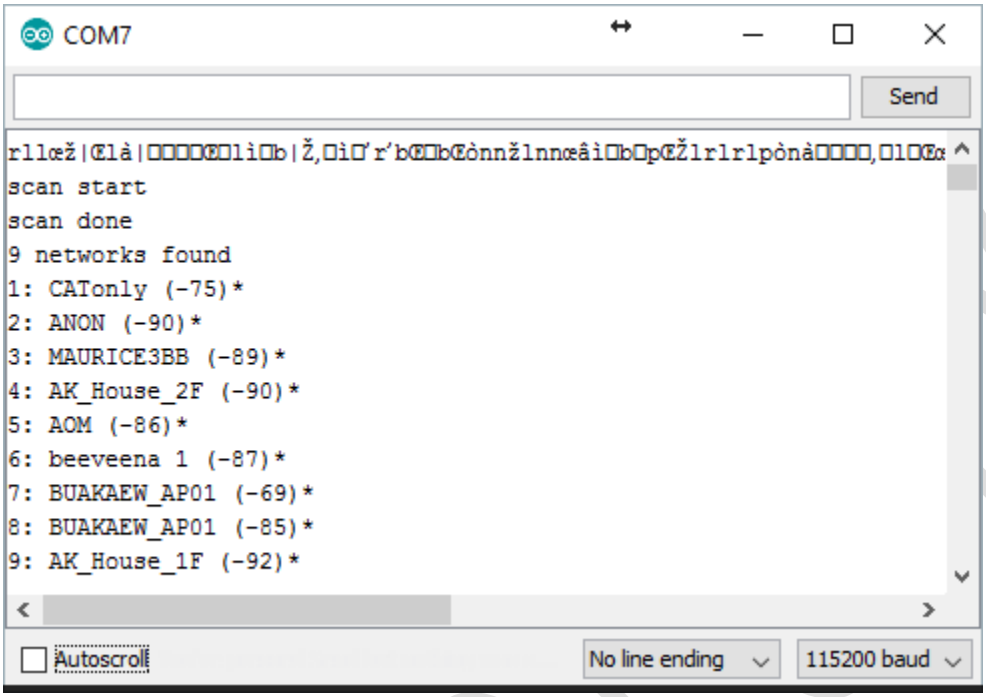
จะได้ code ของ Wi-Fi scan ให้เราทำการคอมไพล์และอัพโหลดเข้าไปในตัว NodeMCU



เมื่อเราทำการอัพโหลดเสร็จสิ้นแล้วในส่วนของ status box จะแสดงข้อความว่า Done uploading



ให้เราเปิด serial monitor จะแสดงรายการของ wifi ที่เปิดอยู่ที่สำคัญคือการเลือกบอร์ดเรทของ serial monitor ต้องตรงกับบอร์ดเรทของ NodeMCU ที่เราใช้ คือ 115200



ถ้าขึ้นตามภาพเป็นอันเสร็จสิ้นกระบวนการเราสามารถใช้งาน NodeMCU ได้

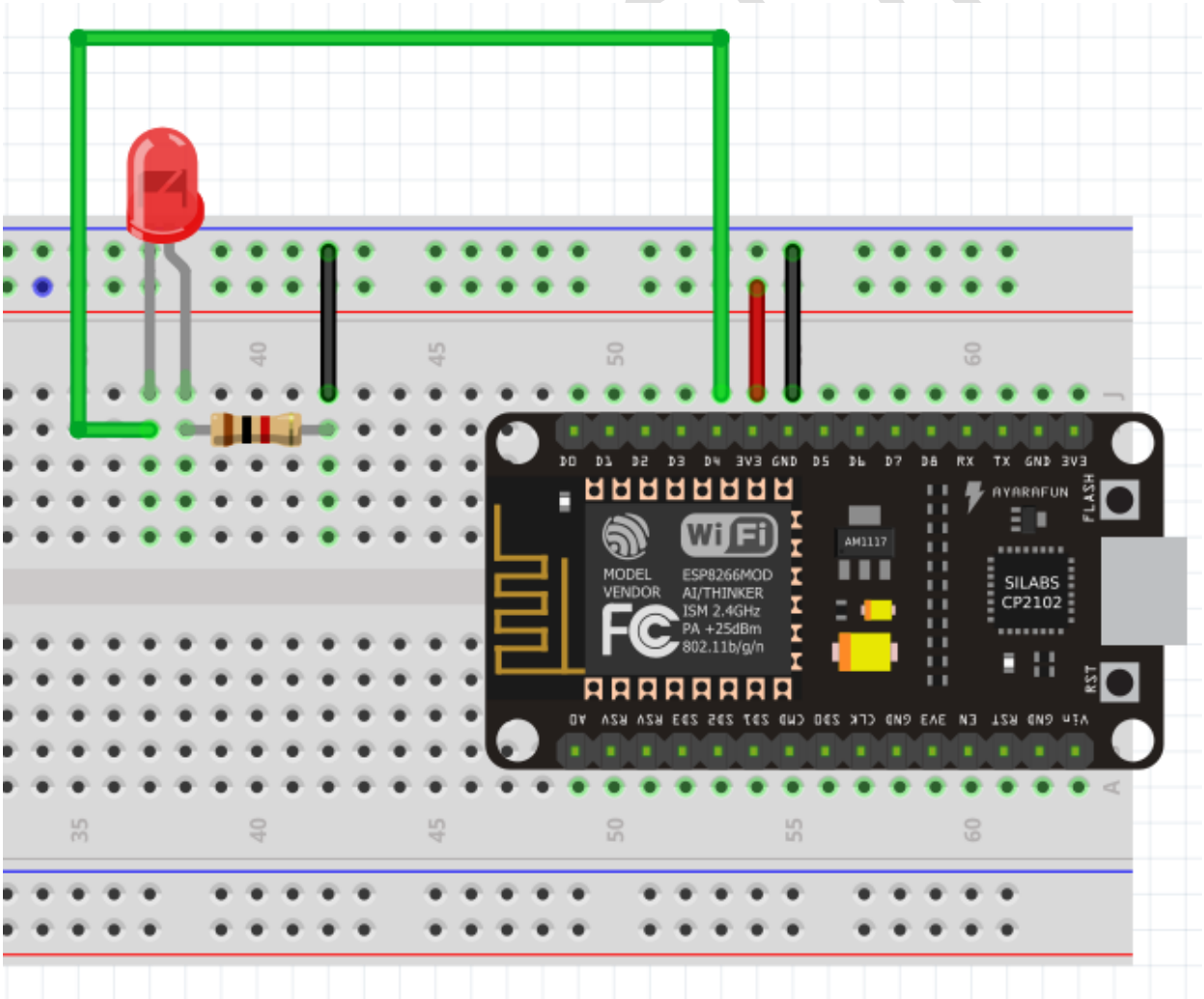
DO NOT COPY

ทดลองใช้งาน NodeMCU

ในการใช้การเชื่อมต่อขาสัญญาณ Input Output ต่าง ๆ จากตัว NodeMCU ปกติแล้วเราสามารถอ้างอิงขาสัญญาณต่าง ๆ ของ NodeMCU ได้โดยใช้ d0 d1 d2 หรืออ้างอิงจาก GPIO ของ ESP8266 โดยตรงเลยก็ได้ ซึ่งจะเป็นตัวเลขอย่างเช่น d0 จะเป็นเลข 16, d1 จะเป็นเลข 5, d2 จะเป็นเลข 4 ในการใช้งานจริงนั้นแนะนำให้ใช้เป็นเลขที่ออกมาจากขา GPIO ของ ESP8266 โดยตรง เพราะในกรณีที่เรานำไปใช้เป็น production เราไม่จำเป็นที่จะต้องแก้โค้ดจาก d0 เป็นเลข 16 เวลาเราใช้ production ส่วนใหญ่เราจะนำ ESP8266 ที่เบิร์น firmware และโปรแกรมของเราเอาไว้แล้วไปใช้งาน

LAB01: Digital Output

ทำการต่อวงจรดังภาพ เปิด ArduinoIDE และเลือก Menu File > Example > Basic > Blink แล้วนำ Code มาแก้ได้



หลักคือ ใช้ function ที่ใช้กำหนด pinMode ให้เป็น OUTPUT และใช้ function digitalwrite ส่งค่าออกเป็น LOW หรือ HIGH

```
pinMode(pout, OUTPUT);
digitalWrite(pout, HIGH);   // LOW, HIGH หรือ 0,1 ก็ได้
```

LAB01: Digital Output

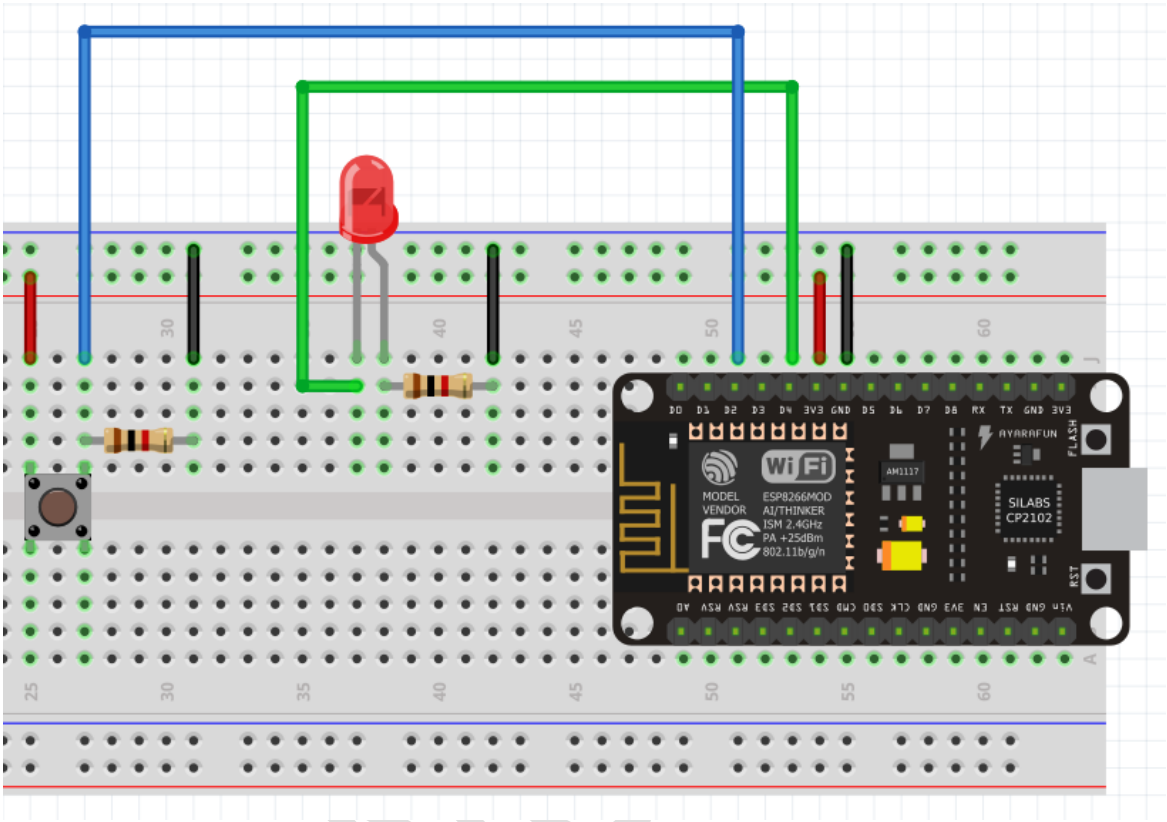
```
const int pout = 2;  // เราสามารถใช้ D4 แทนได้ GPIO 2 นี้จะมีแอลอีดีเชื่อมต่ออยู่บนตัว ESP8266 ด้วย

void setup() {
 pinMode(pOUT, OUTPUT);
}

void loop() {
 digitalWrite(pOUT, HIGH);
 delay(1000);
 digitalWrite(pOUT, LOW);
 delay(1000);
}
```

ผลทดสอบ run ไฟที่ตัวของ ESP8266 จะกระพริบติดดับ 1 วินาที และ LED ที่ต้อไว้จะติดดับสลับกัน

DO NOT COPY

LAB02: Digital Input

ทำการต่อวงจรดังภาพ เปิด ArduinoIDE และเลือก Menu File > Example > Digital > button แล้วนำ Code มาแก้ได้



ในส่วนของ digital in นี้เราจะทดสอบด้วยการต่อ Switch และ LED เมื่อกด Switch แล้ว LED จะติด และเมื่อปล่อย Switch แล้ว LED จะดับ

ข้อควรจำและต้องระวังให้มากที่สุดคือการดีไซน์วงจรใช้งานแล้วทำให้ Short Circuit หรือแหล่งจ่ายไฟไปเชื่อมต่อกับกราวด์ Ground โดยตรงต้องระวังไว้ให้มากถ้าไม่มั่นใจไม่ควรเสียบโดยตรงกับ NodeMCU ให้ทำการต่อวงจรทดสอบข้างนอกก่อน

หลักคือ ใช้ function กำหนด pinMode ให้เป็น INPUT และใช้ function digitalread จากพอร์ตที่ต้องการอ่านค่า ซึ่งจะได้ค่าออกมาเป็น logic LOW หรือ HIGH

```
pinMode(ledPin, INPUT);
digitalWrite(buttonPin);   //ค่าที่ได้จะออกมาเป็น logic LOW, HIGH
```

LAB02: Digital Input

```
const int buttonPin = 0;     // the number of the pushbutton pin
const int ledPin =  2;       // the number of the LED pin
int buttonState = 0;         // variable for reading the pushbutton status

void setup() {
 pinMode(ledPin, OUTPUT);     // initialize the LED pin as an output:
 pinMode(buttonPin, INPUT);   // initialize the pushbutton pin as an input:
}

void loop() {
 buttonState = digitalRead(buttonPin);    // read the state of the pushbutton value:
 if (buttonState == HIGH) {               // if it is, the buttonState is HIGH:
   digitalWrite(ledPin, HIGH);            // turn LED on:
 } else {
   digitalWrite(ledPin, LOW);             // turn LED off:
 }
}
```

DO NOT COPY

LAB03: การส่งค่าออก Serial

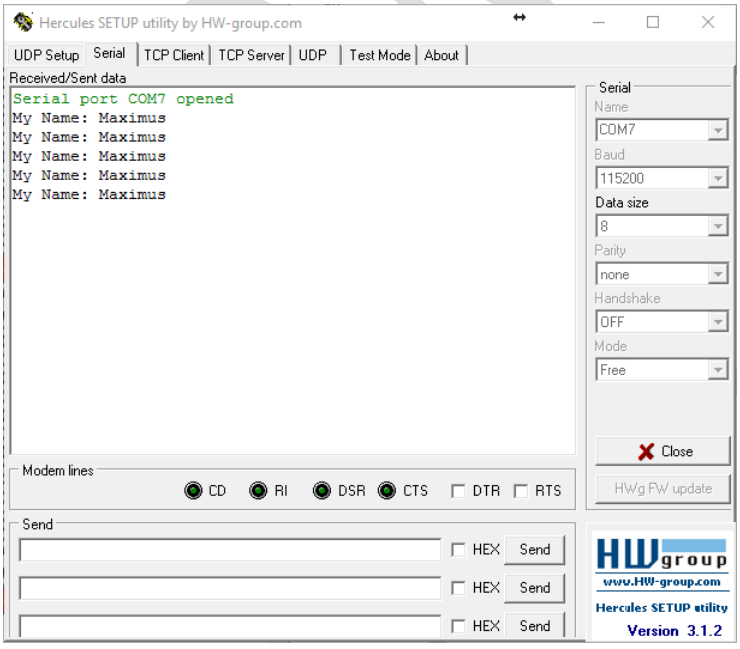
ในการพัฒนาโปรแกรมนั้นหลายครั้งที่เราจะต้องส่งค่าผ่าน serial เพื่อให้เรารู้การทำงานของโปรแกรมที่เราใส่ไว้ใน NodeMCU เราจึงจำเป็นต้องรู้ลักษณะการกำหนดค่า serial และการใช้งาน serial เบื้องต้น

อย่างแรกที่จำเป็นต้องกำหนดคือการกำหนด Baud Rate ของ serial ส่วนใหญ่เรามักจะกำหนดให้ฟังก์ชัน setup ในที่นี้ Baud Rate ที่จะใช้กับ NodeMCU คือ 115200 และส่วนที่ 2 คือการนำค่าของ serial มาแสดงออกทาง serial monitor ด้วยคำสั่ง serial print โดยจะมี 2 ฟังก์ชันหลักคือ serial.print() คือการแสดงผลโดยไม่ขึ้นบรรทัดใหม่ และ serial.println() คือการแสดงผลโดยขึ้นบรรทัดใหม่

LAB03: Send to Serial

```
void setup() {
 Serial.begin(115200);      // initialize serial communication at 115200 bits per second:
}
void loop() {
 Serial.print("My Name: ");  // แสดงผลโดยไม่ขึ้นบรรทัดใหม่
 Serial.println("Maximus");  // แสดงผลแล้วขึ้นบรรทัดใหม่
 delay(5000);        // delay in between reads for stability
}
```

ผลที่ได้คือเมื่อเปิดโปรแกรม hercules จะขึ้นข้อความ “My Name: Maximus” แล้วขึ้นบรรทัดใหม่ไปได้เรื่อยๆ



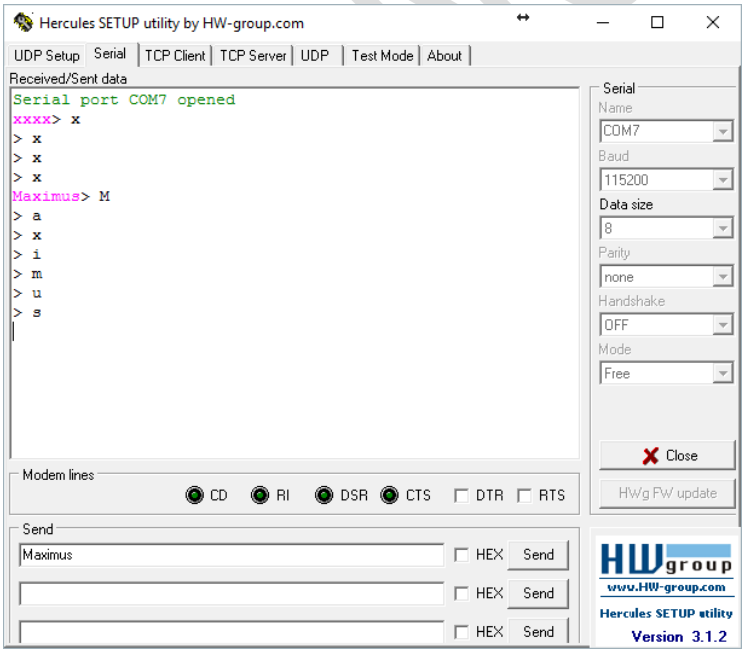
LAB04: การอ่านค่าจาก Serial

การอ่านค่าจาก Serial หลังจากที่เราส่งออกไปได้แล้ว มาทดลองรับค่าจากคอมพิวเตอร์บ้าง วิธีการรับข้อมูลเข้าจาก คอมพิวเตอร์ จะใช้ฟังก์ชั่น Serial.Read() ซึ่งในตัวอย่างนี้ช่วยให้เข้าใจมากขึ้น หลังจาก upload แล้วเปิด Serial Monitor serial.read() จะเป็นฟังก์ชั่น ที่คืนค่ารับมาจาก serial ซึ่งในที่นี้เป็นตัวอักษร จะนำมาใช้งานเราจึง ต้องประกาศตัวแปร มารับค่า และนำค่าไปประมวลผลต่อไป สำหรับในโปรแกรมนี้ เราได้รับค่า และพิมพ์ค่า ออก Serial Monitor โดยเพิ่ม “>” เข้าไปข้างหน้าด้วย

LAB04: Read from Serial

```
void setup() {  Serial.begin(115200);
 Serial.println("Hello Please Type"); }
void loop() {
 if (Serial.available() > 0)  // Function available มีหน้าที่ตรวจสอบว่ามีการส่งค่าออกมาจาก serial ไหม ถ้ามีการส่งค่ามาจะ return value ที่มีค่ามากกว่า 0
 {  char incomingByte = Serial.read();  //  serial.print() ทำหน้าที่อ่านค่าจาก serial  ทีละตัวอักษร
   Serial.print ("> ");
   Serial.println (incomingByte);  // แสดงค่าตัวอักษรที่รับมาจาก serial
} }
```

ผลที่ได้เมื่อเราส่งข้อความออกไปตัว NodeMCU จะส่งค่ากลับมาทีละ Byte โดยใช้โปรแกรม Hercules



LAB05: ทดลอง NodeMCU รับค่าจาก serial เพื่อเปิดปิดไฟ

การใช้งาน NodeMCU ควบคุม LED ผ่าน Serial ถือเป็นการเรียนรู้การรับส่งข้อมูลระหว่าง คอมพิวเตอร์กับ NodeMCU หากสามารถเข้าใจการรับส่งข้อมูลแบบ Serial แล้ว การนำไปประยุกต์ใช้กับงาน ต่าง ๆ ก็จะสามรถทำได้ง่ายมากยิ่งขึ้น

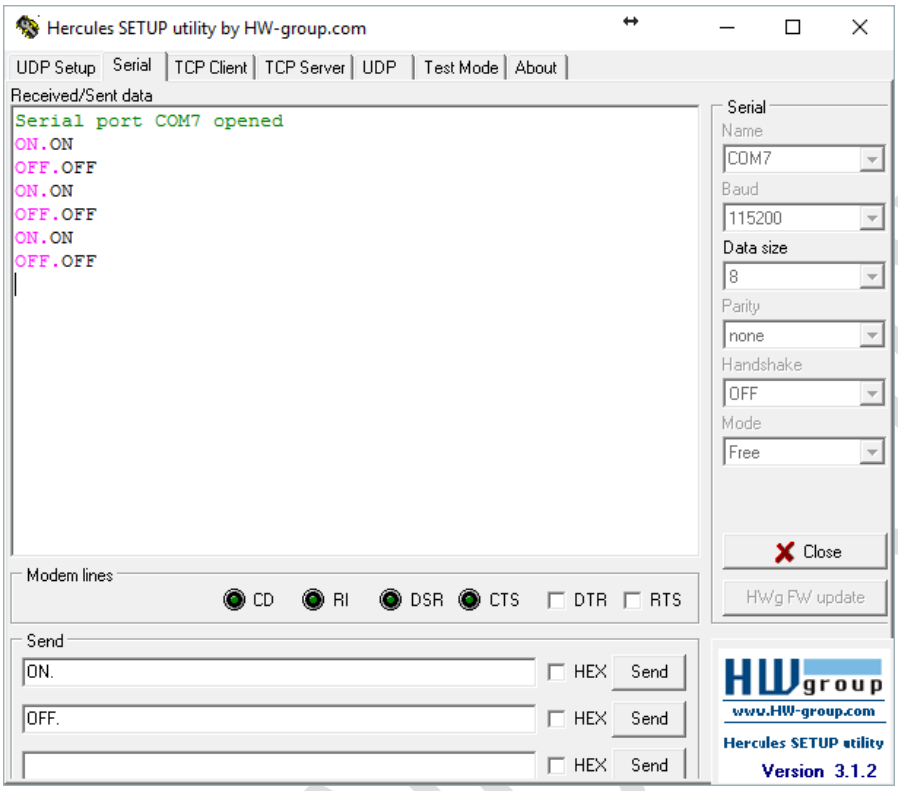
LAB05: Control LED by Serial

```
#define LED1_Pin 2
String SerialGET = "";
void setup() {
 Serial.begin(115200);
 pinMode(LED1_Pin, OUTPUT);
}

void loop(){
while (Serial.available()) {
   char c = Serial.read();
   if (c == '.') {
       Serial.println(SerialGET);
       if (SerialGET.indexOf("ON") >= 0)
           digitalWrite(LED1_Pin, HIGH);
       else if (SerialGET.indexOf("OFF") >= 0)
           digitalWrite(LED1_Pin, LOW);
       SerialGET = "";
   }else
   SerialGET += c;
  }
}
```

ผลการทดลอง เมื่อเราทำการ Run ให้เปิด Serial Monitor ขึ้นมา แล้วพิมพ์ “ON.” กดปุ่ม Send เพื่อเปิด LED และพิมพ์ “OFF.” เพื่อปิด LED การรับข้อมูลมาจาก Serial จะรับมาทีละตัวอักษรผ่านคำสั่ง Serial.read() หลักการคือการวนลูปเพื่อเก็บทุกตัวอักษรที่ได้จากคำสั่ง Serial.read() โดยนำไปเก็บไว้ในตัวแปร

DO NOT COPY

ชนิดสตริง เพื่อง่ายต่อการนำไปใช้ งานหลักการตรวจว่ารับข้อมูลมาครบหรือยัง คือการตรวจหา “.” เพื่อกำหนดสิ้นสุดตัวอักษร การส่ง ข้อมูลผ่าน Serial จะต้องมีการส่งตัวอักษร “.” ทุกครั้ง เมื่อตรวจหา “.” เจอ นั่นหมายความว่าข้อมูลได้ รับมาครบเรียบร้อยแล้ว แล้วจึงนำข้อมูลมาประมวลผลต่อได้เลย



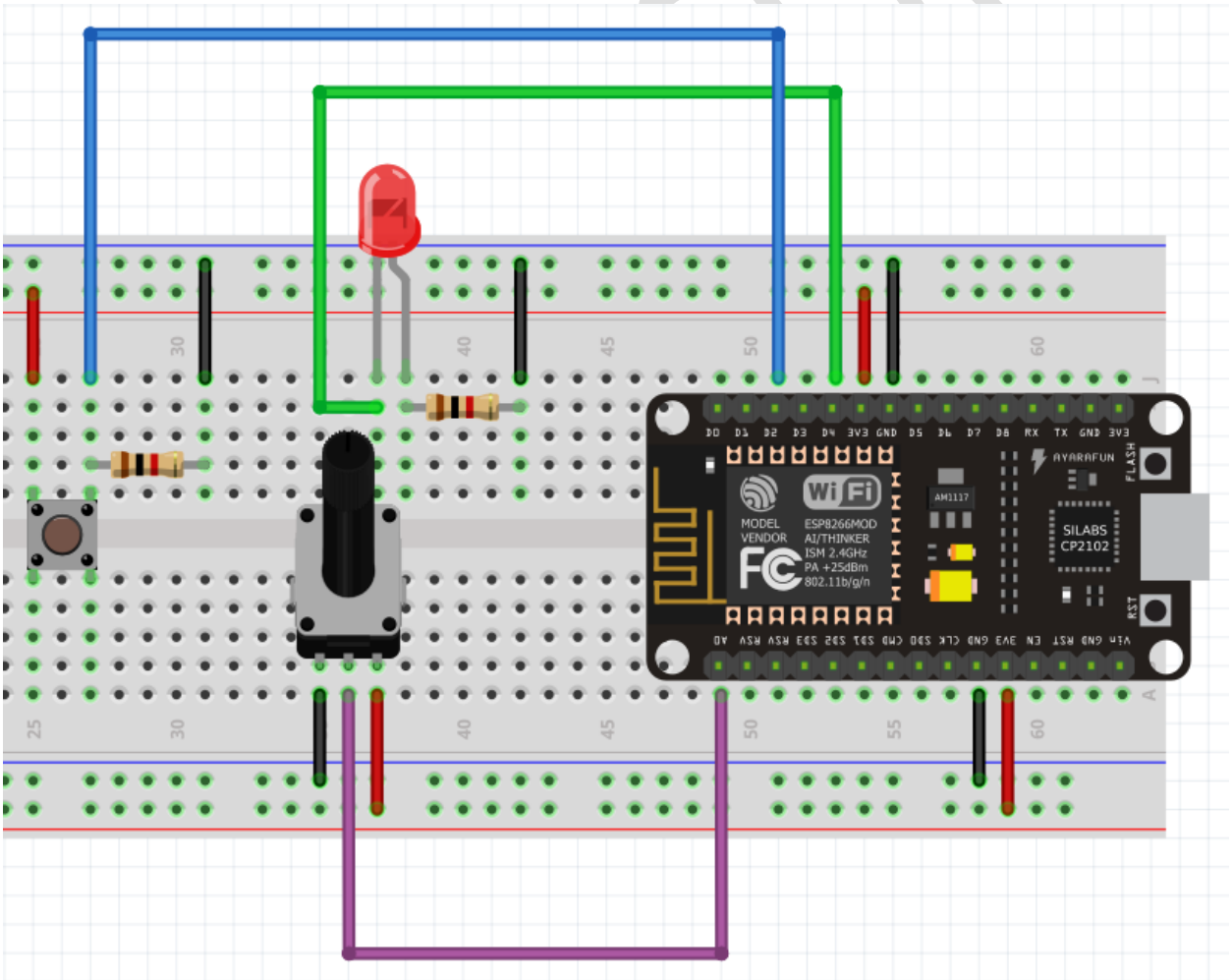
DO NOT COPY

LAB06: Analog Input

เนื่องจาก ESP8266 นั้นใช้ไฟเลี้ยงอยู่ที่ 3.3v และที่ analog pin รับ input ได้เพียง 0-1v เท่านั้น ต่างจาก arduino ที่ใช้ไฟเลี้ยงที่ 5v และ analog pin ก็สามารถรับ input ได้ถึง 0-5v ซึ่ง sensor ทั่วไปนั้นส่วนมาก Vout ของ sensor จะมีแรงดันอยู่ที่ประมาณ 0-5v ถ้าเป็น Analog Pin ทั่วไปของ arduino สามารถรับ input ได้สบาย ๆ แต่ในส่วนของ ESP8266 นั้นจำเป็นต้องต่อวงจรแบ่งแรงดันคำนวณหาค่าแรงดันที่ไม่เกิน 1v มาก เพื่อที่ ADC Pin ของ ESP8266 สามารถอ่านค่าได้อย่างปลอดภัย

เมื่อ NodeMCU V1 ออกมา ซึ่งทางผู้ผลิตก็ได้คำนึงถึงจุดนี้ ภายในบอร์ดของ NodeMCU นั้นจึงได้ต่อวงจรแบ่งแรงดันไว้ภายในบอร์ด ซึ่งผู้ใช้งานไม่ต้องมานั่งต่อวงจรเอง สามารถต่อสายสัญญาณจาก Vout ของ sensor เข้าไปที่ Analog Pin ของบอร์ดตรง ๆ ได้เลย

