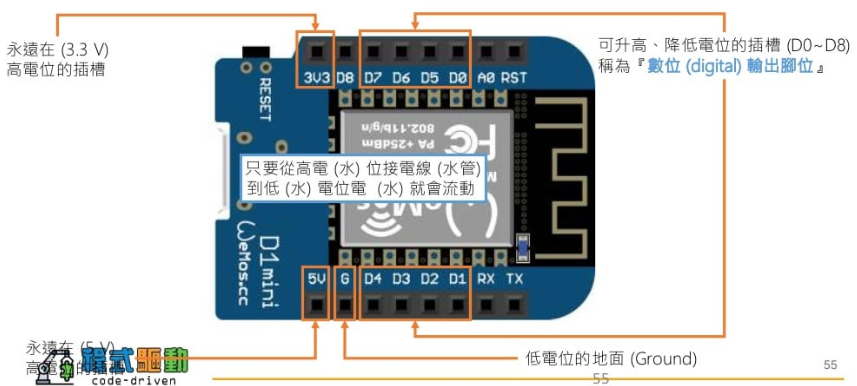
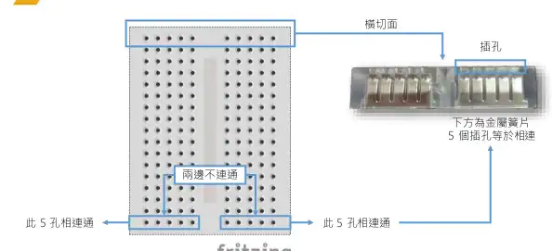
* AIoT物聯網：透過將程式寫入具wifi功能的晶片，讓遠端接受器接收訊號或產生行為，可應用在長照或是智慧家電產品上。
* 製作RJ45網路線：

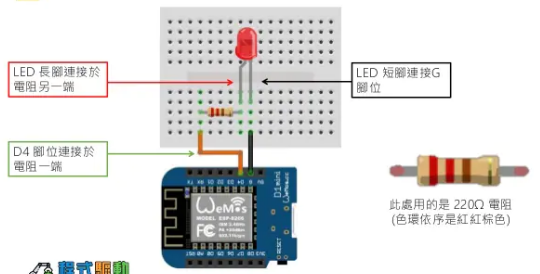
去掉兩端的皮 → 先套上保護套 → 將內部的線依照顏色由右至左排列（白橙 → 橙 → 白綠 → 藍 → 白藍 → 綠 → 白咖 → 咖）塞入集線端（透明）→壓線



* Arduino D1 mini – (ESP-8266)
* 4MB
* 自帶wifi訊號且可被寫入程式的小晶片
* 需先銲接在麵包板（電路板）上 → 接上電線、電阻、LED燈泡(長腳＋ / 短腳－)形成電路 → 插上USB接上電腦 → 利用積木程式寫入code

(D1 mini)

(麵包板)

(電路接法)

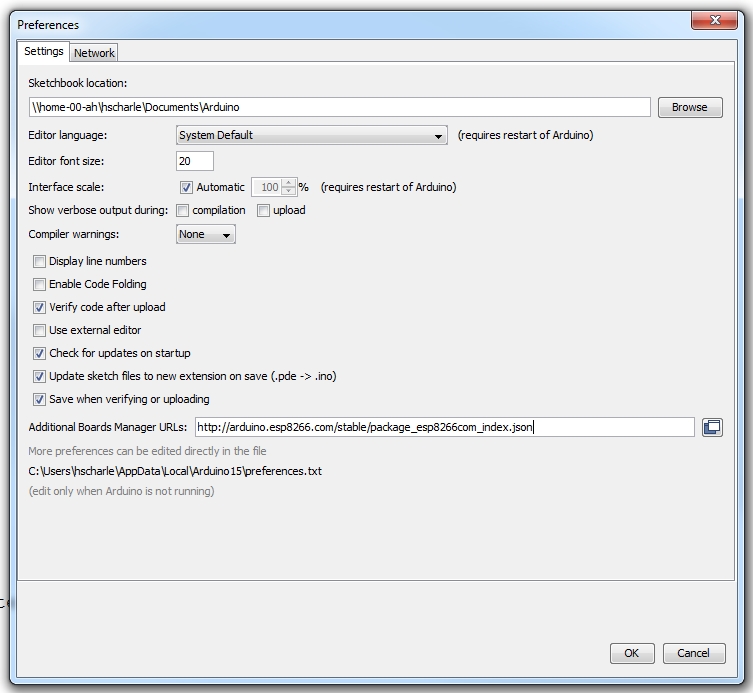
(程式寫法)

