

# Rabboni with python

## Python 安裝與前置作業

1. 請先至 Python 官方網站下載最新版 python(windows)  
<https://www.python.org/downloads/>
2. 使用 python 安裝 rabboni 專屬的 library
  - i. 使用系統管理員開啟命令提示字元(CMD)
  - ii. 輸入: `pip install rabboni==1.73` (目前最新版本 1.73)

```
系統管理員: 命令提示字元
Microsoft Windows [版本 10.0.17134.765]
(c) 2018 Microsoft Corporation. 著作權所有, 並保留一切權利。
C:\WINDOWS\system32>pip install rabboni==0.7
Collecting rabboni==0.7
  Downloading https://files.pythonhosted.org/packages/d1/53/2df27a57cddf84cd061fb7445dfef4bca438aa015ac9d61a439a
Requirement already satisfied: pygatt in c:\program files (x86)\python37-32\lib\site-packages (from rabboni==0.7)
Requirement already satisfied: matplotlib in c:\program files (x86)\python37-32\lib\site-packages (from rabboni=
Requirement already satisfied: pywinusb in c:\program files (x86)\python37-32\lib\site-packages (from rabboni=0
Requirement already satisfied: pyserial in c:\program files (x86)\python37-32\lib\site-packages (from pygatt->ra
Requirement already satisfied: enum-compat in c:\program files (x86)\python37-32\lib\site-packages (from pygatt-
Requirement already satisfied: numpy>=1.10.0 in c:\program files (x86)\python37-32\lib\site-packages (from matpl
Requirement already satisfied: kiwisolver>=1.0.1 in c:\program files (x86)\python37-32\lib\site-packages (from m
Requirement already satisfied: cyclopy>=0.10 in c:\program files (x86)\python37-32\lib\site-packages (from matplo
Requirement already satisfied: pyparsing!=2.0.4,!=2.1.2,!=2.1.6,>=2.0.1 in c:\program files (x86)\python37-32\li
Requirement already satisfied: python-dateutil>=2.1 in c:\program files (x86)\python37-32\lib\site-packages (fro
Requirement already satisfied: setuptools in c:\program files (x86)\python37-32\lib\site-packages (from kiwisolv
Requirement already satisfied: six in c:\users\sui\appdata\roaming\python\python37\site-packages (from cyclopy>=0
Installing collected packages: rabboni
Successfully installed rabboni-0.7
You are using pip version 19.0.3, however version 19.1.1 is available.
You should consider upgrading via the 'python -m pip install --upgrade pip' command.
C:\WINDOWS\system32>
```

## Rabboni 介紹



1. 內含 6 軸 sensor, 3-axis 加速度與 3-axis 陀螺儀
  - i. 規格: ICM-20689
  - ii. 可使用 Python 藍芽與 USB API 將值讀出來
2. 內置 counter 計數功能
  - i. 可使用 Python 藍芽與 USB API 將值讀出來
  - ii. 可使用 Python 藍芽與 USB API 將 counter 歸零
3. 內建兩種連接模式
  - i. **藍芽模式:**
    1. 將 dongle 插上電腦 USB
    2. 開啟 rabboni, 按下藍芽廣播鍵
    3. 執行 python 藍芽相關 code(如 example code)
  - ii. **USB 模式:**
    1. 將 rabboni 使用 USB 連上電腦
    2. 執行 python USB 相關 code(如 example code)



## Rabboni 藍芽連接 — Python API Example code

1. 以下為藍芽的 example code，會做到以下結果
  - i. 掃描藍芽裝置並連上裝置(會自動檢查裝置是否為 Rabboni)
  - ii. 將連上後的六軸資料印出來
  - iii. 依照設定的條件重置 counter
  - iv. 將結果儲存成圖檔與 csv 檔
  - v. 斷開連接
2. 相關 API 請看下一頁 “Rabboni 藍芽連接 — python API”

```
# -*- coding: UTF-8 -*-
from rabboni import *

rabbo = Rabboni(mode = BLE) #先宣告一個物件
rabbo.scan() #掃描所有藍芽 Device
rabbo.print_device() # 列出所有藍芽 Device
rabbo.connect("D1:FA:F0:F2:12:29")#依照 MAC 連接
rabbo.discover_characteristics()#掃描所有服務 可略過
rabbo.print_char()#列出所有服務 可略過
# print (rabbo.characteristics)
# print (rabbo.Status)

rabbo.set_sensor_scale(acc_scale = 16, gyr_scale = 2000) ## 設定加速度跟陀螺儀的最大範圍 建議
先設定完離線在進行檔案錄製

rabbo.read_data()#讀取資料 必跑

try:
    while True:#一直打印資料 直到結束程式
        rabbo.print_data()#print 資料
        if rabbo.Cur_Cnt == 100:
            rabbo.rst_count(mode = "Both")
except KeyboardInterrupt:#結束程式
    print('Shut done!')
    print (rabbo.Accx_list)#印出到結束程式時的所有 Accx 值
    rabbo.stop()#停止 dongle
    rabbo.write_csv(data = rabbo.Accx_list,file_name ="AccX")#將 Accx 寫出 csv 檔
    rabbo.plot_pic(data = rabbo.Accx_list,file_name = "AccX",show = True)#將 Accx 畫出圖案並存檔
```

