

# Auto-py-to-exe使用説明



# 相關連結

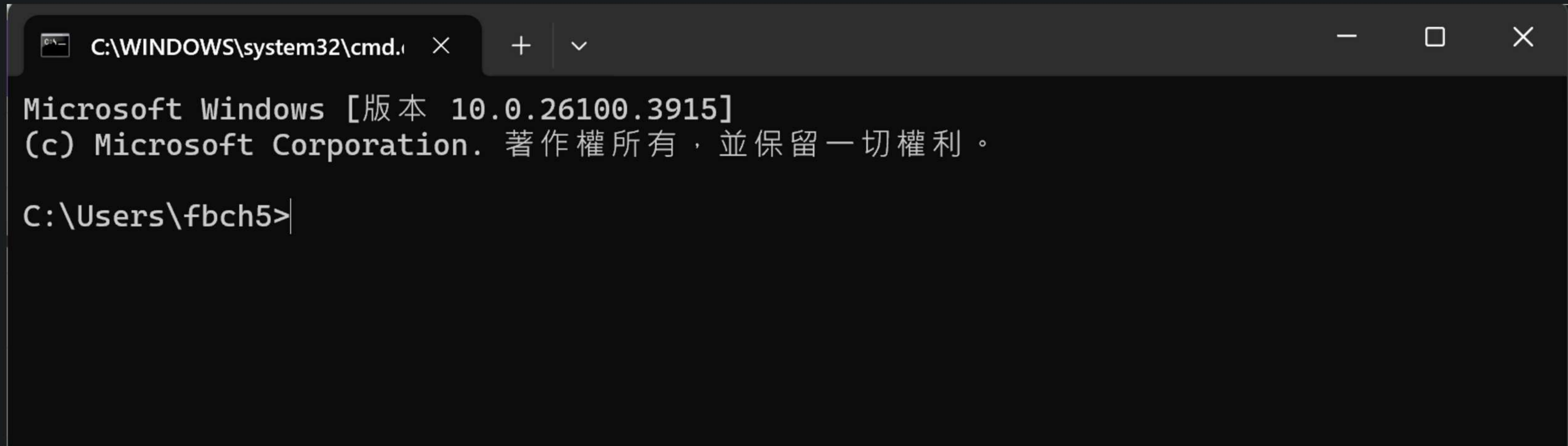
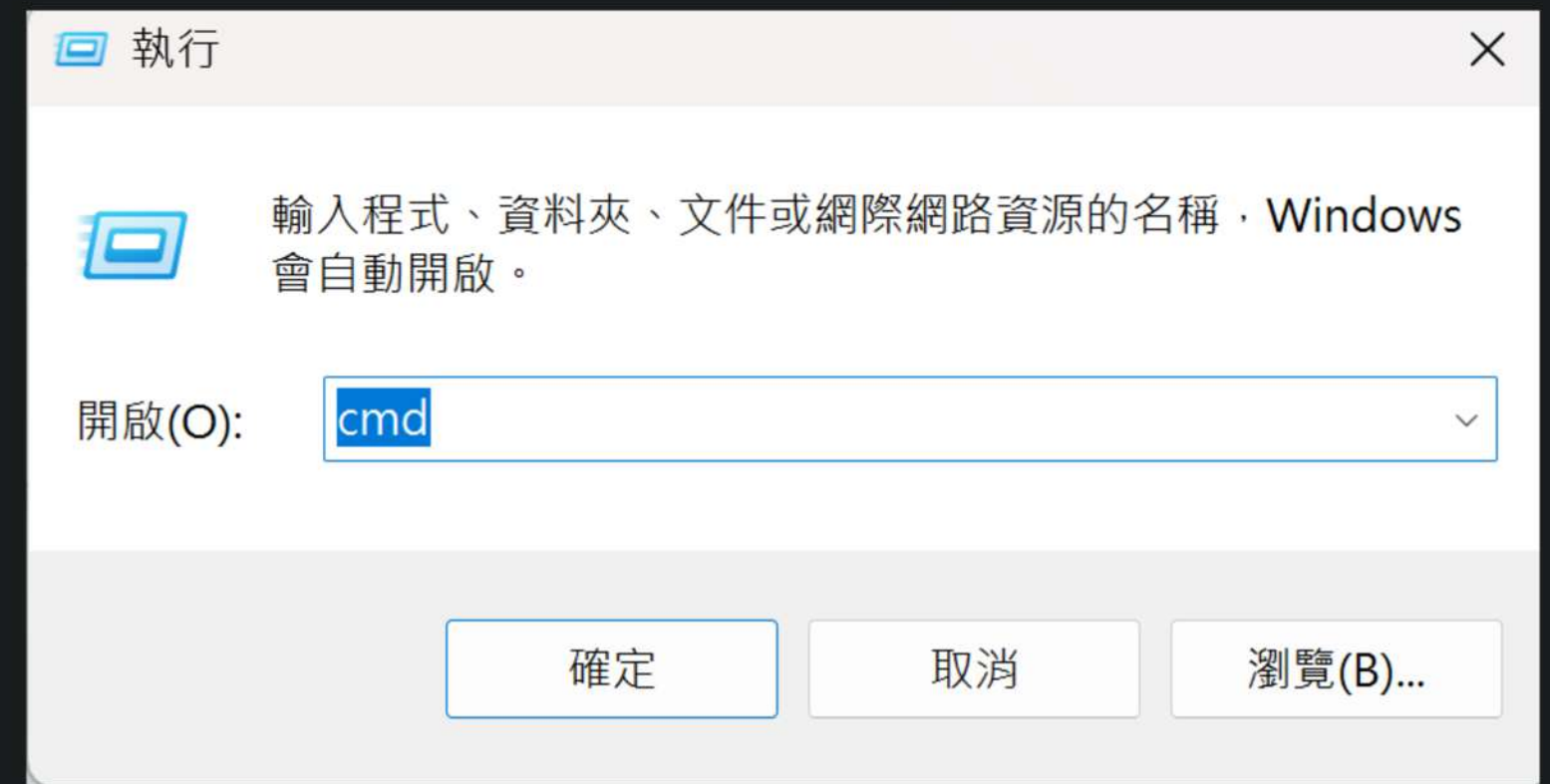
<https://www.python.org/downloads/>  
<https://pypi.org/project/auto-py-to-exe/>

