

程式設計 (一)

Line Bot 教學

Ming-Hung Wang 王銘宏

tonymhwang@cs.ccu.edu.tw

Department of Computer Science and Information Engineering
National Chung Cheng University

Fall Semester, 2021

本章目錄

1. 環境建置
2. Github
3. Heroku
4. Line Developers
5. Connect
6. Code

環境建置

利用 python 建置 lineBot，需擁有放置程式碼的地方及部署的位置。
本教學介紹如何使用 python 來建置一個簡易的 Line 機器人。
會用到的網站資源如下，均可免費註冊。

- Github
- Heroku
- Line Developers
- Connect

Github

Github

Github 是一個可以在上面存放 建立和分享程式碼的平台。也提供下載 code 的功能。請先至網站免費註冊一個帳號。

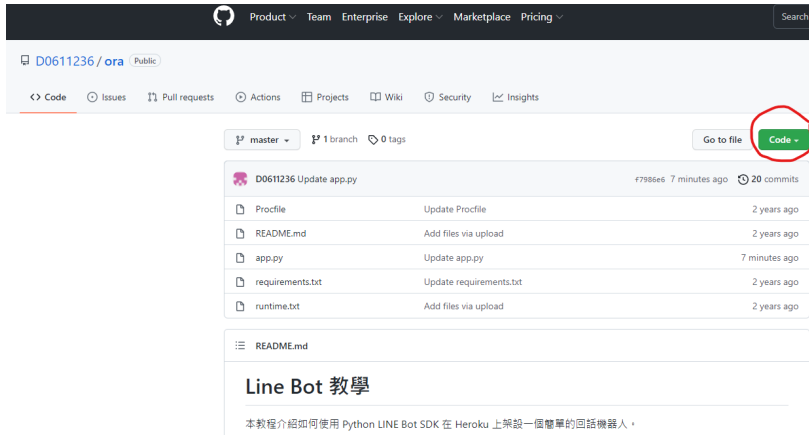
<https://github.com/>



在建立 lineBot 前，可在 Github 上面，抓取一個簡易的 sample。
在此提供一個簡單的 sample 下載，裡面包含使用的語言及建置環境。

<https://github.com/D0611236/Yu>

點入該網址後，會出現以下畫面。點擊右邊綠色方塊 [Code]。



The screenshot shows the GitHub interface for the repository `D0611236/ora`. The repository is public. The navigation bar includes links for Code, Issues, Pull requests, Actions, Projects, Wiki, Security, and Insights. The repository has 1 branch and 0 tags. A red circle highlights the green 'Code' button. Below the repository information, a table lists the files in the repository:

File	Description	Time
Profile	Update Profile	2 years ago
README.md	Add files via upload	2 years ago
app.py	Update app.py	7 minutes ago
requirements.txt	Update requirements.txt	2 years ago
runtime.txt	Add files via upload	2 years ago

Below the file list, the README.md file is expanded, showing the title 'Line Bot 教學' and the description: '本教程介紹如何使用 Python LINE Bot SDK 在 Heroku 上架設一個簡單的回話機器人。'