* 積木程式([FlagsBlock](https://www.flag.com.tw/download/FlagsBlock.exe))

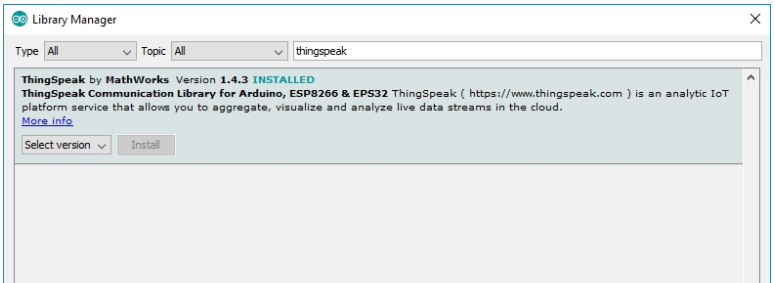
1. 開啟”TEN” → 確認已插上USB → ”安裝驅動程式” － ”D1 mini” → ”install”
2. 去我的電腦右鍵 ”內容”－ ”裝置管理員” → 查看 ”連接阜” 的USB是COM? → 去 ”設定” 調整連接阜和開發板
3. 打好code → 燒錄(此時D1 mini上的藍色LED燈會快速閃爍) → 燈泡開始動作(此時藍色LED燈閃爍節奏會與紅色LED燈互補)

* D1 mini wifi 訊號偵測並上傳至ThingSpeak監控

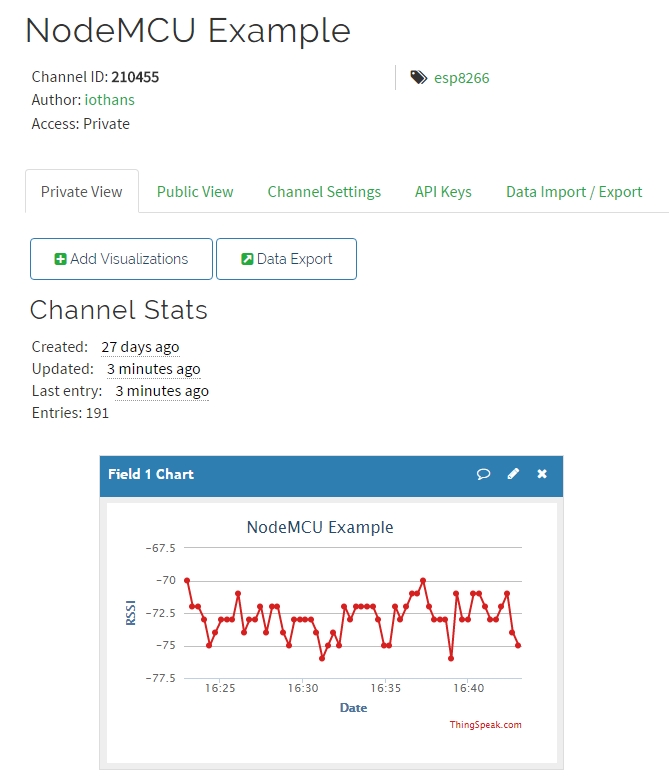
1. 在積木程式上找到偏好設定，貼上URL： http://arduino.esp8266.com/stable/package\_esp8266com\_index.json



1. 打開 ”Arduino IDE草稿碼” → 匯入程式庫 → 管理程式庫 → 搜尋 “ThingSpeak” 並安裝 (1.4.3 version)



