連接7688DUO的時候，請依照下面的步驟開始執行：

1. 拿一條USB接到電腦上
2. 然後去聯發科下載driver
3. 下載Arduino IDE 1.8以上
4. 安裝好Arduino後，進入Arduino打開偏好設定，在下面一個開發板管理員位置輸入：

<http://download.labs.mediatek.com/package_mtk_linkit_smart_7688_test_index.json>

1. 然後重新開啟Arduino，(可能)會提示有需要更新的部分，如果沒有，就去工具→開發板→開發板管理員中搜尋 Media 7688，如果有的話，就做更新。
2. 更新完了以後進入裝置管理員，點LinkIt smart7688做更新。
3. 更新完後，在Arduino工具→開發板找到是否有LinkIt smart 7688
4. 確認有之後再繼續找工具→序列埠是否有連到裝置
5. 都有之後Arduino這設定玩了，接下來要設定網路
6. 對著7688上的wifi按20秒後放開，然後等個2分鐘看連接的筆電上是否有LinkItsmart7688什麼鬼的
7. 有了之後連進去，並開啟瀏覽器打192.168.100.1進入頁面
8. 進入頁面之後會要求設定一次密碼，於是就隨便設定一個密碼，輸入完後進行登入
9. 登入後會有各種訊息，右上角有語言可以選擇，然後點選上方的”網路”選項
10. 先測試設定AP mode設定一個wifi密碼(8位)，設定完按CONFIGUREandRESTART，然後再等個2分鐘
11. 看到自己WIFI上有這個Linksmart什麼鬼的之後進行連線，這個時候就會要求輸入密碼了，就是剛剛設定的WIFI密碼
12. 上面步驟都正常之後，回到192.168.100.1，登入繼續設定網路Station mode
13. 選一個要連接的WIFI網路，如果這個WIFI有密碼的話，需要輸入WIFI密碼。然後一樣按CONFIGURE and RESTART繼續等個2分鐘重啟
14. 等好了之後基本上就設定OK了，如果到這裡沒有連接上的話，就……
15. 沒連接上就各種玩裝置了，按看看RESET鍵，或是重新插入7688，或是按幾次WIFI之類的。
16. 用Advanced IP Scanner來查看是否有這個裝置，通常這個裝置的名稱是LinkIt之類的，如果沒有就重新19的步驟。
17. 步驟20確認有這個裝置之後就完成所有設定了，結束。

上述步驟結束後只是最初設定完成，接下來要安裝內部套件以便往後的程式開發。

先用 opkg update更新

1. npm install mcsjs套件
2. opkg install nano套件
3. Requests套件，先把Requests檔案移到Linkit 7688內

然後進入存放安裝檔的根目錄下輸入python setup.py install

1. npm install bluebird –save
2. opkg install fswebcam
3. opkg install openssh-sftp-server

4跟5是用在攝影機上的套件，6是方便FTP連接7688所安裝的，因為Requests套件需要把安裝檔移到7688內，所以安裝這個套件方便檔案交流。

3的Requests是用在7688post跟get方法上的，也可以使用內建的httplib跟urllib模塊。只是後來為了取得MCS的資料使用了Requests的get方法。

2的nano 套件是方便程式撰寫用的編輯器

1的mcsjs是js連接mcs必須的套件，沒有引入這個模塊無法與MCS做溝通。

下面的是webcam的程式：

var mcs = require('mcsjs');//引入各種包

var exec = require('child\_process').exec;

var Promise = require('bluebird');

var fs = Promise.promisifyAll(require("fs"));

var myApp = mcs.register({

deviceId: 'DCj0Rnam',

deviceKey: 'kPiH1PR1RVE6qWfk',

host: 'api.mediatek.com'

});