**先決條件:**

**電腦內已安裝python(建議版本:3.11.9)**

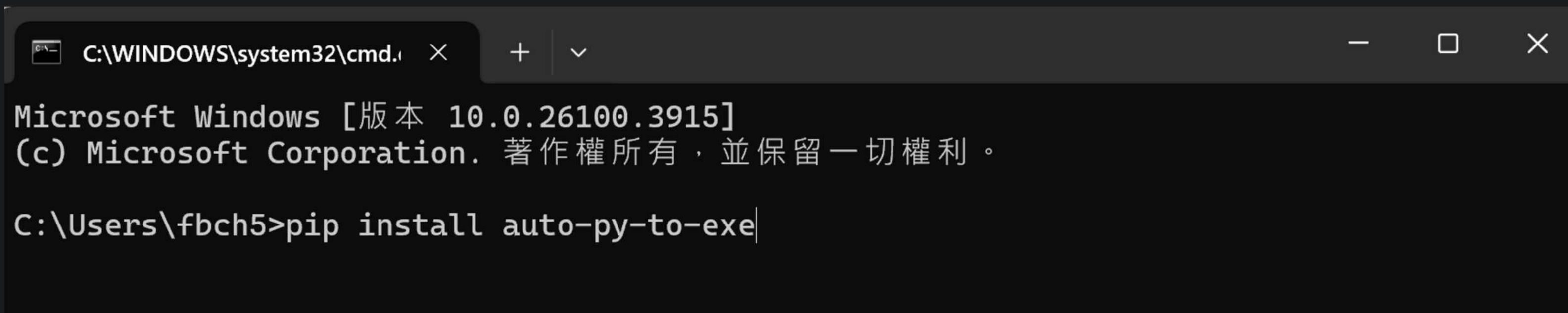
**Add Python to PATH = true**

# 1.打開command:



## 2.輸入指令並enter

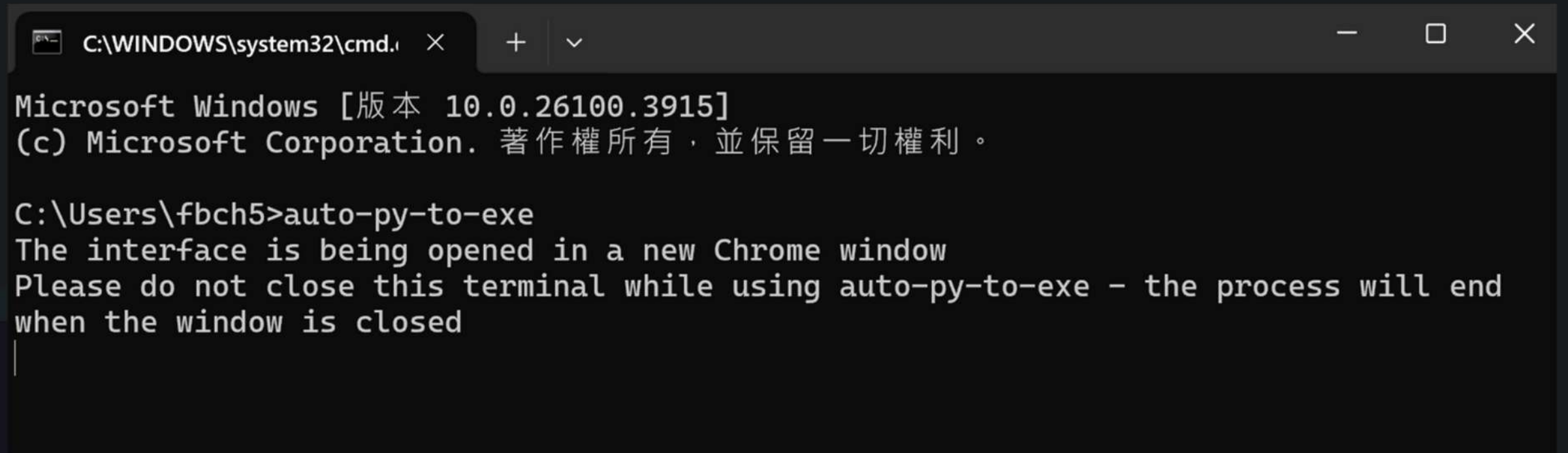
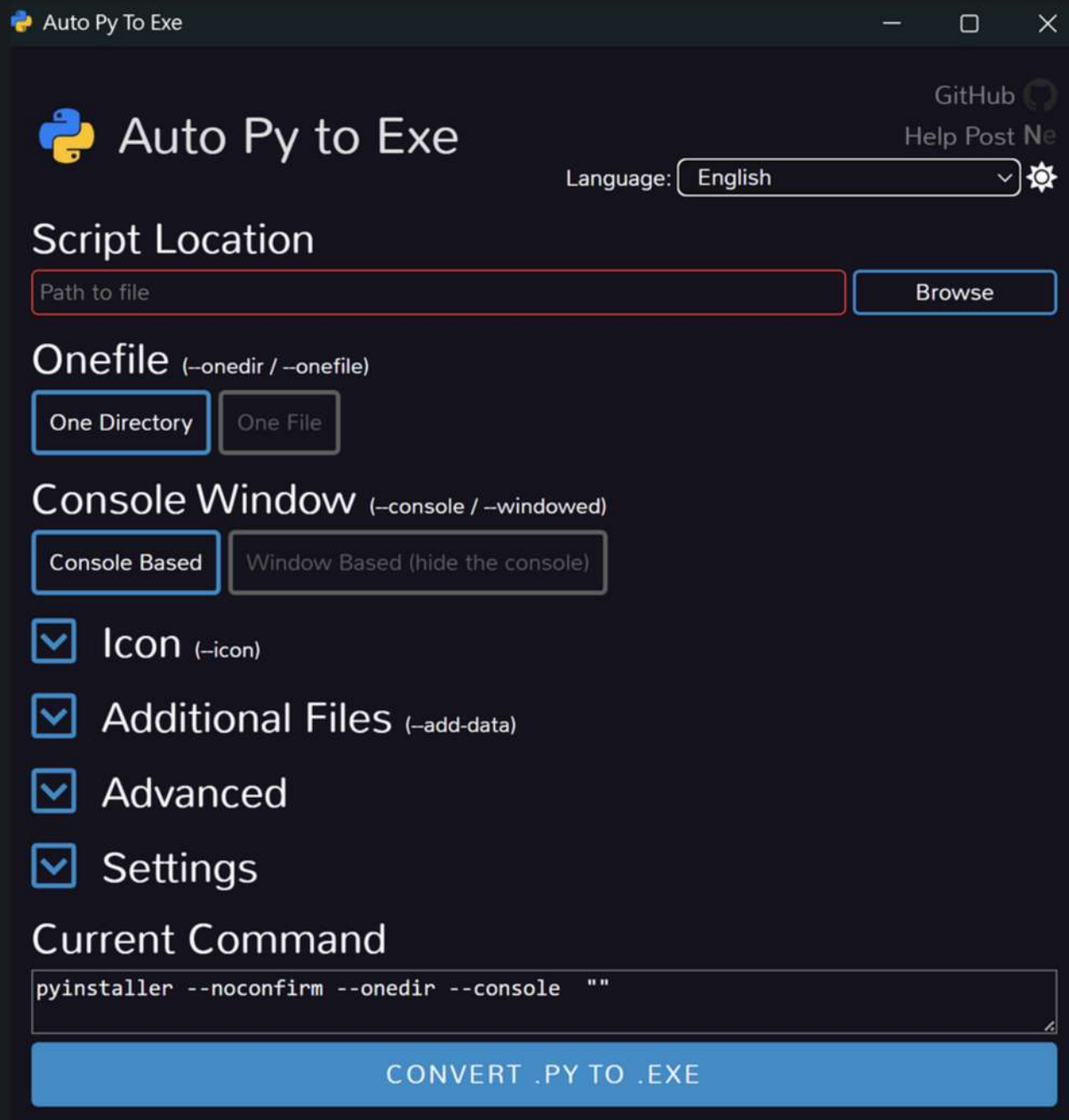
**pip install auto-py-to-exe**



```
C:\WINDOWS\system32\cmd.exe
Microsoft Windows [版本 10.0.26100.3915]
(c) Microsoft Corporation. 著作權所有，並保留一切權利。
C:\Users\fbch5>pip install auto-py-to-exe
```