ทำการต่อวงจรเพิ่มตามรูป โดยใช้ Potentiometer เพิ่มเข้ามา



ค่าที่อ่านได้จาก Analog Pin นั้น ถ้าเราจ่ายไฟที่ 0v ค่าที่อ่านได้จาก analogRead() ก็จะได้เท่ากับ 0 ถ้าหากจ่ายไฟไปที่ analog pin 3.3v ค่าที่อ่านได้จาก analogRead จะได้เท่ากับ 1023 หมายความว่าแรงดันที่อ่านจะอยู่ระหว่าง 0-3.3v ที่ขา Analog Pin ของ NodeMCU

LAB06: Analog Input

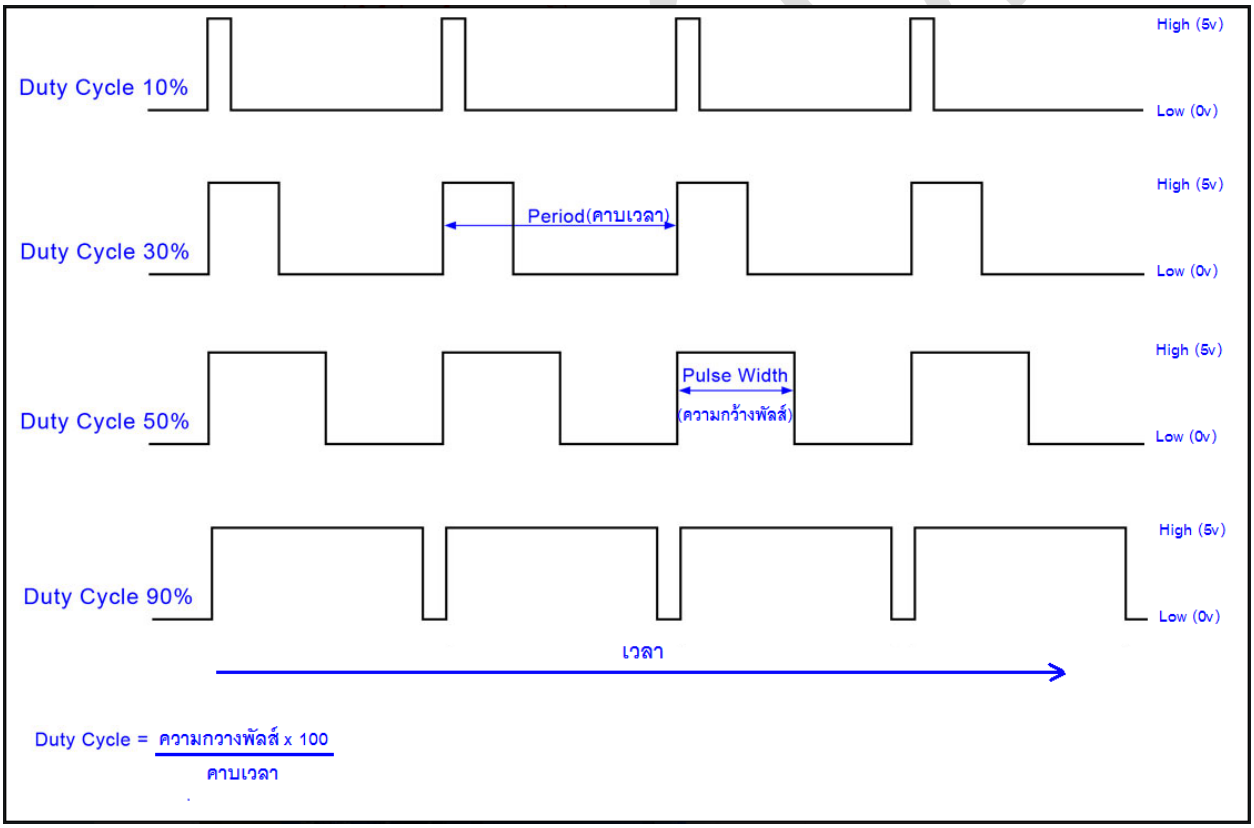
```
void setup() {
 Serial.begin(115200);
}

void loop() {
 int sensorValue = analogRead(A0);  // read the input on analog pin 0:
 Serial.println(sensorValue); // print out the value you read:
 delay(1000);         // delay in between reads for stability
}
```

ผลที่ได้เมื่อเราหมุนปรับค่า Potentiometer ค่าที่ Serial ก็จะเปลี่ยนตามด้วย ซึ่งค่าที่ได้จะอยู่ในช่วง 0 - 1023

DO NOT COPY

LAB07: Pulse Width Modulation (PWD)

โดยปกติแล้ว Digital Port จะสามารถมีได้แค่ 2 สถานะ คือ HIGH (5 โวล์ท กับ (LOW (0 โวล์ท(เท่านั้น จึงทำให้สร้างค่าสัญญาณ Logic ได้เพียง เปิดหรือปิด)1 หรือ 0 , มีไฟหรือไม่มีไฟแค่นั้น (ซึ่งการใช้เทคนิค PWM นั้น จะเป็นการทำให้ Digital Port สามารถเขียนค่าได้มากกว่า HIGH หรือ LOW โดย ทำให้สามารถเขียนค่าเป็นแบบ Analog ได้ NodeMCU มีค่าตั้งแต่ 0-1023 โดยวิธีการนั้นจะใช้การปรับสถานะของสัญญาณ Logic HIGH / LOW สลับกันไปมาด้วยคาบเวลาหนึ่งๆ โดยค่าที่ได้นั้นจะขึ้นอยู่กับ สัดส่วนเวลาของสัญญาณในช่วงเวลาที่มีสถานะเป็น HIGH กับช่วงเวลาที่เป็น LOW โดย ช่วงเวลาทั้งหมดที่สัญญาณมีสถานะเป็น HIGH นั้นเราจะเรียกว่าเป็น "ความกว้าง Pulse (Pulse Width)" โดยสัญญาณพัลส์ เมื่อเทียบ ของช่วงเวลาที่เป็น %HIGH (หรือก็คือ ของ %Pulse Width) กับ) ของคาบเวลา %Period) ของพัลส์ลูกนั้น ๆ เราจะเรียกว่า Duty Cycle เพื่อความเข้าใจสามารถดูได้จากตัวอย่างด้านล่าง



ถ้าจะสร้างสัญญาณเอาต์พุตแบบ PWM (Pulse Width Modulation) ที่ปรับค่า Duty Cycle ได้ ก็สามารถใช้คำสั่ง analogWrite() ตัวอย่างหนึ่งต่อไปนี้ สาธิตการทำให้ LED เพิ่มหรือลดความสว่าง โดยใช้หลักการที่เรียกว่า PWM-based LED Dimming โดยการสร้างสัญญาณ PWM ที่สามารถเปลี่ยนค่า Duty Cycle ได้ โดยสามารถเลือกขา GPIO 0..15 และค่า Duty Cycle ได้ในช่วง 0..1023 (0% ถึง 100%)

เปิด Menu File > Example > Analog > AnalogInOutSerial แล้วนำมาแก้ไขได้

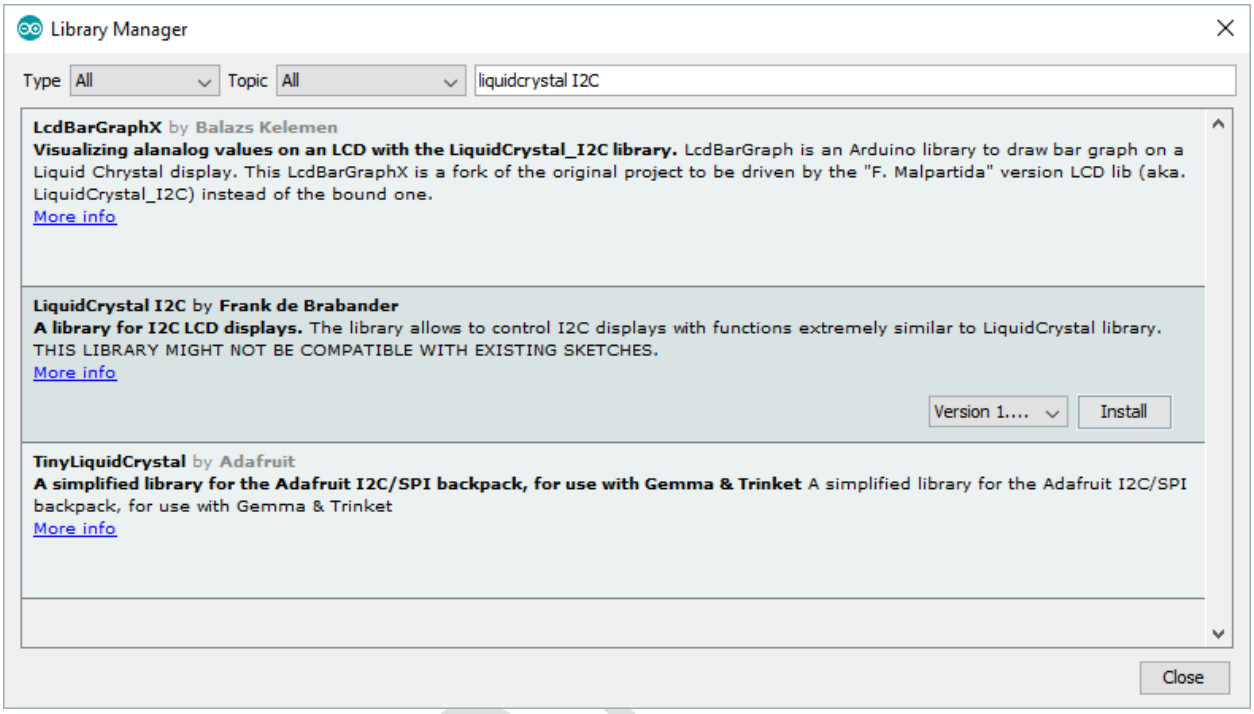
LAB07: PWM

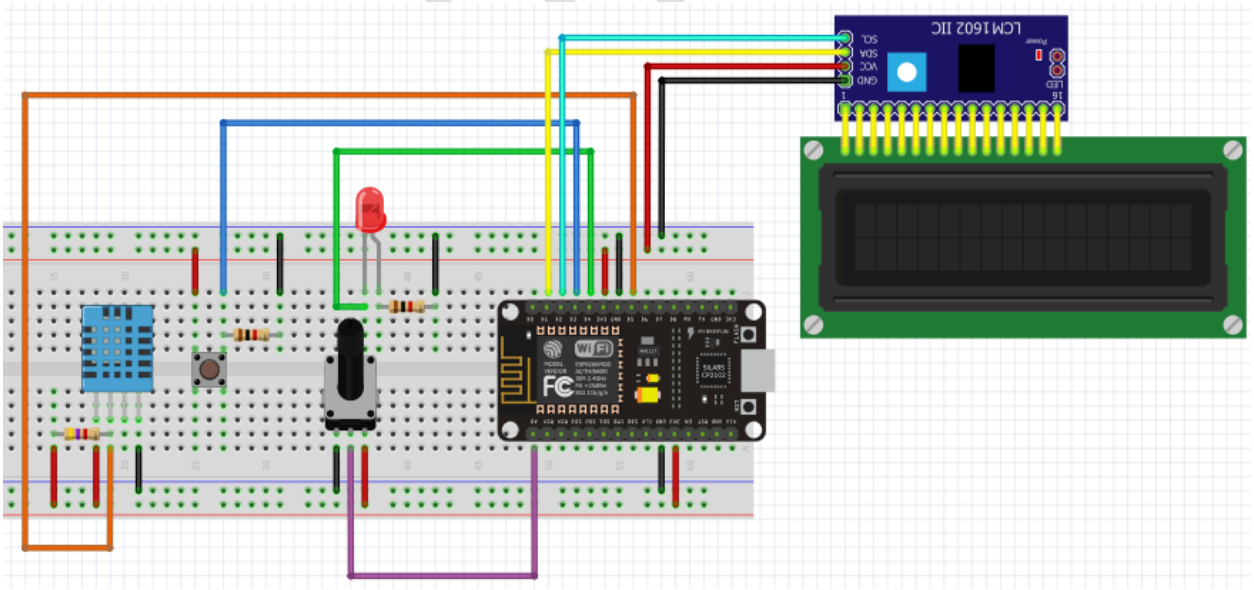
```
#define analogInPin A0  // Analog input pin that the potentiometer is attached to
#define analogOutPin  2 // Analog output pin that the LED is attached to

int sensorValue = 0;        // value read from the pot
int outputValue = 0;        // value output to the PWM (analog out)

void setup() {
 Serial.begin(115200);      // initialize serial communications at 115200 bps:
}

void loop() {
 sensorValue = analogRead(analogInPin);  // read the analog in value:
 outputValue = map(sensorValue, 0, 1023, 0, 512); // map it to the range of the analog out:
   analogWrite(analogOutPin, outputValue);   // change the analog out value:

 // print the results to the serial monitor:
 Serial.print("sensor = ");
 Serial.print(sensorValue);
 Serial.print("\t output = ");
 Serial.println(outputValue);

 delay(2);
}
```

DO NOT COPY

LAB08: การแสดงข้อความ LCD display I2C

ก่อนอื่นต้องติดตั้ง Library LiquidCrystal I2C ที่ Library Manager...

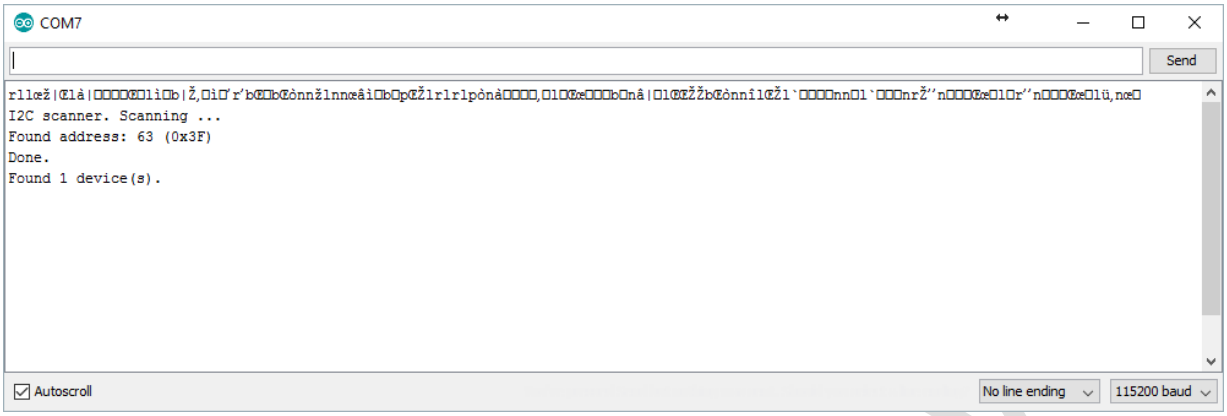




LAB08-1: Scan I2C Address

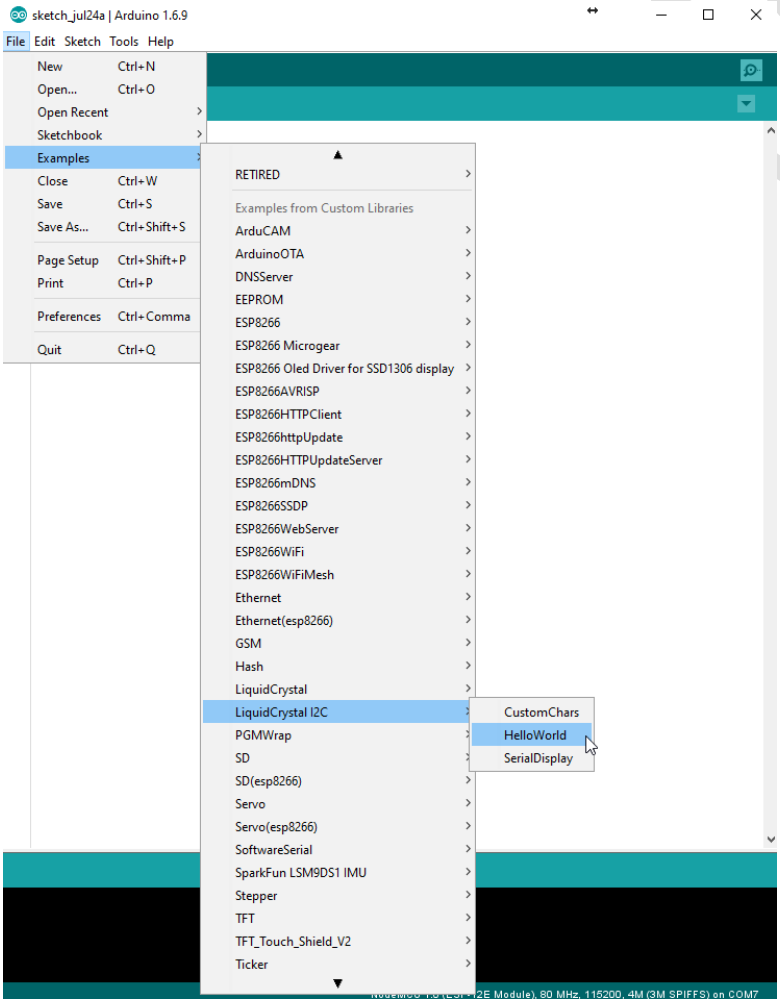
```
#include <Wire.h>
void setup() {
Serial.begin (115200);
while (!Serial) { } // Leonardo: wait for serial port to connect
Serial.println ();
Serial.println ("I2C scanner. Scanning ...");
byte count = 0;
Wire.begin();
for (byte i = 8; i < 120; i++)
{
   Wire.beginTransmission (i);
   if (Wire.endTransmission () == 0)
   {
      Serial.print ("Found address: ");
      Serial.print (i, DEC);
      Serial.print (" (0x");
      Serial.print (i, HEX);
      Serial.println (")");
      count++;
      delay (1); // maybe unneeded?
   } // end of good response
} // end of for loop
Serial.println ("Done.");
Serial.print ("Found ");
Serial.print (count, DEC);
Serial.println (" device(s).");
} // end of setup
void loop() {}
```

DO NOT COPY



Address ที่ใช้คือ 3F

เปิด Menu Example > LiquidCrystalI2C > HelloWorld



COPY

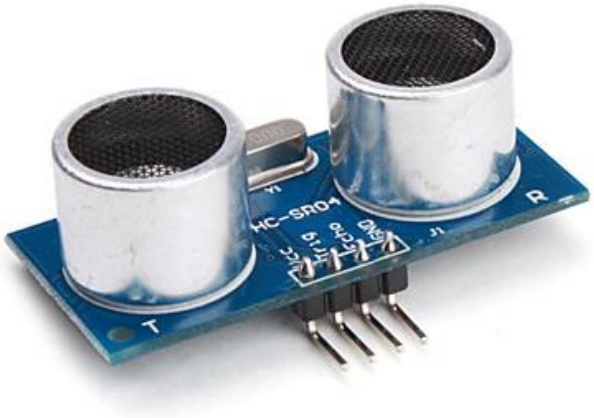
LAB08-2: Show Text on LCD I2C

```
#include <Wire.h>
#include <LiquidCrystal_I2C.h>

LiquidCrystal_I2C lcd(0x3F,16,2);  // set the LCD address to 0x3F for a 16 chars and 2 line

void setup()
{
  lcd.init();                      // initialize the lcd
  lcd.backlight();
  lcd.setCursor(0,0);
  lcd.print("Hello, world!");
  lcd.setCursor(0,1);
  lcd.print("Max Developer");
}


void loop()
{
}
```

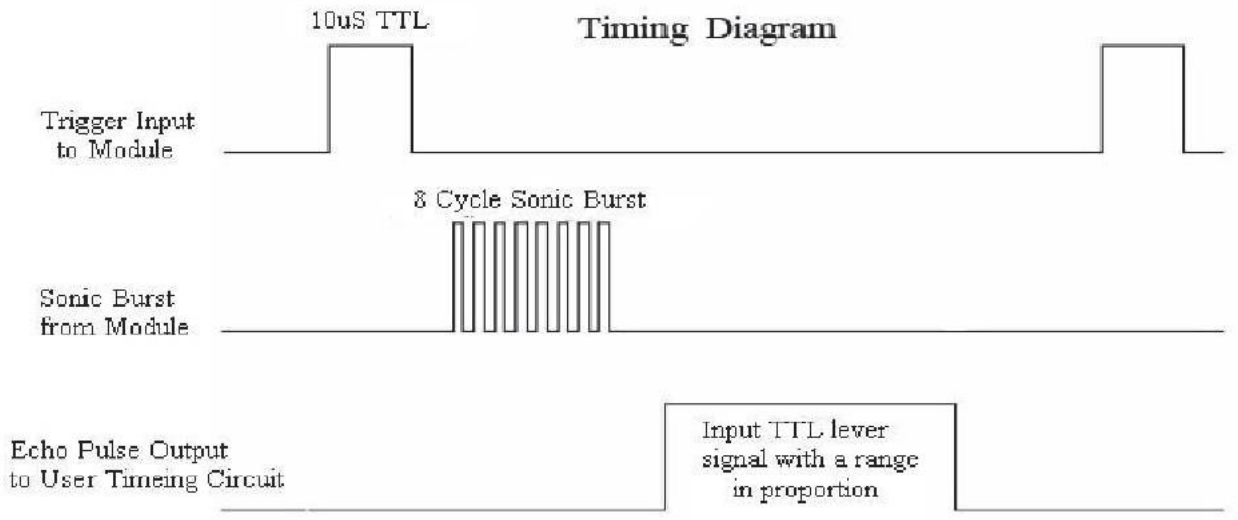
DO NOT COPY

LAB09: การใช้ Ultrasonic วัดระยะทาง

Ultrasonic Ranging Module HC-SR04 โมดูลเพื่อใช้ในการวัดระยะทางโดยไม่มีการสัมผัส



การทำงานของ Ultrasonic sensor ก็เหมือนกับค้างคาวที่บินในเวลากลางคืน คือใช้การส่งคลื่นเสียงที่หูมนุษย์ไม่สามารถได้ยินออกไปสะท้อนวัตถุที่ต้องการวัดระยะ แล้วจับเวลาเสียงสะท้อน เพื่อคำนวณระยะทาง



เพื่อความเข้าใจกันมากขึ้น อุปกรณ์ตัวนี้เริ่มต้นทำงานโดยการส่งสัญญาณเริ่มต้นยาว 10 ไมโครวินาที ไปสั่งให้แหล่งกำเนิดเสียงทำงาน จากนั้นจะส่งคลื่นเสียงความถี่ 40 kHz ออกไป 8 พัสส์ แล้วรอฟังเสียงสะท้อน ตัวซ้ายจะเป็นตัวส่งคลื่นเสียงออกไป ส่วนตัวขวาในรูปจะเป็นตัวรับความถี่ที่สะท้อนกลับมา

เนื่องจากเสียงที่ส่งออกไปถึงแม้จะไม่ได้ยินเพราะเกิน 20 kHZ ที่หูมนุษย์จะรับฟังได้ แต่เนื่องจากยังคงเป็นคลื่นเสียง ดังนั้นความเร็วของเสียงจึงแปรผันตามอุณหภูมิด้วยตามสูตรนี้

$C \approx 331.5 + 0.61\,\theta$ (m/s)

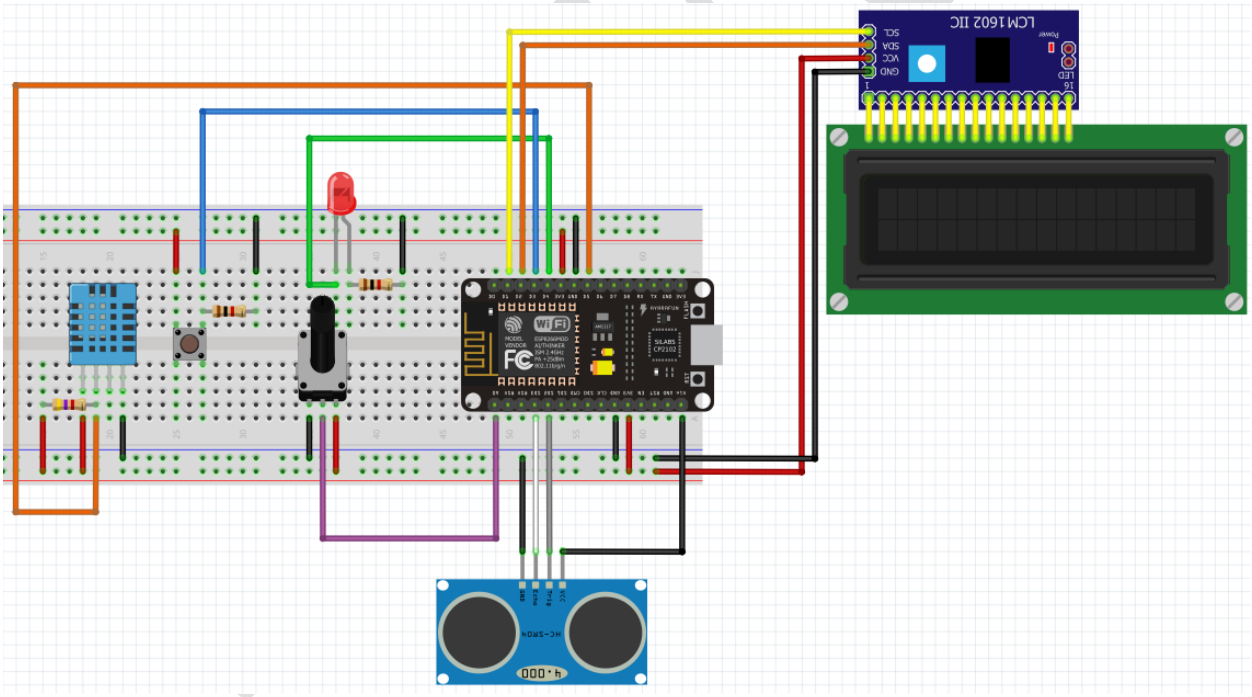
ดังนั้นแน่นอนเวลาผู้ผลิตเขียนโปรแกรมออกแบบไว้ก็อยู่ที่อุณภูมิทำงานที่อาจจะแตกต่างจากบ้านเรา ก็ทำให้ค่าที่วัดได้มีความผิดพลาดไปบ้าง

อีกส่วนที่จะต้องรู้ก็คือช่วงวัด และมุมที่สามารถวัดได้ครับ และเนื่องจากคุณสมบัติของอุปกรณ์ที่ใช้ในการกำเนิดเสียง และรูปร่างของตัวลำโพง)Horn) ก็ทำให้อุปกรณ์ตัวนี้มีมุมวัด 15 องศา)Measuring Angle) โดยสามารถวัดระยะห่างได้ตั้งแต่ 2 ซม จนถึง .4 เมตร

ระยะทางก็คำนวณได้จากสูตรนี้

ระยะทาง ความยาวของสัญญาณสะท้อน =x 340 (m/s) / 2

และแน่นอนครับคุณภาพของสัญญาณ ความแม่นยำก็ขึ้นกับวัสดุ ลักษณะของพื้นผิวที่ใช้วัดด้วย +



ระดับแรงดันที่ใช้คือ 5v

LAB09-1: Basic Ultrasonic

```
#define TRIGGER_PIN  9 // SD2 - GPIO9
#define ECHO_PIN     10  // SD3 - GPI109

void setup() {
  Serial.begin (9600);
  pinMode(TRIGGER_PIN, OUTPUT);
  pinMode(ECHO_PIN, INPUT);
  pinMode(BUILTIN_LED, OUTPUT);
}

void loop() {
  long duration, distance;
  digitalWrite(TRIGGER_PIN, LOW);  // Added this line
  delayMicroseconds(2); // Added this line
  digitalWrite(TRIGGER_PIN, HIGH);
  delayMicroseconds(10); // Added this line
  digitalWrite(TRIGGER_PIN, LOW);
  duration = pulseIn(ECHO_PIN, HIGH);
  distance = (duration/2) / 29.1;
  Serial.print(distance);
  Serial.println(" cm");
  delay(1000);
}
```

DO NOT COPY

ใช้ Library

LAB09-2: Ultrasonic Library

```
#include <Ultrasonic.h>
Ultrasonic ultrasonic(9,10); // (Trig PIN,Echo PIN)
void setup() {
  Serial.begin(9600);
}
void loop() {
  Serial.print(ultrasonic.Ranging(CM)); // CM or INC
  Serial.println(" cm" );
  delay(100);
}
```

Code ที่ปรับใหม่

LAB09-3: Ultrasonic LCD I2C

```
#include <Wire.h>
#include <LiquidCrystal_I2C.h>
#include <Ultrasonic.h>

LiquidCrystal_I2C lcd(0x3F,16,2);  // set the LCD address to 0x3F for a 16 chars and 2 line
Ultrasonic ultrasonic(9,10);

void setup()
{
  lcd.init();                      // initialize the lcd
  lcd.backlight();
  lcd.setCursor(0,0);
  lcd.print("Ulteasonic");
  lcd.setCursor(0,1);
  lcd.print("Sensor >");
```

DO NOT COPY

```
  delay(1000);
  lcd.print(">");
  delay(1000);
  lcd.print(">");
  delay(1000);
  lcd.print(">");
  delay(1000);
  lcd.print(">");
  delay(1000);
  lcd.print(">");
  delay(1000);
  lcd.print(">");
  delay(1000);
  lcd.print(">");
}



void loop()
{
  lcd.clear();
  lcd.setCursor(0, 0);
  lcd.print(ultrasonic.Ranging(CM)); // CM or INC
  lcd.print(" cm");
  lcd.setCursor(0, 1);
  lcd.print(ultrasonic.Ranging(INC)); // CM or INC
  lcd.print(" inc");
  delay(100);
}
```

