

# Python期末報告 109403533 / 林采璇 109403048 / 林子恩

## 主題: Random Foodpanda Linebot 隨機點餐機(中央)

提醒老師:因為我們的程式有與line結合，如老師不方便操作，有附上單純爬蟲的程式碼，給老師參考，檔名為:python期末\_foodpanda爬蟲.ipynb

### 【創作動機】

生活在中央大學內，平日最煩惱的不是學業，而是今天要吃甚麼。宵夜街跟後門都吃膩了，又碰到疫情不想出門，這時我們突然想到為何不做個foodpanda推薦機器人呢？我們希望透過每日定時透過Line傳送推薦美食，讓我們從此不必煩惱要吃什麼，隨機點餐機都幫我們想好了！

### 【實作邏輯】

每日定時三餐時間，爬取foodpanda網站，接著random抽取其中店家，並播送抽取出來的店家至line訊息。

### 【作品介紹】

- part1\_爬取foodpanda網站與Random餐廳
  - 原先使用在課堂所教的bs4 及 selector 方法爬取但是被伺服器擋下來，如下圖
  - 透過檢查封包，從Headers 查看請求的方式與參數，以此爬取json格式檔案  
網址使用 :<https://disco.deliveryhero.io/listing/api/v1/pandora/vendors>

```
import requests
from bs4 import BeautifulSoup
res = requests.get("https://www.foodpanda.com.tw/restaurants/new?lat=24.9681558&lng=121.1952988&vertical=restaurants")

soup = BeautifulSoup(res.text,'html.parser')
print (soup.title.string)

Access to this page has been denied.
```

### 利用POST方法，並帶入參數如經緯度數值爬取資料

```
'name': '惡魔炸蛋蔥油餅',
'payment_types': [],
'post_code': '320011',
'primary_cuisine_id': 214,
'rating': 5.0,
'redirection_url': 'https://foodpanda.com.tw/',
'review_number': 4,
'review_with_comment_number': 0,
'score': 0.0,
'service_fee_percentage_amount': 0,
'service_tax_percentage_amount': 0,
'tag': 'NEXTGEN_NEW_TAG',
'tags': [
    {
        'code': 'NEW',
        'text': 'NEXTGEN_NEW_TAG'
    }
],
'url_key': 'e-mo-zha-dan-cong-you-bing',
'vat_percentage_amount': 0,
'characteristics': {
    'cuisines': [
        {
            'id': 214,
            'name': '小吃',
            'url_key': 'twsnacks',
            'main': True
        }
    ],
    'food_characteristics': [],
    'primary_cuisine': {
```

```
'has_delivery_provider': True,
'hero_image': 'https://images.deliveryhero.io/image/fd-tw/LH/zskw-hero.jpg',
'hero_listing_image': 'https://images.deliveryhero.io/image/fd-tw/LH/zskw-listing.jpg',
'is_new_until': '2022-05-01T00:00:00Z',
'premium_position': 0,
'latitude': 24.95486235,
'longitude': 121.2250732,
'loyalty_percentage_amount': 0.0,
'loyalty_program_enabled': False,
'maximum_express_order_amount': 0,
'metadata': {
    'has_discount': False,
    'timezone': 'Asia/Taipei',
    'close_reasons': [],
    'available_in': None,
    'events': [],
    'is_delivery_available': True,
    'is_pickup_available': True,
    'is_dine_in_available': False,
    'is_express_delivery_available': False,
    'is_temporary_closed': False,
    'is_flood_feature_closed': False
}
```

○ 程式碼截圖(程式碼於:/randomfoodlinebot/randomcrawl.py)

■ 爬蟲主程式(使用套件 requests)

```
1  # 爬中央大學附近的 foodpanda 美食
2  import requests
3  import random
4
5
6  class Randomcrawlrestaurant:
7      def __init__(self):
8          pass
9
10     def crawlrestaurant(self):
11         url = 'https://disco.deliveryhero.io/listing/api/v1/pandora/vendors'
12         query = {
13             'longitude': 121.1952988, # 中央大學經度
14             'latitude': 24.9681558, # 中央大學緯度
15             'language_id': 6,
16             'include': 'characteristics',
17             'dynamic_pricing': 0,
18             'configuration': 'Variant1',
19             'country': 'tw',
20             'budgets': '', # 低中高123
21             'cuisine': '',
22             'sort': 'rating_desc', # 評分最高
23             'food_characteristic': '',
24             'use_free_delivery_label': False,
25             'vertical': 'restaurants',
26             'limit': 50,
27             'offset': 0,
28             'customer_type': 'regular'
29         }
30         headers = {
31             'x-disco-client-id': 'web',
32         }
33         r = requests.get(url=url, params=query, headers=headers)
```