## Rabboni 藍芽連接 — python API

前置 API		
API 名稱	解釋	Return 值
Rabboni(mode = "BLE")	Class 名稱 一開始先宣告一個物件給它 並宣告模式	None
Rabboni.scan()	掃描附近藍芽 device	所有 device 的名稱與相關資訊
Rabboni.print_device()	印出所有	None
Rabboni.connect(MAC)	填入 MAC，連接 device	Device
Rabboni.discover_characteristics()	抓出 device 所有的服務	None
Rabboni.print_char()	列出所有服務(需先 discover)	None
Rabboni.disconnect()	斷開聯絡	None
Rabboni.stop()	關掉 dongle，建議最後都要關掉，以免下次開啟出錯	None
操作、讀取 API		
API 名稱	解釋	參數解釋
Rabboni.read_data()	讀取所有 sensor 資料，必須先跑此行，所有的參數才會出來	None
Rabboni.print_data()	列出 read_data()所讀到的資料 Acc_x,y,z Gyr_x,y,z Count	None
Rabboni.rst_count(mode)	重置裝置紀錄的 count	mode: "Both","Store_cnt","Cur_Cnt" 將不同的 count 進行 reset，Default 為 Both
Rabboni.set_sensor_scale(acc_scale,gyr_scale)	決定 sensor 量取範圍(+/-範圍)	acc_scale:必填,2,4,8,16 gyr_scale:必填, 250,500,100,2000 建議設定與錄製分開進行，設定完後離線再進行錄製。
Rabboni.write_csv(data,file_name)	將 data(list)傳進去寫出 csv 檔案，並以 file_name 命名	data:必填，資料 file_name:必填，檔名

Rabboni.plot_pic(data,file_name,show)	將 data(list)傳進去畫出圖片，並畫圖存起來	data:必填，資料 file_name:選填，檔名，若沒傳入值則不會存檔 show：預設為 True 會顯示圖片，False 則不會顯示圖片
參數 API		
API 名稱	解釋	Return 值
Rabboni.Status	是否連接上	0:disconnected 1:connected
Rabboni.Hex_data	當下 sensor data 的 16 進位值	string
Rabboni.Accx	當下 Accx 的值	float
Rabboni.Accy	當下 Accy 的值	float
Rabboni.Accz	當下 Accz 的值	float
Rabboni.Gyrx	當下 Gyrx 的值	float
Rabboni.Gyry	當下 Gyry 的值	float
Rabboni.Gyrz	當下 Gyrz 的值	float
Rabboni.Cur_Cnt	當下 Cur_Cnt 的值	int
Rabboni.Store_Cnt	裝置中紀錄的 Store_Cnt	Int
Rabboni.Acc_char	目前 Acc (量測)的範圍	int
Rabboni.Gyr_char	目前 Gyro (量測)的範圍	int
Rabboni.Accx_list	紀錄連接後直到結束的 Accx	list
Rabboni.Accy_list	紀錄連接後直到結束的 Accy	list
Rabboni.Accz_list	紀錄連接後直到結束的 Accz	list
Rabboni.Gyrx_list	紀錄連接後直到結束的 Gyrx	list
Rabboni.Gyry_list	紀錄連接後直到結束的 Gyry	list
Rabboni.Gyrz_list	紀錄連接後直到結束的 Gyrz	list
Rabboni.Cnt_list	紀錄連接後直到結束的 Cur_cnt	list