DO NOT COPY

LAB10: การใช้ Motion Sensor

PIR Motion Sensor Getting Started

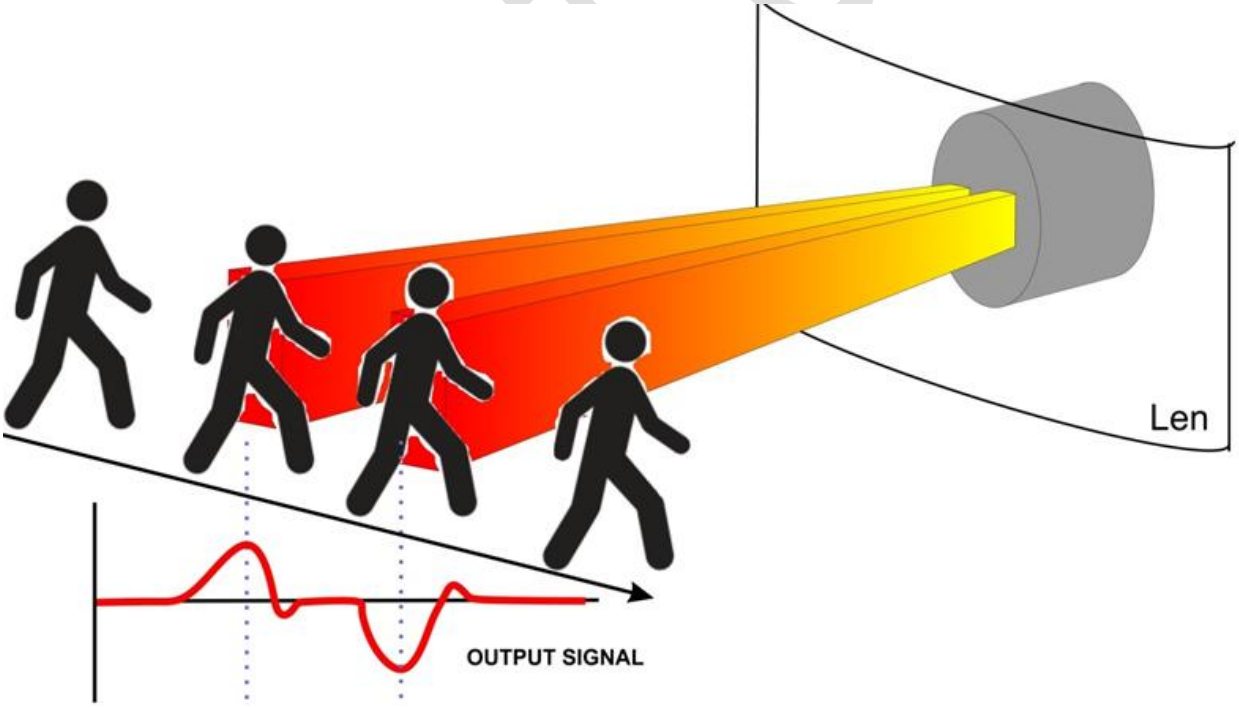
PIR (Passive infrared)

คือ อุปกรณ์ตรวจจับคลื่นรังสี Infrared จากวัตถุ ผ่านอุปกรณ์รวมแสง มายังตัว Pyro Electric ซึ่งจะเปลี่ยนพลังงานความร้อน จากรังสี Infrared เป็นพลังงานไฟฟ้า แม้จะมีประมาณ Infrared แค่เพียงเล็กน้อย จึงทำให้ PIR สามารถตรวจจับ คลื่นรังสี Infrared และ อุณหภูมิได้

PIR Motion Sensor

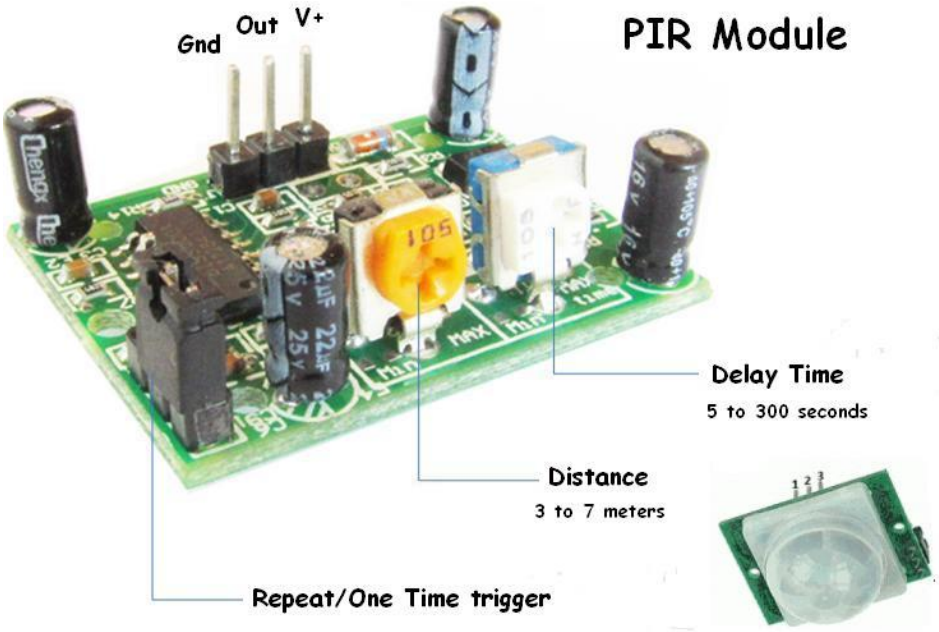
คือ อุปกรณ์ Sensor ชนิดหนึ่งที่ใช้ตรวจจับคลื่นรังสี Infrared ที่แพร่จาก มนุษย์ หรือ สัตว์ ที่มีการเคลื่อนไหว ทำให้มีการนำเอา PIR มาประยุคใช้งานกันเป็นอย่างมากใช้เพื่อตรวจจับการเคลื่อนไหวของสิ่งมีชีวิต หรือ ตรวจจับการบุกรุกในงานรักษาความปลอดภัย

การทำงานของ PIR Sensor

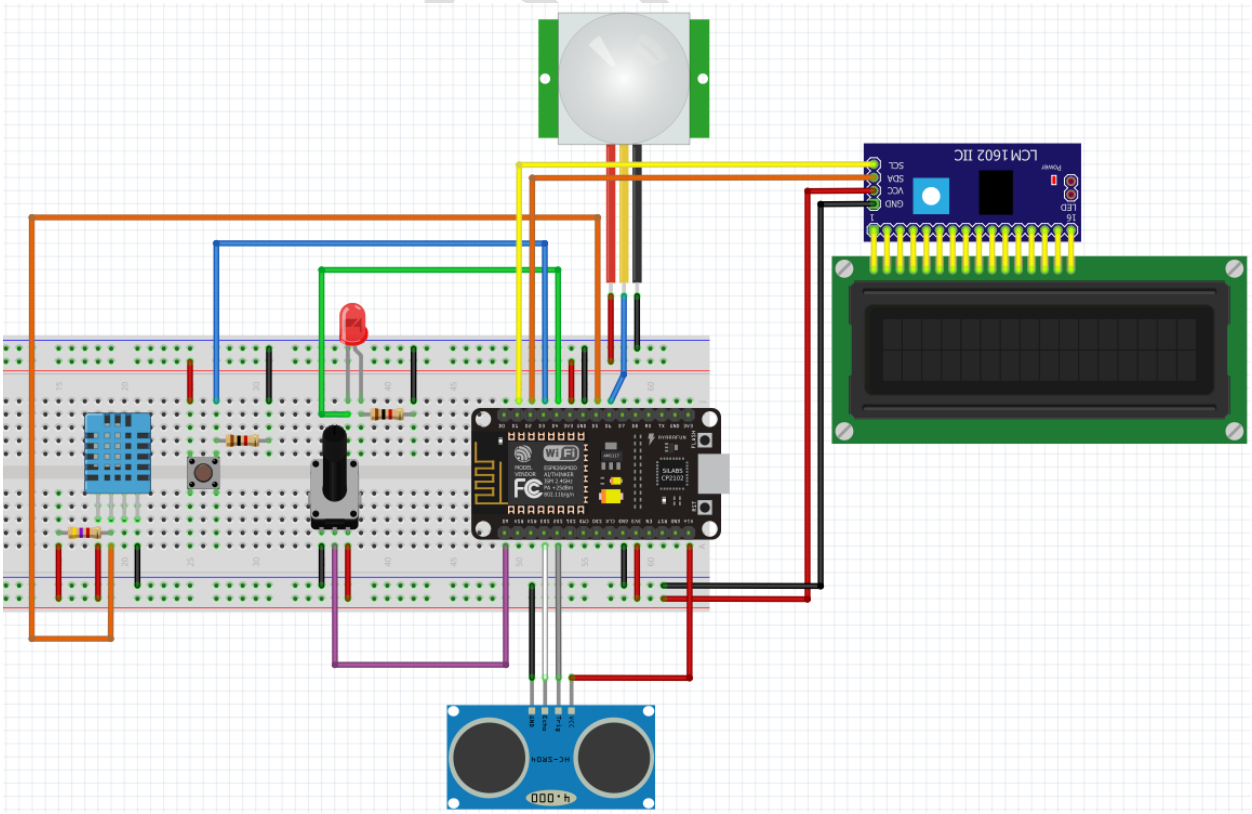


ภายใน PIR จะมีอุปกรณ์ตรวจจับรังสี Infrared อยู่ 2 ชุดด้วยกันดังรูป เมื่อมี คน หรือ สัตว์ ที่มีความอบอุ่นในร่างกายเคลื่อนที่ผ่านเข้ามาใน พื้นที่โซนที่ PIR สามารถตรวจจับคลื่นรังสี Infrared ที่แพร่ออกมาจากสิ่งมีชีวิตได้ PIR จะเปลี่ยนคลื่นรังสี Infrared ให้กลายเป็น กระแสไฟฟ้าดังรูป จะเห็นว่าเมื่อมีสิ่งมีชีวิต เคลื่อนที่ผ่าน อุปกรณ์

ตรวจจับรังสี Infrared ตัวที่ 1 จะได้สัญญาณ Output ออกมาสูงกว่าแรงดันปรกติ และ เมื่อสิ่งมีชีวิตเคลื่อนที่ผ่านอุปกรณ์ตรวจจับรังสี Infrared ตัวที่ 2 จะได้แรงดัน Output ต่ำกว่าค่าแรงดันปรกติ



วงจรที่ใช้



LAB10-1: Basic Motion Sensor

```
const int PIRPin = 12;
const int ledPin =  2;
int PIRState = 0;
void setup()
{
pinMode(ledPin, OUTPUT);
  pinMode(PIRPin, INPUT);
  Serial.begin(9600);
}

void loop()
{
  PIRState = digitalRead(PIRPin);
if(PIRState == HIGH)
      {
            Serial.print("\r\nALARM");
      }else{
           Serial.print("\r\nNORMAL");
      }
  digitalWrite(ledPin,PIRState);
  delay (500);
}
```

DO NOT COPY

Code ที่ปรับแล้ว

LAB10-2: Advance Motion Sensor

```
//the time we give the sensor to calibrate (10-60 secs according to the datasheet)
int calibrationTime = 30;

//the time when the sensor outputs a low impulse
long unsigned int lowIn;

//the amount of milliseconds the sensor has to be low
//before we assume all motion has stopped
long unsigned int pause = 5000;

boolean lockLow = true;
boolean takeLowTime;

int pirPin = 12;    // D6 the digital pin connected to the PIR sensor's output
int ledPin = 2;     // D4



////////////////////////////
//SETUP
void setup(){
  Serial.begin(9600);
  pinMode(pirPin, INPUT);
  pinMode(ledPin, OUTPUT);
  digitalWrite(pirPin, LOW);

  //give the sensor some time to calibrate
  Serial.print("calibrating sensor ");
    for(int i = 0; i < calibrationTime; i++){
```

DO NOT COPY

```
      Serial.print(".");
      delay(1000);
      }
    Serial.println(" done");
    Serial.println("SENSOR ACTIVE");
    delay(50);
  }

////////////////////////////
//LOOP
void loop(){

    if(digitalRead(pirPin) == HIGH){
      digitalWrite(ledPin, HIGH);   //the led visualizes the sensors output pin state
      if(lockLow){
        //makes sure we wait for a transition to LOW before any further output is made:
        lockLow = false;
        Serial.println("---");
        Serial.print("motion detected at ");
        Serial.print(millis()/1000);
        Serial.println(" sec");
        delay(50);
        }
        takeLowTime = true;
      }

    if(digitalRead(pirPin) == LOW){
      digitalWrite(ledPin, LOW);  //the led visualizes the sensors output pin state

      if(takeLowTime){
       lowIn = millis();          //save the time of the transition from high to LOW
```

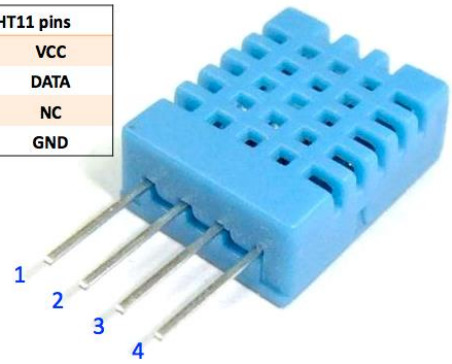
DO NOT COPY

```
      takeLowTime = false;       //make sure this is only done at the start of a LOW phase
      }
     //if the sensor is low for more than the given pause,
     //we assume that no more motion is going to happen
     if(!lockLow && millis() - lowIn > pause){
         //makes sure this block of code is only executed again after
         //a new motion sequence has been detected
         lockLow = true;
         Serial.print("motion ended at ");      //output
         Serial.print((millis() - pause)/1000);
         Serial.println(" sec");
         delay(50);
         }
     }
}
```

LAB11: การวัดอุณหภูมิ และ ความชื้นด้วย DHT11

DHT11 Humidity and Temperature Sensor กับบอร์ด NodeMCU ใช้ในการวัดอุณหภูมิกับความชื้นในอากาศที่มี Specification ตามนี้

DHT11 pins	
1	VCC
2	DATA
3	NC
4	GND



- ย่านวัดความชื้น 20-90% RH โดยมีค่าความแม่นยำ +- 5% RH ความละเอียดในการวัด 1 % แสดงผลแบบ 8 บิต

DO NOT COPY

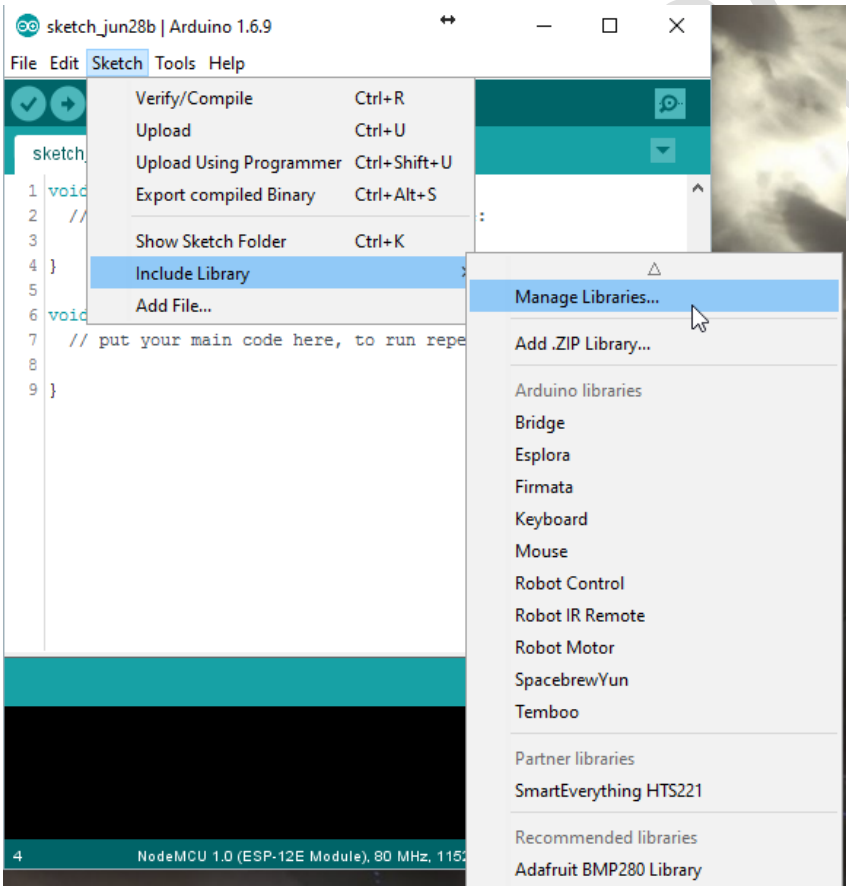
- ย่านวัดอุณหภูมิ 0 -50 องศาเซลเซียส โดยมีค่าความแม่นยำ +- 2 องศาเซลเซียส ความละเอียดในการวัด 1 องศาเซลเซียส แสดงผลแบบ 8 บิต

- กินกระแส 0.5 - 2.5 mA (ขณะทำการวัดค่า) ที่ระดับแรงดัน 3 - 5.5 VDC

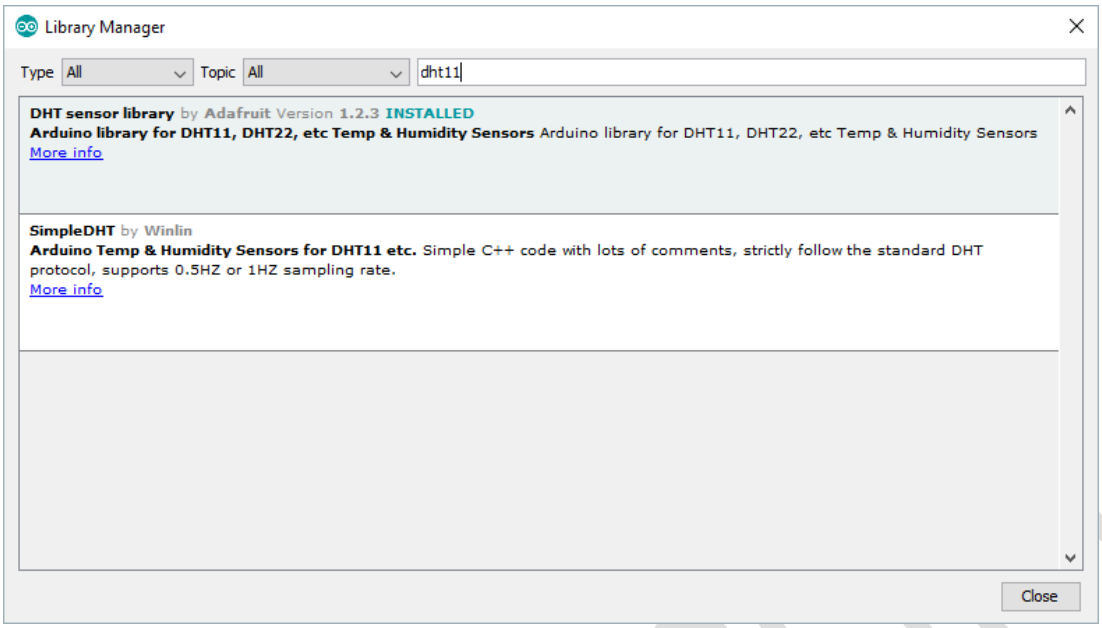
- อ่านค่าสัญญาณ (Sample Rate) ทุก 1 วินาที

- การสื่อสารกับ MCU ของเราด้วยวิธี Single-wire Two-way Serial interface คือการสื่อสารแบบอนุกรมด้วยสายสัญญาณเส้นเดียว

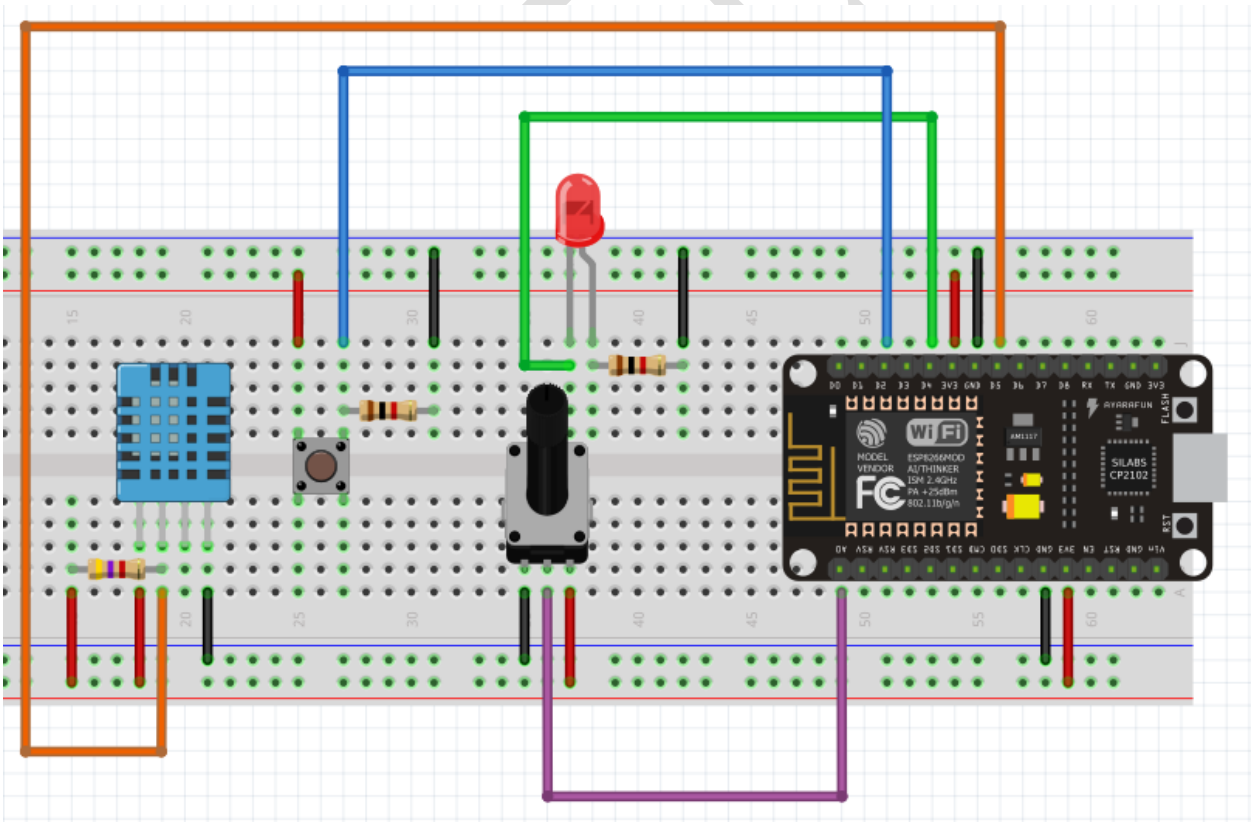
ในการ install library ของ Arduino IDE เราสามารถไปที่ Menu Sketch > Include Library > Manage Library และพิมพ์ DHT11 หลังจากนั้นกดปุ่ม install เราก็จะได้ library มาใช้ได้อย่างง่ายดายหรืออีกวิธีนึงเราสามารถ install ในลักษณะที่เป็น zip file ได้อีกด้วย



COPY



ทำการต่อวงจรดังภาพ เปิด ArduinoIDE และเลือก Menu File > Example > DHT sensor library > DHTTester แล้วนำ Code มาแก้ได้



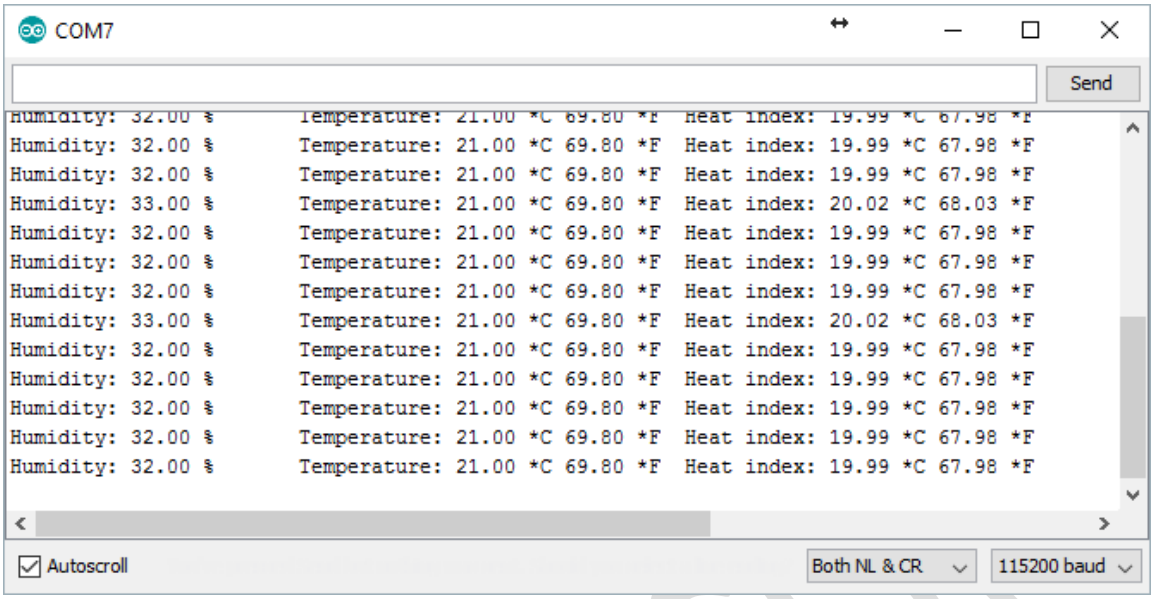
LAB11: DHT11 Temperature and Humidity Sensor

```
#include "DHT.h"

#define DHTPIN 14     // what digital pin we're connected to D5

// Uncomment whatever type you're using!
#define DHTTYPE DHT11   // DHT 11
//#define DHTTYPE DHT22   // DHT 22  (AM2302), AM2321
//#define DHTTYPE DHT21   // DHT 21 (AM2301)
// Connect pin 1 (on the left) of the sensor to +5V
// NOTE: If using a board with 3.3V logic like an Arduino Due connect pin 1
// to 3.3V instead of 5V!
// Connect pin 2 of the sensor to whatever your DHTPIN is
// Connect pin 4 (on the right) of the sensor to GROUND
// Connect a 10K resistor from pin 2 (data) to pin 1 (power) of the sensor
// Initialize DHT sensor.
// Note that older versions of this library took an optional third parameter to
// tweak the timings for faster processors.  This parameter is no longer needed
// as the current DHT reading algorithm adjusts itself to work on faster procs.
DHT dht(DHTPIN, DHTTYPE);

void setup() {
 Serial.begin(115200);
 Serial.println("DHT11 test!");
 dht.begin();
}

void loop() {
 delay(2000);

 // Reading temperature or humidity takes about 250 milliseconds!
```

DO NOT COPY

```
 // Sensor readings may also be up to 2 seconds 'old' (its a very slow sensor)
 float h = dht.readHumidity();
 // Read temperature as Celsius (the default)
 float t = dht.readTemperature();
 // Read temperature as Fahrenheit (isFahrenheit = true)
 float f = dht.readTemperature(true);

 // Check if any reads failed and exit early (to try again).
 if (isnan(h) || isnan(t) || isnan(f)) {
   Serial.println("Failed to read from DHT sensor!");
   return;
 }
 // Compute heat index in Fahrenheit (the default)
 float hif = dht.computeHeatIndex(f, h);
 // Compute heat index in Celsius (isFahreheit = false)
 float hic = dht.computeHeatIndex(t, h, false);

 Serial.print("Humidity: ");
 Serial.print(h);
 Serial.print(" %\t");
 Serial.print("Temperature: ");
 Serial.print(t);
 Serial.print(" *C ");
 Serial.print(f);
 Serial.print(" *F\t");
 Serial.print("Heat index: ");
 Serial.print(hic);
 Serial.print(" *C ");
 Serial.print(hif);
 Serial.println(" *F");
}
```

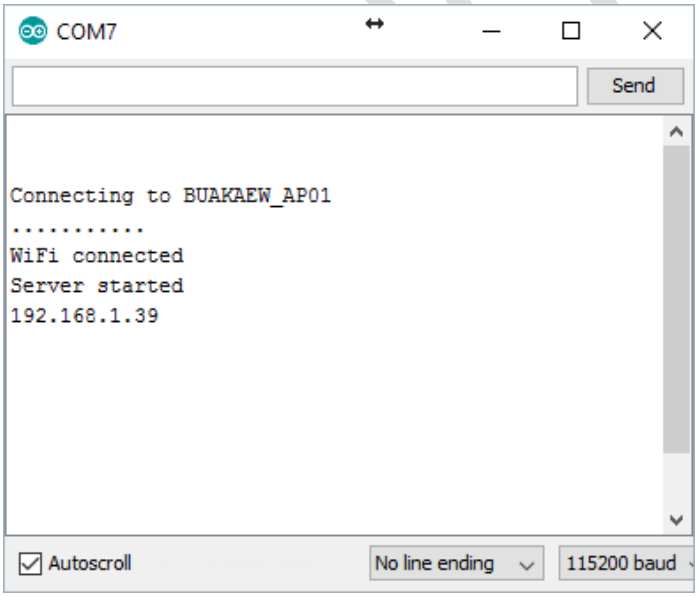
DO NOT COPY

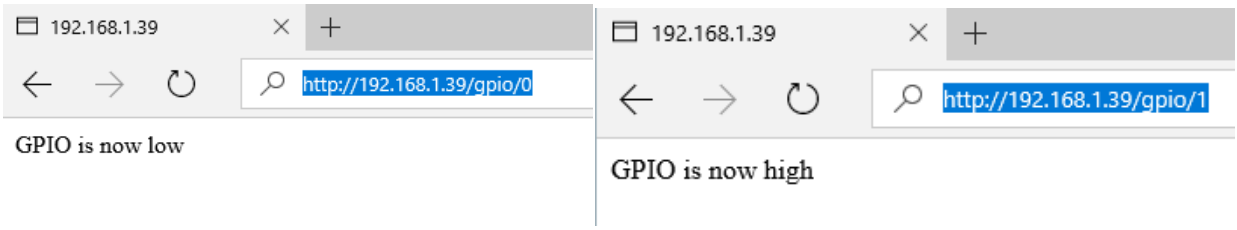


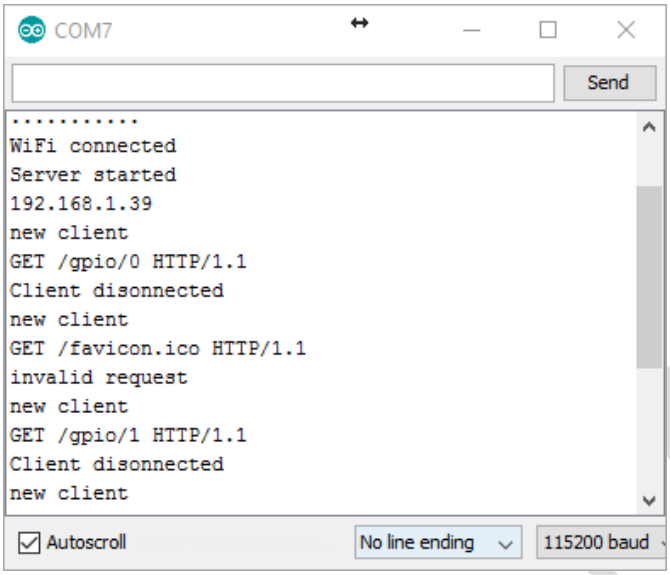
LAB12: ทำ NodeMCU เป็น Web Server

สั่งเปิดปิดอุปกรณ์ผ่าน URL

ไปที่ Menu File > Example > ESP8266 Wifi > WifiWebServer แล้วแก้ไข SSID และ Password ให้ตรง