//為了讓程式循環，使用setInterval來讓webcam可以定時拍照，但是在js中的程式執行，有可能會導致程式阻塞，所以正常情況很少這樣用。

var intervalId = setInterval(function(DataChannelId){

child = exec('fswebcam -i 0 -d v4l2:/dev/video0 --no-banner -p YUYV --jpeg 95 --save /tmp/test.jpg', function (error, stdout, stderr) {//拍照的指令

console.log('stdout: ' + stdout);//拍照後的細項用console.log打印出來

console.log('stderr: ' + stderr);//同上

if (error !== null) {//如果拍照時有出現異常

console.log('exec error: ' + error);//打印出異常

}

fs.readFileAsync('/tmp/test.jpg')//讀取圖片，使用同步方法

.then(function(data) {//因使用同步，所以會等到執行完上面才開始下面

myApp.emit(DataChannelId,'', new Buffer(data).toString('base64'));

});

});

},10000,"DataImage");//這裡的10000是指10秒，後面傳入的是DataChannelId

以下是判斷數值是否超過設定：

因為從PLC 接收到的資料格式需要做轉換才能比較，所以在下面的

data = tn.read\_very\_eager()後再判斷的時候都會轉換型態。

那又因為上傳時都為字串，所以判斷後要使用str(data)轉成字串型態，並加入payload中，這個payload就是上傳時的字串，最後也要判斷，如果payload是空的就不進行上傳動作，減少上傳動作的使用。

while True:

tn=telnetlib.Telnet(HOST,PORT)

print tn

payload = ""

tn.write("rd cm703\r\n")

time.sleep(1)

data=tn.read\_very\_eager()

if float(data)>=30 :

fltWaterWheel\_Value1=str(data)

payload += "DataString,," + fltWaterWheel\_Value1 + "\n"

tn.write("rd cm704\r\n")

time.sleep(1)

data=tn.read\_very\_eager()

fltWaterWheel\_Value2=int(data)

if fltWaterWheel\_Value2 >=30 :

IntData = "DataInteger,," + str(fltWaterWheel\_Value2)

payload += IntData + "\n"

tn.write("rd cm705\r\n")

time.sleep(1)

data=tn.read\_very\_eager()

fltWaterWheel\_Value3=float(data)

if fltWaterWheel\_Value3 >=30 :

FloatData = "DataOther,," + str(fltWaterWheel\_Value3)

payload += FloatData + "\n"

if payload != "" :

post\_to\_mcs(payload)

time.sleep(7)

以上是判斷數值大小的片段程式碼。

以下是IFTTT對7688做連結的方法，可以用此方法取得的網址做控制7688，例如開燈：

先創好原型跟測試裝置，還有資料通道，之後點右上角的人物頭像，選個人檔案，在選服務提供者。然後創建一個APP服務。輸入的網址之後會說到到這邊就先停住。

到IFTTT創帳號，然後點My Applets然後點右上角的new applet，this的地方選webhooks，然後點Receive的按鈕，也只有這個按鈕可以選，取一個Event名稱，這個Event名稱很重要，之後會用上，然後點確定，在that的地方一樣用webhooks然後選request的按鈕，反正也只有這個可以選，進入後輸入URL，這個URL要輸入操控對象的API網址，例如MCS的開關就是https://api.mediatek.com/mcs/v2/devices/DCj0Rnam/datapoints

如果是其他產品就使用對應能操作的網址，然後下面選用POST，資料TYPE用application/json，最下面的body欄位填入以下。

{

"appId":"{your appId}",

"appSecret": "{your appSecret}",

"datapoints":[

{

"dataChnId":"switch",

"values":{ "value":"1" }

}

]

}

在第一次連接webhooks的時候他會給一個KEY，從IFTTT上方點My Applets點Service，然後點webhooks再點右上角的Documentation，進入後他會有一個網址，先將這個網址的event的部分填上\*，然後複製網址貼到最開始的步驟需要輸入網址的那個地方，按下確定就會獲得appId與appSecret，將這兩個複製貼到上面body需要輸入的地方，然後再把dataChnId後的資料通道ID改成自己的。

