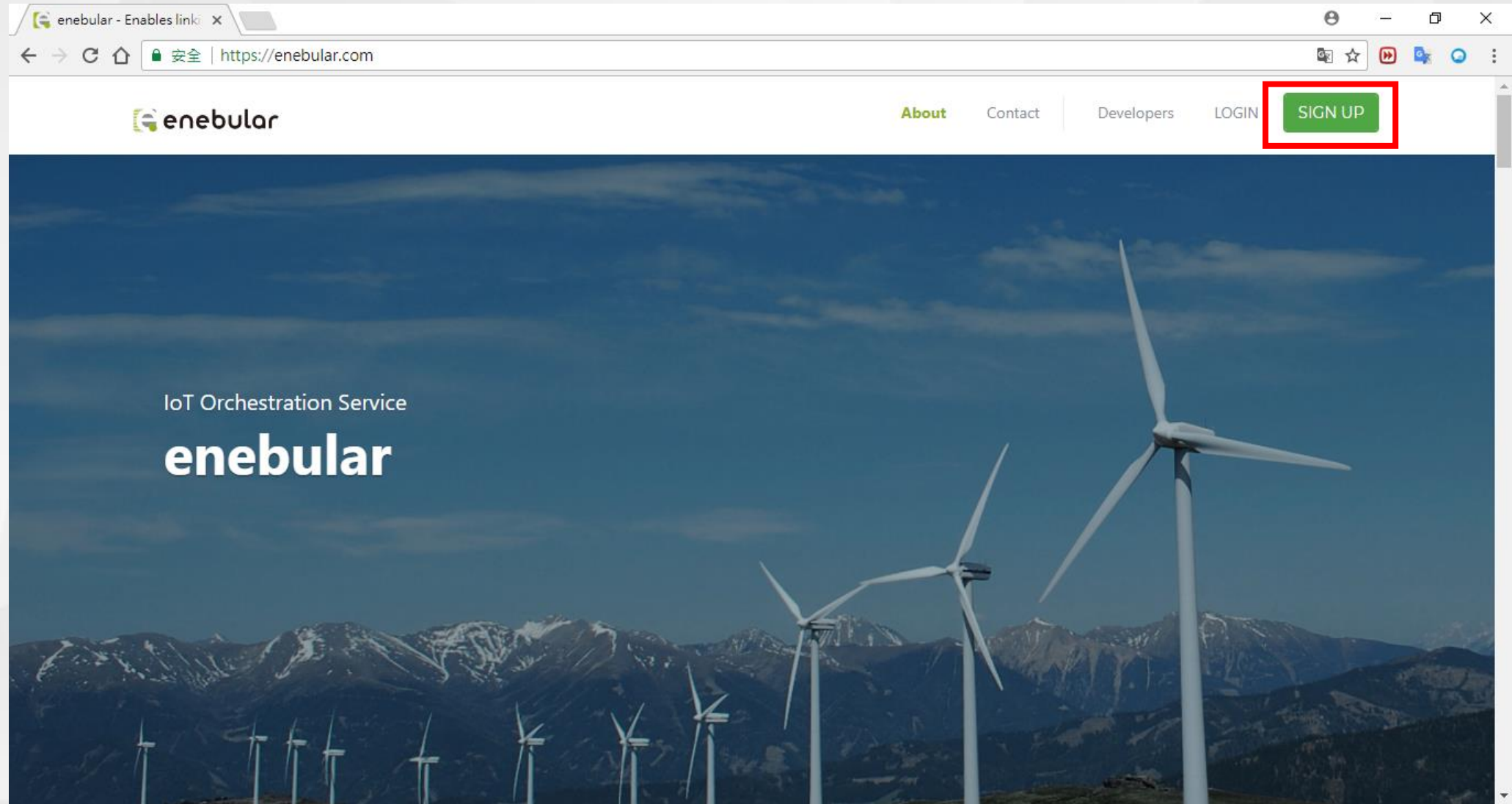


enebular 平台

`https://enebular.herokuapp.com/`



註冊帳號

enebular - Enables linkir X

安全 | https://enebular.herokuapp.com

enebular

About Contact Developers LOGIN SIGN UP

SIGN UP

Sign up with Github

Sign up with Salesforce

Sign up with Heroku

Or

E-mail

Password

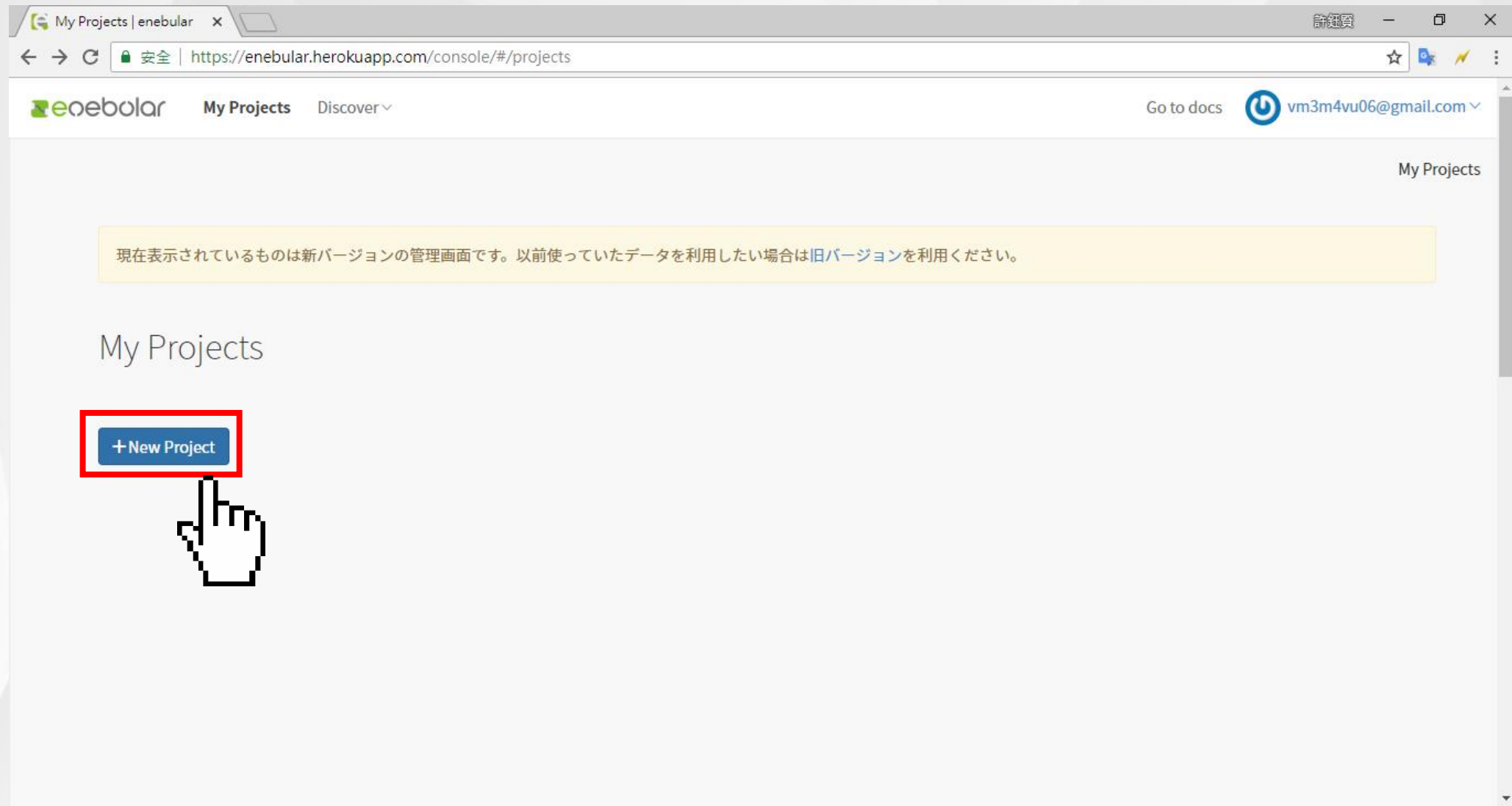
☐ I agree to the [privacy policy](#)