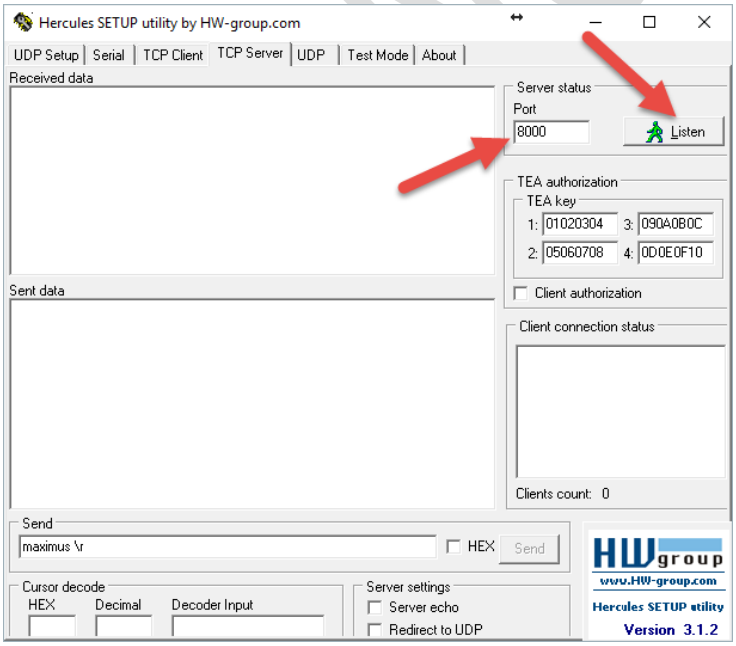






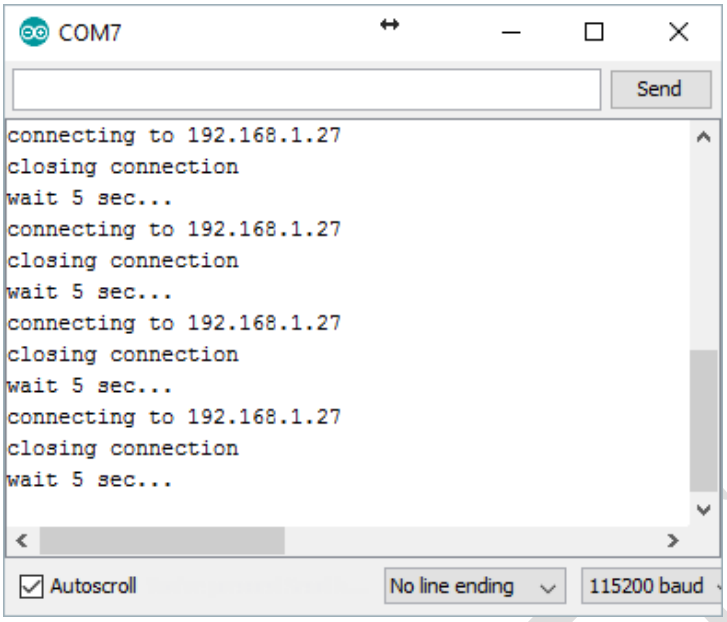
LAB13: ทำ NodeMCU เป็น Web Client

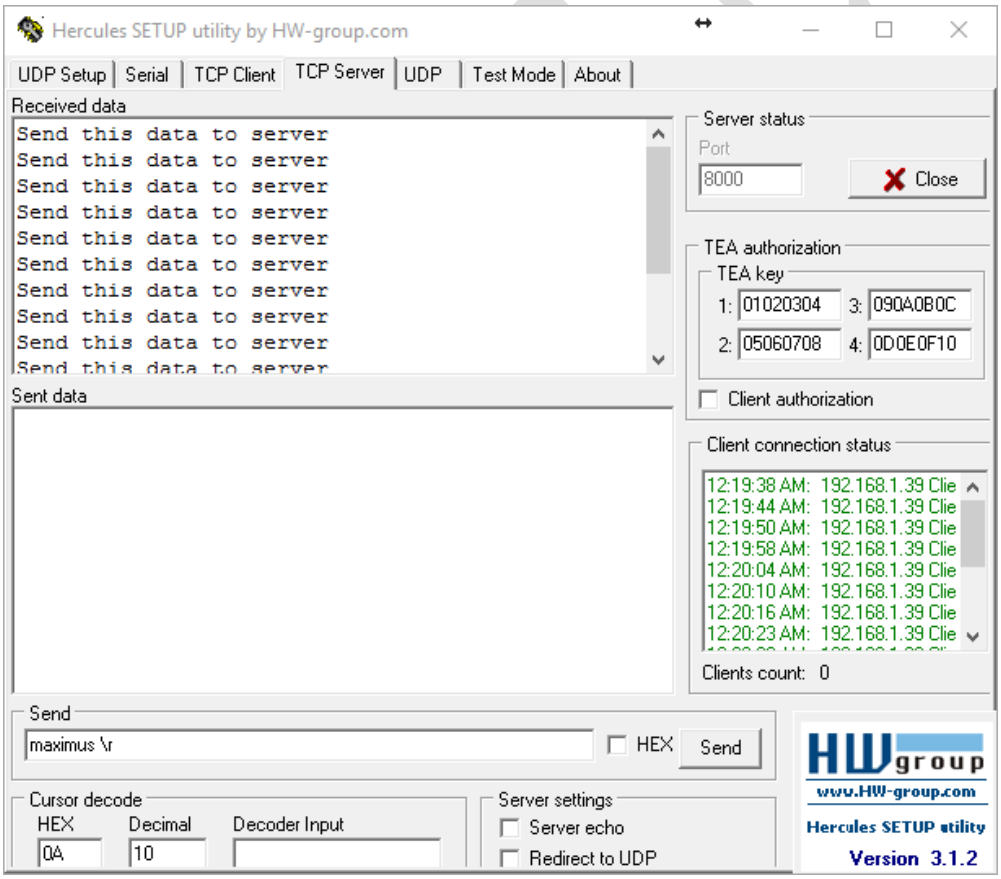
ทดสอบโดยใช้ Hercules เป็น Server



NOT COPY

Arduino IDE ไปที่ Menu File > Example > ESP8266 Wifi > WifiClientBasic แล้วแก้ไข SSID และ Password ให้ตรง





การใช้งาน Cloud Broker สำหรับงาน IoT

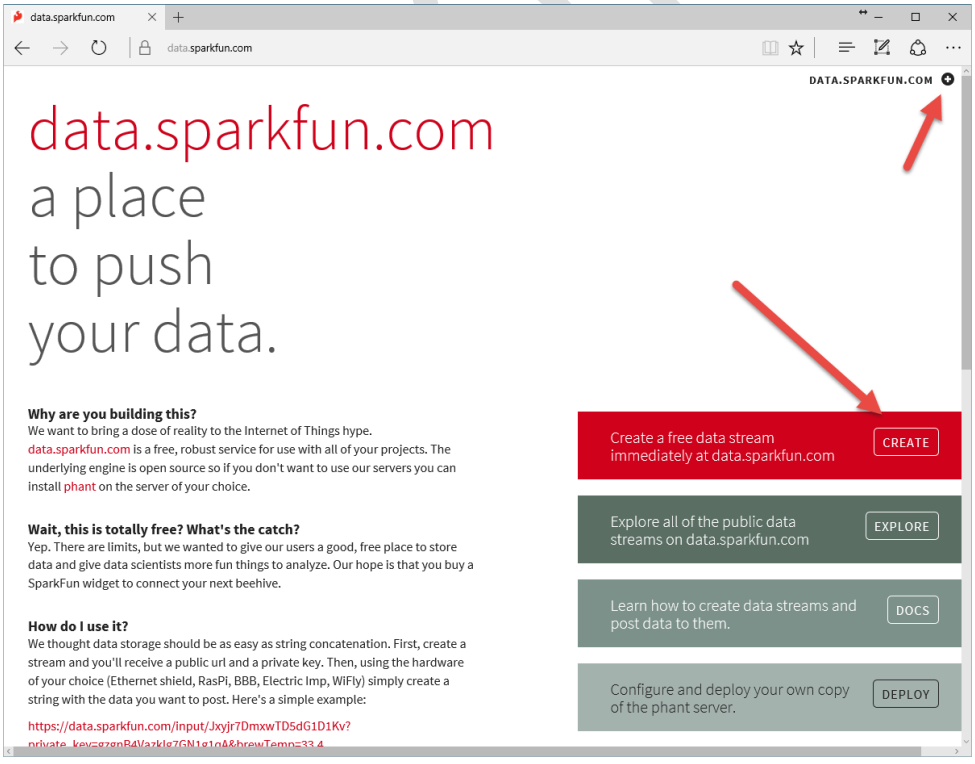
ในการทำ IoT นั้นเราจะใช้ Cloud เป็นตัวกลางในการสื่อสารข้อมูลระหว่าง Node กับ server ซึ่ง Cloud ที่เปิดให้บริการ IoT อยู่นั้นส่วนใหญ่จะเป็นแค่ตัวกลางในการสื่อสารหรือบางตัวสามารถช่วยในการเก็บข้อมูลและแสดงผล โดยที่ตัวมันเองนั้นจะทำเสมือนเป็น broker ซึ่งจะไม่มีการตัดสินใจหรือว่าเงื่อนไขในการทำงานใดๆอยู่ที่ตัว broker

นั่นหมายความว่าหากเราต้องการให้ระบบ IoT ของเรานั้นมีเงื่อนไขในการทำงานอย่างเช่นอุณหภูมิที่ตรวจจับจากเซนเซอร์วัดอุณหภูมิมีค่ามากกว่า 30 องศาให้เปิดพัดลม แต่หากน้อยกว่า 25 องศาให้พัดลมหยุดทำงานการทำงานในลักษณะนี้ตัว broker ไม่สามารถทำได้ดังนั้นเราจึงจำเป็นต้องมี server สำหรับช่วยในการทำเงื่อนไขต่าง ๆ broker เป็นเพียงสื่อกลางในการรับส่งข้อมูลหรืออย่างดีที่สุดก็ช่วยในการเก็บข้อมูลเพื่อส่งต่อและแสดงผลออกมาบน Cloud ได้เลย

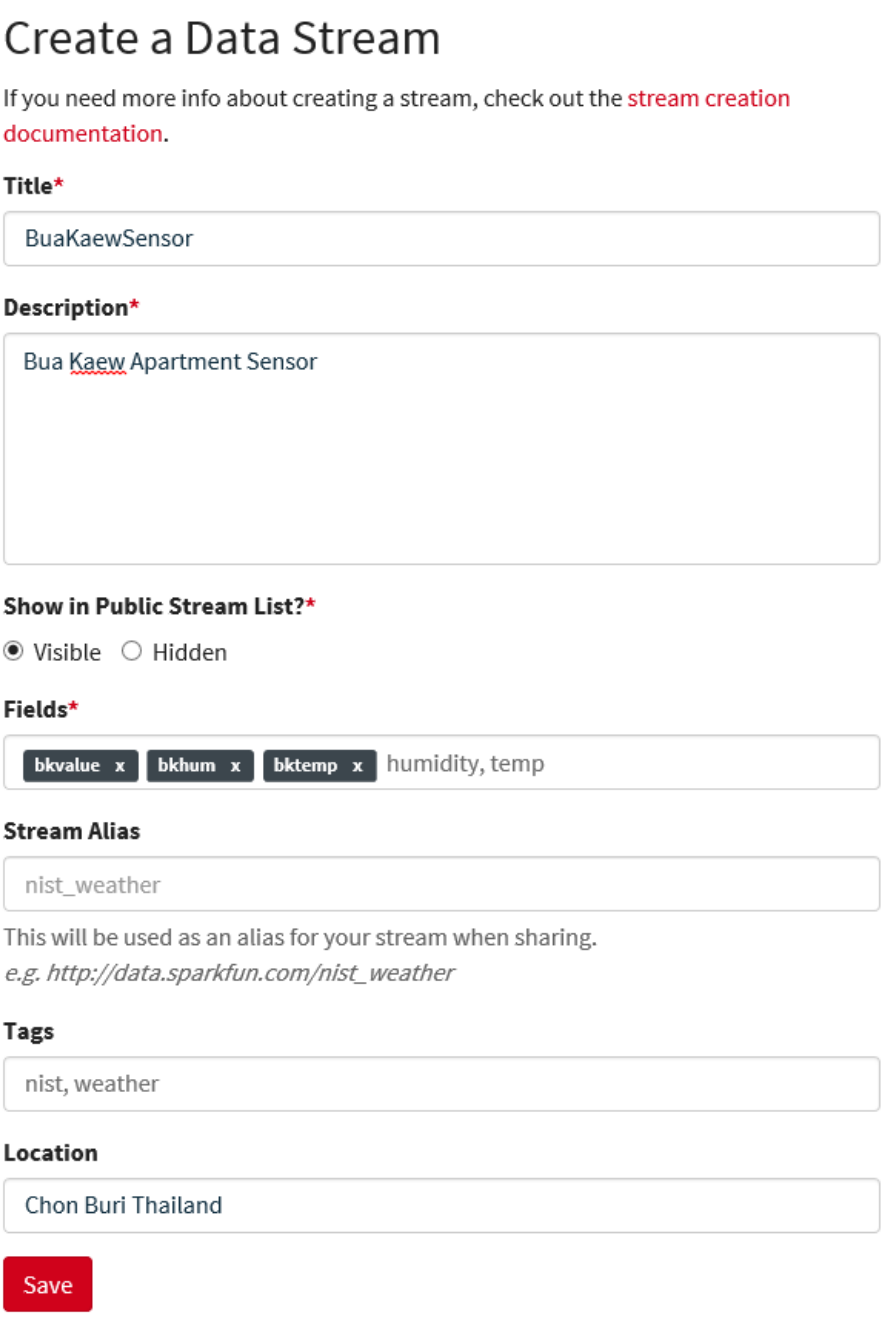
LAB14: ทำ Data Logger โดยส่ง Data ไปที่ SparkFun

SparkFun เป็น Broker ที่ทำหน้าที่เก็บข้อมูลของตัวอุปกรณ์ IoT ต่าง ๆ ที่ส่ง Data มาเก็บไว้บน cloud เราสามารถใช้งานโดยไม่ต้อง register โดยเข้าไปที่ sparkfun.com และเลือก create

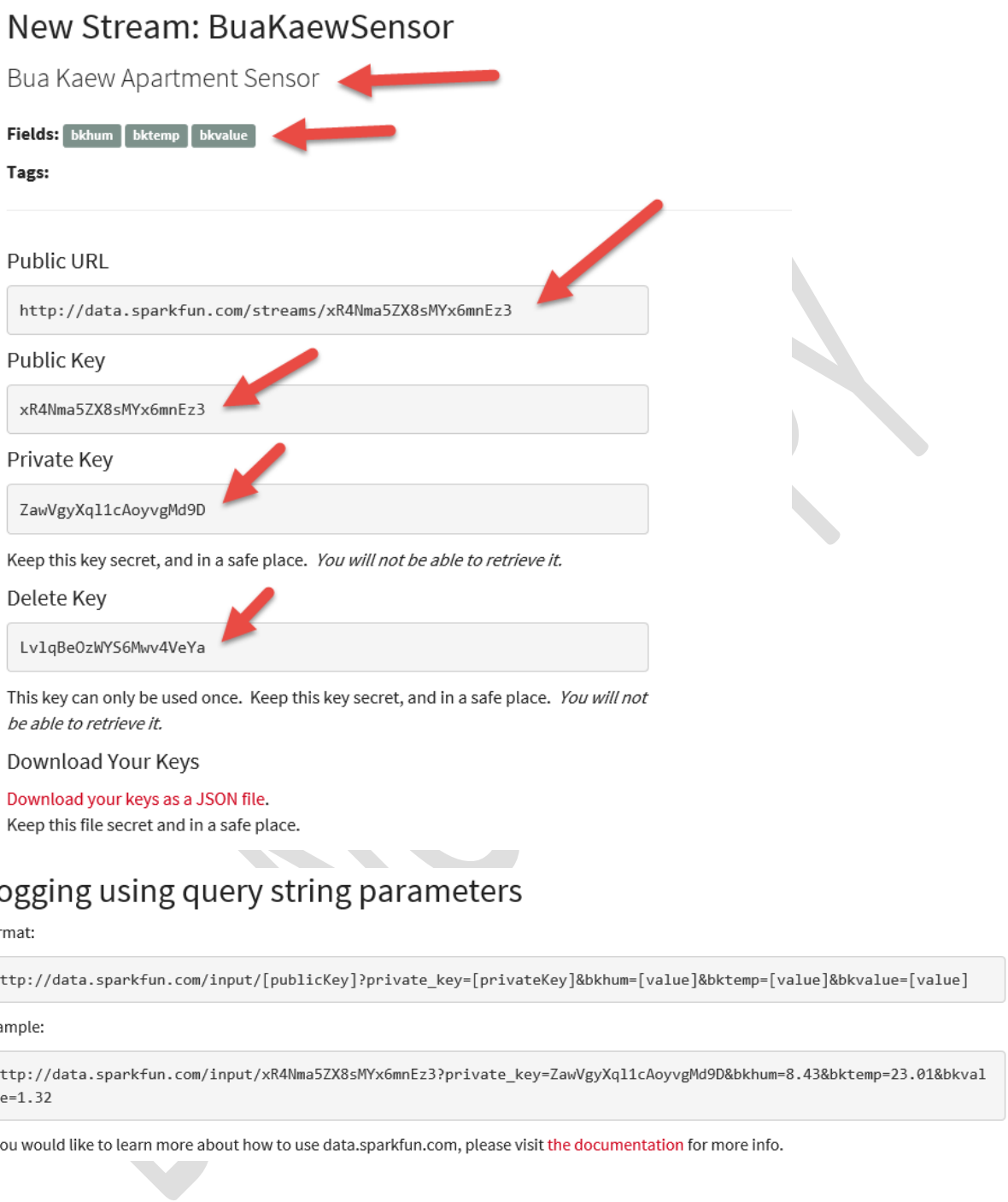
https://data.sparkfun.com/



ใส่รายละเอียดในการสร้าง deta stream ที่สำคัญคือ Title และ Fields



เมื่อบันทึกเรียบร้อยแล้วจะเข้ามาหน้าที่แสดงคีย์ต่างๆที่สำคัญก็คือ public url, public key และ private key รวมถึง Field Parameter ที่เราสร้างเอาไว้เพื่อให้ node mcu ส่งมาด้วยชื่อที่ตรงกัน โดยลักษณะการส่งนั้นจะเป็นการส่งแบบ url Parameter ตามตัวอย่างที่ให้มาด้านล่าง



ในส่วนของ Arduino IDE ไปที่ Menu File > Example > ESP8266WIFI > WIFI Client ให้เราทำการใส่ ssid และ password ในส่วนของ Stream ID ให้ใส่ public key ในส่วนของ private key ก็ใส่ข้อมูลของ private key และในส่วนของ Value ใส่ให้ตรงกับ Parameter ที่เราได้สร้างเอาไว้

WiFiClient §

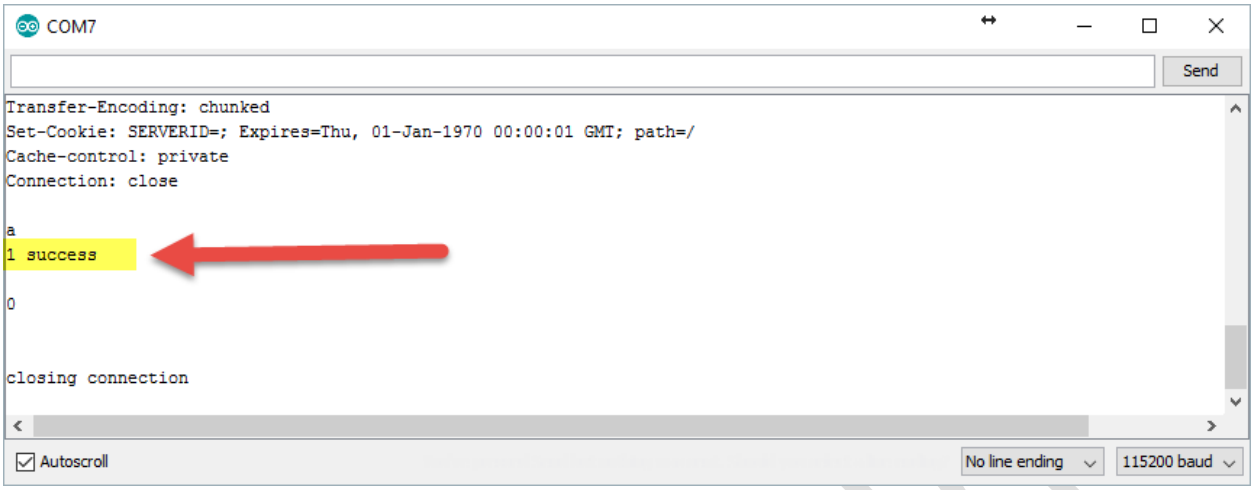
```
#include <ESP8266WiFi.h>

const char* ssid     = "BUAKAEW_AP01";
const char* password = "BuaKaew159357";

const char* host = "data.sparkfun.com";
const char* streamId   = "xR4Nma5ZX8sMYx6mnEz3";
const char* privateKey = "ZawVgyXql1cAoyvgMd9D";

void setup() {
  Serial.begin(115200);
  delay(10);
```

WiFiClient §

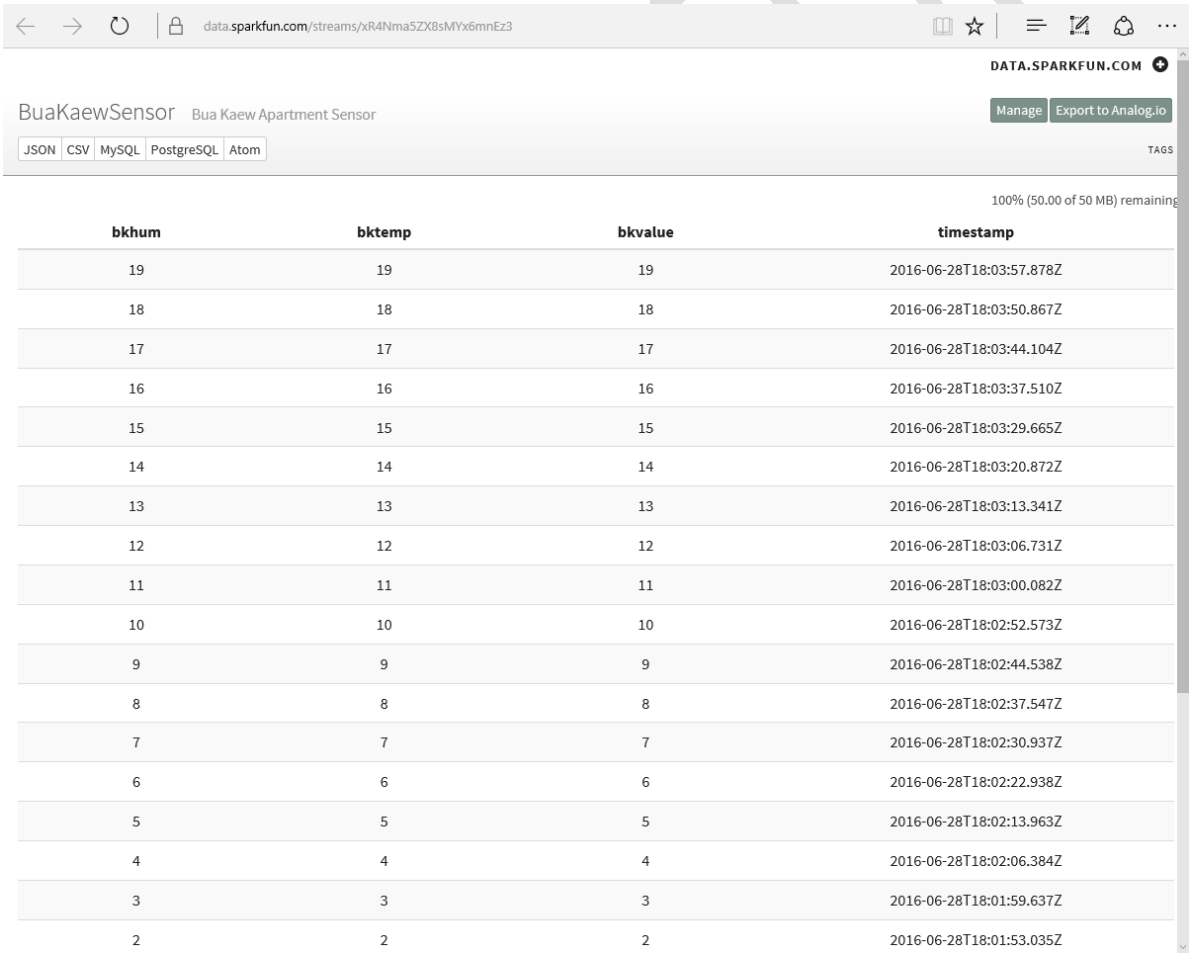
```
  // Use WiFiClient class to create TCP connections
  WiFiClient client;
  const int httpPort = 80;
  if (!client.connect(host, httpPort)) {
    Serial.println("connection failed");
    return;
  }

  // We now create a URI for the request
  String url = "/input/";
  url += streamId;
  url += "?private_key=";
  url += privateKey;
  url += "&bkhum=";
  url += value;
  url += "&bktemp=";
  url += value;
  url += "&bkvalue=";
  url += value;

  Serial.print("Requesting URL: ");
  Serial.println(url);

  // This will send the request to the server
  client.print(String("GET ") + url + " HTTP/1.1\r\n" +
               "Host: " + host + "\r\n" +
               "Connection: close\r\n\r\n");
  unsigned long timeout = millis();
  while (client.available() == 0) {
    if (millis() - timeout > 5000) {
      Serial.println(">>> Client Timeout !");
      client.stop();
      return;
    }
  }
```

COPY

หากมีการแจ้งว่า success แสดงว่าข้อมูลที่ node mcu ส่งไปนั้นได้อัพเดทเข้าไปยัง spark fun เรียบร้อยแล้ว



เราสามารถเข้าไปดูข้อมูลได้จาก public url และข้อมูลเหล่านี้สามารถ export ออกมาในรูปแบบต่างๆได้ไม่ว่าจะเป็น JSON, csv, mysql, ProtgreSQL, atom ซึ่งจะมีค่าข้อมูลต่างๆพร้อมทั้ง TimeStamp



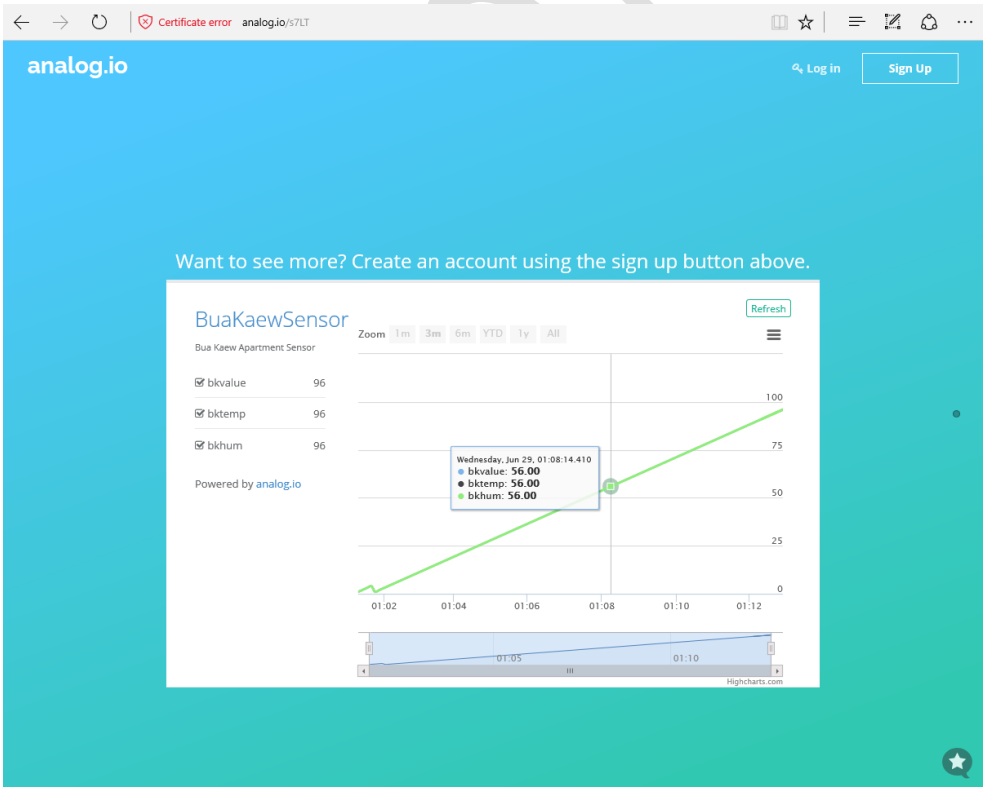
bkhum	bktemp	bkvalue	timestamp
19	19	19	2016-06-28T18:03:57.878Z
18	18	18	2016-06-28T18:03:50.867Z
17	17	17	2016-06-28T18:03:44.104Z
16	16	16	2016-06-28T18:03:37.510Z
15	15	15	2016-06-28T18:03:29.665Z
14	14	14	2016-06-28T18:03:20.872Z
13	13	13	2016-06-28T18:03:13.341Z
12	12	12	2016-06-28T18:03:06.731Z
11	11	11	2016-06-28T18:03:00.082Z
10	10	10	2016-06-28T18:02:52.573Z
9	9	9	2016-06-28T18:02:44.538Z
8	8	8	2016-06-28T18:02:37.547Z
7	7	7	2016-06-28T18:02:30.937Z
6	6	6	2016-06-28T18:02:22.938Z
5	5	5	2016-06-28T18:02:13.963Z
4	4	4	2016-06-28T18:02:06.384Z
3	3	3	2016-06-28T18:01:59.637Z
2	2	2	2016-06-28T18:01:53.035Z