然後按下create，這樣一個能控制MCS開關的服務就結束了。

至於前面提到那個很重要的名稱，在你輸入\*的那串網址，這次把\*改成自己設定的名稱，則這串網址就成為能夠開關MCS的網址了。

他主要是判斷\*位置的名稱，只要我們創多個不同名稱的IFTTT服務，就可以用多個服務去操控MCS上的元件了。

以下是使用thread去非同步執行方法：

先引包，如果發現執行時跳出httplib或urllib錯誤，很有可能是版本上不支持，有兩個解決方法：

1. 進入7688控制介面192.168.0.X，然後reset整個裝置。
2. 按照網路上的方法把httplib,urllib做修改，調到可以支持的版本。

這裡因為使用thread線程來執行，所以使用Queue存取回傳值。

如果有多個裝置的MCS需要分別上傳，在下面的deviceId跟deviceKey可以做修改，

import time

import sys

import httplib, urllib

import json

import random

import threading

from Queue import Queue

sys.path.insert(0, '/usr/lib/python2.7/bridge/')

deviceId = ["DCj0Rnam","deviceId2","deviceId3", "deviceId4","deviceId5"]

deviceKey=["kPiH1PR1RVE6qWfk","deviceKey2",“deviceKey3" ,"deviceKey4" ,"deviceKey5"]

address=["192.168.0.50","192.168.0.51","192.168.0.52","192.168.0.53","192.168.0.54",]

PORT = 8501

global results

results = []

print\_lock = threading.Lock()

def post\_to\_mcslite(payload):#如果有多個MCS裝置要上傳，這裡的傳入參數就要多deviceId,deviceKey

headers = {"Content-type": "text/csv", "deviceKey": deviceKey}

not\_connected = 1

while (not\_connected):

try:

conn = httplib.HTTPConnection("api.mediatek.com:80")

conn.connect()

not\_connected = 0

except (httplib.HTTPException, socket.error) as ex:

print "Error: %s" % ex

time.sleep(5)

conn.request("POST", "/mcs/v2/devices/" + deviceId + "/datapoints.csv", payload, headers)

response = conn.getresponse()

print( response.status, response.reason, payload, time.strftime("%c"))

data = response.read()

conn.close()

#\*\*\*\*\*\*\*\*telnet to plc\*\*\*\*\*\*\*\* 這裡還未用host連線，要連接PLC的話

def telnetplc(worker):

payload = ""

global results

time.sleep(0.5)

integer = random.randint(0,60)

if integer >=30 :

payload += "DataInteger,," + str(integer) + "\n"

float = random.uniform(1,10)

if float >= 5 :

payload += "DataOther,," + str(float) + "\n"

if payload != "" :

with print\_lock :

payload += worker

results.append(payload)

else :

#time.sleep(3)

return telnetplc(worker)

def threader():

while True :

worker = q.get()

telnetplc(worker)

q.task\_done()

q = Queue()

#上傳的裝置不同的話，需要修改post\_to\_mcslite傳入值新增deviceId,deviceKey

while True :

results = [] # 這裡初始化回傳的值

for x in range(5) :#thread amount

t = threading.Thread(target = threader)#thread method

t.daemon = True

t.start()

for i in range(len(address)) : #把PLC的連結放進q中，

q.put(address[i])

q.join()#這行代表執行完上面的才會開始執行下面的程式

for i in range(len(results)) : #因為上面已經用join等待所有程序執行完畢，所以這裡的results必定有值，而以下就是判斷哪一個results是哪一個PLC的程式

if results[i].find(address[0]) != -1 :

payload1 = results[i][:results[i].find(address) -1]

elif results[i].find(address[1]) != -1 :

payload2 = results[i][:results[i].find(address) -1]

elif results[i].find(address[2]) != -1 :

payload3 = results[i][:results[i].find(address) -1]

print results[i]#這行是除錯用的，上面都沒寫到上傳的部分，如果需要加入上傳就在這裡加上post\_to\_mcslite(payload12345)

time.sleep(10)