Sign Up

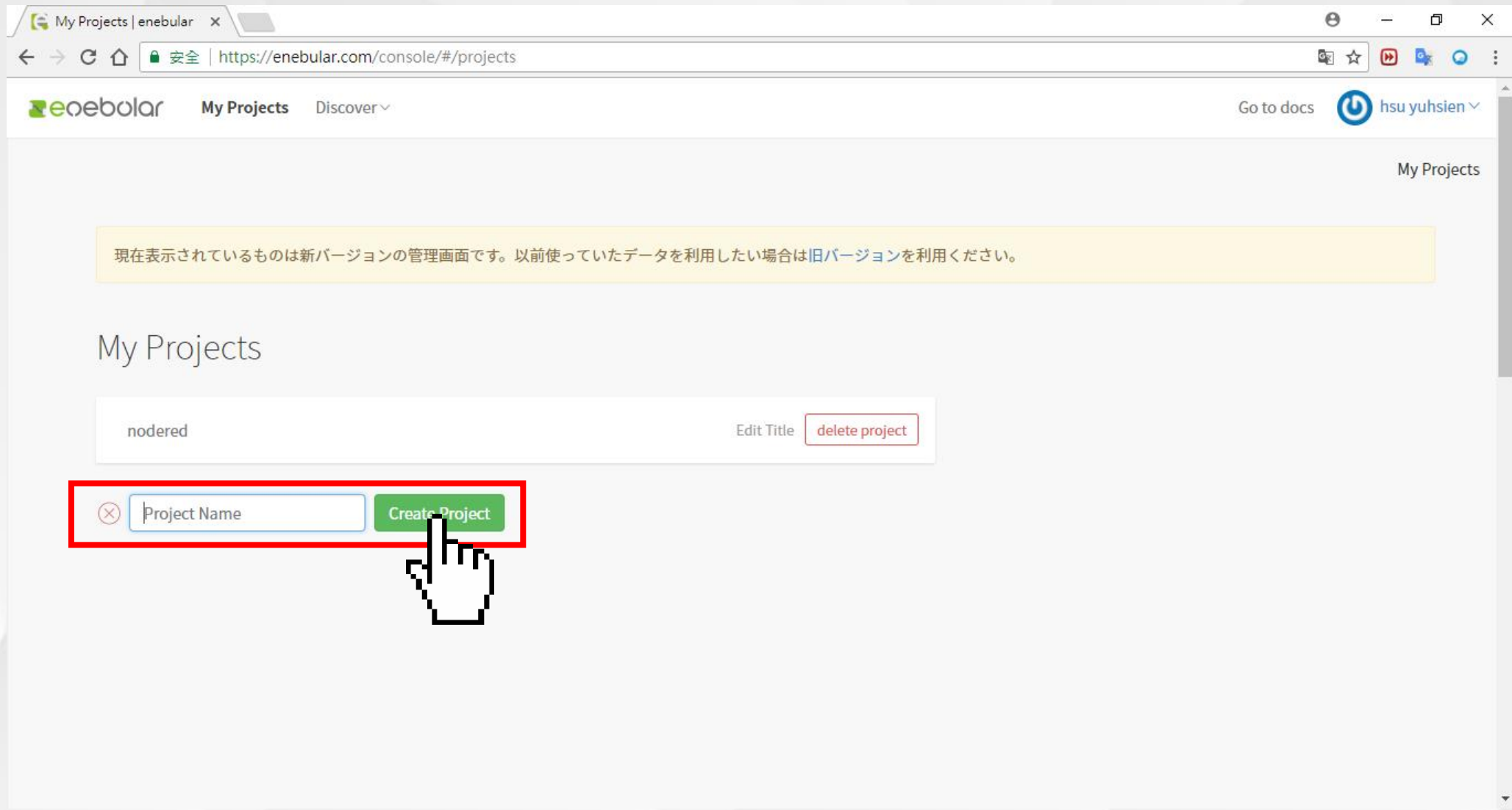
IoT Orchestration Service

enebular

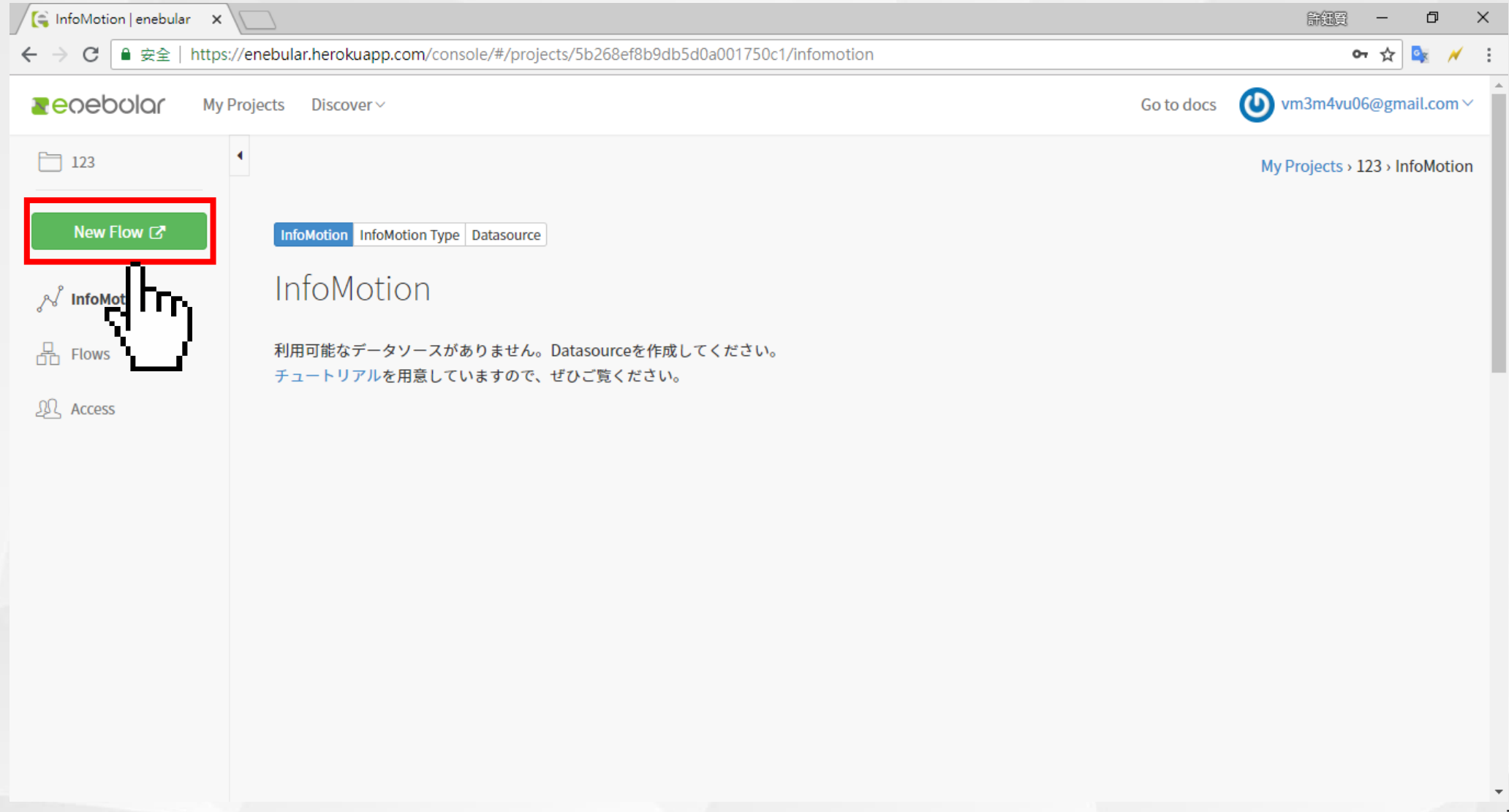
新增專案New Project



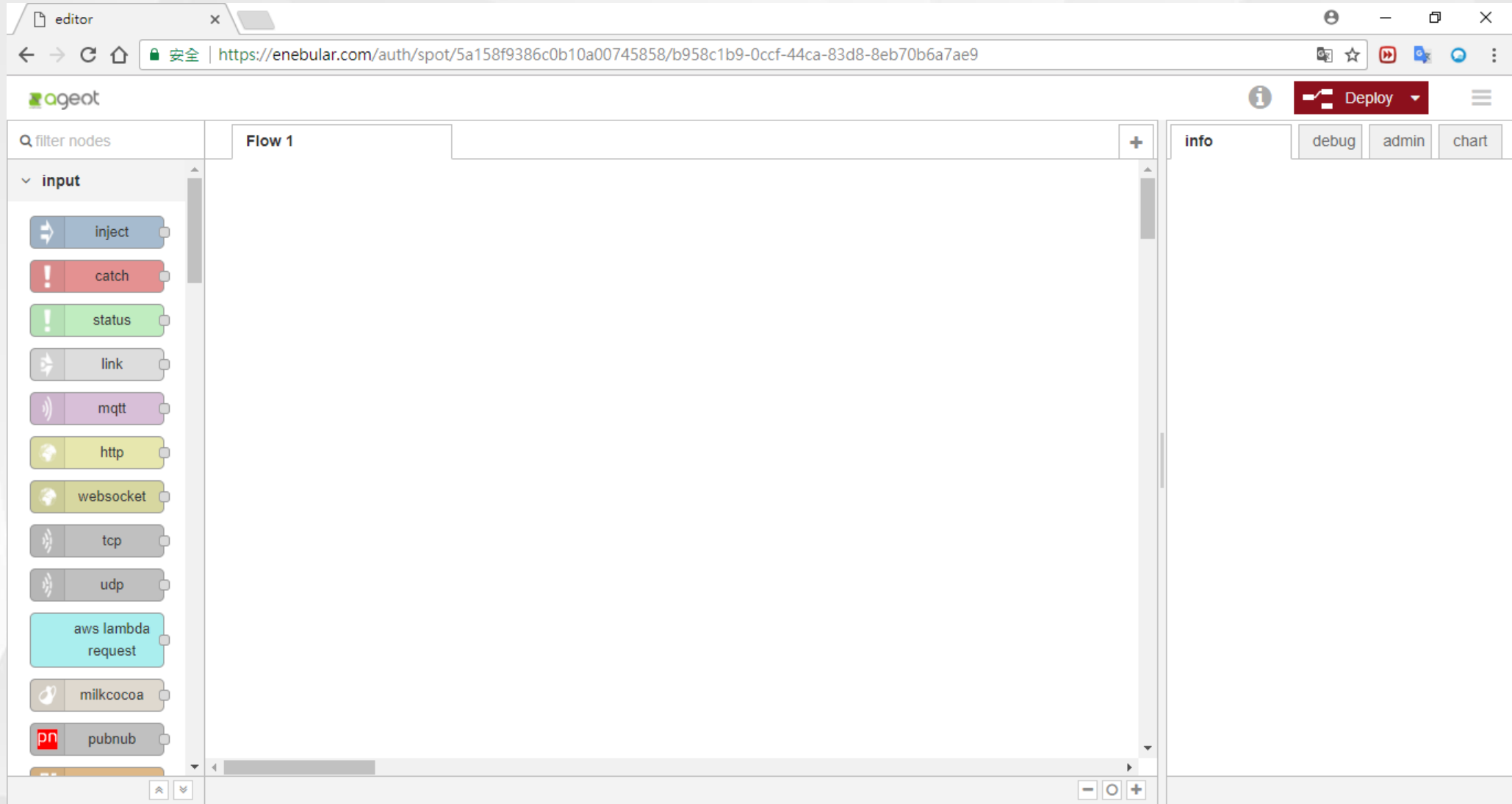
新增專案Create Project



新增編輯Flow



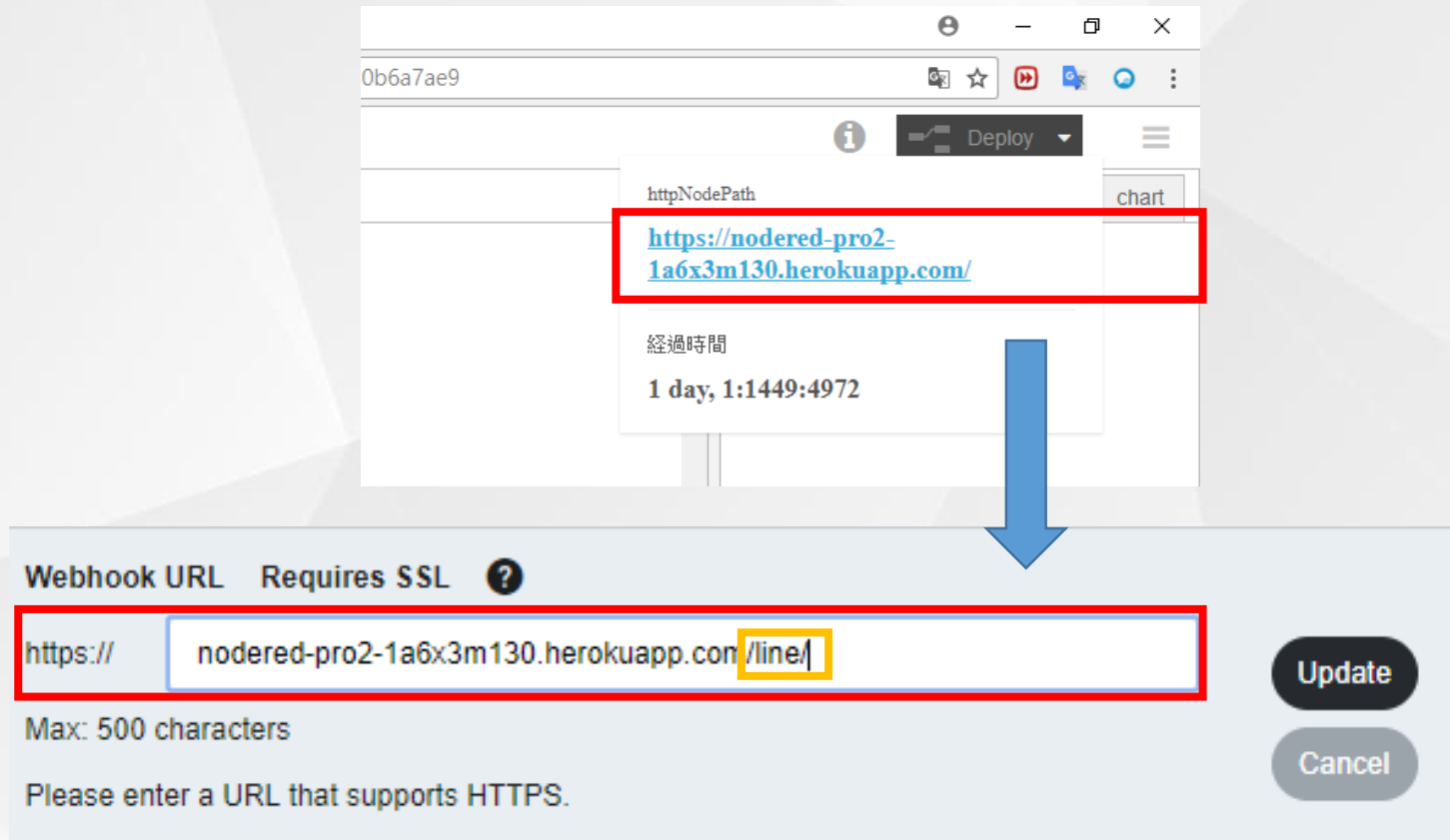
編輯介面



REPLY MESSAGE

REPLY MESSAGE

更改 Webhook URL: enebular -> Line



0b6a7ae9

Deploy

httpNodePath

<https://nodered-pro2-1a6x3m130.herokuapp.com/>

chart

經過時間

1 day, 1:1449:4972

Webhook URL Requires SSL ?

