-

**Flask Server 確認**

localhost:5000 (原電腦server連線)

**Ngrok建置**

**定義**

讓本地端具有一個公共網址->讓外部機器(裝置)有IP可以訪問連線

1. 下載Ngrok軟體(https://ngrok.com/download)
2. 解壓縮得到ngrok.exe
3. 執行cmd使用命令列操作Ngrok:
4. 在命令列中切換至ngrok.exe所在目錄(或在ngrok.exe所在資料夾下開cmd)
5. 在cmd中輸入語法「ngrok.exe http 5000 -region ap 」 (5000是本地Flask web server的port號，所以需與flask的port號一致)
6. 正式執行，切勿關閉cmd
7. Forwarding <http://xxxxxxxx.ngrok.io> (連結server的網址)

**Flask結合Ngrok測試**

可使用其他裝置來嘗試上方的連結網址

**Line對話機器人原理介紹**

本地端server ←→ Line ←→ 用戶

**LineBot建立與介面一覽**

1. Line 機器人帳號創建 (Line messaging api)
2. 介面一覽
   1. 廠商數位化
3. channel\_access\_token (功能:讓Line辨認)

#Messaging API-> Channel access token (記得issue)

Channel\_secret (功能:篩選輸入的訊息)

#Basic settings -> Channel secret

**LineBotSDK套件安裝**

<https://github.com/line/line-bot-sdk-python>

看到範例程式碼，有看，但不知道他寫啥

套件名

Line-bot-sdk "line-bot-sdk"="1.16.0"

Channel\_access token: 給LineBotApi(‘’)用，# Messaging API : Channel access token (long-lived)

channel\_secret:給handler =WebhookHandler(‘’)，# lin page 的Basic settings/ Channel secret

**LineBot Server程式碼結構解析**

生活化案例 - 消息寄過來了，接收，驗證，轉交

webhook, app, route, handler, linebotAPI

WebHandler與LineBotAPI 兩大主角

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| Messenage | Line | Webhook | route | Handler | Custom def | Line bot Api | Line |
| 信 | 郵局 | 地址 | 大門 | 總機 | 業務部 | 收發人員 | 郵局 |

Handler→ Channel secret

Line bot Api→ Channel access token

**第一次 - 自動回覆文字消息體驗**

handler.add

line\_bot\_api.reply\_message()

TextMessage與TextSendMessage

**回家作業:明後天手打一遍樣板程式碼，確認機器人能正常運作。**

=======

關注事件

1.引用套件 FollowEvent

2.handler.add(關注事件) (若收到關注事件， 則取得人家個資，並打印出來)

@handler.add(FollowEvent)

3.寫方法

具體業務流程開發(參考目標)

問卷

詢問是男是女

若是男生，則將個資存到男生個資記事本

若是女生，則將個資存到女生個資記事本

關注事件

1. 取個資

#取得用戶個資

user\_profile=line\_bot\_api.get\_profile(event.source.user\_id)

#開啟一個檔案，將用戶個資轉成json格式，存入檔案內

with open ("./user.txt","a") as myfile :

myfile.write (json.dumps(vars(user\_profile)))

#新資料換行

myfile.write("\r\n")

1. 回傳文字消息
2. 練習 - 回傳圖片消息

用戶發送文字消息

1. 接收文字消息
2. log紀錄 (BI商業視覺化)
   1. 先暫時以with方式
   2. 未來要自己去研究log模組
   3. 為什麼要研究
3. 回傳模板消息
   1. 練習 - 官方方法

MessageAction體驗

URIAction體驗

PostbackAction體驗

* 1. linebotdesigner的json轉換方式
     1. linebotdesigner設計消息，縮圖記得要選「否」，最後取得json文字
        1. URI Action的神奇之處
     2. 將json手動存成檔案，副檔名「.json」
     3. 讀取檔案，並轉成json
     4. json轉換物件
     5. line\_bot\_api發送
     6. 練習 - flexSendMessage以json轉換
  2. 隱藏data設計
     1. sex=male
     2. sex=female

用戶發送Postback事件

1. 將data打印出來
2. data的內容格式探討 - querystring
3. 解析querystring的方式
4. 練習 - 解析sex內容，並取得個資，存到不同檔案
   1. 男性檔案名為male\_profile.txt
   2. 女性檔案名為female\_profile.txt