出現選單，選擇 Download，並解壓縮。

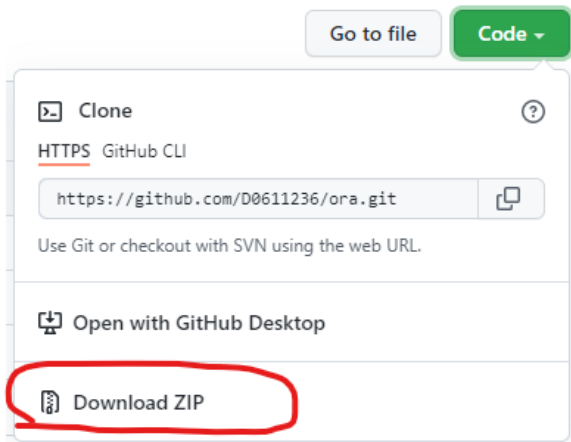
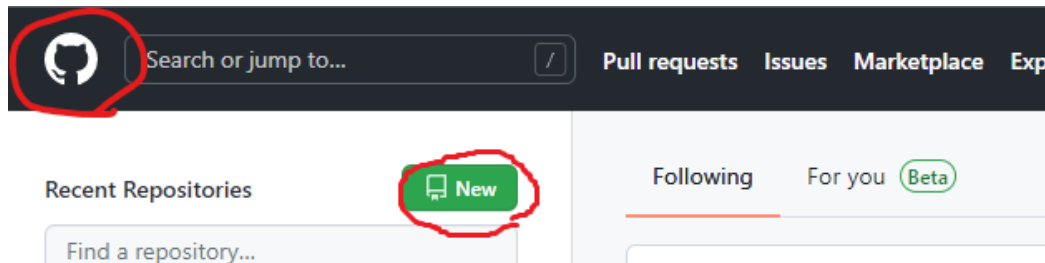




图: Caption

接下來，示範如何在 Github 建立專案。點選左上角回到主頁面，並點選 [New]。




輸入專案名稱，並建立。


Owner * Repository name *

 D0611236 / Yu 

Great repository names are short, unique, and descriptive. Your new repository will be created as Yu-...on? How about [solid-meme](#)?

Description (optional)

☒  **Public**
Anyone on the internet can see this repository. You choose who can commit.

☐  **Private**
You choose who can see and commit to this repository.

Initialize this repository with:

Skip this step if you're importing an existing repository.

☐ **Add a README file**
This is where you can write a long description for your project. [Learn more.](#)

☐ **Add .gitignore**
Choose which files not to track from a list of templates. [Learn more.](#)

☐ **Choose a license**
A license tells others what they can and can't do with your code. [Learn more.](#)

[Create repository](#)

選擇建立好的專案，並選擇上傳，將剛剛的 sample 全數丟上去。

D0611236 / Yu Public

<> Code Issues Pull requests Actions Projects Wiki Security Insights Settings

Quick setup — if you've done this kind of thing before

Set up in Desktop or HTTPS SSH `https://github.com/D0611236/Yu.git`

Get started by [creating a new file](#) or [uploading an existing file](#). We recommend every repository include a [README](#), [LICENSE](#), and [.gitignore](#).

...or create a new repository on the command line

```
echo "# Yu" >> README.md
git init
git add README.md
git commit -m "first commit"
git branch -M main
git remote add origin https://github.com/D0611236/Yu.git
git push -u origin main
```

...or push an existing repository from the command line

```
git remote add origin https://github.com/D0611236/Yu.git
git branch -M main
git push -u origin main
```

...or import code from another repository

You can initialize this repository with code from a Subversion, Mercurial, or TFS project.

Import code

D0611236 / Yu Public

[Code](#) [Issues](#) [Pull requests](#) [Actions](#) [Projects](#) [Wiki](#) [Security](#) [Insights](#) [Settings](#)






 **main**  **1 branch**  **0 tags**


[Go to file](#)

[Add file](#) 

[Code](#) 

 **D0611236** Add files via upload 6e2e1e2 now  **1 commit**

 Procfile	Add files via upload	now
 README.md	Add files via upload	now
 app.py	Add files via upload	now
 requirements.txt	Add files via upload	now
 runtime.txt	Add files via upload	now