<https://nodered-pro2-1a6x3m130.herokuapp.com//line/>

Max: 500 characters

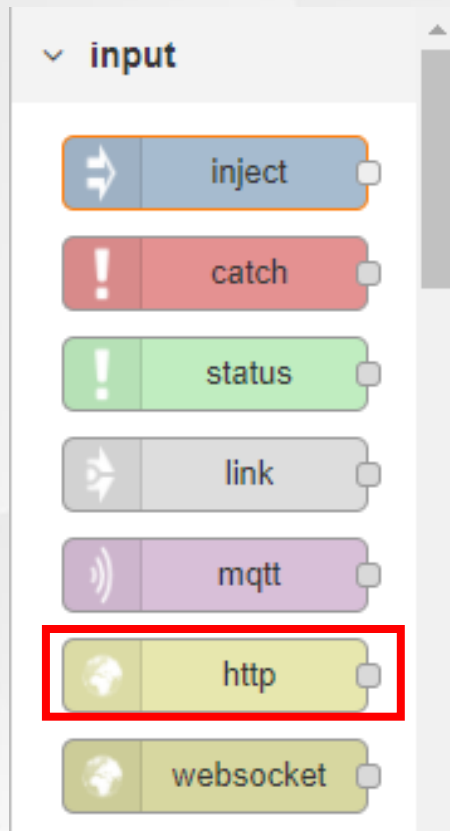
Please enter a URL that supports HTTPS.

Update

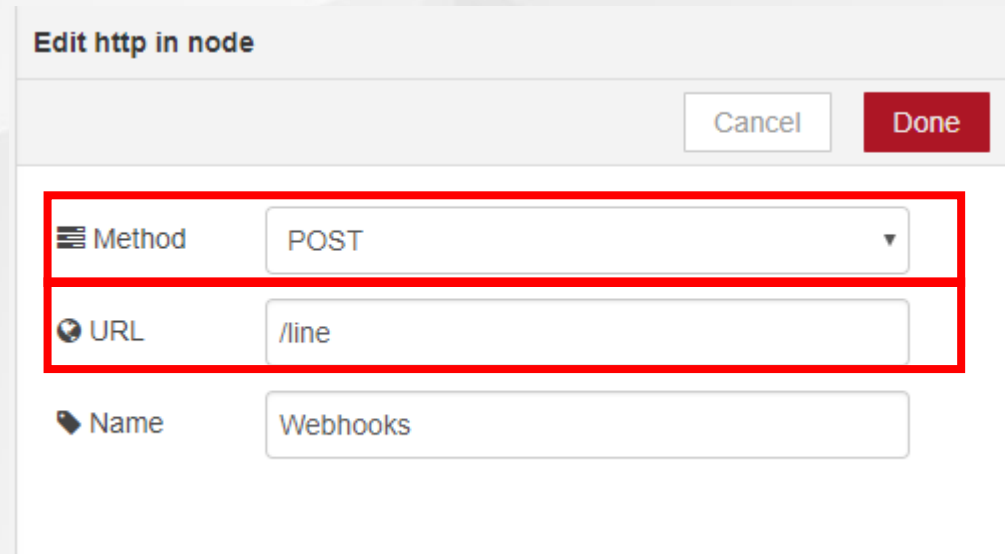
Cancel

REPLY MESSAGE

Input->選http

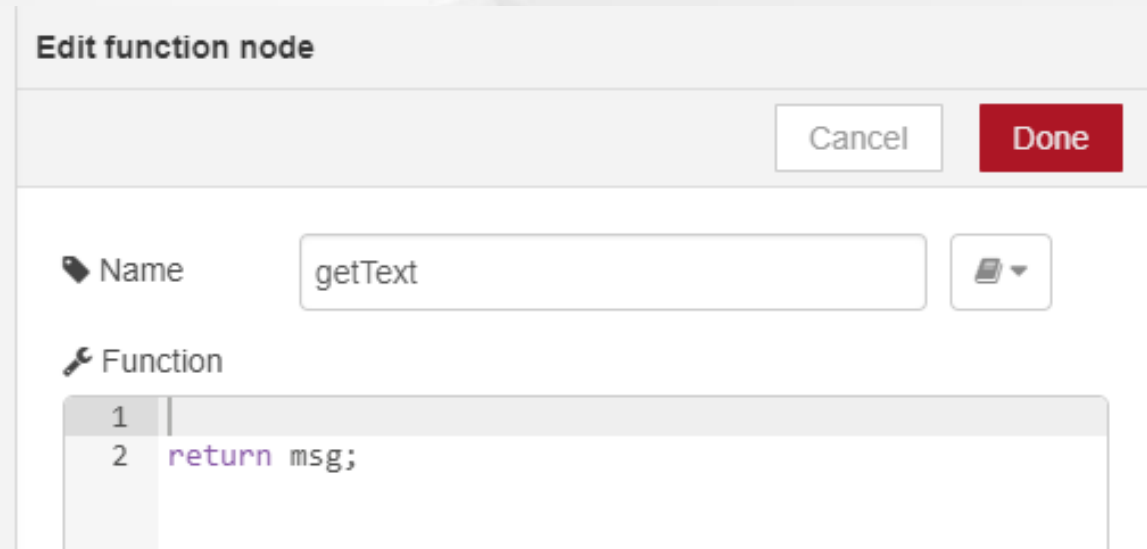


Method->選POST
URL->輸入/line



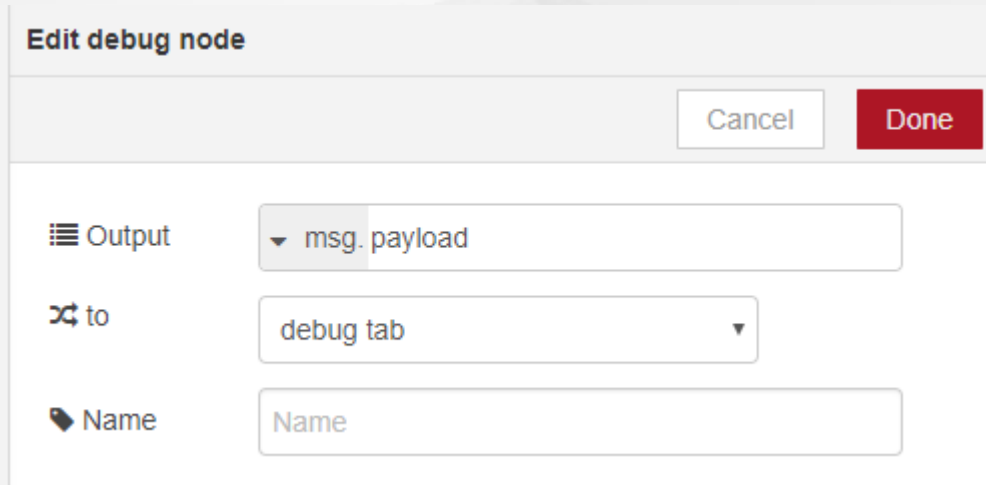
REPLY MESSAGE

function->選function



REPLY MESSAGE

output->選debug



REPLY MESSAGE



info

debug

admin

chart

all flows

current flow

2017/11/22 下午11:50:39 60542f5a.e74fd

msg.payload : Object

{ "events": [{ "type": "message",
"replyToken":
"4ab9fe8f5cfd4a869b23a6ed80658ed2",
"source": { "userId":
"U04cdfd354c708da33c03e3fbcc61f73f",
"type": "user" }, "timestamp":
1511365830541, "message": { "type": "text",
"id": "7030257215853", "text": "你好" } }] }

REPLY MESSAGE

function->選function



Edit function node

Cancel Done

Name

Function

```
1  
2 return msg;
```

REPLY MESSAGE



```
var post_request = {
  "headers": {
    "content-type": "application/json; charset=UTF-8 ",
    "Authorization": " Bearer " + "{Channel Access Token}"
  },
  "payload": {
    "replyToken": msg.payload.events[0].replyToken,
    "messages": [
      {
        "type": "text",
        "text": msg.payload.events[0].message.text
      }
    ]
  }
};
return post_request;
```

REPLY MESSAGE

function->選http request

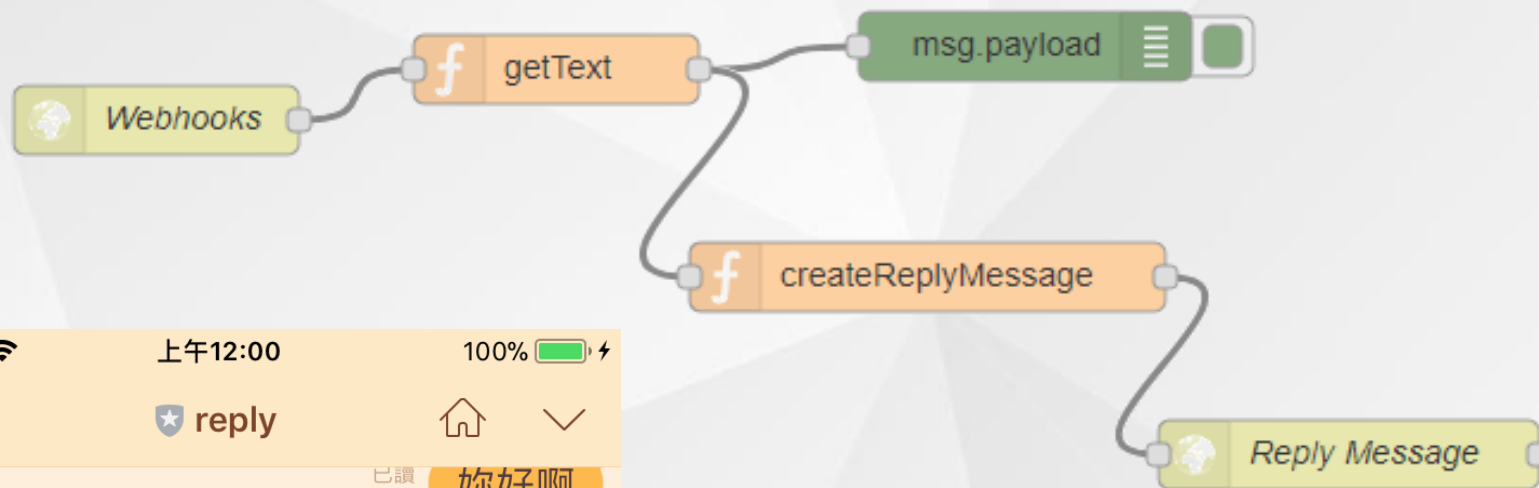
URL->輸入

`https://api.line.me/v2/bot/message/reply`



A screenshot of the 'Edit http request node' dialog box. The 'Method' is set to 'POST'. The 'URL' field contains 'https://api.line.me/v2/bot/message/reply' and is highlighted with a red box. Below the URL, there are two unchecked checkboxes: 'Enable secure (SSL/TLS) connection' and 'Use basic authentication'. The 'Return' dropdown is set to 'a parsed JSON object' and is also highlighted with a red box. The 'Name' field is set to 'Reply Message'. At the bottom, there is a tip: 'Tip: If the JSON parse fails the fetched string is returned as-is.'