LAB15: วิธีเอา Data จาก SparkFun ไปแสดงผลที่ AnalogIO

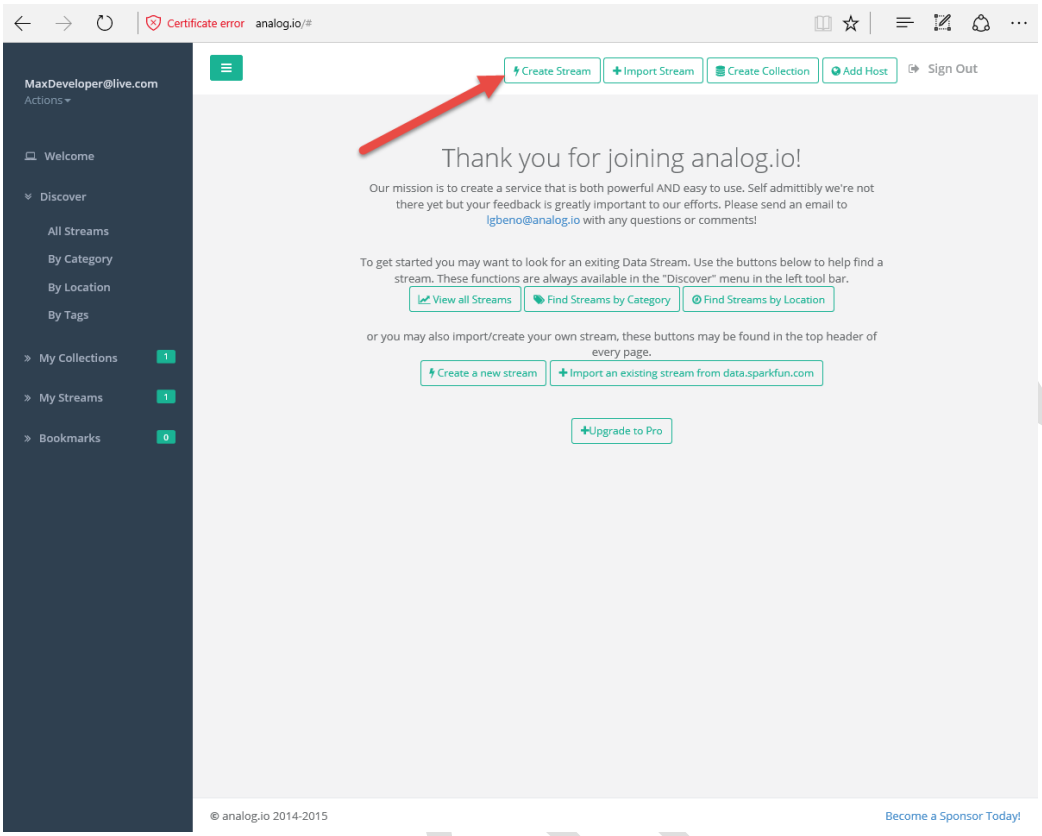
จากการที่ SparkFun ได้ทำการเก็บ data เป็นลักษณะ table มันเป็นการไม่ง่ายเลยที่จะนำข้อมูลเหล่านี้ไปใช้ ดังนั้นเราจึงนำ data นี้ไปแสดงผลในรูปแบบกราฟโดย SparkFun ได้มีอินเตอร์เฟสกับทาง AnalogIO ในการนำข้อมูลเชื่อมต่อและแสดงผลออกมาเป็นรูปแบบของกราฟที่เข้าใจได้ง่าย ในส่วนของหน้า data stream จะมีปุ่มฟังก์ชั่น export to analog io

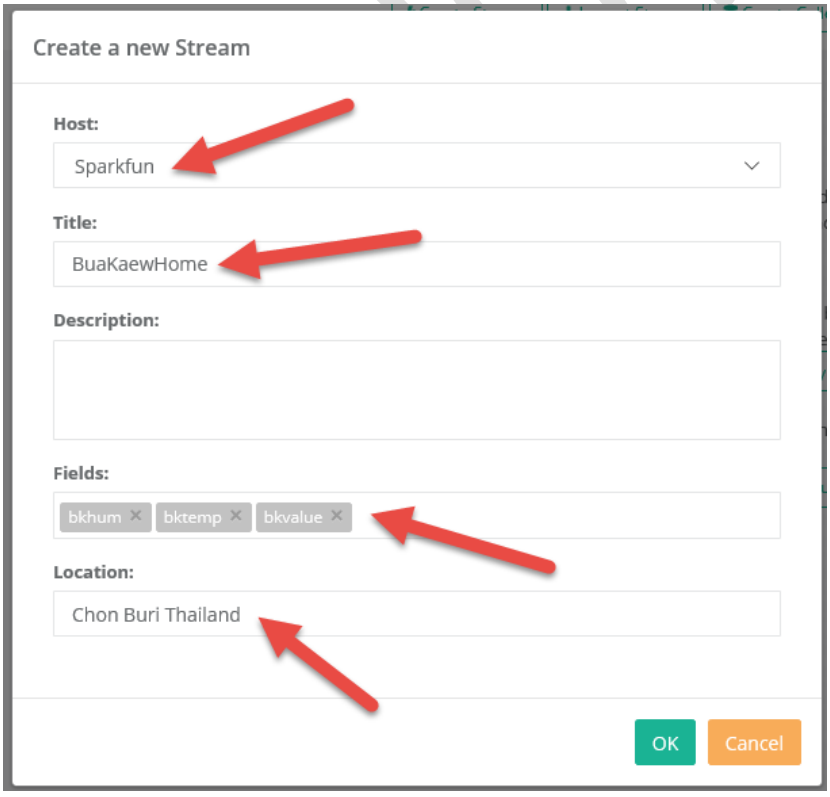
bkhum	bktemp	bkvalue	timestamp
19	19	19	2016-06-28T18:03:57.878Z
18	18	18	2016-06-28T18:03:50.867Z
17	17	17	2016-06-28T18:03:44.104Z

ในส่วนของ analog i o จำเป็นที่จะต้องทำการ register user เพื่อใช้งานในการ manage แต่หากเราไม่ได้ register เราก็สามารถที่จะดูข้อมูลที่สปาร์คฟันส่งมาให้เป็นกราฟได้อย่างง่าย

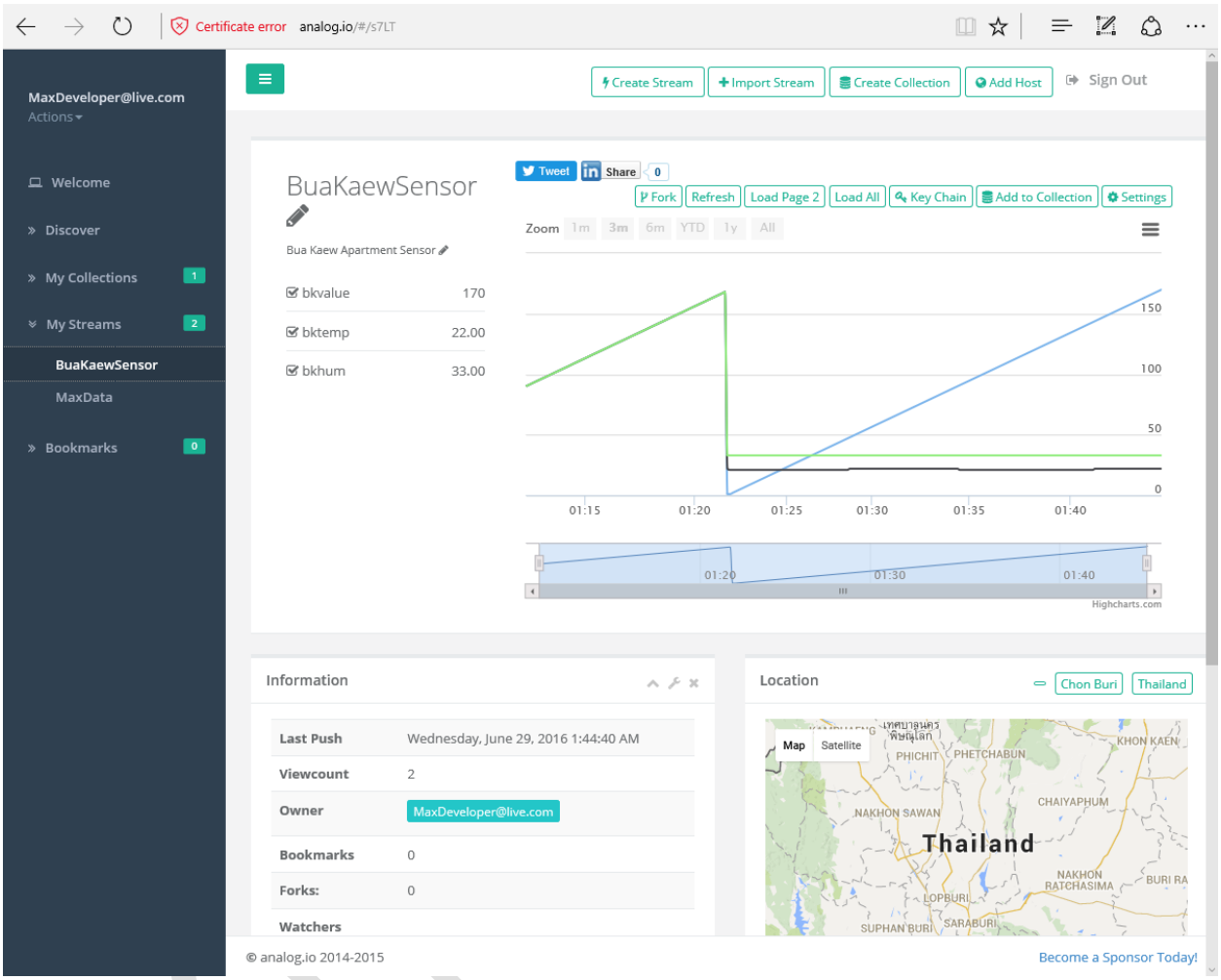


เมื่อทำการ login เข้ามาแล้วเราจะเข้ามายังหน้า manage ของ analog io เราสามารถสร้าง create streamใหม่





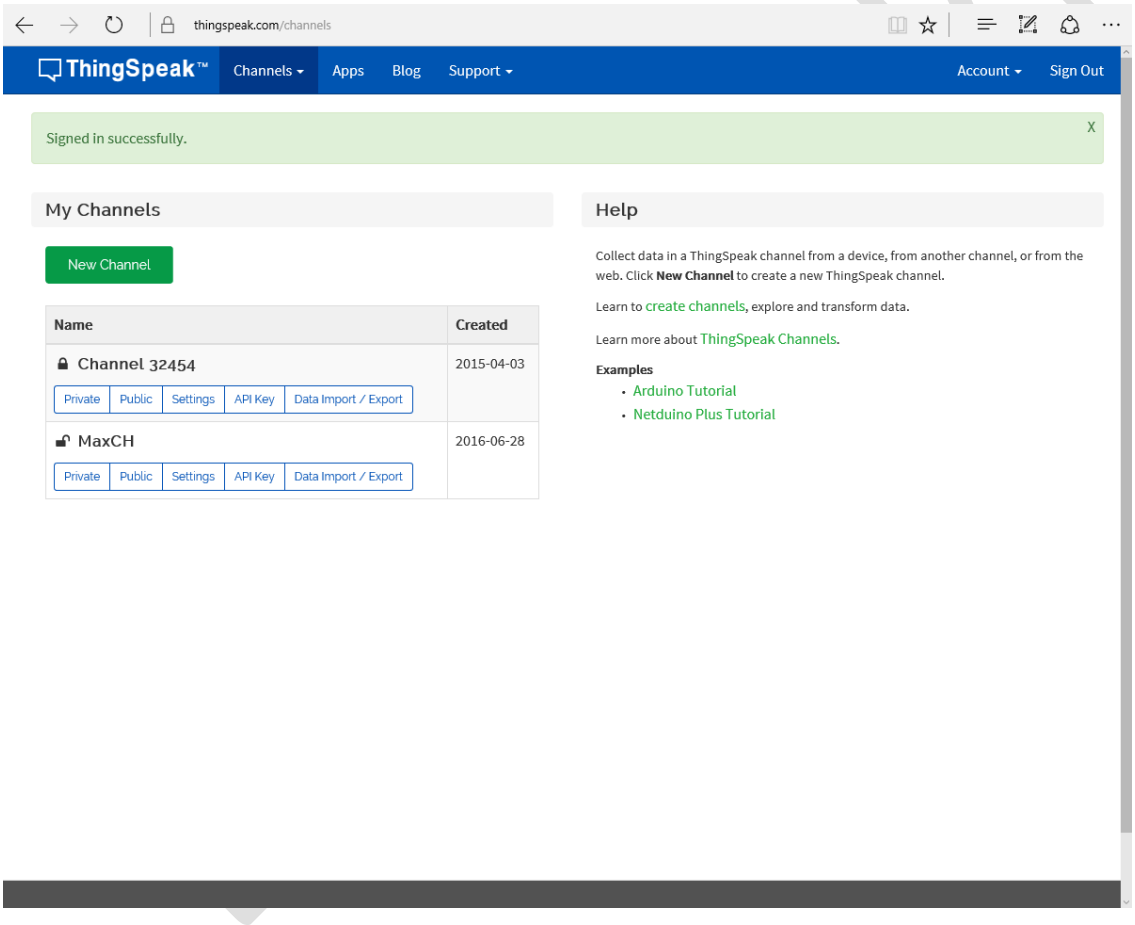
และเมื่อเรา Export มาจาก SparkFun มาที่ analog IO แล้วให้เราเลือก Claim จะขึ้น Dialog มาถามให้ใส่ Private Key เพื่อยืนยัน



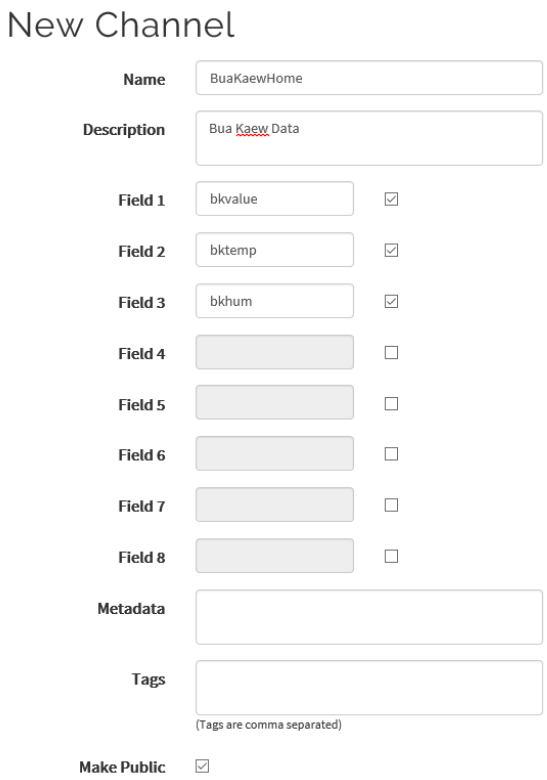
LAB16: ใช้ ThingSpeak เป็น Data Logger พร้อม Plot Graph

Thingspeak เป็นเว็ปที่ให้การบริการในการเก็บข้อมูล และสามารถแสดงข้อมูลแบบ real-time ได้ ซึ่งเราสามารถ 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

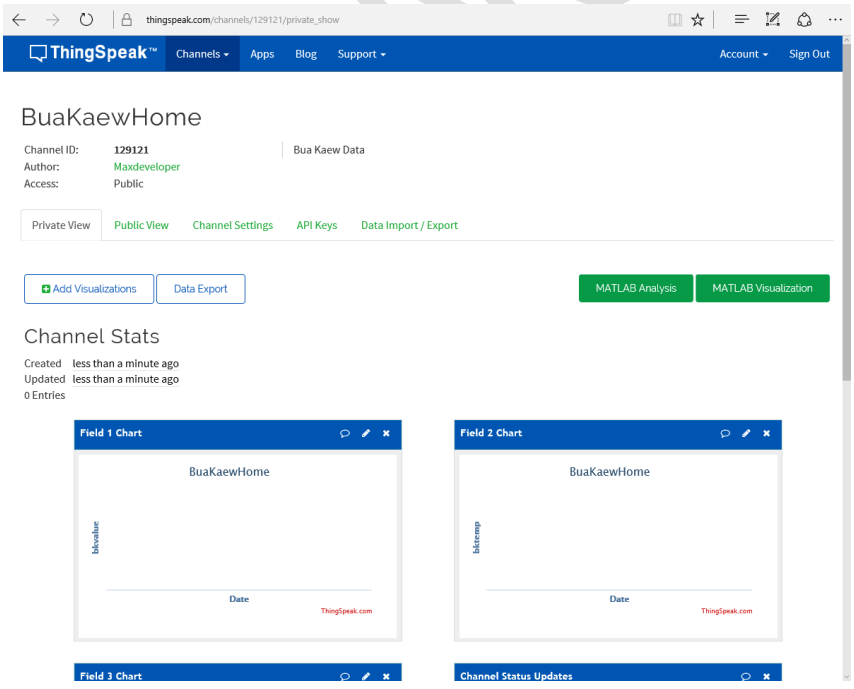
อันดับแรกเลย ให้ทำการสมัครสมาชิกให้เรียบร้อย จากนั้นก็สร้าง channel ขึ้นมา โดยให้เรากดไปที่ My channel แล้วก็ new channel ขึ้นมา



หลังจากที่ new channel ขึ้นมาแล้ว ไปที่ channel setting ก็ป้อนข้อมูลเข้าไป เสร็จแล้วก็กด save channel

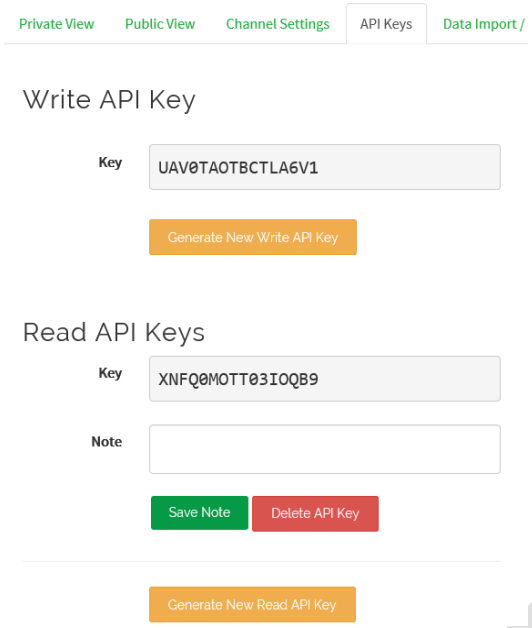


เมื่อสร้างเสร็จแล้ว ก็จะแสดงหน้าต่าง ในหน้าต่างนี้จะแสดงข้อมูลเป็นแบบเส้นกราฟ ซึ่งตอนนี้ยังไม่มีข้อมูลใด ๆ ส่งมาจึงไม่เกิดอะไรขึ้น



NOT COPY

ต่อไปจะทำการ update ข้อมูลไปยัง cloud ผ่าน api ที่ thingspeak มีไว้ให้ อันดับแรกให้ดูที่ API KEY ของเราว่ามันคืออะไร API KEY คือ UAV0TAOTBCTLA6V1 ตามรูป



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

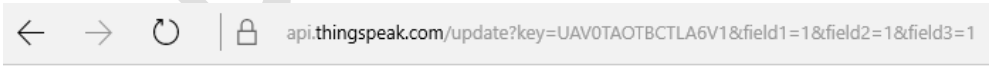
https://api.thingspeak.com/update?key=UAV0TAOTBCTLA6V1&field1=0&field2=0&field3=0

ที่ขีดเส้นไว้คือ

UAV0TAOTBCTLA6V1 คือ API KEY

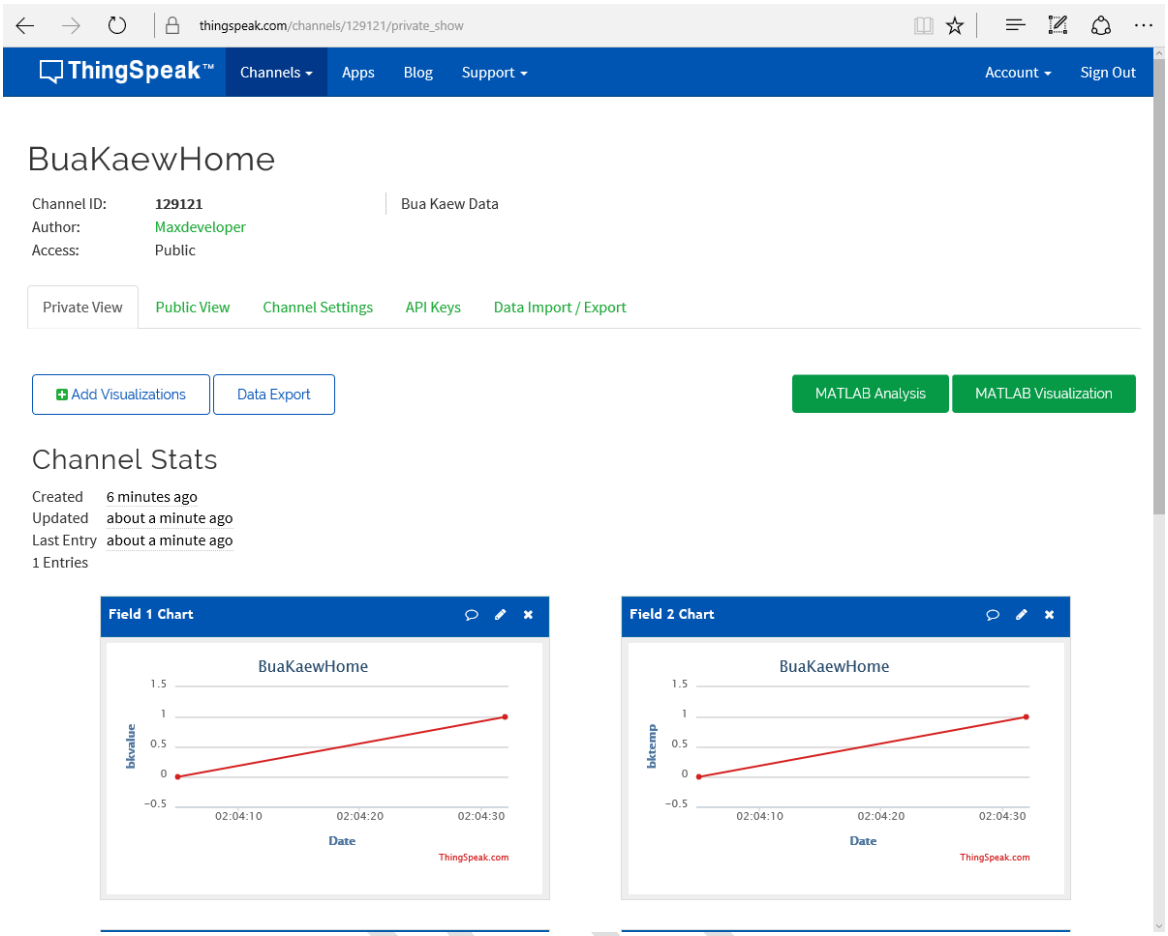
bkvalue=0 คือ field ที่เราต้องการ update ในตัวอย่างคือ field ที่ 1 และกำหนดให้มีค่าเท่ากับ 0

จากรูปตัวอย่างฝั่งขวาคือได้ทำการ update ข้อมูล โดยกำหนดให้ field ที่ 1 มีค่าเท่ากับ 0 และฝั่งซ้ายนั้นได้มีจุดเพิ่มมา 1 จุดก็คือค่าที่เราเพิ่มไปเมื่อซักครู่นั่นเอง



2

DO NOT COPY



ต่อมาเรามาเขียนโปรแกรมให้กับ NodeMcu ให้ส่งข้อมูล update

LAB16: ThingSpeak Data Logger

```
#include <ESP8266WiFi.h>
#include "DHT.h"
#define DHTPIN 14
#define DHTTYPE DHT11
int sensorPin = A0;
int sensorValue = 0;

const char* ssid     = "BUAKAEW_AP01";
const char* password = "BuaKaew159357";
const char* host = "api.thingspeak.com";
const char* APIKey = "UAV0TAOTBCTLA6V1";
```

```
DHT dht(DHTPIN, DHTTYPE);

void setup() {
  Serial.begin(115200);
  dht.begin();
  delay(10);
  // We start by connecting to a WiFi network
  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() {
  delay(10000);
  // Sensor readings may also be up to 2 seconds 'old' (its a very slow sensor)
  float h = dht.readHumidity();
  // Read temperature as Celsius (the default)
  float t = dht.readTemperature();
  // Check if any reads failed and exit early (to try again).
  if (isnan(h) || isnan(t) ) {
```

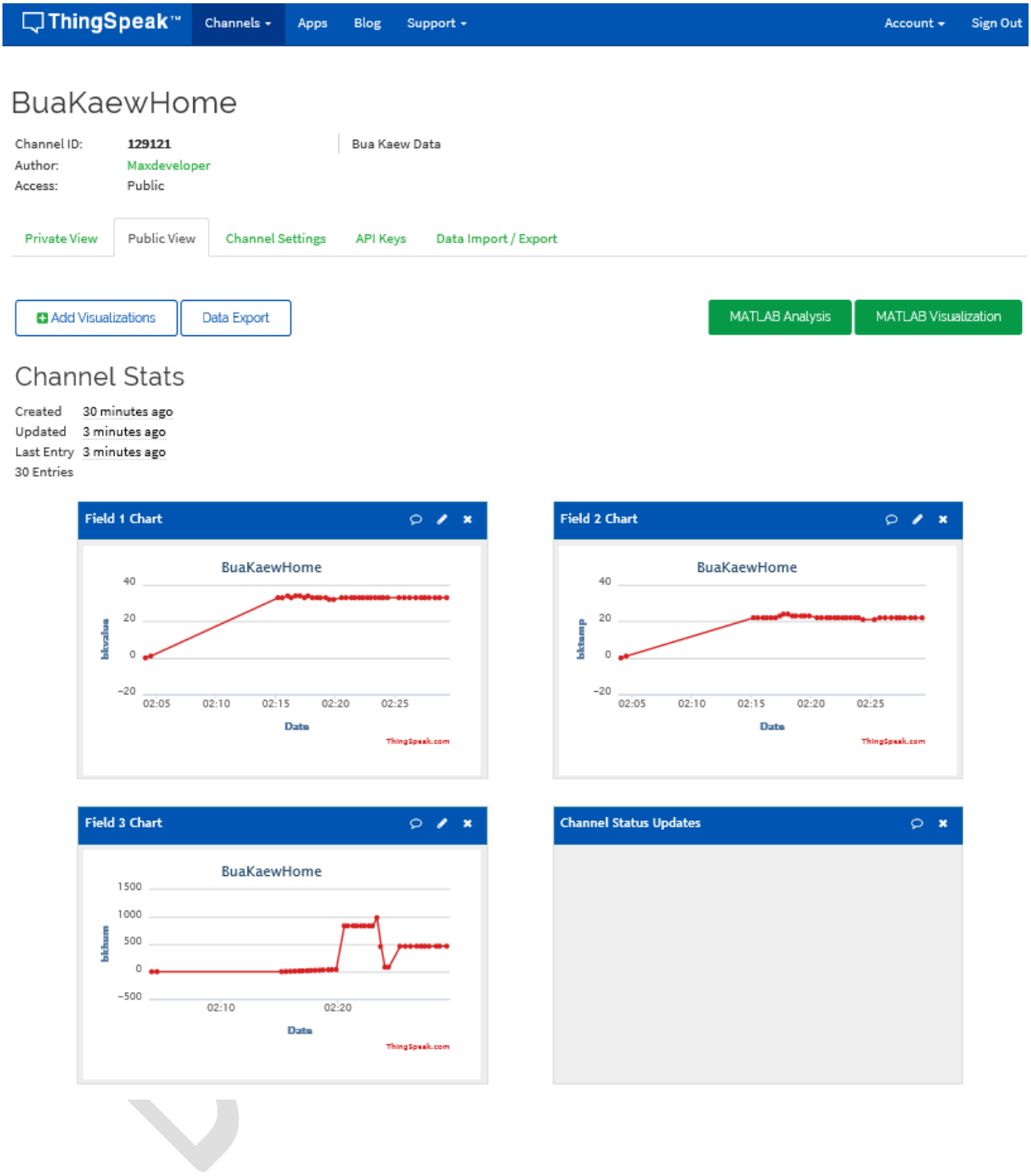
DO NOT COPY

```
  Serial.println("Failed to read from DHT sensor!");
  return;
}
Serial.print("connecting to ");
Serial.println(host);

// Use WiFiClient class to create TCP connections
WiFiClient client;
const int httpPort = 80;
if (!client.connect(host, httpPort)) {
  Serial.println("connection failed");
  return;
}
sensorValue = analogRead(sensorPin);
// We now create a URI for the request
String url = "/update";
url += "?key=";
url += APIKey;
url += "&field1=";
url += h;
url += "&field2=";
url += t;
url += "&field3=";
url += sensorValue;

Serial.print("Requesting URL: ");
Serial.println(url);

// This will send the request to the server
client.print(String("GET ") + url + " HTTP/1.1\r\n" +
             "Host: " + host + "\r\n" +
```