以下是使用thread + socket來做裝置間的互相連線，目的是為了能做到檔案互傳，有了檔案互傳，就可以對7688內的檔案做備份與管理，因為連線部分會有server端與client端，所以檔案會分成兩個部分，一開始線會有占用PORT的情況，後來想盡各種方法還是無法關閉PORT，但是發現能做到使PORT自滅，windows內有預設每一個占用的PORT在斷開連接後有多少的存在時間，比起一開始的無法關閉PORT好多了，但還是有問題，問題在於如果假設1分鐘內連接超過6萬次就會導致PORT佔滿，當然正常情況不可能1分鐘就用掉6萬個PORT。

以下是client端的檔案接收

#持續socket到伺服器拿同一個檔案名稱的資料，time.sleep()可以調整時間，單位是秒

import socket

import time

def Main():

host = '192.168.0.XXX'

port = 5000

while True:

s = socket.socket(socket.AF\_INET,socket.SOCK\_STREAM,0)

s.connect((host,port))

#filename = raw\_input("Filename? -> ")

filename = "test.jpg"

s.send(filename)

data = s.recv(1024)

if data[:6] == 'EXISTS':

#if receive data top 6 words =EXISTS

filesize = long(data[6:])

#get filesize after data 6 words

s.send('OK')

#send "OK", for FileServer line 11,

f = open('' + filename, 'wb')#new a file ,and write type is wb

data = s.recv(1024)

#receive data

total = len(data)

#first run to write filelength

f.write(data)

#write data to newfile

while total < filesize:

data = s.recv(1024)

total += len(data)

f.write(data)

print( "{0:.2f}".format((total / float(filesize))\*100) + "% Done")

print ("Download Complete!")

f.close()

else:

print("File does not Exist!")

s.close()

time.sleep(15)

if \_\_name\_\_ == '\_\_main\_\_':

Main()

以上是FileClient端的程式碼

以下是Server端的程式碼

Server端的程式需要持續開啟，這樣才能讓Client端連接。

import socket

import threading

import os

import time

confirm = "b'OK'"

def RetrFile(name ,sock):

try:

#fname = ""

#file = os.listdir(os.getcwd())

#for f in file :

# if os.path.isfile(f):

# fname += f +"\n"

#sock.send(fname.encode())

filename = sock.recv(1024)

print(filename)

a = filename.rindex(",")

#if len(filename[:a]) == 8:

#print(filename[:a])

#print(filename[a+1:])

#f= open('' + filename , 'wb')

#data = sock.recv(1024)

#totalRecv = len(data)

#f.write(data)

#while totalRecv < filesize

if os.path.isfile(filename):

EXIST = str(os.path.getsize(filename))

sock.send("EXISTS ".encode() + EXIST.encode())

print("data size step ok")

#sock.send(str.encode("EXISTS " + EXIST))

userResponse = sock.recv(1024)

print(userResponse[:2])

#if userResponse[:2] == confirm:#if Client type the Y

if len(userResponse) >1:

with open(filename,'rb') as f:

bytesToSend = f.read(1024)

sock.send(bytesToSend)

f.flush()

#while bytesToSend != "":

while len(bytesToSend) > 0:

bytesToSend = f.read(1024)

sock.send(bytesToSend)

f.flush()

#print(bytesToSend)

f.close()

f.close()

else:

sock.send("ERR".encode())

except:

pass

finally:

sock.close()

def Main():

host = '192.168.0.132'

port = 5000

s = socket.socket(socket.AF\_INET,socket.SOCK\_STREAM,0)

s.setsockopt(socket.SOL\_SOCKET, socket.SO\_REUSEADDR, 1)

s.bind((host,port))

s.listen(5)

print("Server Started")

while True:

c, addr = s.accept()

print("client connected ip:<" + str(addr) +">")

t = threading.Thread(target=RetrFile,args=("retrThread", c))

t.start()

s.close()

if \_\_name\_\_ == '\_\_main\_\_':