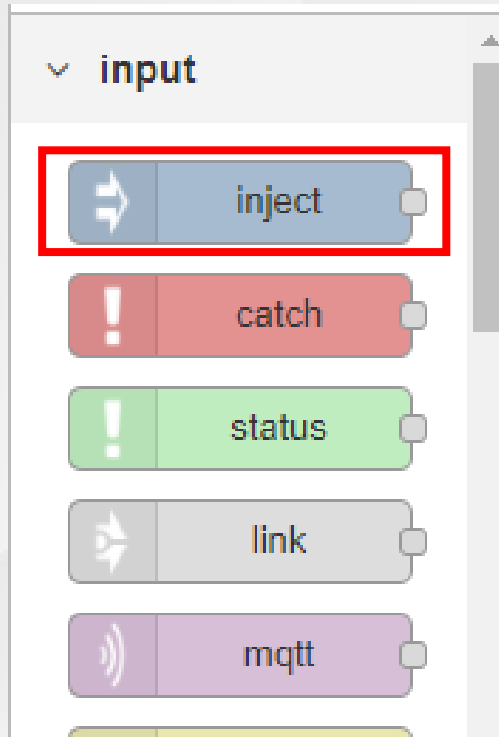
REPLY MESSAGE



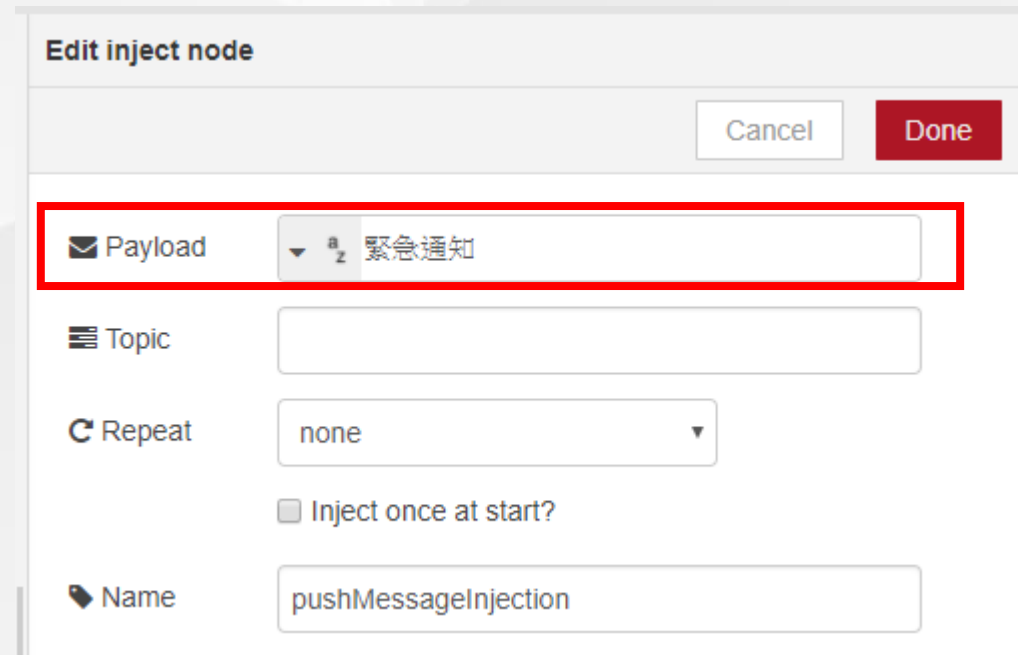
PUSH MESSAGE

PUSH MESSAGE

Input->選inject



Payload->輸入:緊急通知



PUSH MESSAGE

function->選function



Edit function node

Cancel Done

Name

Function

```
1  
2 return msg;
```

PUSH MESSAGE



```
var post_request = {
  "headers": {
    "content-type": "application/json; charset=UTF-8",
    "Authorization": "Bearer " + "{Channel Access Token}"
  },
  "payload": {
    "to": "UserId",
    "messages": [
      {
        "type": "text",
        "text": msg.payload
      }
    ]
  }
};
return post_request;
```

PUSH MESSAGE

function->選http request

URL->輸入

`https://api.line.me/v2/bot/message/push`



Edit http request node

Cancel Done

Method POST

URL `https://api.line.me/v2/bot/message/push`

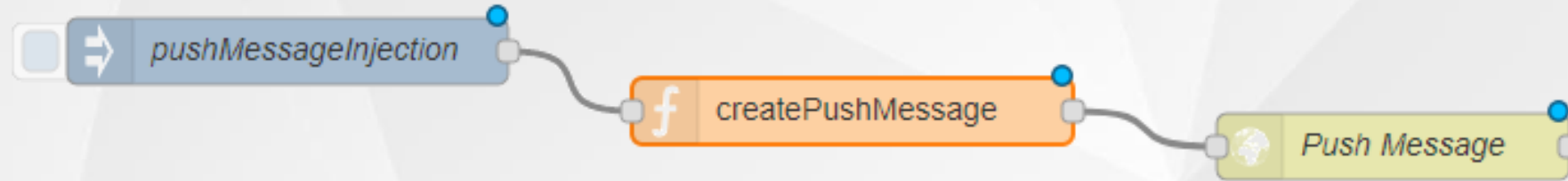
☐ Enable secure (SSL/TLS) connection

☐ Use basic authentication

Return a UTF-8 string

Name Push Message

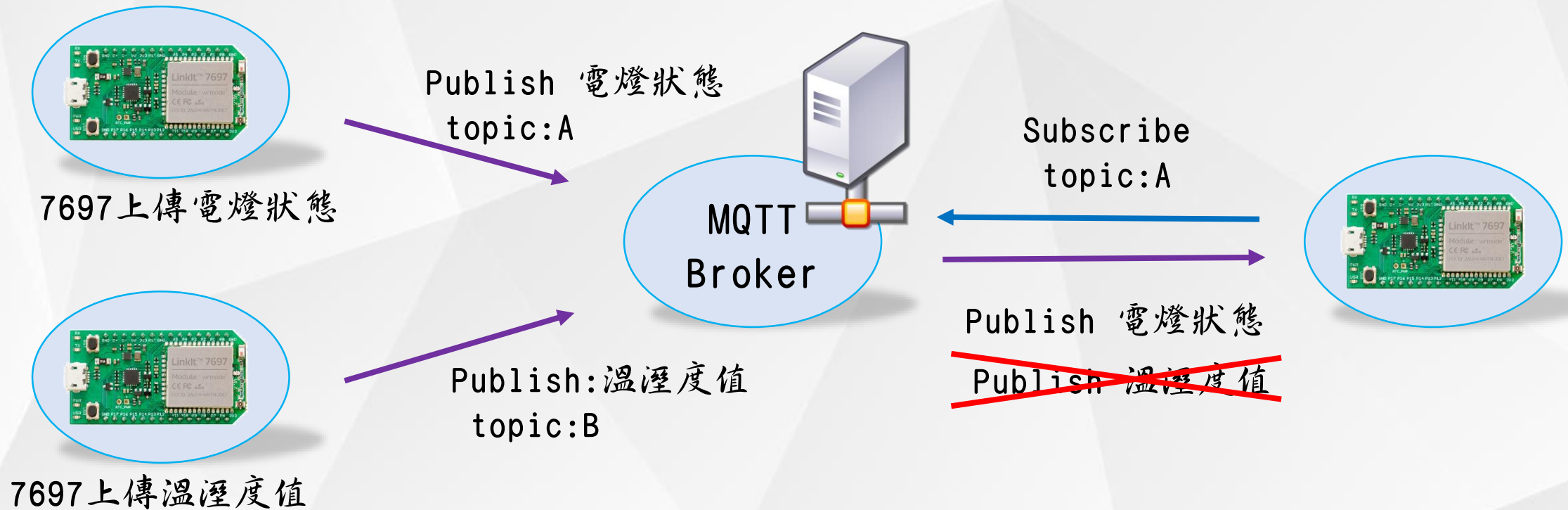
PUSH MESSAGE



MQTT 介紹

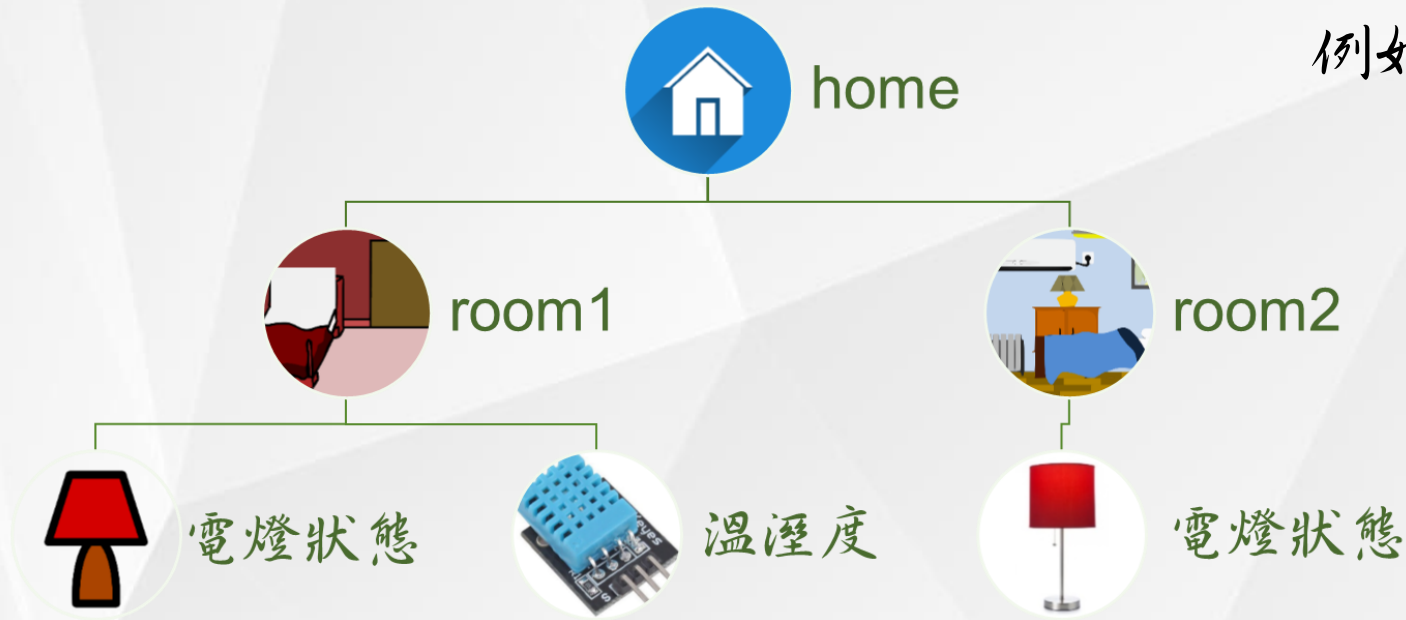
MQTT 介紹

使用TCP/IP作為基本的網路連線，低頻寬、低硬體需求的特性，並且透過publish/subscribe的方式做資料傳送與接收。



MQTT 介紹

Subscribe topic 可結合字元: + 跟 #



例如:

home/+ / 電燈狀態:

home/room1/電燈狀態

home/room2/電燈狀態

home/#: home/room1/電燈狀態

home/room1/溫溼度

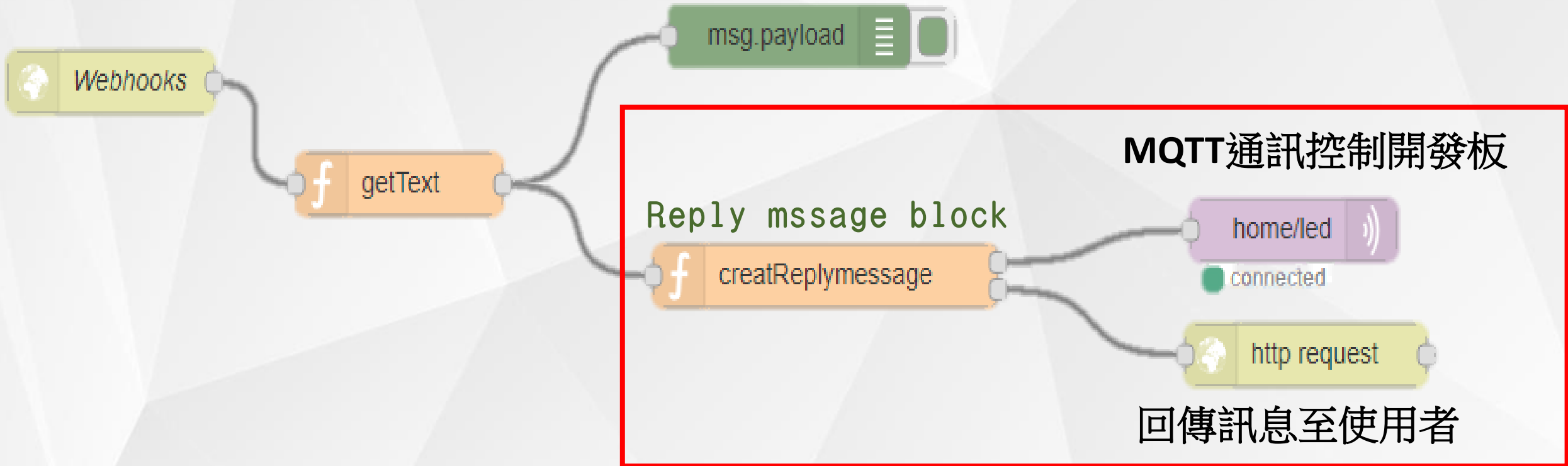
home/room2/電燈狀態

MQTT 訂閱資料

實作 Line bot 控制Usr_led燈

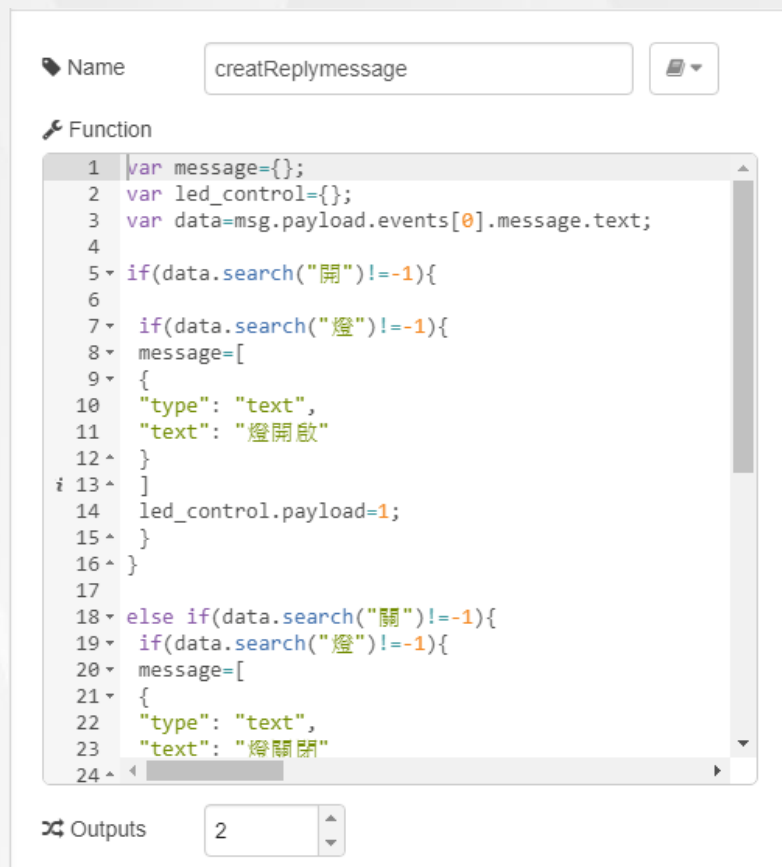
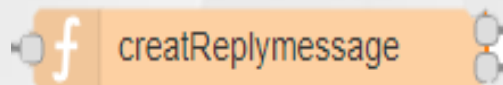


LINE BOT 控制USR_LED (NODE-RED 接線)



LINE BOT 控制USR_LED (NODE-RED 程式)

Reply message block



```
var message={};
var led_control={};
var data=msg.payload.events[0].message.text;
if(data.search("開")!=-1){
    if(data.search("燈")!=-1){
        message=[ { "type": "text", "text": "燈開啟" } ]
        led_control.payload=1; //LED控制訊息
    }
}
else if(data.search("關")!=-1){
    if(data.search("燈")!=-1){
        message=[ { "type": "text", "text": "燈關閉" } ]
        led_control.payload=0; //LED控制訊息
    }
}
}
```

LINE BOT 控制USR_LED (NODE-RED 程式)

Name:

Function

```
21 {
22   "type": "text",
23   "text": "燈關閉"
24 }
25 ]
26 led_control.payload=0;
27 }
28 }
29
30 var post_request = {
31   "headers": {
32     "content-type": "application/json; charset=UTF-8",
33     "Authorization": " Bearer " + "{k2z7jWXeLaEyAY0+0f
34   },
35   "payload": {
36     "replyToken": msg.payload.events[0].replyToken,
37     "messages": message
38   }
39 };
40
41
42 return [led_control,post_request];
43
```

Outputs: 選擇2個 OUTPUT

See the Info tab for help writing functions.

```
var post_request = {
  "headers": {
    "content-type": "application/json; charset=UTF-8 ",
    "Authorization": " Bearer " + "{Channel Access Token}"
  },
  "payload": {
    "replyToken": msg.payload.events[0].replyToken,
    "messages": [
      {
        "type": "text",
        "text": message
      }
    ]
  }
};
return [led_control,post_request];
```

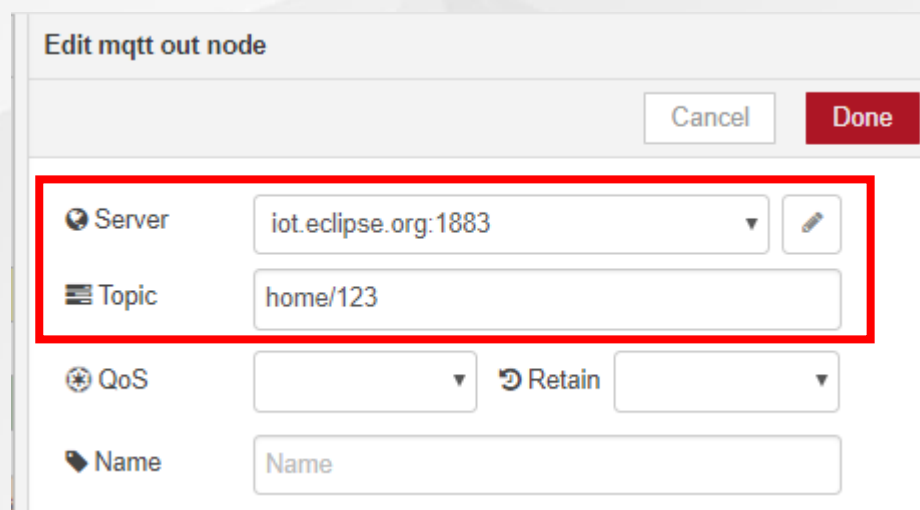
LINE BOT 控制USR_LED (NODE-RED 程式)

Output->選mqtt



Server->輸入iot.eclipse.org

Topic->自行定義

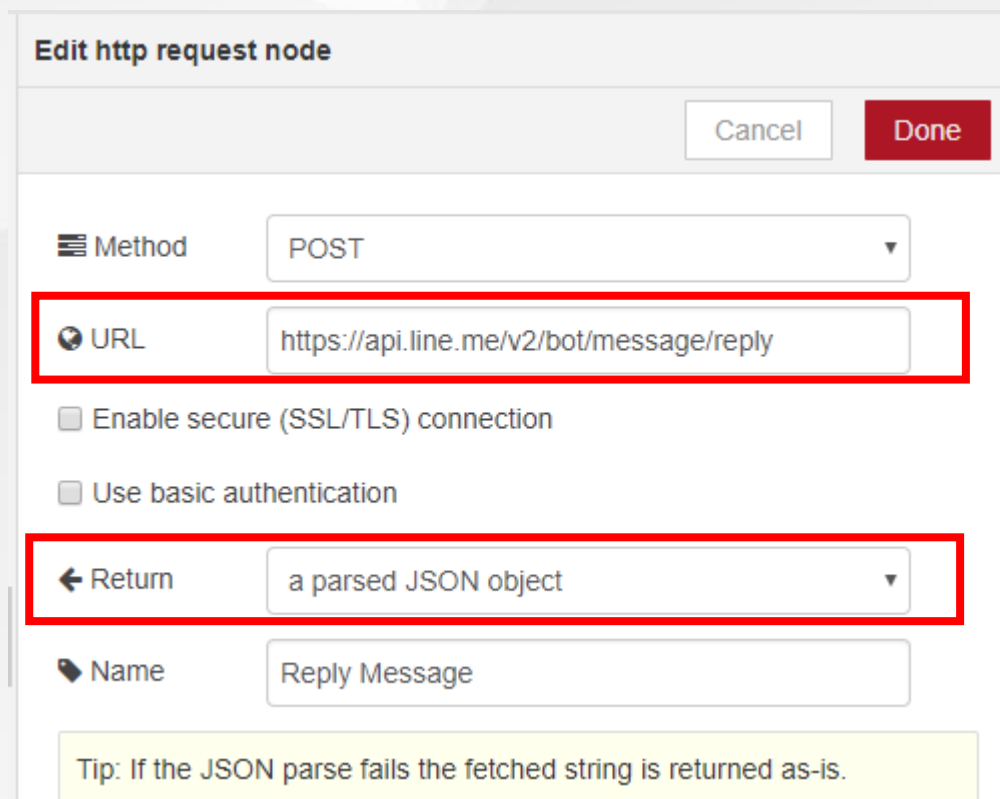


LINE BOT 控制USR_LED (NODE-RED 程式)

function->選http request

URL->輸入

`https://api.line.me/v2/bot/message/reply`

A screenshot of the 'Edit http request node' dialog box in Node-RED. The 'Method' is set to 'POST'. The 'URL' field contains 'https://api.line.me/v2/bot/message/reply' and is highlighted with a red box. Below the URL, there are checkboxes for 'Enable secure (SSL/TLS) connection' and 'Use basic authentication'. The 'Return' field is set to 'a parsed JSON object' and is also highlighted with a red box. The 'Name' field contains 'Reply Message'. At the bottom, there is a tip: 'Tip: If the JSON parse fails the fetched string is returned as-is.'