## Rabboni USB 連接 — Python API Example code

1. 以下為 USB 的 example code，會做到以下結果
  - i. 連上裝置
  - ii. 將連上後的六軸資料印出來
  - iii. 依照設定的條件重置 counter
  - iv. 斷開連接
2. 相關 API 請看下一頁 “Rabboni USB 連接 — python API”

```
from rabboni import *

rabbo = Rabboni(mode = "USB") #先宣告一個物件

rabbo.connect()#連結上 rabboni，若沒插上會報錯

print ("Status:",rabbo.Status)
rabbo.set_sensor_scale(acc_scale = 4, gyr_scale = 2000) ## 設定加速度跟陀螺儀的最大範圍 建議先設定完離線在進行檔案錄製
try:
    rabbo.read_data()
    while True:#一直打印資料 直到結束程式
        rabbo.print_data()#print 資料
        print (rabbo.data_num)
        if rabbo.Cur_Cnt > 10:
            rabbo.rst_count() #重置 count 會 delay 一下
        if rabbo.data_num>100:
            rabbo.stop()#停止運作
            break

except KeyboardInterrupt:#結束程式
    print('Shut done!')
    # print (rabbo.Accx_list)#印出到結束程式時的所有 Accx 值
    rabbo.stop()#停止運作
```

## Rabboni USB 連接 — python API

前置 API		
API 名稱	解釋	Return 值
Rabboni(mode = "USB")	Class 名稱 一開始先宣告一個物件給它 並宣告模式	None
Rabboni.connect()	透過 USB 連接 device，如果抓不到裝置會報錯	None
Rabboni.disconnect()	斷開聯絡，關掉 USB	None
Rabboni.stop()	斷開聯絡，關掉 USB	None
操作、讀取 API		
API 名稱	解釋	參數解釋
Rabboni.read_data()	讀取所有 sensor 資料，必須先跑此行，所有的參數才會出來	None
Rabboni.print_data()	列出 read_data()所讀到的資料 Acc_x,y,z Gyr_x,y,z Count	None
Rabboni.rst_count(mode)	重置裝置紀錄的 count	mode: "Both","Store_cnt","Cur_Cnt" 將不同的 count 進行 reset，Default 為 Both
Rabboni.set_sensor_scale(acc_scale,gyr_scale)	決定 sensor 量取範圍(+範圍)	acc_scale:必填,2,4,8,16 gyr_scale:必填,250,500,100,2000 建議設定與錄製分開進行，設定完後離線再進行錄製。
Rabboni.write_csv(data,file_name)	將 data(list)傳進去寫出 csv 檔案，並以 file_name 命名	data:必填，資料 file_name:必填，檔名
Rabboni.plot_pic(data,file_name,show)	將 data(list)傳進去畫出圖片，並畫圖存起來	data:必填，資料 file_name:選填，檔名，若沒傳入值則不會存檔 show：預設為 True 會顯示圖片，False 則不會顯示圖片

參數 API		
API 名稱	解釋	Return 值
Rabboni.Status	是否連接上	0:disconnected 1:connected
Rabboni.Hex_data	當下 sensor data 的 16 進位值	string
Rabboni.Accx	當下 Accx 的值	float
Rabboni.Accy	當下 Accy 的值	float
Rabboni.Accz	當下 Accz 的值	float
Rabboni.Gyrx	當下 Gyrx 的值	float
Rabboni.Gyry	當下 Gyry 的值	float
Rabboni.Gyrz	當下 Gyrz 的值	float
Rabboni.Cur_Cnt	當下 Cur_Cnt 的值	int
Rabboni.Store_Cnt	裝置中紀錄的 Store_Cnt	Int
Rabboni.Acc_char	目前 Acc (量測)的範圍	int
Rabboni.Gyr_char	目前 Gyro (量測)的範圍	int
Rabboni.Accx_list	紀錄連接後直到結束的 Accx	list
Rabboni.Accy_list	紀錄連接後直到結束的 Accy	list
Rabboni.Accz_list	紀錄連接後直到結束的 Accz	list
Rabboni.Gyrx_list	紀錄連接後直到結束的 Gyrx	list
Rabboni.Gyry_list	紀錄連接後直到結束的 Gyry	list
Rabboni.Gyrz_list	紀錄連接後直到結束的 Gyrz	list
Rabboni.Cnt_list	紀錄連接後直到結束的 Cur_Cnt	list
Rabboni.data_num	上述的 list 總共有幾筆 (單位為 sampling 數)	int