1. 匯入[example code](https://drive.google.com/open?id=1UQp5ehJn8VMdbz8bPEumly5EotGqoEBP)（2個）至Arduino IDE
2. 改掉 “secrets.h” 的 ThinkSpeak Channel ID & WIFI帳密
3. 確認好COM後燒錄 ” RSSI\_to\_ThingSpeak”
4. 可以開啟 ”程式碼監控欄” 看有沒有 ”CONNECTED SUCCESS”
5. 即可在 ” ThinkSpeak” 看到數據



* 遠端點燈(wifi)

1. 下載ESPAsync相關套件
   * [ESPAsyncWebServer](https://github.com/me-no-dev/ESPAsyncWebServer/archive/master.zip)
   * [ESPAsyncTCP](https://github.com/me-no-dev/ESPAsyncTCP/archive/master.zip)
2. 打開**”Arduino IDE” > “草稿碼” > “ 匯入程式庫” > “ 加入.zip程式庫”**
3. 將以下程式碼複製到Arduino IDE

// 載入必須程式庫

#include <ESP8266WiFi.h>

#include <ESPAsyncTCP.h>

#include <ESPAsyncWebServer.h>

// 替換成您的網絡憑據

const char\* ssid = "REPLACE\_WITH\_YOUR\_SSID";

const char\* password = "REPLACE\_WITH\_YOUR\_PASSWORD";

bool ledState = 0;

const int ledPin = 2;

// 在端口80上建立asyncwebserver物件

AsyncWebServer server(80);

AsyncWebSocket ws("/ws");

const char index\_html[] PROGMEM = R"rawliteral(

<!DOCTYPE HTML><html>

<head>

<title>ESP Web Server</title>

<meta name="viewport" content="width=device-width, initial-scale=1">

<link rel="icon" href="data:,">

<style>

html {

font-family: Arial, Helvetica, sans-serif;

text-align: center;

}

h1 {

font-size: 1.8rem;

color: white;

}

h2{

font-size: 1.5rem;

font-weight: bold;

color: #143642;

}

.topnav {

overflow: hidden;

background-color: #143642;

}

body {

margin: 0;

}

.content {

padding: 30px;

max-width: 600px;

margin: 0 auto;

}

.card {

background-color: #F8F7F9;;

box-shadow: 2px 2px 12px 1px rgba(140,140,140,.5);

padding-top:10px;

padding-bottom:20px;

}

.button {

padding: 15px 50px;

font-size: 24px;

text-align: center;

outline: none;

color: #fff;

background-color: #0f8b8d;

border: none;

border-radius: 5px;

-webkit-touch-callout: none;

-webkit-user-select: none;

-khtml-user-select: none;

-moz-user-select: none;

-ms-user-select: none;

user-select: none;

-webkit-tap-highlight-color: rgba(0,0,0,0);

}

/\*.button:hover {background-color: #0f8b8d}\*/

.button:active {

background-color: #0f8b8d;

box-shadow: 2 2px #CDCDCD;

transform: translateY(2px);

}

.state {

font-size: 1.5rem;

color:#8c8c8c;

font-weight: bold;

}

</style>

<title>ESP Web Server</title>

<meta name="viewport" content="width=device-width, initial-scale=1">

<link rel="icon" href="data:,">

</head>

<body>

<div class="topnav">

<h1>ESP WebSocket Server</h1>

</div>

<div class="content">

<div class="card">

<h2>Output - GPIO 2</h2>

<p class="state">state: <span id="state">%STATE%</span></p>

<p><button id="button" class="button">Toggle</button></p>

</div>

</div>

<script>

var gateway = `ws://${window.location.hostname}/ws`;

var websocket;

window.addEventListener('load', onLoad);

function initWebSocket() {

console.log('Trying to open a WebSocket connection...');

websocket = new WebSocket(gateway);

websocket.onopen = onOpen;

websocket.onclose = onClose;

websocket.onmessage = onMessage; // <-- add this line

}

function onOpen(event) {

console.log('Connection opened');

}

function onClose(event) {

console.log('Connection closed');

setTimeout(initWebSocket, 2000);

}

function onMessage(event) {

var state;

if (event.data == "1"){

state = "ON";

}

else{

state = "OFF";

}

document.getElementById('state').innerHTML = state;