Main()

下面是Linebot的設定與佈署，因為程式過於長，word貼不上，所以暫且談如何佈署一個Linebot，這裡使用我自己做的檔案，所以沒提及json 或.ignore或config或HEAD，如果是第一次從頭開始的話上面的檔案要重新自己寫。

第一步：

先進入<https://developers.line.me/en/>

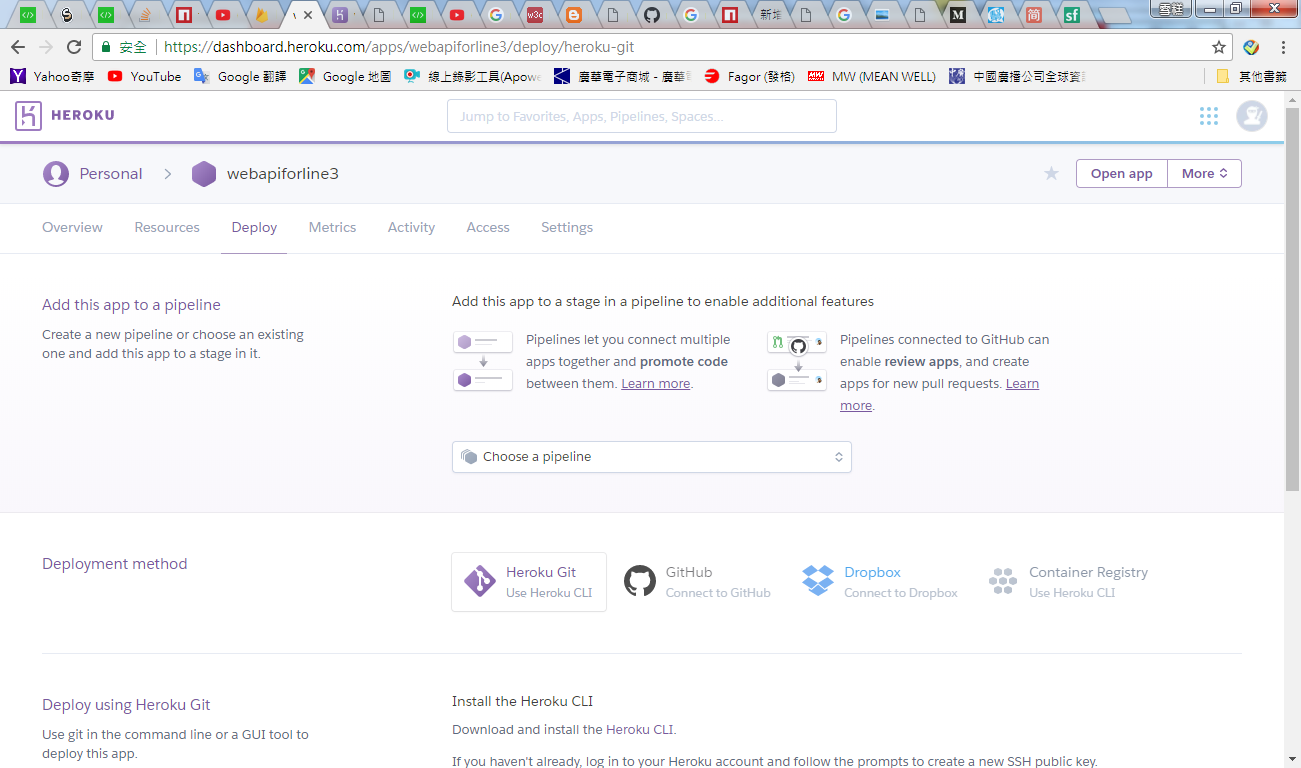
點選右邊的start using Messaging API

填入資料

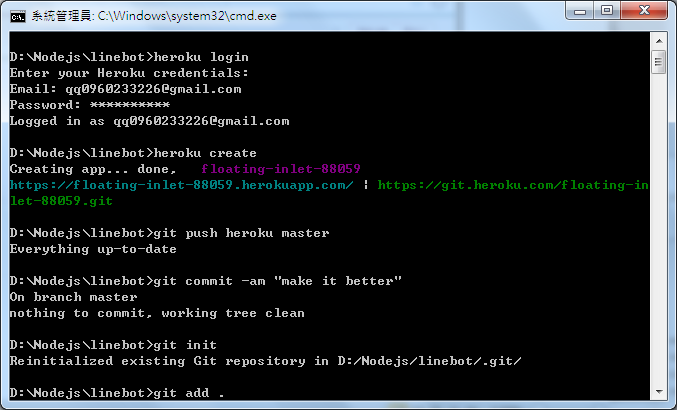
**Select provider的部分就填公司或個人名稱**

創建好一個linebot 後，點進linebot 內記起Channel ID，Channel Secret，Access Token這三個，後面會用上，如果沒有出現Access Token的話，點一下Issue然後直接點Issue就會出現了。

第二步：