 **README.md**

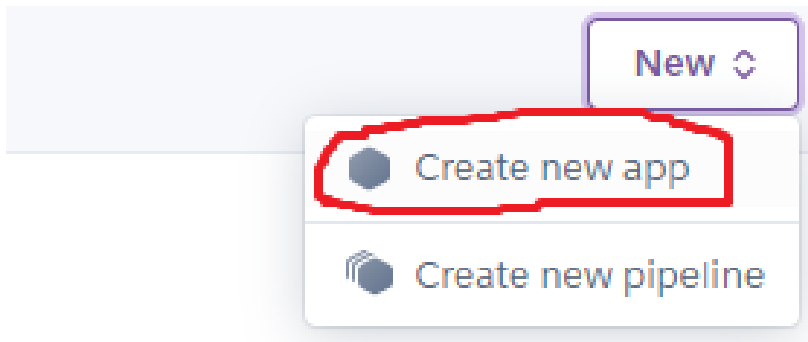


Heroku

Heroku 為一個提供放置網站/伺服器服務的免費平台，且支援多種語言。到以下網址免費註冊一個帳戶。

<https://www.heroku.com/>

建立好帳戶之後，進入主頁面，點選 [New]->[Create New App]。



輸入自己想要的名稱，並建立。

Create New App

App name

yyuuuu



yyuuuu is available

Choose a region



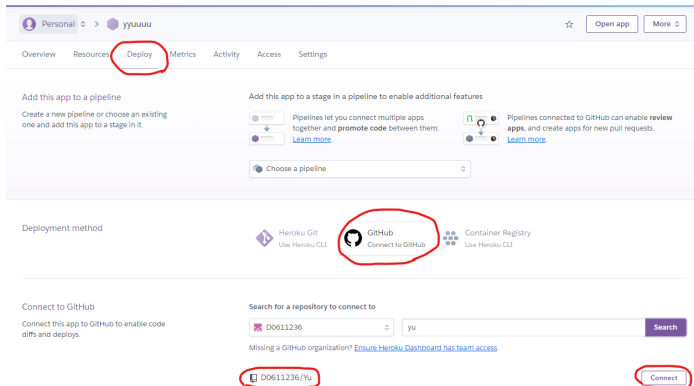
United States



Add to pipeline...

Create app

點選剛建立的專案，並選擇 [Deploy]。
之後再選擇 [GitHub] 之後再找到自己 Github 的專案，並點擊
[Connect]。



再切至 [Settings]，點選 [Add buildpack]，選擇開發用的語言/工具。

Personal > yyuuuu

GitHub 08611236/Yu

Overview Resources Deploy Metrics Activity Access **Settings**

App Information

App Name

Add Buildpack

Enter Buildpack URL

Enter buildpack URL (e.g. heroku/nodejs or https://github.com/heroku/heroku-buildpack-ruby)

Or select from our officially supported buildpacks

nodejs	python	php	ruby	java
go	gradle	scala	clojure	

Save changes

Config Vars

Config vars change the way your app behaves. In addition to creating your own, some add-ons come with their own.

Buildpacks

Buildpacks are scripts that are run when your app is deployed. They are used to install

Add buildpack

Line Developers

Line Developer 提供各式開發的工具以及相關開發文件。

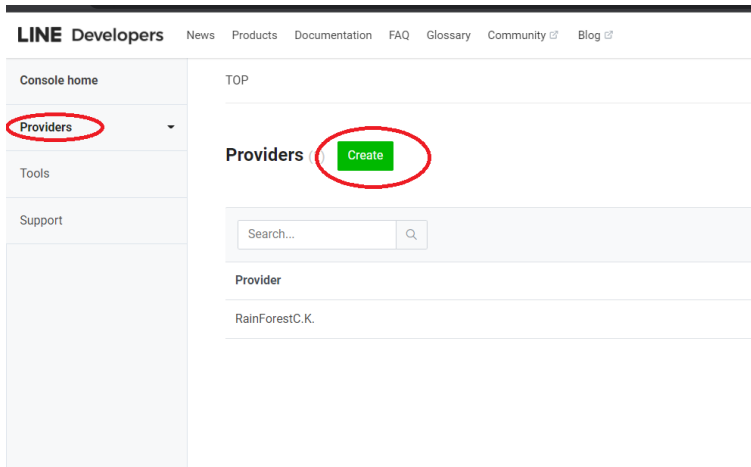
進入此網站開始建立屬於自己的 Linebot。

(用自己的 Line 帳號登入即可，不需要申請商用帳號)