DO NOT COPY

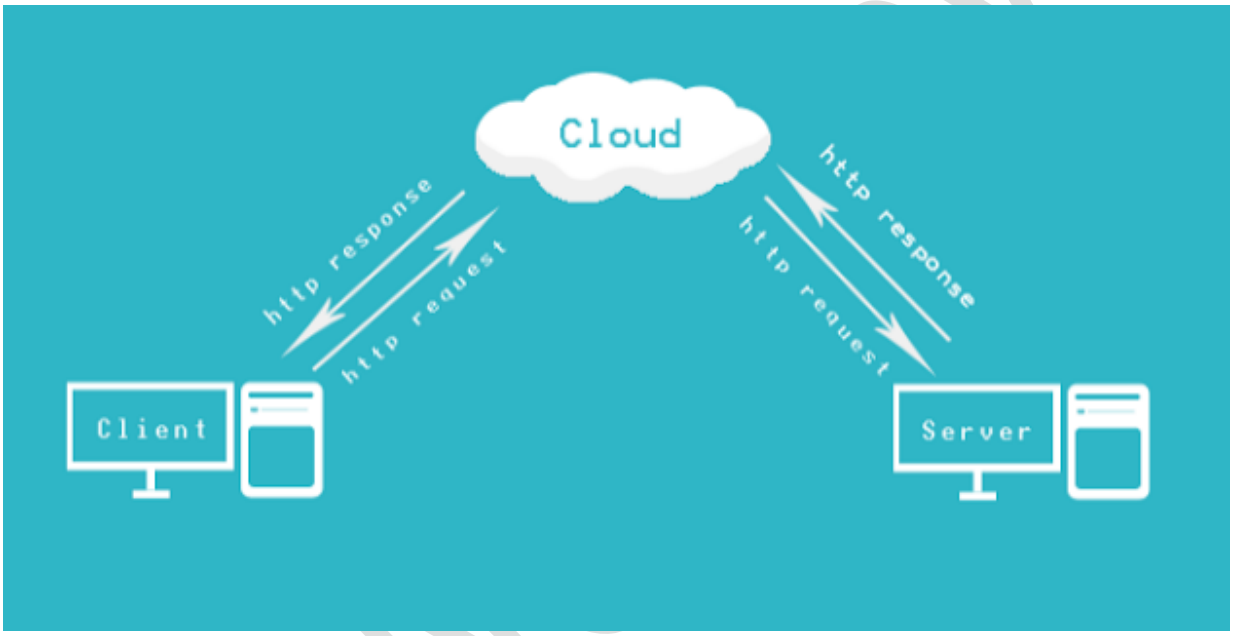
```
                "Connection: close\r\n\r\n");
  unsigned long timeout = millis();
  while (client.available() == 0) {
    if (millis() - timeout > 5000) {
      Serial.println(">>> Client Timeout !");
      client.stop();
      return;
    }
  }

  // Read all the lines of the reply from server and print them to Serial
  while(client.available()){
    String line = client.readStringUntil('\r');
    Serial.print(line);
  }

  Serial.println();
  Serial.println("closing connection");
}
```

DO NOT COPY


ThingSpeak™
Channels
Apps
Blog
Support
Account
Sign Out
BuaKaewHome
Channel ID: 129121
Author: Maxdeveloper
Access: Public
Bua Kaew Data
Private View
Public View
Channel Settings
API Keys
Data Import / Export
Add Visualizations
Data Export
MATLAB Analysis
MATLAB Visualization
Channel Stats
Created 30 minutes ago
Updated 3 minutes ago
Last Entry 3 minutes ago
30 Entries
Field 1 Chart
BuaKaewHome
bkvalue
Date
ThingSpeak.com
Field 2 Chart
BuaKaewHome
bktemp
Date
ThingSpeak.com
Field 3 Chart
BuaKaewHome
bkhum
Date
ThingSpeak.com
Channel Status Updates

LAB17: ใช้ ThingSpeak เป็น ควบคุมตัว NodeMCU

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

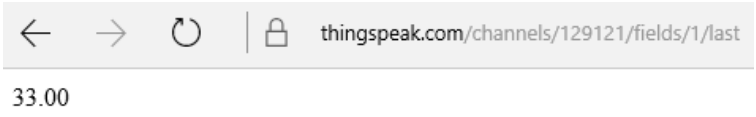


การ get data หรือจะเป็นการ update data เราสามารถศึกษา หรือดู Example ได้จาก Documentation จาก Support ของ ThingSpeak ได้เลย ในส่วนของการ get data นั้นจะใช้วิธีการ get ตามแบบ Example ตัวนี้ ซึ่งเป็นการดึงค่าล่าสุดที่เราบันทึกข้อมูลลง database

จะลอง get data มาดู ซึ่งจะพิมพ์ url นี้ ลงในช่อง search ของ browser

https://api.thingspeak.com/channels/**129121**/fields/1/last

ที่ขีดเส้นใต้ไว้นั้น 129121 คือช่อง channels ของเรานะครับ ส่วน 1 คือ ต้องการดูที่ field ไหน ส่วนเลือกดูที่ field 1



33.00

จากรูปข้างบนก็คือลองเข้าไปยัง url ดังกล่าวผ่าน web browser และเชื่อมต่อไปยัง link นั้น แล้วจะมี response ตอบกลับมาเป็นเลข 2 นั่นก็คือค่าที่เราต้องการที่จะทราบว่า ค่าที่บันทึกล่าสุดของ field ที่ 1 คือค่าอะไร ก็มีการ response กลับมาเป็นค่าดังกล่าว

ตัวอย่างนี้เป็นเพียงการทดสอบดูว่าสามารถ get ค่าจาก server ได้หรือไม่ ต่อมาเราก็มาดูกันว่า เราจะเขียนให้ NodeMcu ของเรานั้นไป get ค่ามาแล้วมาควบคุม LED ได้อย่างไร

LAB17: ThinkSpeak Control

```
#include <ESP8266WiFi.h>

const char* ssid = "BUAKAEW_AP01"; //ใส่ SSID WiFi ที่ต้องการเชื่อมต่อ
const char* password = "BuaKaew159357"; //ใส่ password ของ WiFi ที่ต้องการเชื่อมต่อ
const char* host = "api.thingspeak.com"; // ชื่อ host ที่ต้องการติดต่อ
const char* url = "/channels/129121/fields/1/last"; // path ที่ต้องการเรียก

int lengthData = 0; //ตัวแปรนี้จะเก็บขนาดของข้อมูลที่ทาง Server มีการ response กลับมา
int data = 0; // ตัวแปรนี้จะเก็บค่าข้อมูลที่เราต้องการใช้งาน

String line = ""; // ตัวแปรนี้จะเก็บค่าทั้งหมดที่อ่านได้จาก Server

int led1 = 14; //กำหนดขา GPIO OUTPUT ซึ่ง pin นี้จะตรงกับ D1

const int PORT = 80; // ใช้ port 80

WiFiServer server(80); // สร้าง object จาก คลาส WiFiServer กำหนด port 80

void setup() {
 Serial.begin(115200); //ใช้ buadrate 115200
 delay(10);


 pinMode(led1,OUTPUT); // กำหนด ขา ให้เป็น ขา OUTPUT


 Serial.println("\n\n");
 Serial.print("Connecting to ");
```

DO NOT COPY

```
 Serial.println(ssid);  // แสดงข้อความออกทาง Serial ว่า Connecting to (SSID ที่เชื่อมต่อ)
 WiFi.begin(ssid, password);  // เชื่อมต่อ wifi จาก SSID และ password ที่เรากำหนด

 while (WiFi.status() != WL_CONNECTED){ // ถ้าหากยังเชื่อมต่อกับ wifi ยังไม่ได้ก็จะแสดงข้อความ ... ไป
เรื่อย ๆ
   delay(500);
   Serial.print(".");
 }

 Serial.println();
 Serial.println("WiFi Connected!!");
 Serial.print("IP address : ");
 Serial.println(WiFi.localIP());  // แสดง IP address ที่ได้รับจาก wifi ที่เราเชื่อมต่อ
}

void loop() {
 delay(5000); //delay ไว้ 5 วิ
 Serial.print("Connecting to ");
 Serial.println(host); //แสดงข้อความทาง Serial ว่า Connecting to (host ที่ต้องการเชื่อมต่อ)

 WiFiClient client = server.available(); // สร้าง client ขึ้นมาจาก WiFiClient และกำหนดให้เท่ากับ
Server.available() เพื่อไว้เช็คเวลามีข้อมูลส่งกลับมา

 Serial.print("Requesting URL : ");
 Serial.println(url); // แสดง url ที่เรา request ไป

 if(!client.connect(host,PORT)){
   Serial.print("."); //ในเงื่อนไขนี้เช็คว่าถ้าไม่ได้เชื่อมต่อกับ host อยู่ให้แสดง . ออกมา
   return;
 }else{
       // และถ้าหากเชื่อมต่ออยู่ให้ส่ง http request ไปที่ server
```

DO NOT COPY

```
  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 !!"); // เมื่อทำการเก็บค่าไว้หมดแล้วก็จะแสดง GET SUCCESSFULLY ออกมา

 convertDataToInt(&lengthData , &data ,line); // ใน function นี้เขียนขึ้นมาเพื่อตัดข้อความบางส่วนออก เอาเฉพาะที่เราจำเป็นต้องใช้งานจริง ๆ ก็จะมี lengthData และ data

 String s =(String)"ID "+lengthData+" data "+data; // สร้างตัวแปร String มาดูผลลัพธ์ว่าข้อมูลที่ได้จาก Server
 Serial.println(s); // แสดงข้อมูลที่ผ่านการ filter เรียบร้อยแล้ว
  switch(data){ // แล้วก็ต่อมาจะเป็นเงื่อนไขกันเราก็เช็คกับตัวแปร data ว่าเป็นค่าอะไรบ้าง เพื่อที่เราจะเอามาทำเป็นเงื่อนไขในการเปิดปิด LED
  case 1:
     digitalWrite(led1,HIGH); // ถ้าเป็น 1 led1 ติด
```

DO NOT COPY

```
      break;
    case 0:
       digitalWrite(led1,LOW);// ถ้าเป็น 2 led1 ดับ
      break;
 }

 line = ""; // เมื่อจบ process ทุกอย่างแล้ว ก็เคลียข้อมูลในตัวแปร line ให้เท่ากับ "" เพื่อรอรับค่าใหม่ใน loop หน้า
 Serial.println();
 Serial.println("Closing Connection"); //แสดงข้อความ Closing Connection
}

void convertDataToInt(int* id_ ,int* data_ ,String rawData){ // พารามิเตอร์ ก็จะมี pointer id_ , pointer data_ , ค่าตัวแปรดิบที่อ่านได้จากที่ server response กลับมา
//Serial.println(rawData); // ลบ comment ออก ถ้าต้องการดูว่าค่าที่ response กลับมามีอะไรบ้าง
String dataSub = rawData.substring(rawData.indexOf("close")+9); //สร้างตัวแปรมาเพื่อ substring เฉพาะข้อมูลที่เราต้องการใช้จริง ๆ
 Serial.println(dataSub.length()); // ลบ comment ออก ถ้าต้องการดูขนาดข้อมูลหลังจากที่เรา substring จากข้อมูลทั้งหมดแล้ว
 Serial.println(dataSub); //ลบ comment ออก ถ้าต้องการดูว่าข้อมูลที่เรา substring ออกมามีข้อมูลเป็นอย่างไร

*id_ = dataSub.substring(0,2).toInt();  // ทำการแปลงข้อมูลจาก String เป็น int ไปเก็บข้อมูลไว้ที่ตัวแปร pointer เพื่อเปลี่ยนค่าให้กับตัวแปรที่เราอ้างอิง ก็คือตัวแปร lengthData นั่นเอง
*data_ = dataSub.substring(3 , 3 + *id_).toInt(); // ทำการแปลงข้อมูลจาก String เป็น int ไปเก็บข้อมูลไว้ที่ตัวแปร pointer เพื่อเปลี่ยนค่าให้กับตัวแปรที่เราอ้างอิง ก็คือตัวแปร data
data=dataSub.substring(0,1).toInt();
}
```

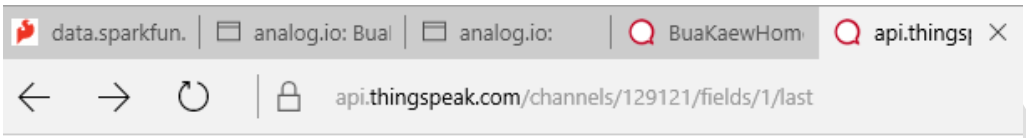
เป็นการ get ค่าจาก Server โดยส่ง http request ขอดูในข้อมูลล่าสุดของ field 1 ถ้าส่ง request ถูกต้อง Server ก็จะส่ง http response กลับมาเป็นค่าที่เราต้องการ เมื่อได้ค่าที่ต้องการแล้วก็ทำการ substring ก็คือตัดเอาข้อความเอาเฉพาะที่จำเป็นมาใช้งาน ตามที่อธิบายไว้แต่ละบรรทัดที่สำคัญ

DO NOT COPY

เมื่อเรา update ค่าเป็น 0

103

ผลที่ได้จาก last ของ field 1 ก็จะเป็น 0 และ LED จะดับ

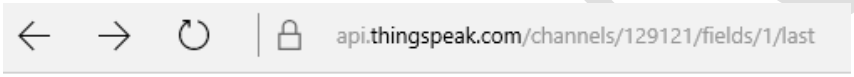


0

เมื่อเรา update ค่าเป็น 1

104

ผลที่ได้จาก last ของ field 1 ก็จะเป็น 1 และ LED จะดับ

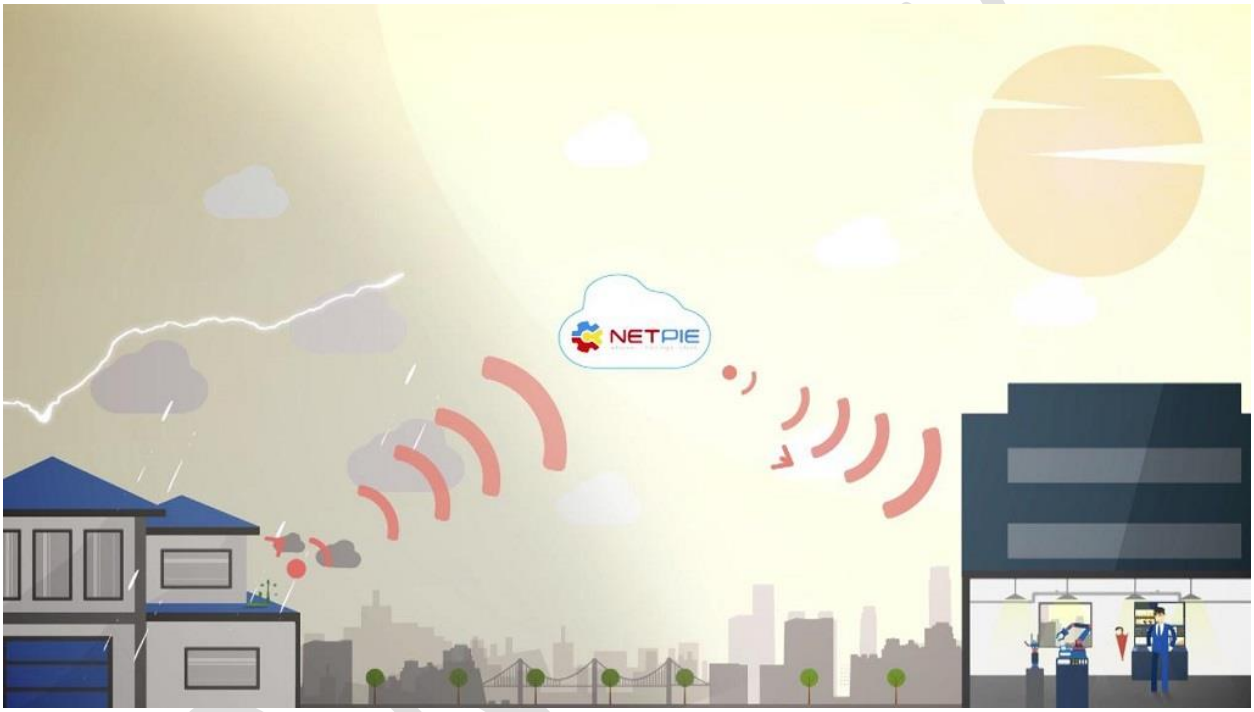


1

DO NOT COPY

NETPIE

“NETPIE แพลตฟอร์ม IoT เพื่อนักพัฒนาและอุตสาหกรรมไทย” โดยทีมพัฒนาของ Nectec ซึ่งเป็นแพลตฟอร์มทางเลือกแรกของนักพัฒนาไทยที่เชื่อมอุปกรณ์และเครื่องมือต่าง ๆ หรือ The Internet of Things (IoT) ระยะแรกเน้นการสนับสนุนนักพัฒนาและอุตสาหกรรมขนาดย่อม(SMEs) เพื่อสร้างขีดความสามารถและความเข้มแข็งให้กับอุตสาหกรรมไทยขนาดใหญ่ของไทย ซึ่งเป็น Platform ที่เราสามารถใช้ฟรีถึง 100 อุปกรณ์ต่อ Account และ Server อยู่ที่เมืองไทยมาลองเล่น NetPIE ไทยทำ ไทยใช้ กัน



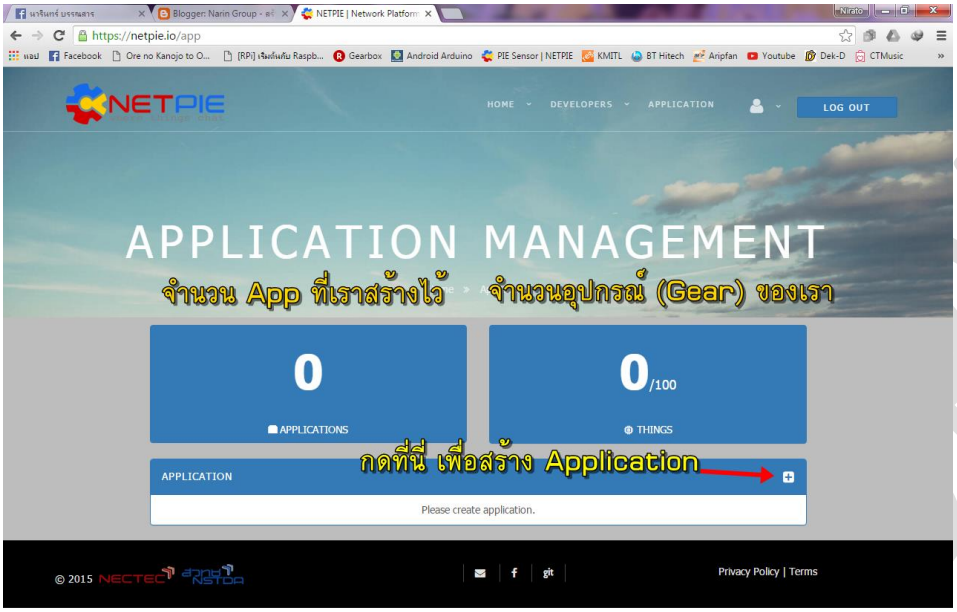
Download Microgear https://github.com/netpieio/microgear-esp8266-arduino

จะได้เป็น zip file ออกมาให้เรากินสะตอเข้าไปใน library

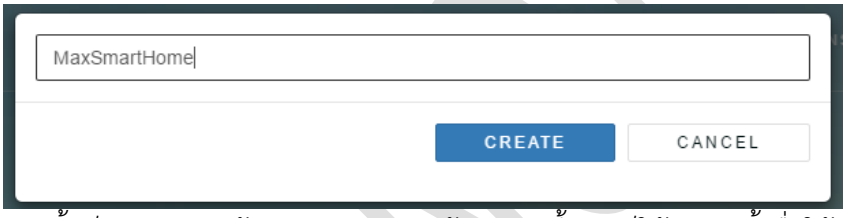
ก่อนเริ่มเราต้องสมัคร NETPIE ก่อน

การสร้าง Application (แอพพลิเคชั่นเป็นตั (วแทนของระบบของเรา ("ระบบสมาร์ทโฮมในบ้าน" เช่น) ในระบบของเราก็จะสามารถ เพิ่มอุปกรณ์เข้าไปได้ โดยอุปกรณ์ของเรานั้นจะถูกเรียกว่า "Gear" (ปัจจุบันใช้คำว่า Things) ซึ่งอุปกรณ์ที่ว่านั้นก็อย่างเช่น ตัว NodeMCU ที่เรากำลังจะทำเป็นตัวควบคุมสวิชต์ไฟ หรือพวกบอร์ดอื่นๆที่มีการต่อเซนเซอร์สำหรับส่งข้อมูลค่าต่างๆ เช่นอุณหภูมิและความชื้น เป็นต้น ใน Application หนึ่งอาจจะมีหลายๆ อุปกรณ์ หลายๆ)Gear ก็ได้ และแต่ละ (Gear ภายในระบบเดียวกัน ภายใน) Application เดียวกันก็ (จะสามารถสื่อสารกันเองระหว่างอุปกรณ์ได้

และแต่ละอุปกรณ์ แต่ละ)Gear) ก็จะมี Key เป็นของตัวเอง ซึ่ง Key นี้ก็จะเป็นเหมือนสิ่งที่ใช้ระบุตัวตนของอุปกรณ์เรา (คล้ายๆกับเลขบัตรประจำตัวประชาชน)แต่ละ Gear ก็จะมี Key เป็นของตัวเอง และก็จะไม่เหมือนกับของ Gear อื่นๆ

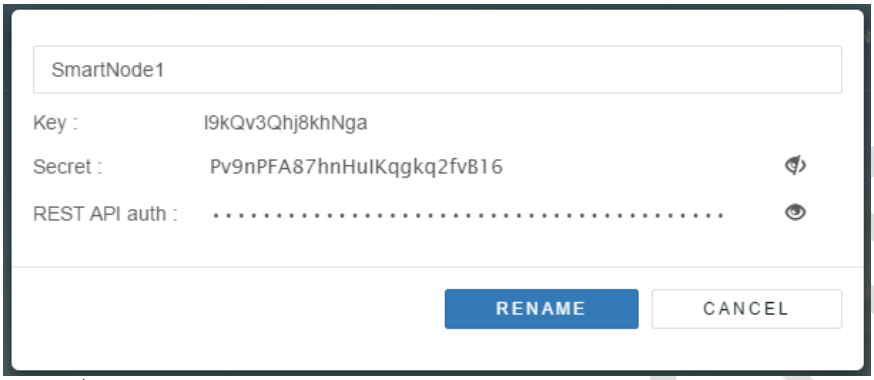


เมื่อเข้ามาใน Application แล้ว ทำการสร้าง Application เช่น MaxSmartHome



จากนั้นก็จะปรากฏหน้าต่างสำหรับการสร้าง Key ขึ้นมา ก็ให้ทำการตั้งชื่อให้อุปกรณ์ของเรา เช่น ตั้งชื่อเป็น SmartNode1 (ประมาณว่ากล่องสวิชต์เปิดปิดไฟของเราชื่อ SmartNode ตัวที่ 1) แล้วก็ทำการเลือก Type ให้เป็น Session Key จากนั้นก็กด CREATE เลยครับ

ทีนี้ก็ให้กดที่ Key ที่สร้างไว้ ก็จะปรากฏข้อมูล Key ขึ้นมาหากจะเปลี่ยนชื่อของอุปกรณ์ ก็พิมพ์ชื่อใหม่ด้านบน แล้วก็กดปุ่ม RENAME



ข้อมูลที่เราจะนำไปใช้ก็จะมี Key กับ SecretKey

DO NOT COPY

LAB18: เปิดปิดไฟด้วย NETPIE ผ่านเว็บ HTML5

Code NodeMCU

LAB18: HTML5 Control by NETPIE

```
#include <AuthClient.h>
#include <MicroGear.h>
#include <MQTTClient.h>
#include <SHA1.h>
#include <Arduino.h>
#include <ESP8266WiFi.h>
#include <EEPROM.h>
const char* ssid     = "BUAKAEW_AP01";
const char* password = "BuaKaew159357";
#define APPID       "MaxSmartHome"
#define GEARKEY     "l9kQv3Qhj8khNga"
#define GEARSECRET  "Pv9nPFA87hnHulKqgkq2fvB16"
#define SCOPE       "MaxPlug1"
WiFiClient client;
AuthClient *authclient;
int relayPin = D4;
MicroGear microgear(client);
void onMsghandler(char *topic, uint8_t* msg, unsigned int msglen) {
  Serial.print("Incoming message --> ");
  Serial.print(topic);
  Serial.print(" : ");
  char strState[msglen];
  for (int i = 0; i < msglen; i++) {
    strState[i] = (char)msg[i];
    Serial.print((char)msg[i]);
  }
```

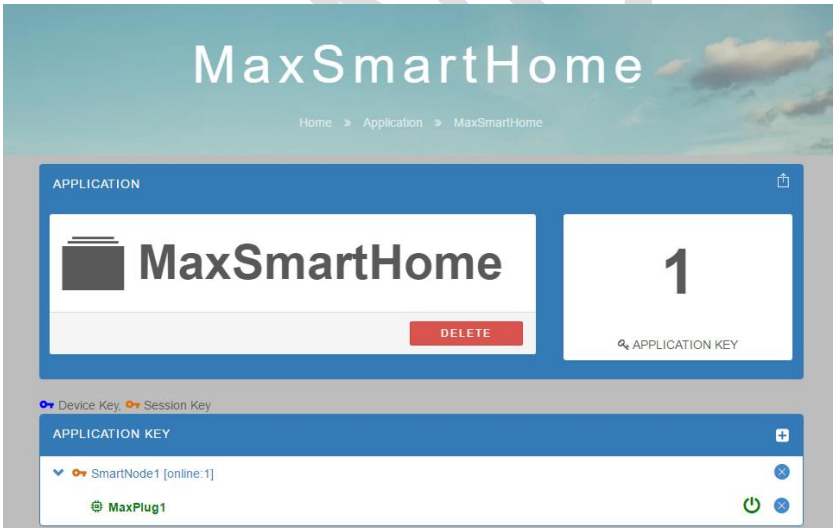
DO NOT COPY

```
  Serial.println();
  String stateStr = String(strState).substring(0, msglen);
  if (stateStr == "ON") {
    digitalWrite(relayPin, LOW);
    microgear.chat("controllerplug", "ON");
  } else if (stateStr == "OFF") {
    digitalWrite(relayPin, HIGH);
    microgear.chat("controllerplug", "OFF");
  }
}
void onConnected(char *attribute, uint8_t* msg, unsigned int msglen) {
  Serial.println("Connected to NETPIE...");
  microgear.setName("pieplug");
}
void setup() {
  Serial.begin(115200);
  Serial.println("Starting...");
  pinMode(relayPin, OUTPUT);
  microgear.on(MESSAGE,onMsghandler);
  microgear.on(CONNECTED,onConnected);

  if (WiFi.begin(ssid, password)) {
    while (WiFi.status() != WL_CONNECTED) {
      delay(500);
      Serial.print(".");
    }
    Serial.println("WiFi connected");
    Serial.println("IP address: ");
    Serial.println(WiFi.localIP());
    //uncomment the line below if you want to reset token -->
```

DO NOT COPY

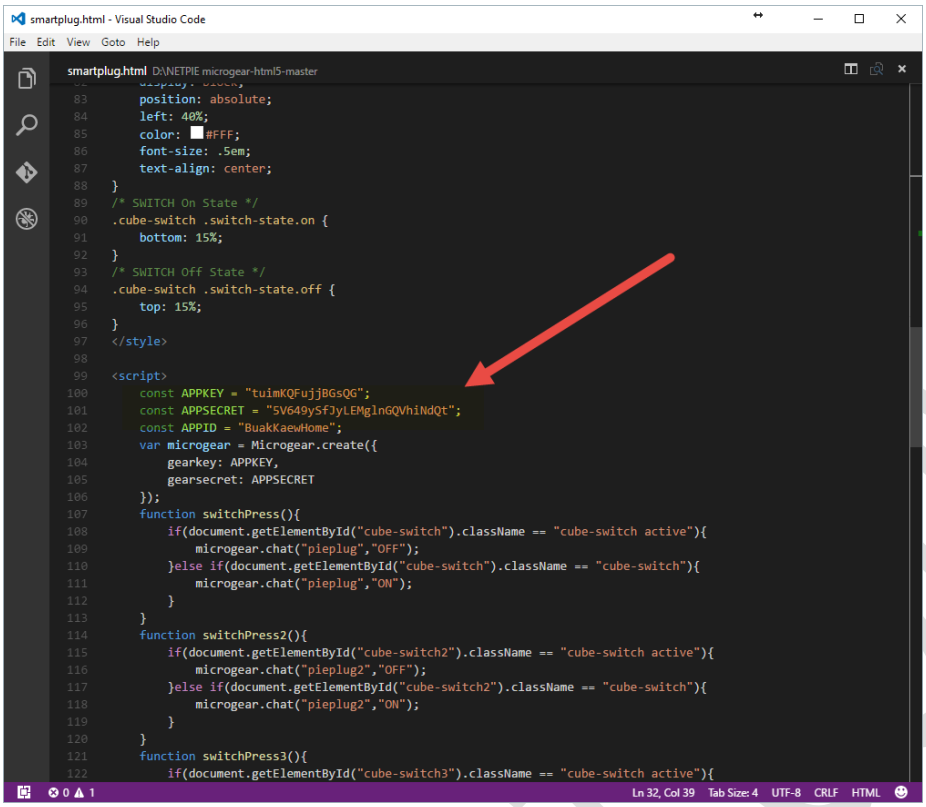
```
    microgear.resetToken();
    microgear.init(GEARKEY, GEARSECRET, SCOPE);
    microgear.connect(APPID);
  }
}

void loop() {
  if (microgear.connected()) {
    microgear.loop();
    Serial.println("connect...");
  } else {
    Serial.println("connection lost, reconnect...");
    microgear.connect(APPID);
  }
  delay(1000);
}
```

หลังจากที่ NodeMCU Connect กับ NETPIE แล้ว จะมี Key ขึ้นในหน้าของ Application



ในส่วนของ HTML5 แก้ไฟล์ smartplug.html แล้วก็ทำการเอา KEY กับ SECRET และ APP ID ที่เราไปสร้างไว้ใน NETPIE มาใส่ลงไป