接著到heroku上申請帳號，申請完之後，隨便創一個專案，然後點進去點選Deploy標籤，會出現選擇上傳方式的頁面，在Install the Heroku CLI的這個地方點下Download and Install the Heroku CLI，下載後安裝。

安裝好之後打開cmd輸入heroku login然後輸入帳號密碼



進入後轉移到存放index.js 的目錄底下，如上圖，移到目錄下後，輸入heroku create ，這個時候heroku 就會幫你創建一個專案並”自動”連接到這個專案。

第三步：

接著下指令

git init

git add .

heroku git:remote –a 專案名稱

git commit –am “說明文字”

git push heroku master

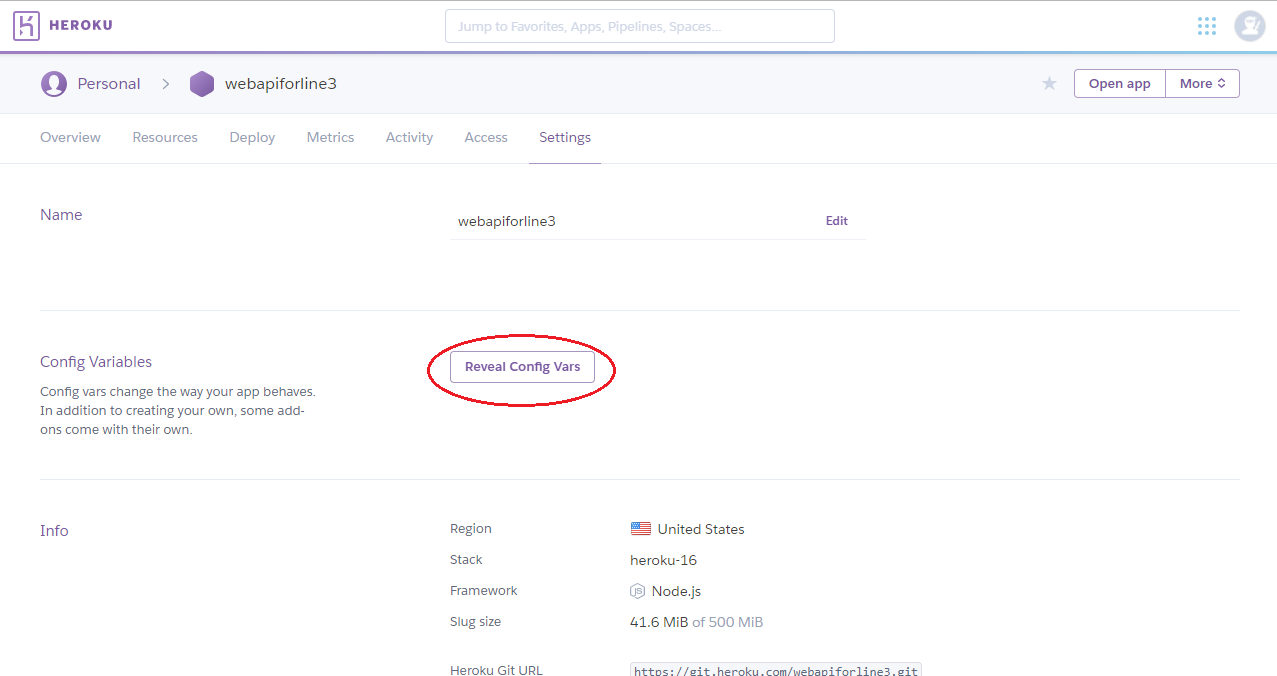
然後就完成了，事後再去heroku網站上手動更改專案名稱會比較簡單，更改後保險一點請再做一次heroku git:remote –a 專案名稱設定。