實作 Line bot 控制Usr_led燈

Arduino 程式撰寫



Line bot 控制Usr_led燈 (1/5)

```
#include<LWiFi.h>  
#include <PubSubClient.h>
```

```
char ssid[] = " ssid ";    // your network SSID (name)  
char pass[] = " password"; // your network password
```

```
int keyIndex = 0;  
int status = WL_IDLE_STATUS;  
WiFiClient client;  
char message[256];  
PubSubClient upload(client);
```

Line bot 控制Usr_1led燈 (2/5)

```
void callback(char* topic, byte* payload,
unsigned int length) {
  if (strcmp(topic,"home/123") == 0) {
    Serial.print("Message arrived [");
    Serial.print(topic);
    Serial.print("] ");
    for (int i=0;i<length;i++) {
      Serial.print((char)payload[i]);
      message[i]=payload[i];
    }
    Serial.println();
  }
```

```
if(atoi(message)==1){
  digitalWrite(7, HIGH);
}
else if(atoi(message)==0){
  digitalWrite(7, LOW);}
}
```

Line bot 控制Usr_led燈 (3/5)

```
void reconnect() {  
    while (!upload.connected()) {  
        Serial.print("Attempting MQTT connection...");  
        // Attempt to connect  
        if (upload.connect("client")) {  
            Serial.println("connected");  
            upload.subscribe("home/#");  
        }  
        else {  
            Serial.print("failed, rc=");  
            Serial.print(upload.state());  
            Serial.println(" try again in 5 seconds");  
            delay(5000);}  
    }  
}
```

Line bot 控制Usr_led燈 (4/5)

```
void setup() {  
  //Initialize serial and wait for port to open:  
  Serial.begin(9600);  
  pinMode(7, OUTPUT);  
  while (!Serial) {  
    }  
  while (status != WL_CONNECTED) {  
    Serial.print("Attempting to connect to SSID: ");  
    Serial.println(ssid);  
    status = WiFi.begin(ssid, pass);  
  }  
  Serial.println("Connected to wifi");  
  printWifiStatus();  
  upload.setServer("iot.eclipse.org", 1883);  
  upload.setCallback(callback);  
}
```

```
void loop() {  
  if (!upload.connected()) {  
    reconnect();  
  }  
  upload.loop();  
}
```

Line bot 控制Usr_led燈 (5/5)

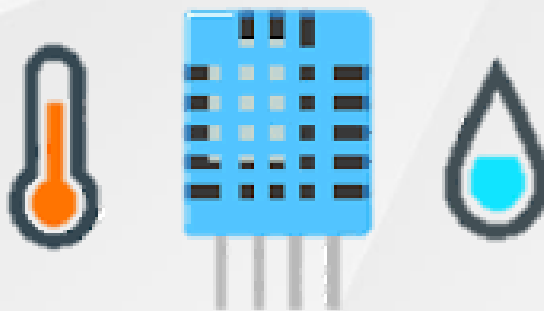
```
void printWifiStatus() {  
    // print the SSID of the network you're attached to:  
    Serial.print("SSID: ");  
    Serial.println(WiFi.SSID());  
  
    // print your WiFi shield's IP address:  
    IPAddress ip = WiFi.localIP();  
    Serial.print("IP Address: ");  
    Serial.println(ip);  
  
    // print the received signal strength:  
    long rssi = WiFi.RSSI();  
    Serial.print("signal strength (RSSI):");  
    Serial.print(rssi);  
    Serial.println(" dBm");  
}
```

MQTT 上傳資料

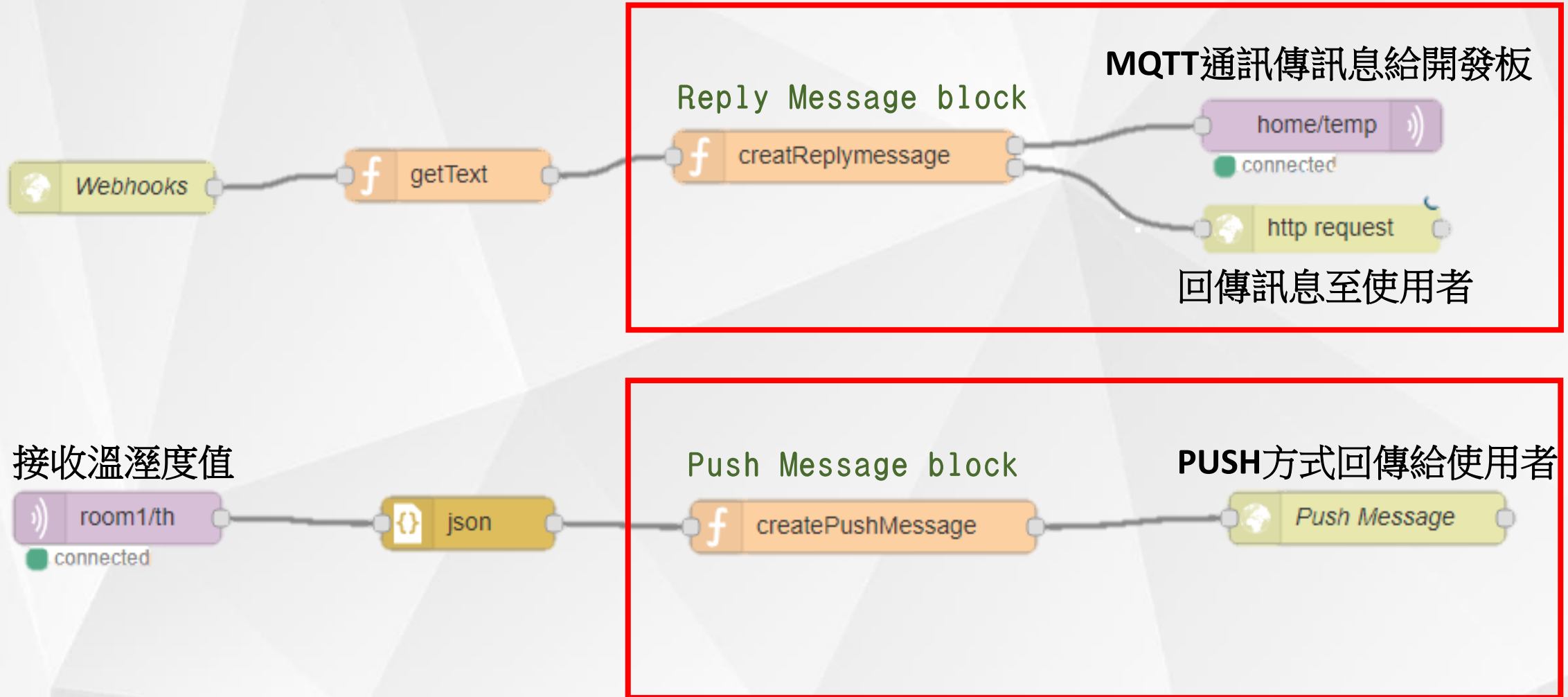
實作 Line bot 獲得溫溼度

溫溼度模組

如何獲得溫溼度呢？



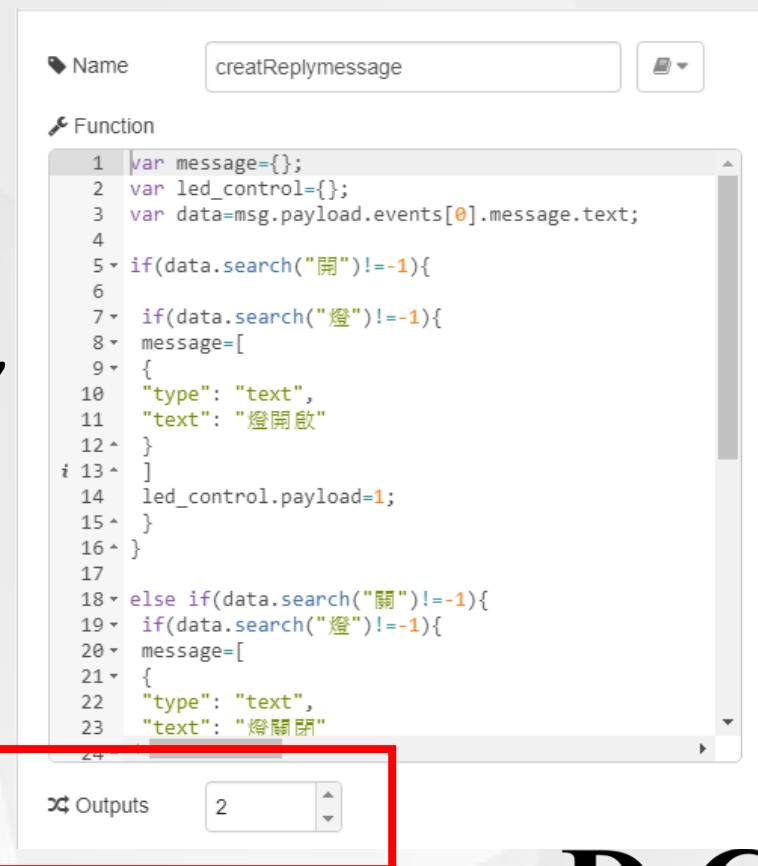
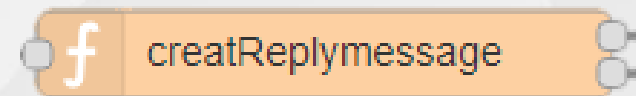
Line bot 獲得溫溼度 (Node-Red 接線)



Line bot 獲得溫溼度 (Node-Red 程式)

```
var message={};
var th_control={};
var data=msg.payload.events[0].message.text;
if(data.search("溫度")!=-1||data.search("濕度")!=-1){
  th_control.payload=1; //溫溼度觸發訊息
}
var post_request = {
  "headers": { "content-type": "application/json; charset=UTF-8",
  "Authorization": " Bearer " + "{Channel Access Token}" },
  "payload": {
  "replyToken": msg.payload.events[0].replyToken,
  "messages": message
  }
};
return [th_control,post_request];
```

Reply Message block

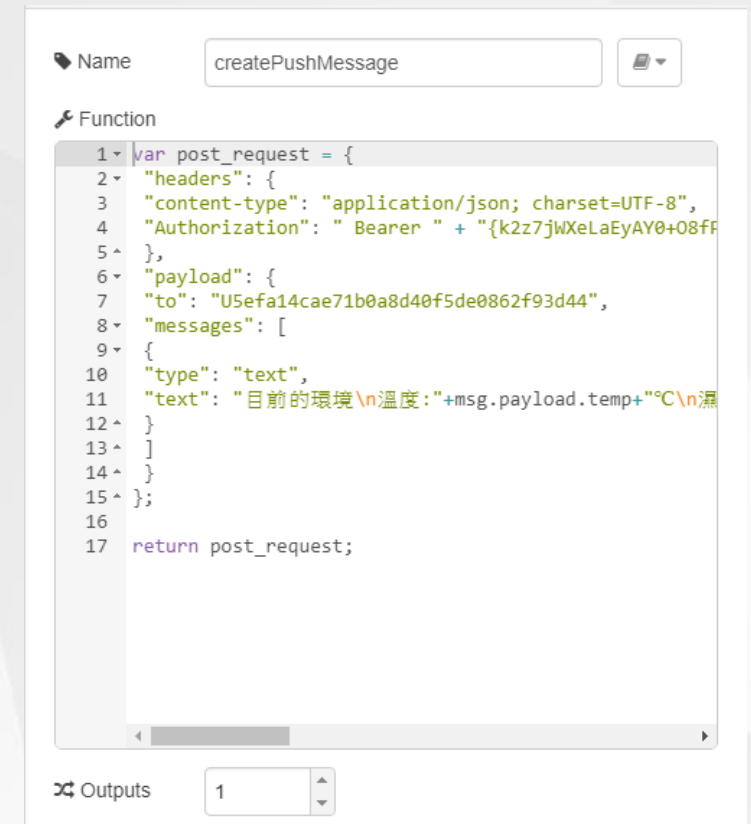
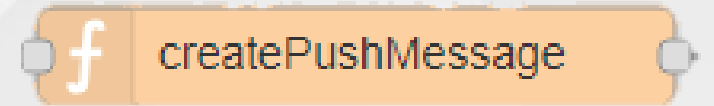


選擇2個 OUTPUT

Line bot 獲得溫溼度 (Node-Red 程式)

```
var post_request = {  
  "headers": { "content-type": "application/json; charset=UTF-8",  
  "Authorization": " Bearer " + "{Channel Access Token}" },  
  "payload": {  
    "to": "your userid",  
    "messages": [  
      {  
        "type": "text",  
        "text": "目前的環境\n溫度:"+msg.payload.temp+  
        "°C\n濕度:"+msg.payload.humid+"%"  
      }  
    ]  
  }  
};  
return post_request;
```