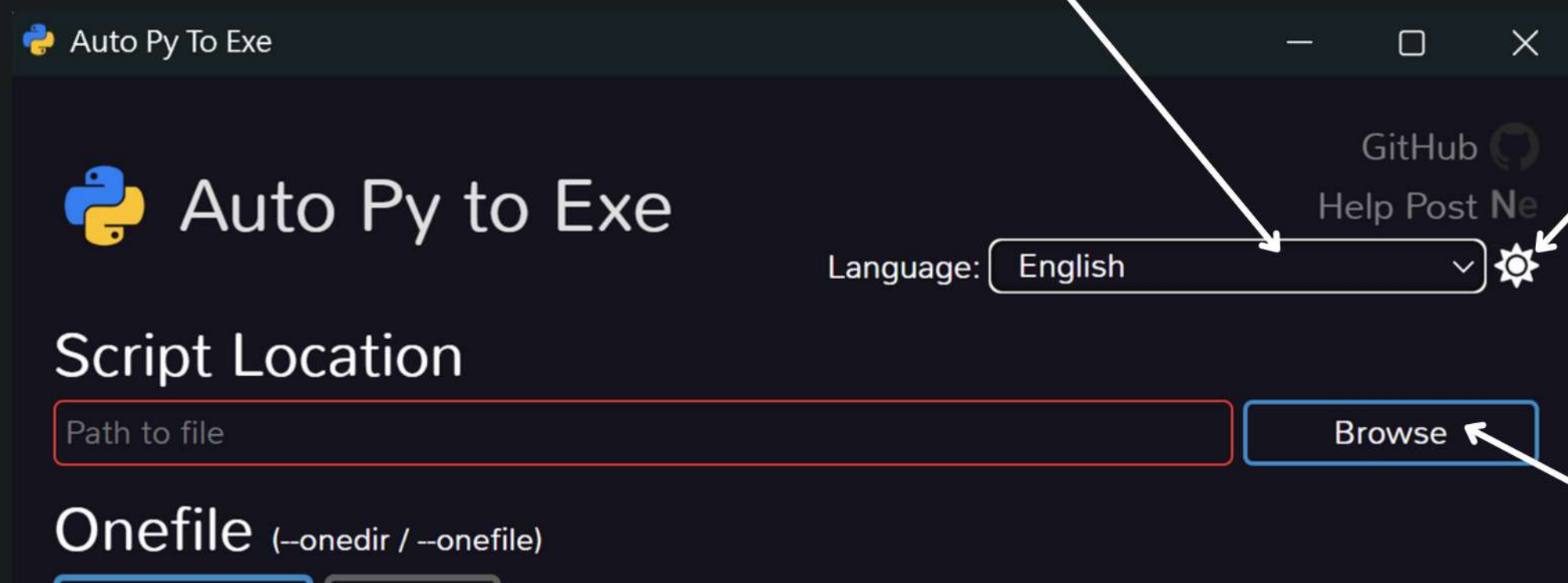
# 3.執行auto-py-to-exe

## auto-py-to-exe



更改語言

更改背景顏色



Language:

English



Script Location

Path to file

Browse

Onefile (--onedir / --onefile)

選擇檔案



# Onefile (--onedir / --onefile)

One Directory

One File

## One Directory

--onedir（一個資料夾打包）：

- 定義：將 Python 程式和所有相關的依賴文件打包成一個資料夾。該資料夾內會包含一個可執行檔案，以及所有 Python 依賴的庫、資源檔案等。
- 優點：
  - 可執行檔和依賴文件分開存放，通常可以減少執行檔案的大小。
  - 更新或修改依賴比較方便，不需要重新打包整個程序。
- 缺點：
  - 用戶需要將整個資料夾一同發佈和移動，會有額外的文件夾結構。

## OneFile

onefile（單一檔案打包）：

- 定義：將 Python 程式和所有依賴文件打包成一個單一的可執行檔案。這樣生成的 .exe 文件內部會包含所有的庫和資源，並且在執行時會臨時解壓縮到內存中。
- 優點：
  - 只有一個單一的檔案，方便發佈和移動，對用戶來說更加簡單。