雖然中途還有一些設定，不過不是很難，上網找一下不用5分鐘就能得到答案了，上述是比較複雜的部份

如果沒有安裝git 的話，建議在網路上找一下git 來安裝

安裝後要設定git 一下

第四步：

第三步完成後就等於佈署好webapi了，回到linebot 頁面把一開始記下的三串文字複製好，回到heroku的專案內點入setting 點Reveal Config Vars。 

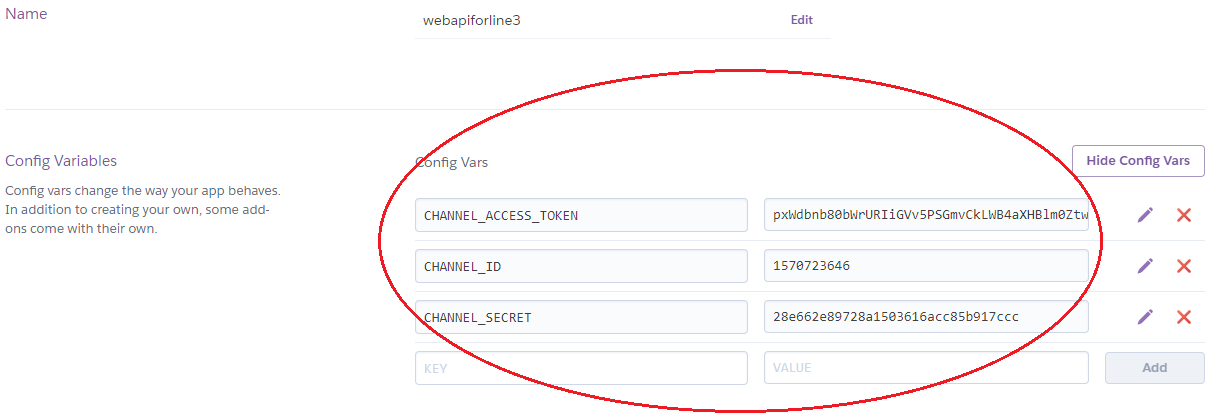
然後在上面輸入以下

CHANNEL\_ACCESS\_TOKEN 這個是自己的token

CHANNEL\_ID 這個是自己的Channel ID

CHANNEL\_SECRET 這個是自己的Channel Secret

就是最開始記起來的那三個東西辣



然後在heroku右上角點 open App，打開後在網址上最後面加上/linewebhook

然後將整個網址複製貼到linebot的Webhook URL中，記得Webhook URL上面的USE Webhook要開啟，然後測試發送，沒問題之後整個linebot就架設好了。