■ Random抽取結果(使用套件 random)

透過random 從50筆資料選出一筆，內容有餐廳名、評分及該餐廳在foodpanda的連結

```
33     r = requests.get(url=url, params=query, headers=headers)
34
35     if r.status_code == requests.codes.ok:
36         data = r.json()
37
38         restaurants = data['data']['items']
39
40         # for restaurant in restaurants[:50]:
41         #     print(restaurant['name'])
42         #     print(restaurant['redirection_url'])
43         #     print(restaurant['rating'])
44         #     print(
45         #         '-----')
46
47         # random
48         draw = random.choice(restaurants[:50])
49         content1 = "抽到" + draw['name'] + "\n" + "評價為" + \
50             str(draw['rating']) + "\n" + draw['redirection_url']
51         # print("抽到" + draw['name'])
52         # print("排名為" + str(draw['rating']))
53         # print(draw['redirection_url'])
54         return content1
55     else:
56         content2 = "請求失敗"
57         return content2
58         # print('請求失敗')
```

- part2\_line robot建置

總共可以區分成以下步驟：

#### A. 建立Line Provider與Messaging API channel:

至 Line Developer 建立官方帳號，並設置一個頻道類別為Messaging API的 Channel，設置完成後如下圖：

The screenshot shows the LINE Developers console interface. On the left, there's a sidebar with 'Console home', 'Providers' (which is highlighted in green), 'Search...', 'Admin' (with 'RandomFood' listed), 'Tools', and 'Support'. The main area is titled 'TOP' and has a section for 'Recently visited channels'. It shows a card for a channel named '隨機點餐機(中央)' with an 'Admin' badge, a thumbnail image of various food items, and a green 'Messaging API' icon.

#### B. LINE Bot應用程式

為求快速開發，因為Django是一個開放原始碼的Web應用框架且Django的框架已經定義了專案的基本架構及程式碼撰寫規範，故本次開發使用Django框架，利用Python來建置LINE Bot應用程式(APP)。

首先，如果將Django的框架建置完成，會有以下幾個程式，其功能分別如下：

##### 1) \_\_init\_\_.py

告訴Python, mylinebot是一個Package。

##### 2) asgi.py

全名為Asynchronous Server Gateway Interface,

用來提供非同步的功能。

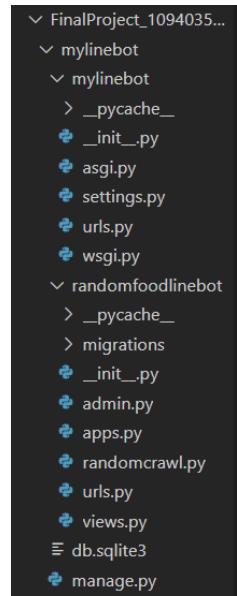
##### 3) settings.py

Django專案的設定檔。

##### 4) urls.py

定義Django專案中，各個應用程式(APP)的網址。

##### 5) wsgi.py



全名為Web Server Gateway Interface, 提供Django網站和伺服器間的標準介面。

#### 6) manage.py

用來管理整個Django專案, 像是啟動本地端伺服器、連接資料庫及建立應用程式(APP)等。

接著透過manage.py建立應用程式(APP), 建置完成, 會有以下幾個檔案, 其功能分別如下:

##### 1) migrations

記錄資料庫與models.py中的欄位同步歷程。

##### 2) \_\_init\_\_.py

顯示Django的每個應用程式(APP)。

##### 3) admin.py

定義或客製化這個應用程式(APP)在Django後台(Django Administration)的欄位顯示方式。

##### 4) apps.py

Django應用程式的設定檔。

##### 5) views.py

負責接收瀏覽器的請求, 進行邏輯的處理後, 回傳執行結果給瀏覽器。

## C. 安裝Ngrok

Ngrok 是一個轉發的伺服器, 他能夠把外界的請求轉發到指定的 Port。我們透過Ngrok, 將本機的 Port對外公開, 執行結果如下:

```
命令提示字元 - ngrok http 8000
ngrok
Hello World! https://ngrok.com/next-generation

Session Status          online
Account                s109483533@g.ncu.edu.tw (Plan: Free)
Version                3.0.4
Region                Japan (jp)
Latency               91ms
Web Interface          http://127.0.0.1:4040
Forwarding             https://20ab-180-177-9-164.jp.ngrok.io -> http://localhost:8000

Connections            ttl     opn     rt1     rt5     p50     p90
                         1       0      0.00    0.00   307.82   307.82

HTTP Requests
-----
POST /randomfoodlinebot/callback 200 OK
POST /randomfoodlinebot/callback 200 OK
```