<https://developers.line.biz/zh-hant/>

Line Developers

登入後，選擇 [Providers]，並選擇 [Create]，創建一個 Provider。



之後選擇剛建立的 Provider，並選擇 [Create a Messaging API channel]。

yun

Channels

Roles

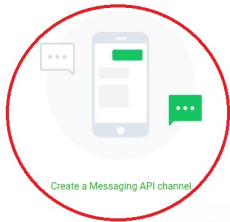
Settings

This provider doesn't have any channels yet

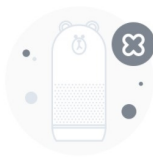
To create one, choose a channel type below



Create a LINE Login channel



Create a Messaging API channel



Create a CLOVA Skill channel



Create a Blockchain Service channel

Line Developers

選擇相應的類別 地區及依自己喜好更換圖片 (圖片也留空)。

Create a new channel

Channel type

Messaging API

選擇 channel type

✓ Don't leave this empty

Provider

yun

✓ Don't leave this empty

Company or
owner's country or
region ⓘ

Taiwan

選擇所在地區

Companies should select their company's country or region. Individuals should

✓ Don't leave this empty

Channel icon

optional

Register

點此可更換機器人圖像

✓ File type must be one of: PNG,JPG,JPEG,GIF,BMP

✓ File must be no larger than 3 MB

填入機器人名稱 描述 類別及信箱相關資訊。

Channel name	ID好難想	機器人命名
<small>Note: The channel name can't be changed for seven days.</small>		
<small>✓ Don't leave this empty</small>		
<small>✓ Don't use special characters (4-byte Unicode)</small>		
<small>✓ Enter no more than 20 characters</small>		

Channel description	隨便打就好	機器人描述
<small>✓ Don't leave this empty</small>		
<small>✓ Don't use special characters (4-byte Unicode)</small>		
<small>✓ Enter no more than 500 characters</small>		

Category	音樂	機器人類別
<small>✓ Don't leave this empty</small>		

Subcategory	經紀、娛樂公司	機器人子類別
<small>✓ Don't leave this empty</small>		

Email address ①	@gmail.com	個人信箱
<small>✓ Don't leave this empty</small>		
<small>✓ Enter a valid email address</small>		
<small>✓ Enter no more than 100 characters</small>		

勾選同意條款，並點選 [Create]

Privacy policy URL

optional

Enter privacy policy URL

- ✓ Enter a valid HTTPS URL
- ✓ Enter no more than 500 characters

Terms of use URL

optional

Enter terms of use URL

- ✓ Enter a valid HTTPS URL
- ✓ Enter no more than 500 characters

☒ I have read and agree to the [LINE Official Account Terms of Use](#) 

☒ I have read and agree to the [LINE Official Account API Terms of Use](#) 

✓ Select the checkbox after reading the related document

Create

進入剛創立的機器人，並切換至 [Messaging API]。

TOP > yun > ID好難想 > Messaging API



ID好難想

Admin

Messaging API

Basic settings

Messaging API

LIFF

Security

Statistics

Roles

將 Greeting 與自動回復功能關閉。(之後可視各自需求開啟)

TOP > yun > ID好難想 > Messaging API

Use webhook ⓘ



LINE Official Account features

Edit the message text and other settings for these features in the LINE Official

Allow bot to join
group chats ⓘ Disabled

Auto-reply
messages ⓘ

Disabled

Greeting messages ⓘ

Disabled

下滑找到 [Channel access token]，點擊 [Issue] 產生 token。

TOP > yun > ID好難想 > Messaging API



Channel access token

Channel access token (long-lived) ?