  - 沒有額外的文件夾結構。
- 缺點：
  - 檔案通常比較大，因為所有的依賴都打包在一起。
  - 初次執行時，會進行解壓縮，因此啟動速度可能會稍慢。



## Console Window (--console / --windowed)

Console Based

Window Based (hide the console)

### Console Based


#### --console (基於控制台)

- 定義：程式會在運行時顯示控制台窗口，這通常用於需要顯示輸出訊息或進行調試的命令行應用。
- 用途：適合命令行工具或腳本，讓用戶可以看到程式的輸出（例如打印的訊息）。
- 視覺效果：運行時會看到一個命令行窗口顯示程式的執行過程。

### Windows Based (hide the console)

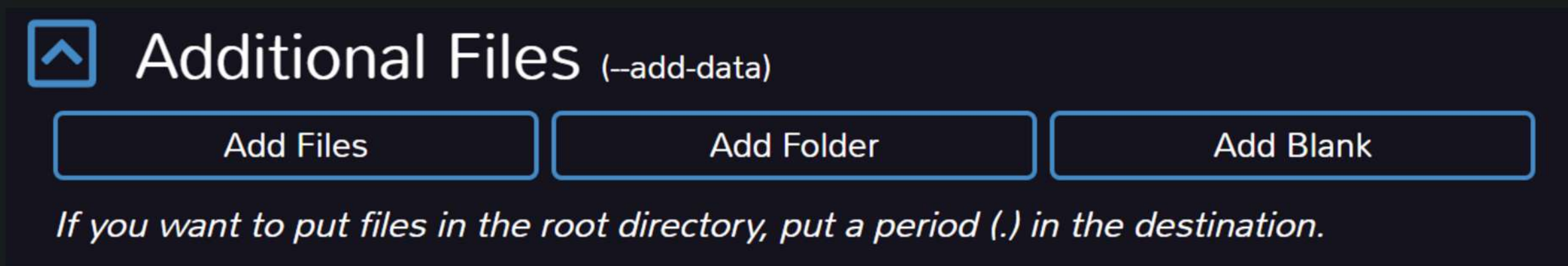
#### --windowed (基於視窗 / 隱藏控制台)

- 定義：程式在運行時不顯示控制台窗口。這通常用於 GUI 應用（例如使用 Tkinter、PyQt 等框架開發的應用），使得應用僅顯示 GUI 視窗，而不是一個控制台窗口。
- 用途：適合圖形界面應用，這些應用不需要顯示控制台，只顯示視窗界面。
- 視覺效果：運行時不會顯示控制台窗口，只有應用的 GUI 視窗顯示出來。