#### D. 設定程式碼中LINE Bot憑證與程式碼執行時URL的設定

(程式碼於 :/mylinebot/settings.py)

將Ngrok 產生的網址填入ALLOWED\_HOSTS , LINE Bot憑證分別填入下方的  
LINE\_CHANNEL\_ACCESS\_TOKEN, 與LINE\_CHANNEL\_SECRET

```
28 v ALLOWED_HOSTS = [
29     '20ab-180-177-9-164.jp.ngrok.io' # 允許的網域名稱
30 ]
31
32 # Application definition
33
34 # Line Developer 的憑證設定 (TopSecret!)
35 LINE_CHANNEL_ACCESS_TOKEN = 'Cn...gJf...1763725906...N...5...BTA...E...L...U...L...G...9...'
36 LINE_CHANNEL_SECRET = 'L...E...L...F...C...L...O...A...L...C...P...E...5...'
```

#### E. LINE Webhook URL設定

按照Ngrok 產生的網址，填入Line Developer的管理介面

The screenshot shows the LINE Developers Management Interface. On the left, there's a sidebar with 'Console home', 'Providers' (which is expanded), 'Admin', 'RandomFood', 'Tools', and 'Support'. The main area is titled 'Webhook settings' for a 'RandomFood' provider. It shows a 'Webhook URL' field containing 'https://20ab-180-177-9-164.jp.ngrok.io/randomfoodlinebot/callback' with 'Verify' and 'Edit' buttons. A 'Use webhook' toggle switch is turned on. Below it, a 'Webhook redelivery' toggle switch is turned off.

- part3\_定時播送（程式碼於:/randomfoodlinebot/views.py）

```

50 def morningontimemessage():
51     getfoodontime = Randomcrawlrestaurant()
52     line_bot_api.broadcast(TextSendMessage(
53         text="早安!起床囉!" + '\n' + "這是每日早上7:30定時播送，現在正在測試中...敬請期待^^" + \
54             '\n' + getfoodontime.crawlrestaurant()))
55
56
57 def noonontimemessage():
58     getfoodontime = Randomcrawlrestaurant()
59     line_bot_api.broadcast(TextSendMessage(
60         text="午安!今天過一半了喔!" + '\n' + "這是每日中午11:30定時播送，現在正在測試中...敬請期待^^" + \
61             '\n' + getfoodontime.crawlrestaurant()))
62
63
64 def eveningontimemessage():
65     getfoodontime = Randomcrawlrestaurant()
66     line_bot_api.broadcast(TextSendMessage(
67         text="該吃晚餐囉!" + '\n' + "這是每日中午19:00定時播送，現在正在測試中...敬請期待^^" + \
68             '\n' + getfoodontime.crawlrestaurant()))
69
70
71 schedule.every().day.at("07:30").do(morningontimemessage)
72 schedule.every().day.at("11:30").do(noonontimemessage)
73 schedule.every().day.at("19:00").do(eveningontimemessage)
74 # 測試用
75 # schedule.every(30).seconds.do(ontimemessage)
76
77 while True:
78     schedule.run_pending()
79

```

### ➤ LINE API

line有提供API供程式開發者使用，我們使用了broadcast的API，能夠在任何時間推播訊息到所有的使用者，方便搭配定時播送的功能。

### ➤ 物件導向呼叫爬蟲程式

將爬蟲程式包裝，以物件導向的方式，在需要的時候程式碼的時候呼叫，主要語法程式碼如下：

```

from .randomcrawl import Randomcrawlrestaurant
getfoodontime = Randomcrawlrestaurant()
getfoodontime.crawlrestaurant()

```

### ➤ Schedule

為了定時播送，我們使用了python的schedule套件，語法程式碼如71~78行。因為schedule不是 Python 標準函式庫裡面的模組，在使用前，要先安裝，安裝指令如下：

```
$ pip install schedule
```

## 【執行結果】

- 因為我們設定7:30、11:30、19:00推播，故執行結果如下：

The screenshot shows a messaging interface with a header bar featuring a profile picture, a shield icon, the text '隨機點餐機(中央)', a volume icon, a search icon, and a menu icon. Below the header, there are three message bubbles, each containing a scheduled message from the bot.