Push Message block

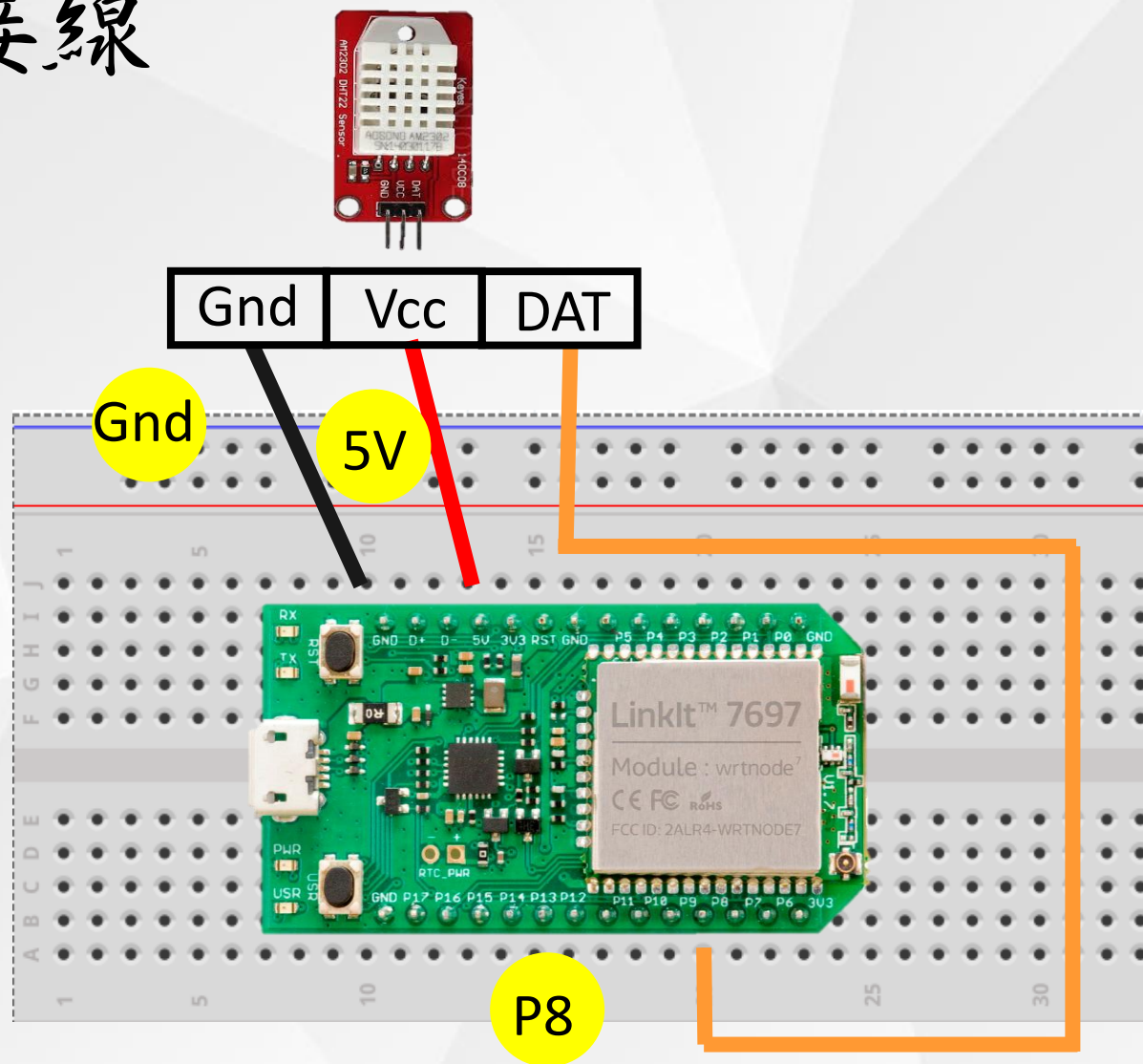


實作 Line bot 獲得溫溼度

Arduino 程式撰寫



溫溼度模組接線



Line bot 獲得溫溼度 (1/6)

```
#include<LWiFi.h>
#include <PubSubClient.h>
#include <dht.h>
#define dht_dpin 8
dht DHT;
char ssid[] = "your ssid";    // your network SSID (name)
char pass[] = "your password"; // your network password
int keyIndex = 0;
int status = WL_IDLE_STATUS;
WiFiClient client;
char message[256];
PubSubClient upload(client);
/*****/
char data[256];
char load[256];
```

Line bot 獲得溫溼度 (2/6)

```
void callback(char* topic, byte* payload, unsigned int length) {  
  
    if (strcmp(topic, "home/temp") == 0) {  
        Serial.print("Message arrived [");  
        Serial.print(topic);  
        Serial.print("] ");  
        for (int i = 0; i < length; i++) {  
            Serial.print((char)payload[i]);  
            load[i]=payload[i];  
        }  
        Serial.println();  
        if(atoi(load)==1){  
            th();  
        }  
    }  
}
```


Line bot 獲得溫溼度 (3/6)

```
void reconnect() {  
  while (!upload.connected()) {  
    Serial.print("Attempting MQTT connection...");  
    // Attempt to connect  
    if (upload.connect("client")) {  
      Serial.println("connected");  
      upload.subscribe("home/#");  
    } else {  
      Serial.print("failed, rc=");  
      Serial.print(upload.state());  
      Serial.println(" try again in 5 seconds");  
      // Wait 5 seconds before retrying  
      delay(5000);  
    }  
  }  
}
```

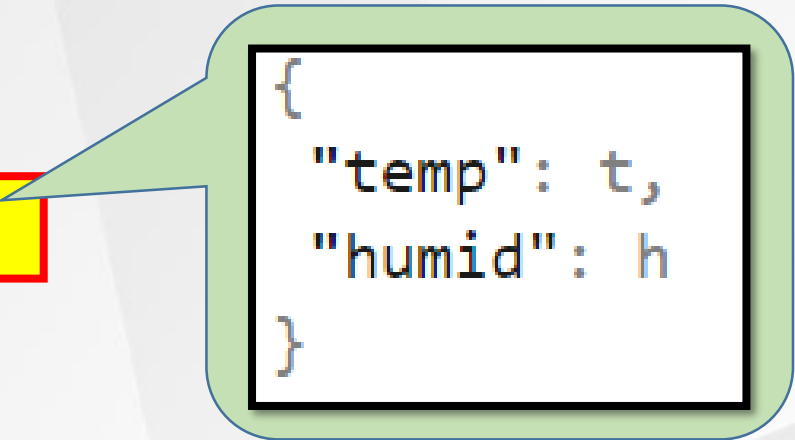
Line bot 獲得溫溼度 (4/6)

```
void setup() {  
  Serial.begin(9600);  
  while (!Serial) {}  
  // wait for serial port to connect. Needed for native USB port only  
}  
while (status != WL_CONNECTED) {  
  Serial.print("Attempting to connect to SSID: ");  
  Serial.println(ssid);  
  status = WiFi.begin(ssid, pass);  
}  
Serial.println("Connected to wifi");  
printWifiStatus();  
  
upload.setServer("iot.eclipse.org", 1883);  
upload.setCallback(callback);  
}
```

Line bot 獲得溫溼度 (5/6)

```
void loop() {  
  if (!upload.connected()) {  
    reconnect();  
  }  
  upload.loop();  
}
```

```
void th(){  
  DHT.read11(dht_dpin);  
  String h= String(DHT.humidity);  
  String t= String(DHT.temperature);  
  String load = "{\"temp\":\""+ t +" , \"humid\":\""+h+"}\"";  
  load.toCharArray(data, (load.length() + 1));  
  upload.publish("room1/th",data);  
}
```



```
{  
  "temp": t,  
  "humid": h  
}
```

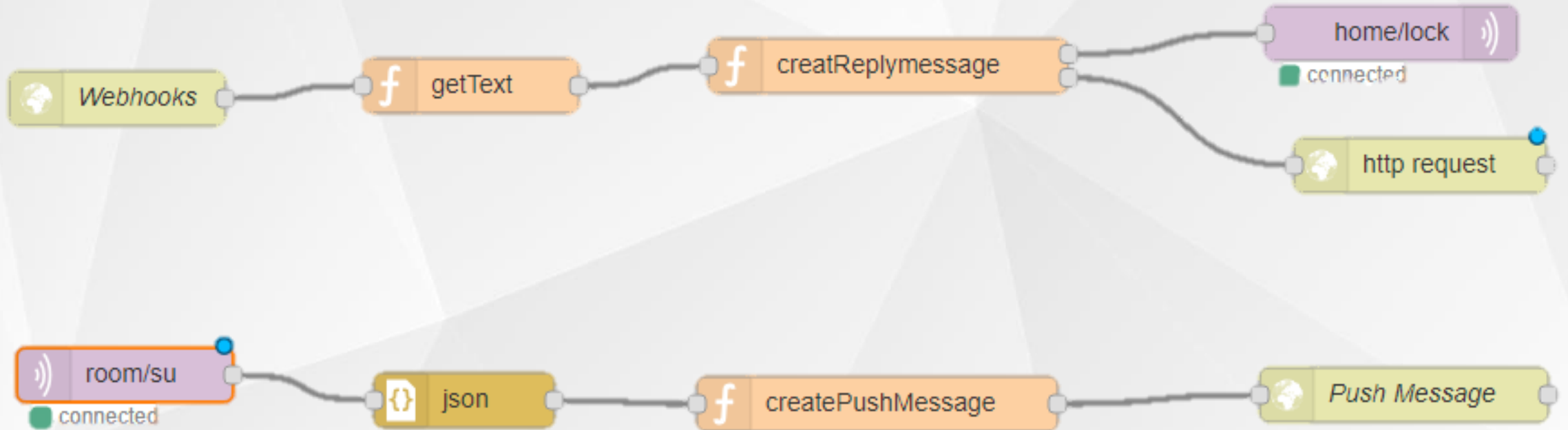
Line bot 獲得溫溼度 (6/6)

```
void printWifiStatus() {  
    // print the SSID of the network you're attached to:  
    Serial.print("SSID: ");  
    Serial.println(WiFi.SSID());  
  
    // print your WiFi shield's IP address:  
    IPAddress ip = WiFi.localIP();  
    Serial.print("IP Address: ");  
    Serial.println(ip);  
  
    // print the received signal strength:  
    long rssi = WiFi.RSSI();  
    Serial.print("signal strength (RSSI):");  
    Serial.print(rssi);  
    Serial.println(" dBm");  
}
```

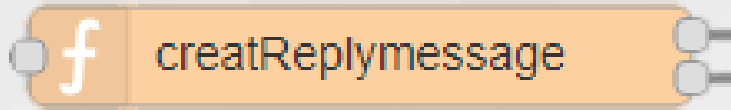
實作 Line bot 保全燈



Line bot 保全燈 (Node-Red 接線)



Reply Message block



Name:

Function

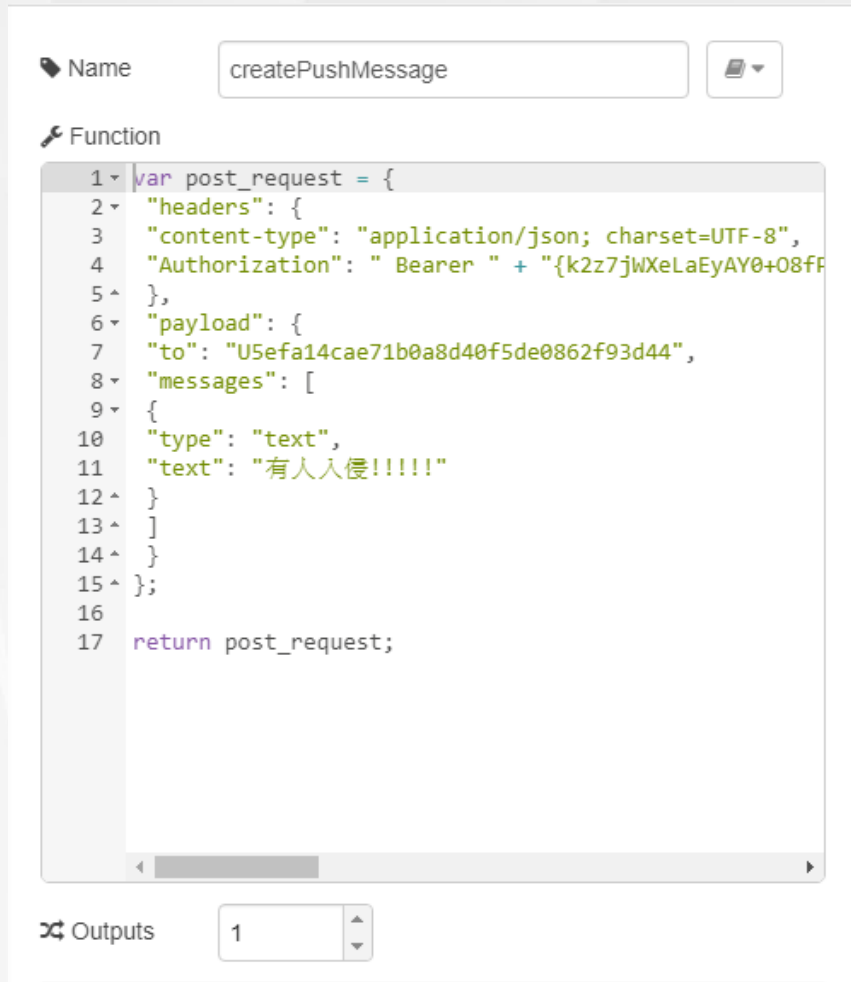
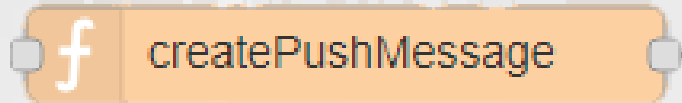
```
1 var message={};
2 var su_control={};
3 var data=msg.payload.events[0].message.text;
4
5
6
7 if(data.search("開")!==-1){
8
9   if(data.search("保全")!==-1){
10    message=[
11    {
12      "type": "text",
13      "text": "已開啟保全!"
14    }
15  ]
16  su_control.payload=1;
17 }
18
19
20 }
21
22 else if(data.search("關")!==-1){
23   if(data.search("保全")!==-1){
24     message=[
25
```

Outputs:

選擇2個 OUTPUT

```
var message={};
var su_control={};
var data=msg.payload.events[0].message.text;
if(data.search("開")!==-1){
  if(data.search("保全")!==-1){
    message=[ { "type": "text", "text": "已開啟保全!" } ]
    su_control.payload=1; }
  }
  else if(data.search("關")!==-1){
    if(data.search("保全")!==-1){
      message=[ { "type": "text", "text": "已關閉保全!" } ]
      su_control.payload=0; }
    }
    var post_request = {
      "headers": { "content-type": "application/json; charset=UTF-8",
        "Authorization": " Bearer " + "{YOUR TOKEN}" },
      "payload": { "replyToken": msg.payload.events[0].replyToken,
        "messages": message
      }
    };
    return [su_control,post_request];
```

Push Message block



```
var post_request = {  
  "headers": {  
    "content-type": "application/json; charset=UTF-8",  
    "Authorization": " Bearer " + "{Channel Access Token}" },  
  "payload": { "to": "your userid",  
    "messages": [ { "type": "text", "text": "有人入侵!!!!!"  
    }  
  ]  
}  
};  
return post_request;
```

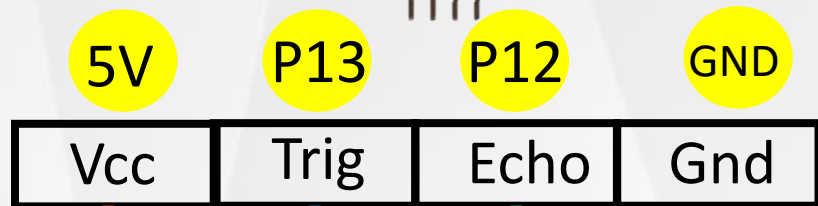

實作 Line bot 保全燈

Arduino 程式撰寫



保全燈硬體接線

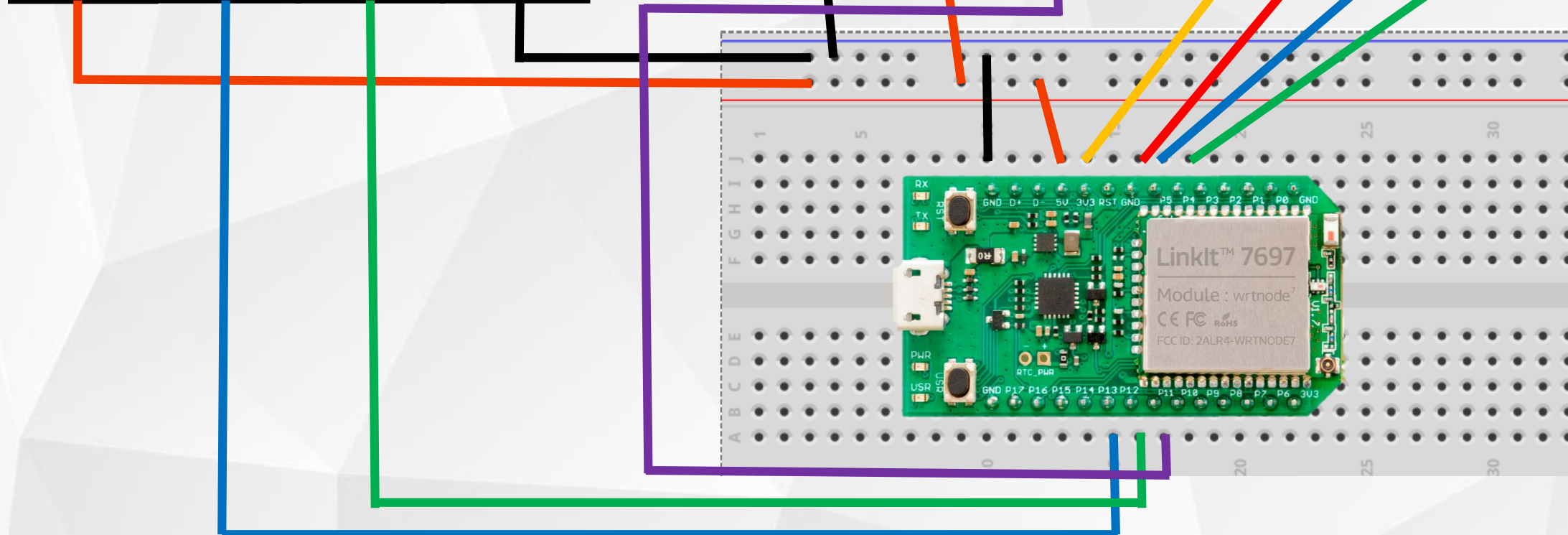
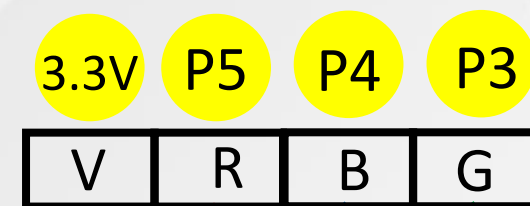
HC-SR04



蜂鳴器



RGB LED



Line bot 保全燈 (1/7)

```
#include<LWiFi.h>
#include <PubSubClient.h>
char ssid[] = " your ssid";    // your network SSID (name)
char pass[] = " your password"; // your network password
int keyIndex = 0;
int status = WL_IDLE_STATUS;
WiFiClient client;
int R=5,G=4,B=3;
#define trigPin 13
#define echoPin 12
int lock;
int piezoPin = 11;
int su=0;
char load[256];
PubSubClient upload(client);
```

Line bot 保全燈 (2/7)

```
void callback(char* topic, byte* payload, unsigned int length) {  
  if (strcmp(topic, "home/lock") == 0) {  
    Serial.print("Message arrived [");  
    Serial.print(topic);  
    Serial.print("] ");  
    for (int i = 0; i < length; i++) {  
      Serial.print((char)payload[i]);  
      load[i]=payload[i];  
    }  
    Serial.println();  
    if(atoi(load)==1){  
      lock=1;  
      digitalWrite(R, HIGH);  
      digitalWrite(G, LOW);  
      digitalWrite(B, HIGH);  
    }  
  }  
}
```

Line bot 保全燈 (3/7)

```
else if(atoi(load)==0){  
    lock=0;  
    su=0;  
    digitalWrite(R, LOW);  
    digitalWrite(G, LOW);  
    digitalWrite(B, LOW);  
    noTone(piezoPin);  
}  
}  
}
```

```
void reconnect() {  
    while (!upload.connected()) {  
        Serial.print("Attempting MQTT connection...");  
        if (upload.connect("linelock")) {  
            Serial.println("connected");  
            upload.subscribe("home/#");  
        } else {  
            Serial.print("failed, rc=");  
            Serial.print(upload.state());  
            Serial.println(" try again in 5 seconds");  
            // Wait 5 seconds before retrying  
            delay(5000);  
        }  
    }  
}
```

Line bot 保全燈 (4/7)

```
void setup() {  
  Serial.begin(9600);  
  pinMode(trigPin, OUTPUT);  
  pinMode(echoPin, INPUT);  
  pinMode(R, OUTPUT);  
  pinMode(G, OUTPUT);  
  pinMode(B, OUTPUT);  
  digitalWrite(R, LOW);  
  digitalWrite(G, LOW);  
  digitalWrite(B, LOW);  
  pinMode(piezoPin, OUTPUT);  
  noTone(piezoPin);
```

```
  while (status != WL_CONNECTED) {  
    Serial.print("Attempting to connect to SSID: ");  
    Serial.println(ssid);  
    status = WiFi.begin(ssid, pass);  
  }  
  Serial.println("Connected to wifi");  
  printWifiStatus();  
  
  upload.setServer("iot.eclipse.org", 1883);  
  upload.setCallback(callback);  
}
```

Line bot 保全燈 (5/7)

```
void loop() {  
  if (!upload.connected()) {  
    reconnect();  
  }  
  upload.loop();  
  long duration, distance;  
  digitalWrite(trigPin, LOW);  
  delayMicroseconds(2);  
  digitalWrite(trigPin, HIGH);  
  delayMicroseconds(10);  
  digitalWrite(trigPin, LOW);  
  duration = pulseIn(echoPin, HIGH);  
  distance = (duration/2) / 29.1;  
  Serial.print(" distance = "); Serial.print(distance);  
  Serial.println("cm ");  
  delay(1000);  
}
```

Line bot 保全燈 (6/7)

```
if((lock==1)&&(distance<4)){  
    su=1;  
    upload.publish("room/su","1");  
    digitalWrite(R, LOW);  
    digitalWrite(G, HIGH);  
    digitalWrite(B, HIGH);  
}  
if(su==1){  
    tone(piezoPin, 1000, 100);delay(200);  
    tone(piezoPin, 1000, 100);delay(200);  
    tone(piezoPin, 1000, 100);delay(200);  
    tone(piezoPin, 1000, 100); noTone(piezoPin);  
    delay(1000);  
}  
}
```