**最重要的一點是一定要加linebot好友**

以下是其他細項提醒：

1. Webcam拍照的程式是使用nodejs去編寫的，所以必須要裝nano才能夠執行，並且因為nodejs執行是非同步方式，所以webcam使用的是阻塞式的方法去讓他循環跑程式，其餘nodejs程式不建議使用此方法。
2. 而因為我自身原因沒辦法使用Azure所以才會使用heroku平台當作Linebot的服務器，如果有信用卡可以綁定且有足夠資金可以運用的話Azure可能會比較方便，並且在開發C#上會比較方便。
3. 使用socket做連接的那程式，如果不要太時間連接都沒問題，如果每秒都要做連結的話可能就會出問題了，並且PORT盡可能不要佔用太多。
4. 在7688內部安裝的部分，Requests套件不是一定要安裝，只是如果需要從MCS上取得數值的話才需要安裝，或是想要更方便post到MCS。
5. 經過中途大約1個月多的時間沒碰過7688，再次碰到的時候發現他的video0裝置掃描不到，這是一個很嚴重的問題，如果在這些時間中有無意間改動它內部的設定，那可能還很正常，但如果是放著他會出現這樣的異常的話，代表7688以後每幾個月也會出現同一個情況，當然也有可能是我這7688剛剛好就是比較有問題的，正常的不會出現此現象的可能性也有，至少我在MCS討論區上沒有人回答我對此的提問。
6. 針對第5點發生時如何解決？正常人可能會按板子上的任何reset按鍵，但經過我測試，按了也沒用，若是我將SD卡擴充進去導致的，那拔出SD卡就好了？錯，拔出SD卡一樣沒有video0裝置，目前我使用到的方法是到7688控制台去Reset這塊板子，但這樣做會倒置7688內的資料全部消失，所以在點Reset前三思。
7. 在擴充SD卡上可能會去參考網路上的方法，但網路上的方法不一定是最好的，仔細看網路上教學的方法，會發現他們的擴充方法是基於”備份”的形式，也就是說他把”整個”7688的資料都複製進SD卡內，讓你有種空間被擴充的感覺，但如果拔出SD卡，內部的資料會全部消失，因為都存在SD中。
8. 另外，聯發科可能會對7688做出韌體的更新，這裡要注意！因為在做韌體更新之後7688內部的資料也會”全部消失”，他與Reset不同，Reset還會留下一些安裝過的檔案或套件，但韌體更新會全方面Reset，所以要注意。
9. 還有在公司做7688連線設定的時候發現，7688會連接到192.168.0.XXX以外的地點，例如192.168.50.XXX，而連接到這個50可能會導致尋找不到7688，或是地址突然做變更 例如從192.168.50.XXX 突然轉成192.168.0.XXX，在這個轉換的過程中，如果程式有在運行，可能會報錯或是停止運行，這也是我初步推估會變成無法長期上傳資料的問題之一，當然問題二就是因為公司的電斷掉了，所以無法連上，當然也有可能是筆電的睡眠待機等原因導致。
10. 要安裝各種套件之前先用opkg update 做更新，沒做更新的話可能會無法安裝某些套件。
11. Linebot一定要加入好友才能夠發送訊息或回覆訊息，如果沒有加入好友，即使有對方的ID也無法傳送訊息。
12. IFTTT的部分除了可以控制7688外，也可以使用MCS提供的觸發條件與動作，然後將IFTTT的webhook得到的網址貼上去，但是這裡還是有個問題，確實這樣就能做到Line的提醒，但是根據我的經驗，他觸發條件的速度極度慢，有很多時候都是MCS資料超過數值半小時或是幾小時後他才會發警訊到Line上，很不理想，我也不知道是不是MCS的服務器速度較慢或是其他原因。
13. 其實一開始連接的步驟有點多了，正確步驟大概是連接到電腦，然後點7688的WIFI進192.168.100.1設定密碼並輸入密碼，然後到WIFI中設定要連上的網路，並輸入網路的密碼，最後他會重新連上網路等到他燈慢慢閃的時候就代表連上了，但有時候連上了卻發現找不到LinkIt7688的IP，這時候估計就是7688連接上了錯誤的網段，所以這個時候就重新讓他在連一次網路直到可以為止。