**Message 1 (上午 7:30):**

早安! 起床囉!  
這是每日早上7:30定時播送, 現在正在測試中... 敬請期待^^  
抽到晨間廚房 (平鎮壢新店)  
評價為4.7  
<https://foodpanda.com.tw/restaurant/n2qe/chen-jian-chu-fang-ping-zhen-li-xin-dian>

**Message 2 (上午 11:30):**

午安! 今天過一半了喔!  
這是每日中午11:30定時播送, 現在正在測試中... 敬請期待^^  
抽到七盞茶 (中壢中華店)  
評價為4.8  
<https://foodpanda.com.tw/restaurant/r176/qi-zhan-cha-zhong-li-zhong-hua-dian>

**Message 3 (下午 7:00):**

該吃晚餐囉!  
這是每日中午19:00定時播送, 現在正在測試中... 敬請期待^^  
抽到熊好茶 (中壢民族店)  
評價為4.8  
<https://foodpanda.com.tw/restaurant/v492/xiong-hao-cha-zhong-li-min-zu-dian>

## 【未來展望】

- 使用line API 中的reply\_message API 接收訊息

(程式碼註解於 :/randomfoodlinebot/views.py)

```
18 # 未來可開發方向，依照傳入訊息回覆
19 # 可以讓使用者輸入所在地區、食物喜好種類等等
20
21 @csrf_exempt
22 def callback(request):
23     if request.method == 'POST':
24         signature = request.META['HTTP_X_LINE_SIGNATURE']
25         body = request.body.decode('utf-8')
26
27     try:
28         events = parser.parse(body, signature) # 傳入的事件
29     except InvalidSignatureError:
30         return HttpResponseForbidden()
31     except LineBotApiError:
32         return HttpResponseBadRequest()
33
34     for event in events:
35         if isinstance(event, MessageEvent): # 如果有訊息事件
36             getfood = Randomcrawlrestaurant()
37             line_bot_api.reply_message( # 回復傳入的訊息文字
38                 event.reply_token,
39                 TextSendMessage(text='仍在開發測試中...敬請期待^^' + \
40                                 '\n' + getfood.crawlrestaurant())
41
42             # TextSendMessage(text=event.message.text)
43         )
44     return HttpResponse()
45 else:
46     return HttpResponseBadRequest()
```

- 目前透過使用者端傳遞訊息並加以回復的功能尚未完成，或許未來可以加入傳入所在地址、想要食物類別，也期待之後有時間有想法可以加入新的功能。

今天

已讀  
下午 2:31

安安



仍在開發測試中...敬請期待^^  
抽到MUMU咖啡 中壢中山店  
評價為4.8  
<https://foodpanda.com.tw/restaurant/f7ol/mumuka-pei-zhong-li-zhong-shan-dian>

下午 2:31

## 【開發困境】

- 1) 在爬取過程中，使用課堂教學selector的爬蟲方法被伺服器擋下來(前面有提到)，後用參考資料中的方法解決，但foodpanda網站似乎有更新，格式與範例不盡相同。
- 2) 目前的程式，因為是定時播送，所以程式碼需要一直在主機上跑(一直開著程式)，可以考慮尋求代管資源。

## 【參考資料】

- foodpanda 爬蟲  
[https://blog.jiatoold.com/posts/foodpanda\\_spider/](https://blog.jiatoold.com/posts/foodpanda_spider/)
- line developer and account manager  
<https://developers.line.biz/zh-hant/>  
<https://tw.linebiz.com/login/>
- line robot 建置  
<https://www.learncodewithmike.com/2020/03/django-install.html>  
<https://www.learncodewithmike.com/2020/03/django-create-app.html>  
<https://www.learncodewithmike.com/2020/06/python-line-bot.html>
- 定時播送  
python schedule  
<https://www.readfog.com/a/1648168828734115840>  
<https://pypi.org/project/schedule/>  
line api overview  
<https://github.com/line/line-bot-sdk-python>  
<https://developers.line.biz/en/docs/messaging-api/overview/>  
line broadcast api  
<https://developers.line.biz/en/reference/messaging-api/#send-broadcast-message>  
line reply message api  
<https://developers.line.biz/en/reference/messaging-api/#send-reply-message>