Issue

點選Issue

Channel access token (long-lived) ? 點選之後會產生一堆代碼

nYao7F31TfQDIPrVEiYQz00ECU3ETwal6CsaPl9DEwS5lslimgUEPWSeOxcxl10a+
Ekk3LXZ0vN27vddAdB04t89/10/w1c



Connect

本節將介紹如何串聯程式碼與機器人。

先前在 Heroku 的章節，已將 Heroku 與 Github 連結。在此，要再將 Github 的程式碼與 Heroku 應用至創建的 LineBot 上。

至 Line Developers，並切換至 [Messaging API]，找到 [Webhook settings]->[Webhook URL]，點選 [Edit]，填寫 https://你在 Heroku 的 App 名稱.herokuapp.com/callback。並開啟 [Use webhook]。

TOP > yun > ID好難想 > Messaging API

Webhook settings

你在Heroku的App名稱

Webhook URL ?

https://yyyyuuu.herokuapp.com/callback

Verify

Edit

Use webhook ?



至 Line Developers，複製 [Channel access token] 及 [Channel secret]。

TOP > yun > ID好難想 > Basic settings

Channel secret ⓘ

041b97080b541192476e8735



Channel access token (long-lived) ⓘ

nYao7F31TfQDIPrVEiYQz00ECU3ETwal6CsaPl9DEwS5lslimgUEPWSeOxcx11Oa+Ekk3LXZ0vN27vddAdB04t89/1O/w1c



前往 Github 將剛剛的 [Channel access token] 及 [Channel secret]，放置 app.py 裡的對應位置。

Yu /

app.py

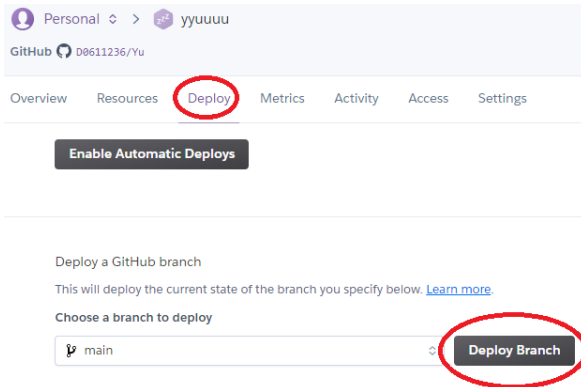
in main

<> Edit file

Preview changes

```
17 # Channel Access Token
18 line_bot_api = LineBotApi('你的Channel access token')
19 #or line_bot_api = 'Channel_token'
20
21 # Channel Secret
22 handler = WebhookHandler('你的Channel secret')
23 #or handler = 'Channel_secret'
24
```

之後前往 Heroku，進入你的 APP，之後點選 [Deploy]，點擊 [Deploy Branch]，每次更新完 Github 上的 app.py 之後，都須點擊 [Deploy Branch]，才能完成同步更新。



可在 [Activity] 查看是否成功完成同步更新，若失敗可點擊 [View build log]，查找出錯點。



The screenshot displays the GitHub Connect user interface. At the top, the breadcrumb navigation shows 'Personal' followed by a dropdown arrow, a right-pointing chevron, a repository icon, and the name 'yyuuuu'. Below this, the GitHub logo and the identifier 'D0611236/Yu' are visible. A horizontal menu contains several tabs: 'Overview', 'Resources', 'Deploy', 'Metrics', 'Activity', 'Access', and 'Settings'. The 'Activity' tab is highlighted with a red circle. Below the menu, the 'Activity Feed' section shows a recent event. On the left of the event is a hammer icon with a red 'x' inside a circle. To its right is a circular icon containing a person. The event text reads '[redacted]@gmail.com: Build failed' in red, followed by the Chinese characters '失敗' in red. Below the event text, the timestamp 'Mar 30 at 9:45 PM' is shown, followed by a blue link 'View build log' which is also circled in red.