NOT COPY

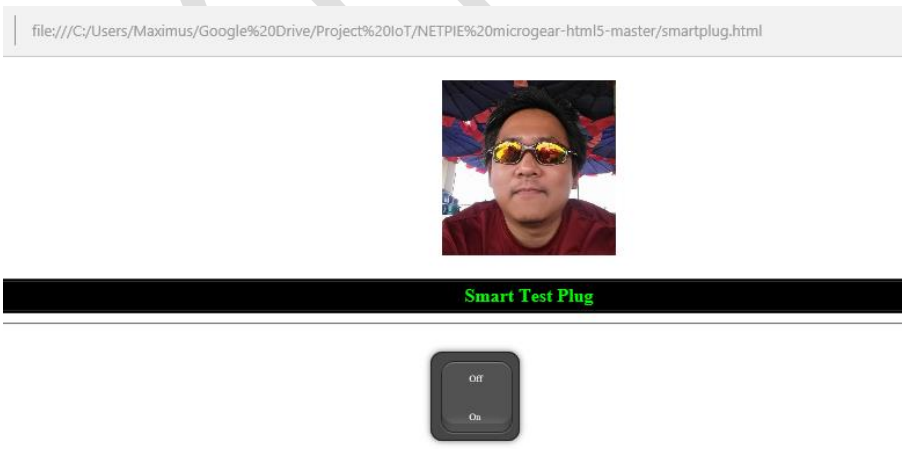


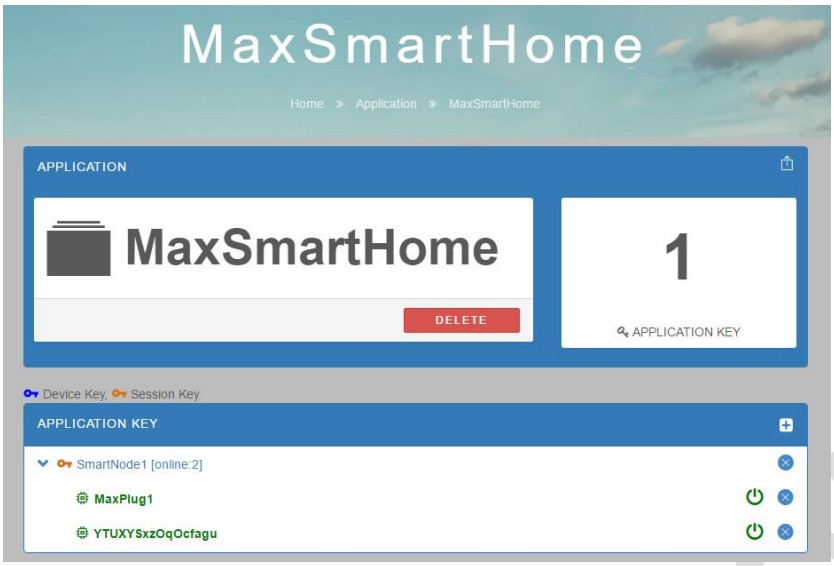
เสร็จแล้วก็กด File > Save เลยครับ

ที่สำคัญคือ นั่นคือ Microgear Library (ของฝั่ง HTML 5) ซึ่งไฟล์นี้จะต้องแยกมาดาวน์โหลดในภายหลัง เนื่องจากทาง NECTEC จะมีการพัฒนาและแก้ไขปรับปรุง Library นี้อยู่เรื่อยๆก็ให้ทำการ Download ไฟล์ Microgear Library สำหรับ HTML5 จากลิ้งค์ด้านล่างนี้ มาลงไว้ในโฟล์เดอร์ที่มีไฟล์ HTML

https://github.com/netpieio/microgear-html5

ผลที่ได้ คือ สามารถเปิดปิดไฟบน Web โดยผ่าน NetPie





LAB19: NETPIE Freeboard

Freeboard นั้นเป็น Web Application ที่ทำให้เราสามารถสร้าง Dashboard (แดชบอร์ดสำหรับ (ระบบIoT ของเราขึ้นมาได้ ถ้าจะให้พูดง่ายๆ ก็คือคล้ายๆกับการสร้าง กระดาน IoT ส่วนตัวที่เราสามารถวางปุ่มกด,สวิชต์ ไว้ใช้สำหรับควบคุมอุปกรณ์ หรือวางหน้าปัดไว้สำหรับแสดงผลข้อมูลต่างๆที่ได้จากอุปกรณ์ในระบบ IoT ของเรา นอกจากนี้ยังสามารถนำข้อมูลที่ได้มาไปแสดงผลเป็นกราฟและสามารถปรับแต่งได้ตามใจชอบได้ง่ายๆ เพียงแค่ คลิ๊กวาง และมีการป้อนข้อมูลหรือคำสั่งอีกนิดหน่อย-ลาก- ก็สามารถทำงานได้แล้วครับคือง่ายมากๆ ไม่ต้องเสียเวลานั่งงมเขียนเป็น HTML Webpage ให้วุ่นวาย ที่สำคัญคือข้อมูลนั้นมีการอัพเดทแบบ real-time มีความเสถียรและเชื่อถือได้ และเป็น Open-Source ซึ่งทำให้นักพัฒนาสามารถต่อยอดให้ดียิ่งขึ้นได้อีกด้วย

NOT COPY

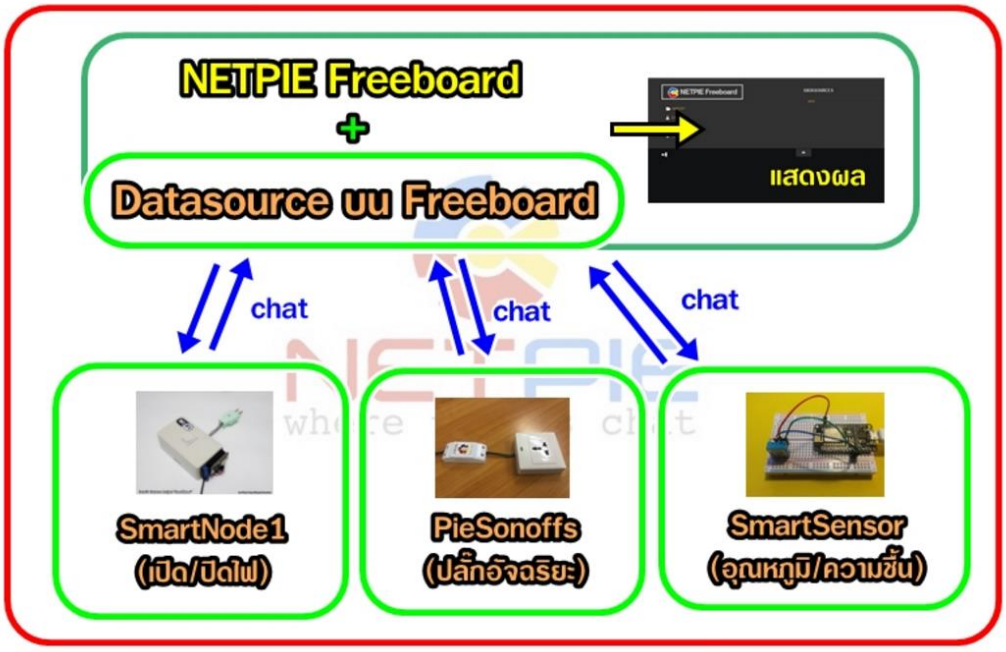
STEP1: ดาวน์โหลดและติดตั้ง NETPIE Freeboard

สำหรับตัว NETPIE Freeboard (และ Freeboard อื่นๆนั้นจะมีระบบไฟล์แบบเดียวกับเว็บเพจต่างๆ (คือจะมีไฟล์ค่อนข้างเยอะหน่อย แต่ถูกจัดเก็บไว้อย่างเป็นระบบในโฟล์เดอร์ต่างๆ ซึ่งไฟล์ส่วนใหญ่นั้นก็จะประกอบด้วยไฟล์เว็บเพจที่เป็น html และไฟล์โค้ด Javascript ต่างๆ โดยไฟล์ทั้งหมดทีมงาน NETPIE ได้อัพขึ้นไว้ที่ Github สามารถเข้าไปดาวน์โหลดได้ตามลิ้งค์ด้านล่างนี้เลย

https://github.com/netpieio/netpie-freeboard

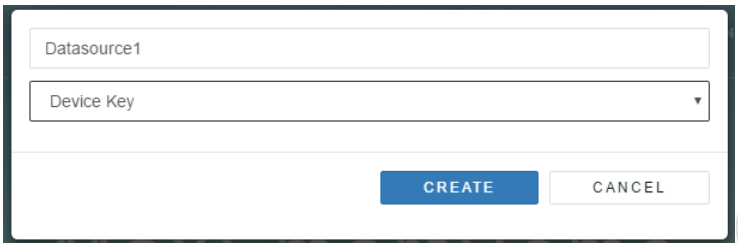
STEP2: ทำความเข้าใจกับ Freeboard Datasource

เนื่องจาก NETPIE Freeboard นั้นจะเป็นหน้าเพจที่แสดงข้อมูลต่างๆ ซึ่งตัวอุปกรณ์)Things) ของเราภายใน AppID เดียวกันนั้นจะมีการ chat เพื่อส่งข้อมูล (ค่าต่างๆจากเซนเซอร์/เช่นค่าอุณหภูมิ)หรือส่งคำสั่งหากัน (ปิด/เปิด)และกัน ซึ่งตัว Freeboard เองก็ต้องทำตัวให้เปรียบเสมือนว่าเป็น Things ตัวหนึ่ง ที่ Things อื่นๆ สามารถ chat ส่งข้อมูลมาหาได้เช่นกัน เพื่อที่จะได้นำข้อมูลเหล่านั้นมาแสดงผลบน Dashboard ของเราตามที่เราต้องการได้ ซึ่งเราจะต้องสร้าง Datasource ขึ้นมา เพื่อเป็นตัวกลางของการสื่อสารรับส่งข้อมูล เป็น) ตัวแทนของFreeboard) ซึ่งสำหรับคนที่ใช้ NETPIE อยู่แล้ว ก็สามารถสร้าง Datasource แบบ NETPIE Microgear ได้เลย โดยใช้)Key จาก NETPIE) หรือสำหรับใครที่ใช้งาน Datasource แบบอื่นก็สามารถเพิ่มได้เช่นกัน โดย NETPIE Freeboard นั้นรองรับ Datasource หลากหลายรูปแบบเลย ทั้ง JSON, Open Weather Map, Dweet.io, Playback, Clock และ Octoblue แต่ในที่นี้ผมจะขอพูดถึง Datasource แบบของ NETPIE Microgear

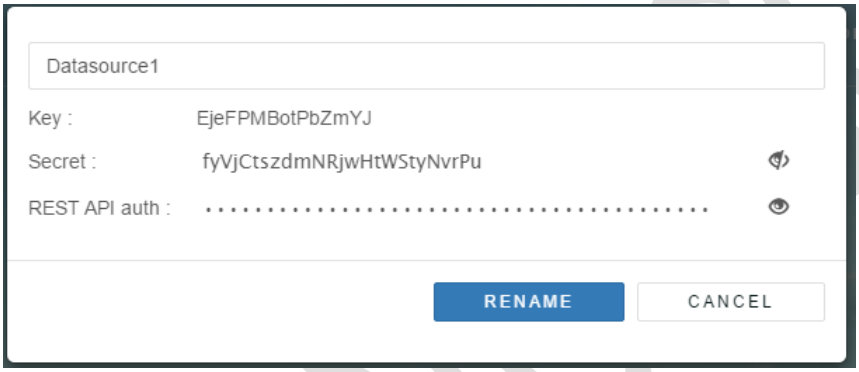


ซึ่งในการเพิ่มDatasource แบบ NETPIE Microgear นั้น ให้ทำการสร้าง Key ใหม่ขึ้นมาใหม่ใน AppID ที่เราต้องการให้ Freeboard แสดงผลและรับส่งข้อมูล)AppID ที่มี Things อื่นๆอยู่ด้วย เพื่อนำ (Key และ Secret ไปใส่ใน Freeboard

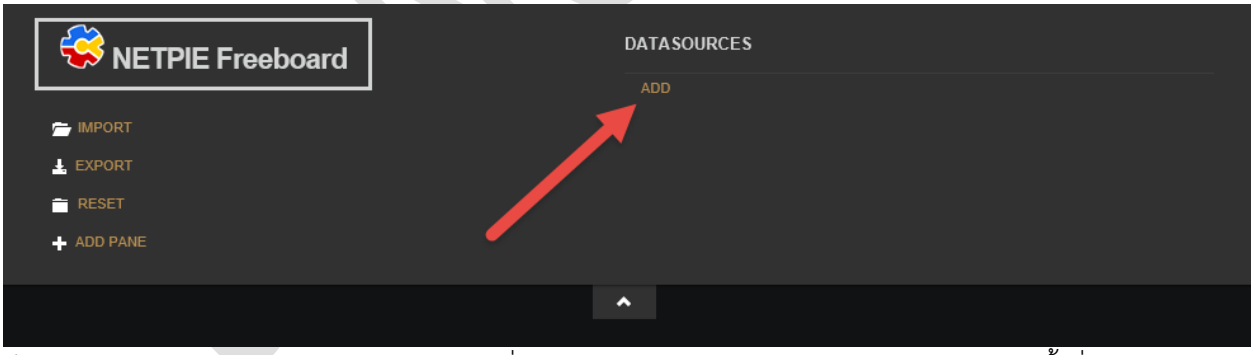
สร้าง Gear ของ Datasource



บันทึกค่า Key เก็บไว้



เปิดไฟล์ index.html เลือก add datasource



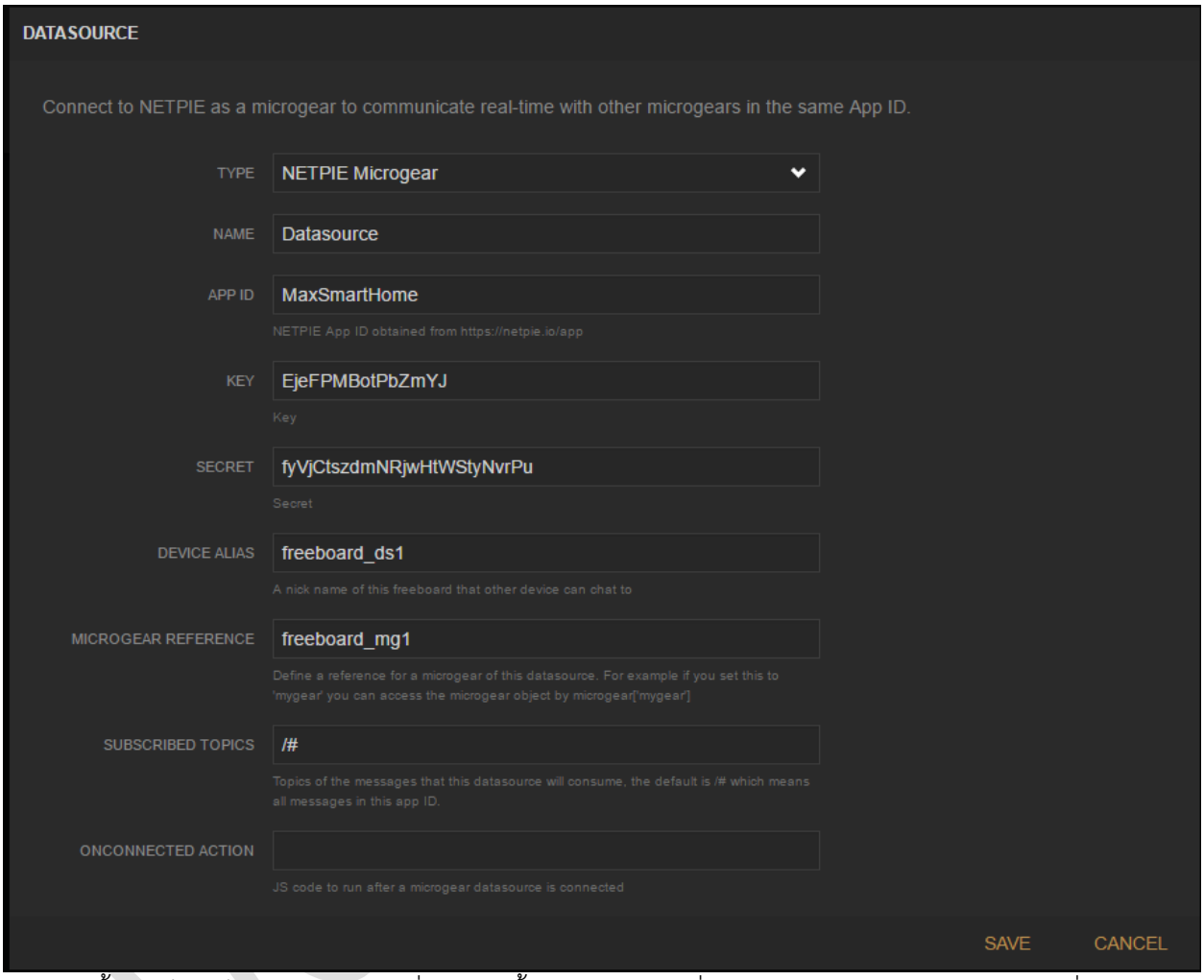
ก็ให้นำข้อมูลจาก Key สำหรับ Datasource ที่เราสร้างไว้ใน NETPIE มากรอกลงไปและทำการตั้งชื่อ Device Alias และ Microgear Reference ไว้ใช้กับคำสั่งต่างๆด้วย (Device Alias จะใช้เป็นเหมือนชื่อเรียกเวลาเราต้องการ chat ไปหา Datasource ผ่านคำสั่งต่างๆ ส่วน Microgear Reference เป็นชื่ออ้างอิง Microgear ครับ เพราะ Datasource จะทำตัวเหมือนกับว่าเป็น Things หนึ่งๆ ทำตัวเป็น)Microgear ตัวหนึ่งแต่ไม่ได้มีตัวตนจับต้องได้จริงๆ(ซึ่งเป็นชื่อที่ Freeboard จะใช้อ้างถึงตัวเอง เวลาจะคำสั่งส่งข้อความหนึ่งๆ โดยจะมีรูปแบบนี้

```
microgear["ชื่อ Microgear Reference"].chat("ชื่อผู้รับ(Alias)","ข้อความ")
```

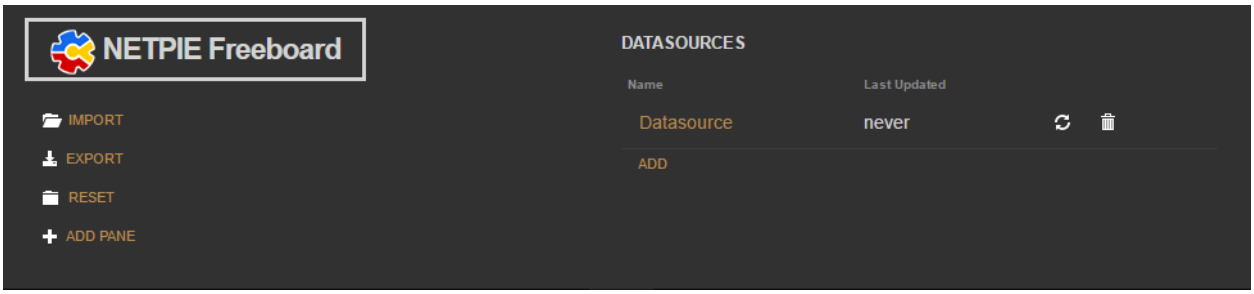
เช่น ผมต้องการใช้ Freeboard ของผม ซึ่งกรอกข้อมูลทุกอย่างดังรูปที่ผ่านมาไปสั่งเปิดไฟหน้าบ้านของ โดยที่ชุดเปิดปิดไฟของผมมี Alias เป็น smartnode1 ก็จะใช้คำสั่งนี้บน Freeboard

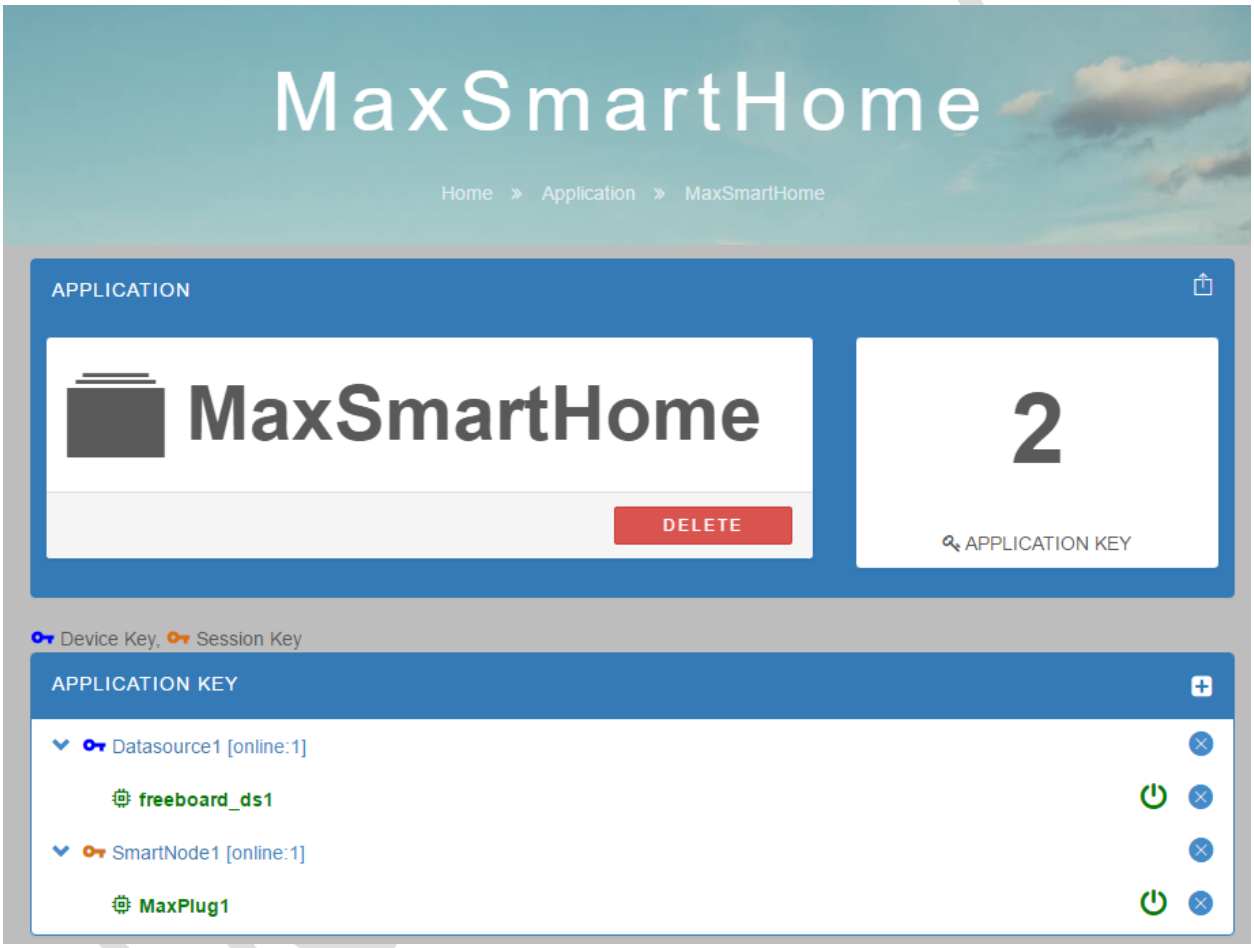
```
microgear["freeboard_mg1"].chat("smartnode1","ON")
```

และเมื่อกรอกข้อมูลเรียบร้อยแล้ว ก็ให้กด Save ด้านล่างเลย



หลังจากนั้นเราก็จะเห็น Datasource ที่เราสร้างขึ้นปรากฏในรายชื่อ Datasource ใน Freeboard หนึ่งจะใช้หลายๆ Datasource ก็ได้ ไม่ได้จำกัดว่าต้องใช้แค่ Datasource เดียวโดยหลังชื่อ Datasource จะแสดงข้อมูล Last Updated ซึ่งจะแสดงเวลาที่มีการอัพเดทข้อมูลล่าสุด (ในตอนนี้ Datasource ของเรายังไม่มีการรับส่ง chat ครับเลยยังเป็น never อยู่(และจะมีปุ่ม Refresh ไว้สำหรับรีข้อมูลใหม่ และปุ่มถังขยะไว้สำหรับลบ Datasource นี้ทิ้งไป


NETPIE Freeboard
IMPORT
EXPORT
RESET
ADD PANE
DATASOURCES
Name
Last Updated
Datasource
never
ADD


MaxSmartHome
Home » Application » MaxSmartHome
APPLICATION
MaxSmartHome
DELETE
2
APPLICATION KEY
Device Key, Session Key
APPLICATION KEY
Datasource1 [online:1]
freeboard_ds1
SmartNode1 [online:1]
MaxPlug1

LAB20: เปิดปิดไฟผ่าน NETPIE Freeboard

ทำการสร้าง KEY ชื่อ TestLED ไว้ใน NETPIE และทำการนำมาใส่ค่าในส่วนที่ผมคอมเมนต์ไว้)7 บรรทัดนั้นให้ (ถูกต้องจากนั้นก็โหลดโค้ดลงบนบอร์ดได้เลย

LAB20: ON/OFF by NETPIE Freeboard

```
#include <AuthClient.h>
#include <MicroGear.h>
#include <MQTTClient.h>
#include <SHA1.h>
#include <Arduino.h>
#include <ESP8266WiFi.h>
#include <EEPROM.h>

const char* ssid     = "BUAKAEW_AP01";      //== 1 == ใส่ชื่อเครือข่าย WiFi ที่จะให้ NodeMCU V2 เชื่อมต่อ
const char* password = "BuaKaew159357";          //== 2 == ใส่รหัสผ่านของเครือข่าย WiFi ที่จะให้ NodeMCU V2 เชื่อมต่อ
#define LEDPin D4                                //== 3 == หากต่อหลอด LED ที่ขาอื่นที่ไม่ใช่ D2 ให้ทำการเปลี่ยนชื่อขาตรงนี้ด้วย
#define APPID       "MaxSmartHome"               //== 4 == ใส่ Application ID (ชื่อ Application) ที่สร้างบน NETPIE
#define GEARKEY     "l9kQv3Qhj8khNga"            //== 5 == ใส่ Key ของอุปกรณ์ ที่ได้จากการสร้างบน NETPIE
#define GEARSECRET  "Pv9nPFA87hnHulKqgkq2fvB16"  //== 6 == ใส่ Secret ของอุปกรณ์ ที่ได้จากการสร้างบน NETPIE
#define ALIAS       "testled"                    //== 7 == ตั้งชื่อ Alias ให้กับ Things นี้  เป็นชื่อเรียกเวลา) chat มาหา(
#define SCOPE       ""
WiFiClient client;
AuthClient *authclient;
MicroGear microgear(client);
```

DO NOT COPY

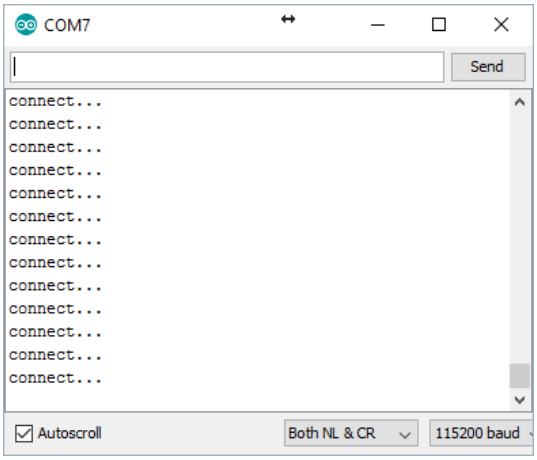
```
void onConnected(char *attribute, uint8_t* msg, unsigned int msglen) {
  Serial.println("Connected to NETPIE...");
  microgear.setAlias(ALIAS);
}
void onMsghandler(char *topic, uint8_t* msg, unsigned int msglen) {
  Serial.print("Incoming message --> ");
  Serial.print(topic);
  Serial.print(" : ");
  char strState[msglen];
  for (int i = 0; i < msglen; i++)   {
    strState[i] = (char)msg[i];
    Serial.print((char)msg[i]);
  }
  Serial.println();
  String stateStr = String(strState).substring(0, msglen);
//=========== ช่วงประมวลผลคำสั่ง ==========
  if (stateStr == "ON")   {
    digitalWrite(LEDPin, HIGH);
    microgear.chat("testledstatus", "ON");       //==== คำสั่ง chat เพื่อบอกส่งค่าสถานะไปยังหลอด
LED บน NETPIE Freeboard
  }
  else if (stateStr == "OFF")   {
    digitalWrite(LEDPin, LOW);
    microgear.chat("testledstatus", "OFF");       //==== คำสั่ง chat เพื่อบอกส่งค่าสถานะไปยังหลอด
LED บน NETPIE Freeboard
  }
//=========== ช่วงประมวลผลคำสั่ง  ==========
}
void setup()
{
  Serial.begin(115200);
```

DO NOT COPY

```
  Serial.println("Starting...");
  pinMode(LEDPin, OUTPUT);
  microgear.on(MESSAGE,onMsghandler);
  microgear.on(CONNECTED,onConnected);
  if (WiFi.begin(ssid, password))  {
    while (WiFi.status() != WL_CONNECTED)    {
      delay(500);
      Serial.print(".");
    }
    Serial.println("WiFi connected");
    Serial.println("IP address: ");
    Serial.println(WiFi.localIP());
    //uncomment the line below if you want to reset token -->
    microgear.resetToken();
    microgear.init(GEARKEY, GEARSECRET, SCOPE);
    microgear.connect(APPID);
  }
}
void loop() {
  if (microgear.connected())  {
    microgear.loop();
    Serial.println("connect...");
  }
  else  {
    Serial.println("connection lost, reconnect...");
    microgear.connect(APPID);
  }
  delay(1000);
}
```