QuickReply

1. 檢討模板消息的缺陷
2. QuickReply介紹
3. QuickReply結構實作
4. 練習 - 引入多個QuickReply動作

n

用戶發送圖片消息

1. log觀看消息內容
2. 取回圖片，並存在本地端。

圖文選單製作

1. 拿設定檔去申請圖文選單
2. 把圖片傳給指定的選單id
3. 綁定用戶與圖文選單
4. 解除綁定
5. 刪除圖文選單

Push篇

主動推送消息給用戶

1. **push\_message**
2. **broadcast**

#### **narrowcast介紹**

#### **你們的未來 - multicast**

自主練習 - 主動推送影像地圖消息給用戶

AWS Simple Storage Service (S3): 資料儲存

EC2: 操作

Identity and Access Management (IAM): 權限管理服務

bucket=硬碟

object=檔案(影片、圖片、文件)

AWS

**概論**

**Iam 應用場景**

**Iam 核心名詞**

[**https://tibame.signin.aws.amazon.com/console**](https://tibame.signin.aws.amazon.com/console) IAM Login console

**帳:student**

**密:student**

****

**帳號建置階段**

**群組化管理aws帳號**

創建iam group

並選擇合適的Policy 附加至Group上。

**創建用戶，並加入群組內**

Password申請

Credential申請

**開發階段**

監控

Access advisor

Credential Report

CloudTrail

Credential補發

**S3(Simple Storage Service)**

應用場景介紹

無限空間的儲存服務

object→ bucket

boto3：S3與python client溝通用的套件

練習

<https://cxcxc-learning.signin.aws.amazon.com/console>

1.建置客戶端

2.上傳檔案

3.結合爬蟲上傳檔案

4.下載檔案

網頁操作S3

程式碼方式操作s3

客戶端建置

上傳

下載

刪除

產生credentials檔

1.安裝 awscli

2.執行 aws configure

**BB機器人佈屬至Heroku**

用Heroku建置個人對話機器人專題

1.開新python project

2.flask server 部署

3.確認在本地正常運作

4.依heroku要求修改

(a)自己的flask PORT號，加入app.run(host="0.0.0.0", port=os.environ['PORT'])

(b)一個自己的bot專用的requirements.txt套件設定檔

(C) 為heroku VM建立 Procfile (txt檔案)，內容寫入「web: python app.py」，指示herok 要執行的程式

5.安裝git

[Git - Downloads - Git SCM](https://git-scm.com/downloads)

(a)初次使用CMD使用Git時，要註冊使用者名稱與email

git config --global user.name "user name"

ex:

$ git config --global user.name "John Doe"

$ git config --global user.email johndoe@example.coml

6.安裝heroku command line並註冊Heroku帳號

[The Heroku CLI | Heroku Dev Center](https://devcenter.heroku.com/articles/heroku-cli)

7.打指令並上傳

(a)本機git登錄

(b) 到heroku上開專案

(c) heroku給的語句

參考說明頁面 [https://dashboard.heroku.com/apps/(你的使用者名稱)/deploy/heroku-git](https://dashboard.heroku.com/apps/tibame-benny/deploy/heroku-git)

1. 使用CMD輸入「heroku login」，跳出瀏灠器頁面後用自己的帳密登入，登入後CMD會產生連線成功的訊息
2. 配合Git指定上傳至Heroku的目錄，輸入指令「heroku git:clone -a (你的使用者名稱)」，

Ps: 只有第一次「才需要」初始化

1. git init
2. heroku git:remote -a (你的app名)
3. 進入放有要上傳程式的目錄，並在該目錄下開啟CMD，使用指令「cd (你的使用者名稱)」
4. 配合Git上傳要上傳的檔案

(1) git add . (「點」表示所有檔案，可輸入檔名上傳特定檔案)

(2) git commit -am "make it better"

(3) git push heroku master

8.用heroku的https網址存取

更新Heroku上的程式碼：

(一)先在本地主機測試程式碼可否正確運作：

1. 公版
2. 把LIne的網址正確貼入
3. Ngrok啟動

(二) 準備把更新的內容上傳至Heroku：

1. 更改內容為Heroku所需的設定 (1.requirement.txt追加套件 2.註解原來的app.run，使用Heroku版本的app.run)
2. 進入CMD利用Git指令上傳程式碼:

(1) git add . (「點」表示所有檔案，可輸入檔名上傳特定檔案)

(2) git commit -am "make it better"

(3) git push heroku master

執行完後，會取回app的執行連結 (ex: https://xxxxx.herokuapp.com/)，再貼回line的Webhook URL讓Line與Heroku連結

1. webhook更改

2020-7-22

----------------------------------------------

Docker 的應用場景介紹

VM ==> Docker

1. 動態資源條配
2. 瞬間開關

Docker

1. 安裝Docker
2. Download 映像檔
3. 啟用