Personal ▾ > yyuuuu

GitHub D0611236/Yu

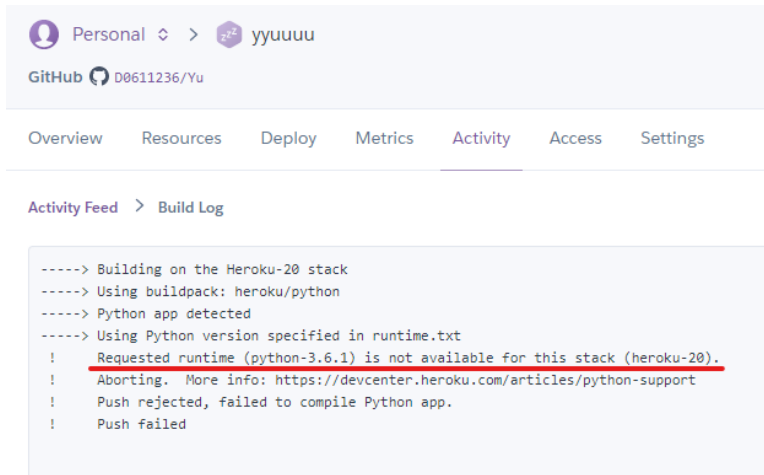
Overview Resources Deploy Metrics **Activity** Access Settings

Activity Feed

  [redacted]@gmail.com: Build failed 失敗

Mar 30 at 9:45 PM [View build log](#)

比如，此處顯示的 bug 為當前的 Requested runtime 無法適用。

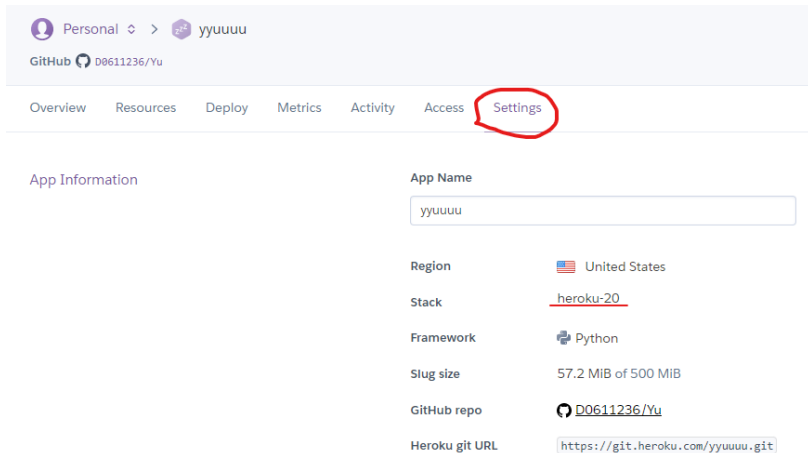


The screenshot shows the Heroku web interface for a personal application named 'yyuuuu'. The 'Activity' tab is selected, displaying the 'Build Log'. The log contains the following text:




```
-----> Building on the Heroku-20 stack
-----> Using buildpack: heroku/python
-----> Python app detected
-----> Using Python version specified in runtime.txt
!   Requested runtime (python-3.6.1) is not available for this stack (heroku-20).
!   Aborting. More info: https://devcenter.heroku.com/articles/python-support
!   Push rejected, failed to compile Python app.
!   Push failed
```

The error message is underlined in red in the original image.

點選 [Settings]，查看當前版本。(此處以 **heroku-20** 為例)

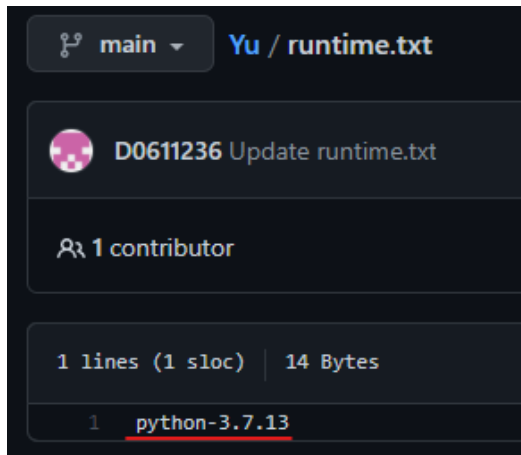


The screenshot shows the Heroku dashboard for a user named 'yyuuuu'. The 'Settings' tab is selected and highlighted with a red circle. Below the navigation bar, the 'App Information' section is visible. It contains a table with the following details:



App Name	yyuuuu
Region	 United States
Stack	<u>heroku-20</u>
Framework	 Python
Slug size	57.2 MiB of 500 MiB
GitHub repo	 <u>D0611236/Yu</u>
Heroku git URL	<code>https://git.heroku.com/yyuuuu.git</code>


Connect

前往 Github，將 runtime.txt，修改到對應的版本。(此處以 **heroku-20** 為例，**python-3.7.13** 可適用於 **heroku-20**)



修改完畢後，再次點擊 [Deploy Branch]，並至 [Activity] 查看是否成功完成同步更新。

 Personal ▾ >  yyuuuu

GitHub  D0611236/Yu

Overview

Resources

Deploy



Metrics

Activity

Access

Settings

Activity Feed

  ricky870921@gmail.com: Build succeeded 成功
Mar 30 at 9:49 PM · [View build log](#)

Code

本章節，主要描述如何藉由修改 `app.py`，來達到特定使用者的內容回復。

注意：更新完程式碼，均需再次點擊 [Deploy Branch]。

`event.message.text` 為使用者傳送的訊息，我們可以用 `get = event.message.text`，將訊息接收至 `get` 裡。

宣告一個 `message`，當作等等須回傳的資訊。

後面的 `TextSendMessage`，則表示回傳的資訊是文字。先使用 `message = TextSendMessage(text = get)`，讓回傳的資訊等於用戶傳送的文字訊息。

最後 `line_bot_api.reply_message(event.reply_token, message)`，則是負責讓機器人送出訊息。

```
def handle_message(event):  
    get = event.message.text  
    #event.gessage.text接收使用者文字訊息
```

```
    message = TextSendMessage(text = get)
```

```
    line_bot_api.reply_message(event.reply_token, message)
```



接下來可以根據個人需求，調整為只有接收到特定文字才會回復。

訊息回復的樣式不單僅限於文字，也可使用圖片 貼圖甚至 url，以下提供幾個常用的訊息格式。詳細內容可參照官方的程式範例。

回復常用格式

TextSendMessage	傳送文字訊息
ImageSendMessage	傳送圖片訊息
StickerSendMessage	傳送貼圖訊息
TemplateSendMessage	傳送模板樣式訊息 (此類含有多種類別)

TemplateSendMessage 的類別，含有許多細項，其中常用的包含：

- ButtonsTemplate
 - 回傳 ButtonsTemplate 訊息
- CarouselTemplate
 - 回傳 CarouselColumnTemplate 訊息 (由多個 ButtonsTemplate 組成)
- ImageCarouselTemplate
 - 回傳 ImageCarouselTemplate 訊息

TextSendMessage 文字訊息範例



有人要吃橘子嗎 (=° ω °) つ 🍊

2:36 PM

Read
2:36 PM

如何

ImageSendMessage 圖片訊息範例



StickerSendMessage 貼圖訊息範例



2:51 PM

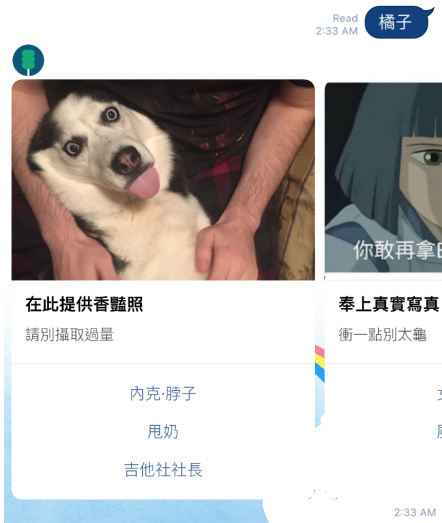
Read
2:51 PM

Ok

TemplateSendMessage 模板樣式訊息範例-ButtonsTemplate



TemplateSendMessage 模板樣式訊息範例-CarouselTemplate



TemplateSendMessage 模板樣式訊息範例-ImageCarouselTemplate