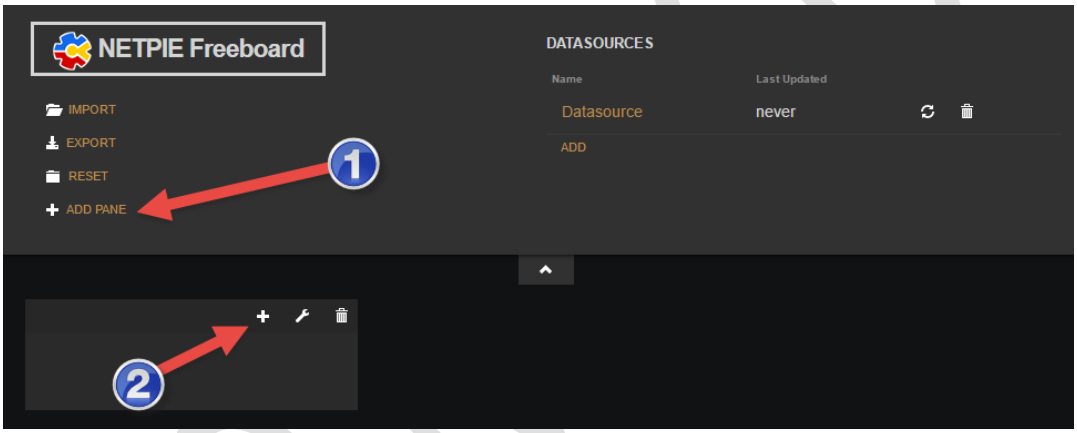
เมื่ออัพโหลดลงแล้ว ก็ให้ลองเปิด Serial Monitor (เลือก Baud Rate เป็น 115200) ดูว่าบอร์ดของเรานั้นเชื่อมต่อเรียบร้อยหรือไม่ ตามรูปด้านล่างนี้

DO NOT COPY

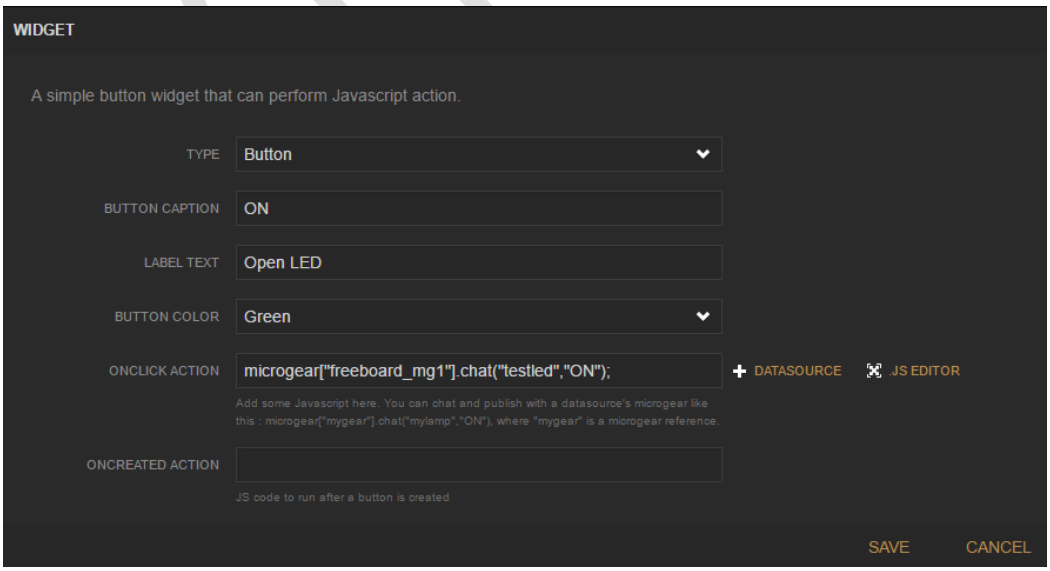


ถ้าหากไม่มีปัญหาอะไรแล้วก็เป็นอันว่าเรียบร้อย เพียงเท่านี้เราก็จะได้ Thing ที่มีหลอด LED หนึ่งหลอด และพร้อมที่จะทดลองการเปิดปิดผ่าน NETPIE Freeboard แล้ว

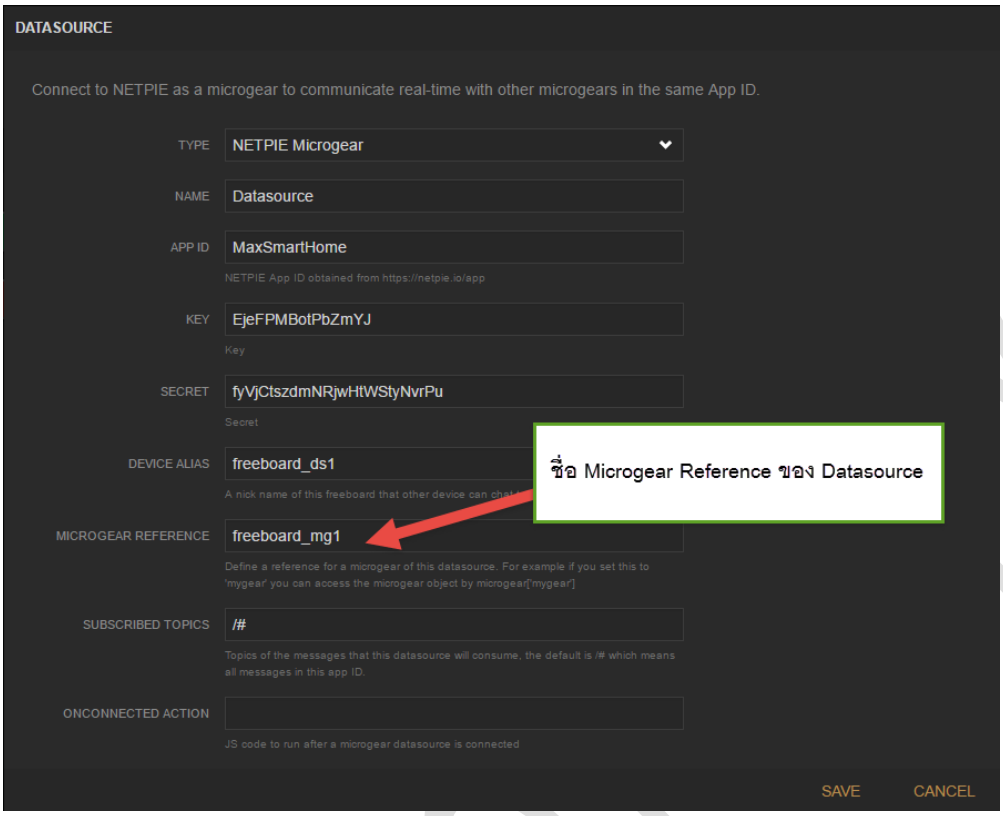
ในส่วนของ Freeboard ให้ทำการเปิด Freeboard ที่เราสร้าง Datasource ไว้ใน Part แรกขึ้นมาจากนั้นก็ให้กดปุ่ม ADD PANE ตามภาพนี้



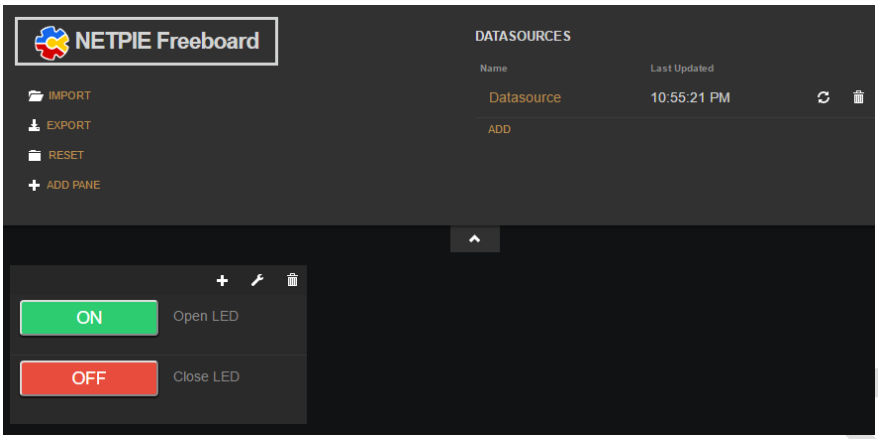
สร้าง Widget เป็นปุ่มเปิดปิด



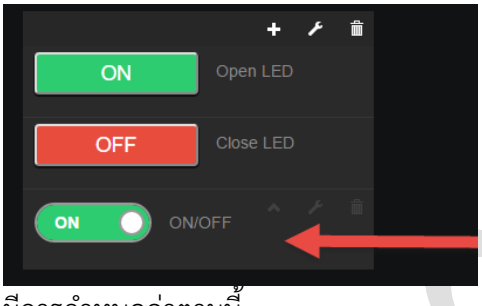
โดยค่าต่าง ๆ ได้จาก



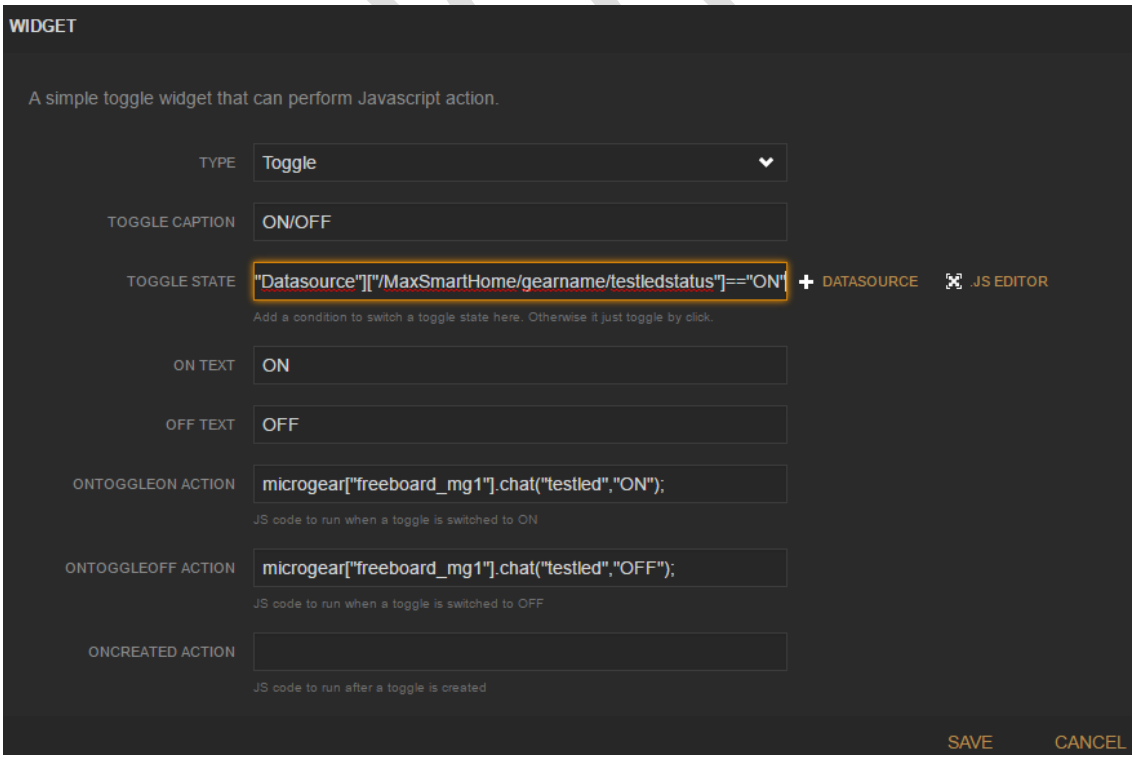
เราสามารถควบคุมการเปิดปิดไฟด้วยปุ่มบน Freeboard ได้



Widget ที่น่าสนใจอีกตัวคือ Toogle



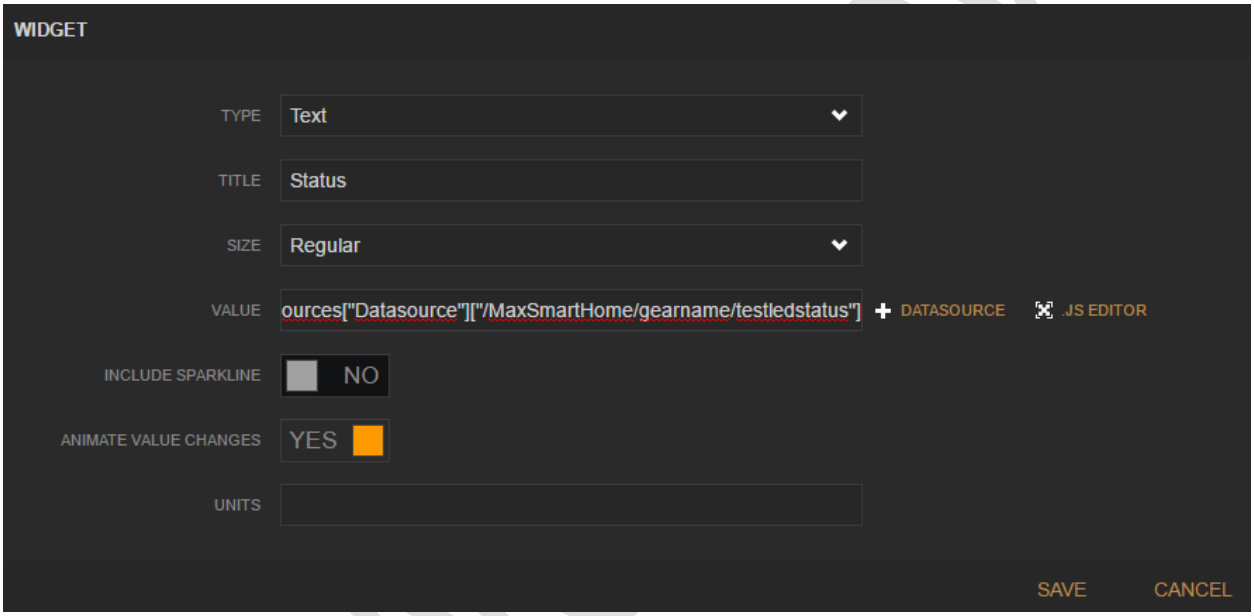
มีการกำหนดค่าตามนี้



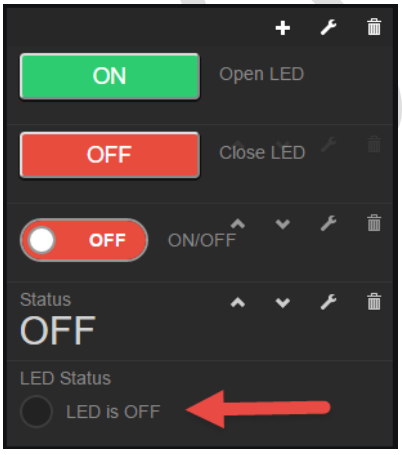
NOT COPY

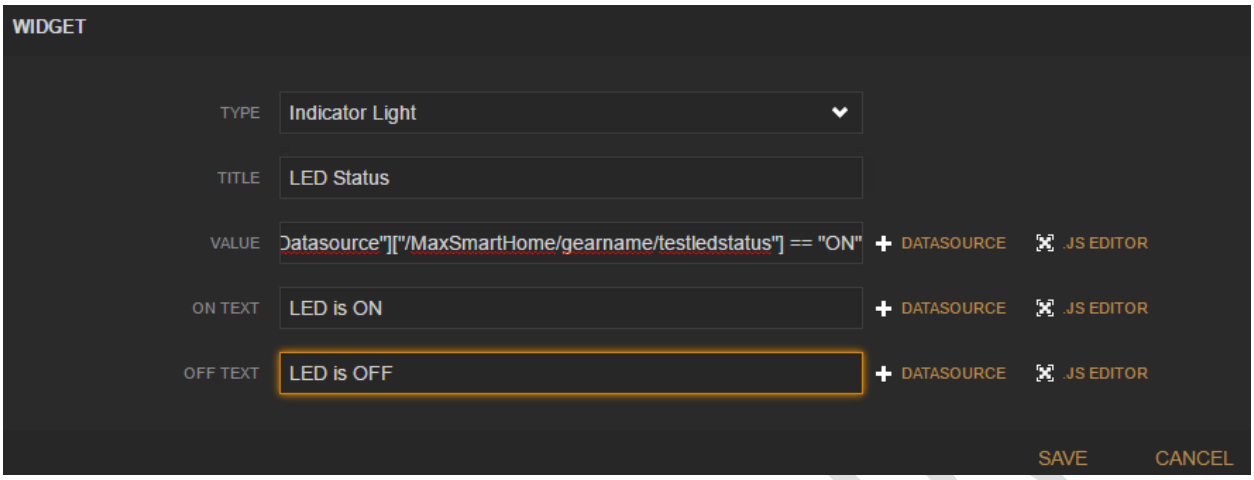
Widget ที่น่าสนใจอีกตัวคือ Text

มีการกำหนดค่าตามนี้



Widget ที่น่าสนใจอีกตัวคือ Indicator Light

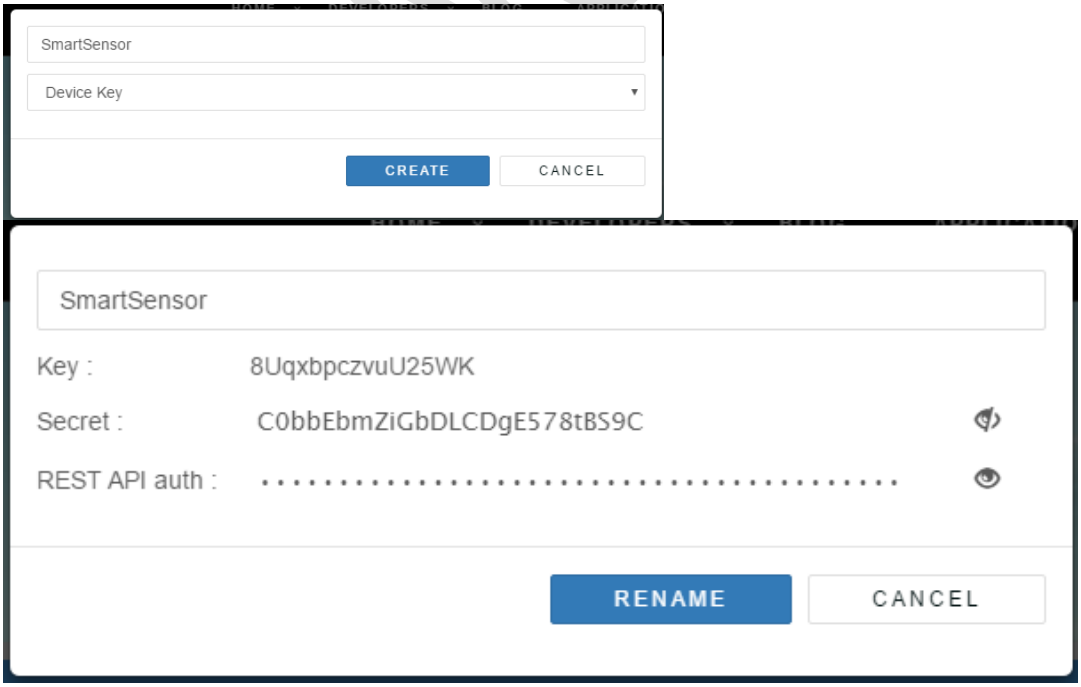




LAB21: วัดอุณหภูมิ ความชื้น ส่งค่ามาแสดงบนฟรีบอร์ด

หลังจาก Part ที่แล้วเราได้ลองเปิดปิดหลอดไฟ)LED) ผ่าน NETPIE กันแล้ว ซึ่งนั่นเป็นรูปแบบของการส่งข้อมูลสถานะ ระหว่าง "ปิด" หรือ "เปิด"Freeboard ของเรากับอุปกรณ์)Things) นั้น คาดว่าหลังจากได้ทดลองลงมือทำดูจริงๆแล้วคงทำให้เข้าใจระบบการทำงาน ของ NETPIE Freeboardกันมากขึ้นแล้ว และใน Part นี้จะเป็นการทดลอง ส่งค่า "ความชื้น" และ "อุณหภูมิ"มาแสดงบน Freeboard ขอเรียกการทดลองนี้ว่า "SmartSensor"

ก่อนอื่นก็ให้สร้าง Key สำหรับอุปกรณ์นี้บน NETPIE ก่อนเช่นของผมสร้างไว้ภายใน AppID ชื่อ MaxSmartHome โดยสร้างเป็น Device Key ชื่อว่า SmartSensor



LAB21: NETPIE Freeboard Read From Sensor DHT11

```
#include "DHT.h"
#include <AuthClient.h>
#include <MicroGear.h>
#include <MQTTClient.h>
#include <SHA1.h>
#include <Arduino.h>
#include <ESP8266WiFi.h>
#include <ESP8266WebServer.h>
#include <ESP8266mDNS.h>
#include <EEPROM.h>

const char* ssid     = "BUAKAEW_AP01";   //ชื่อ SSID ของ WiFi ที่จะเชื่อมต่อ
const char* password = "BuaKaew159357";          //รหัสผ่านของ WiFi ที่จะเชื่อมต่อ

#define APPID   "MaxSmartHome"                  //ใส่ APPID ที่สร้างไว้ใน NETPIE
#define KEY     "8UqxbpczvuU25WK"               //ใส่ KEY ที่สร้างไว้สำหรับ SmartSensor
#define SECRET  "C0bbEbmZiGbDLCDgE578tBS9C"    //ใส่ SECRET ของ KEY ที่สร้างไว้
#define ALIAS   "SmartSensor"                   //ตั้งชื่อให้กับอุปกรณ์นี้  (เช่นตอนนี้ผมตั้งชื่อว่า)
"SmartSensor"
#define DHTPIN  D5          //ระบุขาที่ต่อกับเซนเซอร์ DHT
#define DHTTYPE   DHT11     //ระบุว่าใช้ DHT11 หรือ DHT22 (ในที่นี้ผมใช้เป็น DHT22)
#define RefreshTime   100    //ค่าความถี่ของการส่งข้อมูล เช่นในที่นี้คือ ส่งข้อมูลออกไปทุกๆ 0.1 วินาที
)100 ms)
WiFiClient client;
AuthClient *authclient;

DHT dht(DHTPIN, DHTTYPE);
int timer = 0;
MicroGear microgear(client);
```

DO NOT COPY

```
void onConnected(char *attribute, uint8_t* msg, unsigned int msglen)
{
  Serial.println("Connected to NETPIE...");     //หากการเชื่อมต่อสำเร็จ ให้แสดงข้อความนี้ทาง Serial
Monitor
}

void setup()
{
  Serial.begin(115200);
  Serial.println("Starting...");
  dht.begin();
  microgear.on(CONNECTED,onConnected);
  microgear.init(KEY,SECRET,ALIAS);
  microgear.connect(APPID);

  if (WiFi.begin(ssid, password))
  {
    while (WiFi.status() != WL_CONNECTED)
    {
      delay(500);
      Serial.print(".");
    }
    Serial.println("WiFi connected");
    Serial.println("IP address: ");
    Serial.println(WiFi.localIP());
  }
}

void loop()
{
    if (microgear.connected())
```

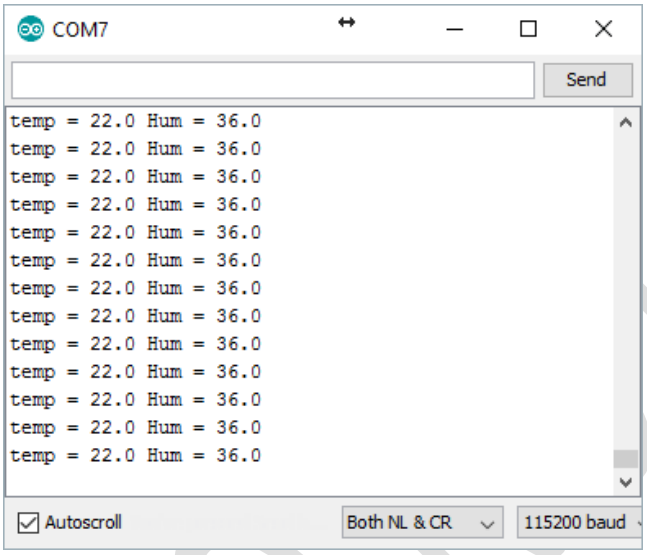
DO NOT COPY

```
  {
    microgear.loop();
    timer=0;
    float tempread = dht.readTemperature();   //เก็บค่าอุณหภูมิที่อ่านได้ไว้เป็นแบบ float ไว้ในตัวแปร "tempread"
    float humidread = dht.readHumidity();     //เก็บค่าความชื้นที่อ่านได้ไว้เป็นแบบ float ไว้ในตัวแปร "humidread"
    char temp[10];     //
    char humid[10];    // สร้างตัวแปรไว้ใช้ส่งข้อมูลทาง microgear.chat (ต้องใช้รูปแบบ char)
    if (isnan(tempread) || isnan(humidread) || tempread > 100 || humidread > 100)     {
        tempread = 0.0;       // ถ้าอ่านค่าไม่ได้ หรืออ่านค่าได้มากกว่า 100
        humidread = 0.0;      // ให้แสดงค่าเป็น 0 แทน
    }

    int tempread_decimal = (tempread - (int)tempread) * 100;      // เก็บส่วนของตัวเลขที่อยู่หลังจุดทศนิยม ไปเป็นแบบ int
    int humidread_decimal = (humidread - (int)humidread) * 100;   // เช่น 35.87 จะเก็บ 87 ไว้ในตัวแปร

    sprintf(temp,"%d.%d", (int)tempread,tempread_decimal);      // ยัดค่าตัวแปร int เข้าไปในตัวแปร char โดยมีรูปแบบ
    sprintf(humid,"%d.%d", (int)humidread,humidread_decimal);   // %d.%d = ส่วนจำนวนเต็ม. จะ) ส่วนทศนิยมchat ตัวแปรนี้ไป(
    Serial.print("temp = ");
    Serial.print(temp);
    Serial.print(" Hum = ");
    Serial.println(humid);
    microgear.chat("SmartSensor/Temperature",temp);     // ทำการ chat ตัวแปรต่างๆ ออกไป
    microgear.chat("SmartSensor/Humidity",humid);       // โดยใช้ชื่อผู้รับเป็น "SmartSensor/Temp,Humid,LightLevel"
  }
```

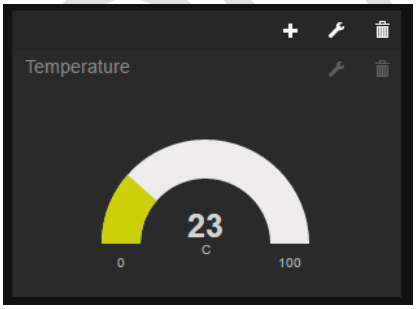
DO NOT COPY

```
    else
    {
      Serial.println("connection lost, reconnect...");
      microgear.connect(APPID);
      delay(timer);
      timer+=100;
    }
    delay(RefreshTime);    // รอเวลาเป็นจำนวนเท่ากับที่ตั้งไว้ในตัวแปร RefreshTime
}
```

แสดงผลดังรูป Serial



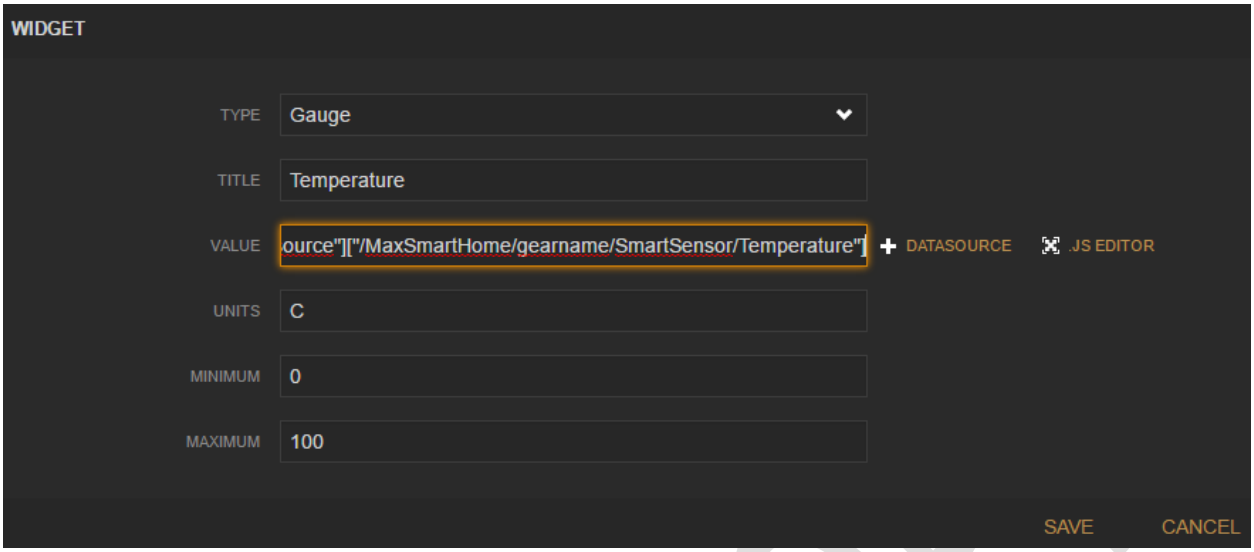
ในส่วนของ Freeboard เร้า Add Widget Gauge



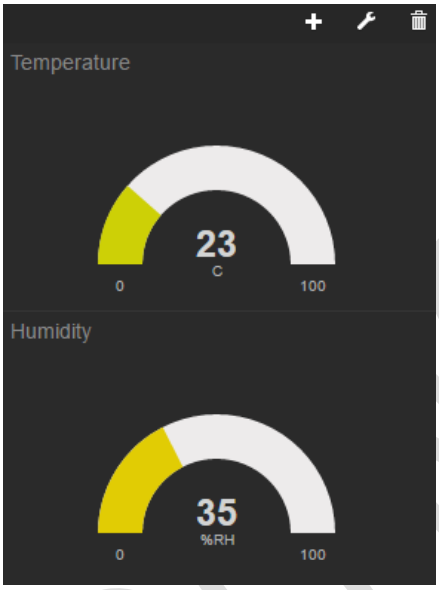
ในส่วนของ Datasource กำหนดเป็น

datasources["Datasource"]["/MaxSmartHome/gearname/SmartSensor/Temperature"]

DO NOT COPY



และ Humidity เช่นกัน



LAB: เขียนโปรแกรมควบคุมผ่านมือถือด้วย APP Inventor

LAB: Facebook BOT API chat NETPIE IoT

DO NOT COPY

Reference

บทความ Arduitronics (http://www.arduitronics.com/article)

บทความ ThaiEasyElec (http://www.thaieasyelec.com/article-wiki-th/electronic-article.html)

NETPIE (https://netpie.io/)

DO NOT COPY