}

function onLoad(event) {

initWebSocket();

initButton();

}

function initButton() {

document.getElementById('button').addEventListener('click', toggle);

}

function toggle(){

websocket.send('toggle');

}

</script>

</body>

</html>

)rawliteral";

void notifyClients() {

ws.textAll(String(ledState));

}

void handleWebSocketMessage(void \*arg, uint8\_t \*data, size\_t len) {

AwsFrameInfo \*info = (AwsFrameInfo\*)arg;

if (info->final && info->index == 0 && info->len == len && info->opcode == WS\_TEXT) {

data[len] = 0;

if (strcmp((char\*)data, "toggle") == 0) {

ledState = !ledState;

notifyClients();

}

}

}

void onEvent(AsyncWebSocket \*server, AsyncWebSocketClient \*client, AwsEventType type,

void \*arg, uint8\_t \*data, size\_t len) {

switch (type) {

case WS\_EVT\_CONNECT:

Serial.printf("WebSocket client #%u connected from %s\n", client->id(), client->remoteIP().toString().c\_str());

break;

case WS\_EVT\_DISCONNECT:

Serial.printf("WebSocket client #%u disconnected\n", client->id());

break;

case WS\_EVT\_DATA:

handleWebSocketMessage(arg, data, len);

break;

case WS\_EVT\_PONG:

case WS\_EVT\_ERROR:

break;

}

}

void initWebSocket() {

ws.onEvent(onEvent);

server.addHandler(&ws);

}

String processor(const String& var){

Serial.println(var);

if(var == "STATE"){

if (ledState){

return "ON";

}

else{

return "OFF";

}

}

}

void setup(){

// Serial port for debugging purposes

Serial.begin(115200);

pinMode(ledPin, OUTPUT);

digitalWrite(ledPin, LOW);

// Connect to Wi-Fi

WiFi.begin(ssid, password);

while (WiFi.status() != WL\_CONNECTED) {

delay(1000);

Serial.println("Connecting to WiFi..");

}

// Print ESP Local IP Address

Serial.println(WiFi.localIP());

initWebSocket();

// Route for root / web page

server.on("/", HTTP\_GET, [](AsyncWebServerRequest \*request){

request->send\_P(200, "text/html", index\_html, processor);

});

// Start server

server.begin();

}

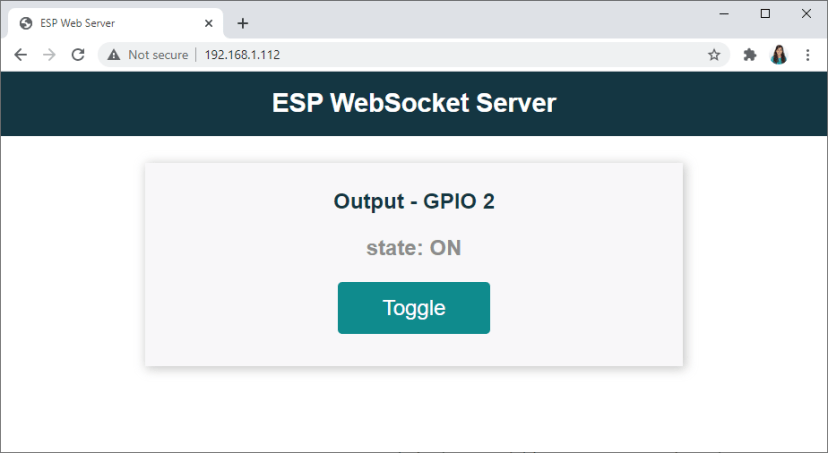
void loop() {

ws.cleanupClients();

digitalWrite(ledPin, ledState);

}

1. 確認好COM後燒錄
2. 開啟 ”程式碼監控欄” 看有沒有 ”CONNECTED SUCCESS”，會出現port
3. 把port用手機貼到網站上查詢，就可以遠端點按開關



* 更多應用

1. 搜尋＂ESP8266 project＂
2. 在積木程式點選 ”範例” → “物聯網感測器大應用”拿現成的程式
3. http://smf.ntc.im/index.php?board=2.0