 **Icon** (--icon)

選擇icon為exe添加icon





### Add Files（添加文件）：

- 這是指將某些文件添加到你打包的應用程序內，這些文件可能是配置檔、圖片、音效或其他資源文件。
- 在 auto-py-to-exe 或 PyInstaller 中，你可以使用 `--add-data` 參數來指定要包括的文件。

### Add Folder（添加資料夾）：

- 將整個資料夾添加到你的打包檔案中。這可以方便地包含資料夾內的多個文件。
- 同樣使用 `--add-data` 參數來指定資料夾。

### Add Blank（添加空白）：

- 這可能指的是創建一個空白的文件或資料夾，然後在打包過程中將其添加進去。
- 這通常在某些特殊情況下使用，比如你需要一個預設的空白資料夾。

### 在根目錄中放置文件：

- 如果你希望將文件放在打包後的可執行文件的根目錄中，可以使用 `.`（點）作為目標位置，這會將文件放在根目錄，而不是內部資料夾。



## Advanced

### General Options

--name ●

--contents-directory ●

--upx-dir ●  Browse for Folder

--clean ●

--log-level ●

### What to bundle, where to search

--add-binary ●

--paths ●

--hidden-import ●

--collect-submodules ●

--collect-data ●

--collect-binaries ●

--collect-all ●

--copy-metadata ●

--recursive-copy-metadata ●

--additional-hooks-dir ●

--runtime-hook ●

--exclude-module ●

auto-py-to-exe) 可用的參數選項，它們提供了高度的自訂化能力，用來控制打包後的可執行檔行為與內容



## Settings

### auto-py-to-exe Specific Options

Output Directory ● C:\Users\fbch5\output

Browse

Increase Recursion Limit ● Enable

### Manually Provide Options

Manual Argument Input ● ARGUMENTS

### Configuration

Import Config From JSON File

Export Config To JSON File

用途：指定打包完成後，輸出檔案儲存的資料夾。  
預設值範例：C:\Users\fbch5\output  
操作：按下 Browse 按鈕可選擇不同的資料夾。

用途：在打包大型專案或使用遞迴大量導入模組時，可避免出現 RecursionError。  
啟用後效果：相當於執行 `sys.setrecursionlimit(5000)`。

Manual Argument Input（參數手動輸入區）  
用途：你可以直接輸入 PyInstaller 支援的參數字串，與上面 GUI 選項類似，但更自由。

用途：可以將目前設定「匯出為 JSON」儲存起來，或從 JSON 檔案「匯入設定」。

實用場景：

你有多個不同的打包設定檔案，可快速切換。

分享或備份設定。



## Current Command

```
pyinstaller --noconfirm --onedir --console ""
```

CONVERT .PY TO .EXE

顯示目前使用的command

執行python轉exe可執行檔

# 文字介面應用程式範例

## 1.複製CharGPT 生成的code

幫我用Python寫一個簡單的文字介面的應用程式

當然可以！這裡是一個用 Python 撰寫的簡單「文字介面應用程式」範例，採用命令列互動方式（CLI）。這個應用程式是一個簡易的待辦清單系統，使用者可以新增、查看和刪除待辦事項：

python

✓ 已複製


✎ 編輯

```
# todo_app.py

def show_menu():
    print("\n===== 簡易待辦清單 =====")
    print("1. 顯示待辦事項")
    print("2. 新增待辦事項")
    print("3. 刪除待辦事項")
```

在桌面建立一個新的資料夾並把code貼到記事本存成副檔名為.py

名稱	修改日期	類型	大小
test.py	2025/5/14 上午 10:39	Python 來源檔案	2 KB



test.py

共用

詳細資料

類型

Python 來源檔案

大小

1.48 KB

檔案位置

D:\Windows路徑用\桌面\新增...

1.48 KB |

編輯

```
tasks.append(task)
print("已新增。")

def delete_task(tasks):
    show_tasks(tasks)
    try:
        index = int(input("請輸入要刪除的編號："))
        if 1 <= index <= len(tasks):
            removed = tasks.pop(index - 1)
            print(f"已刪除：「{removed}」")
        else:
            pass
    except:
        pass
```

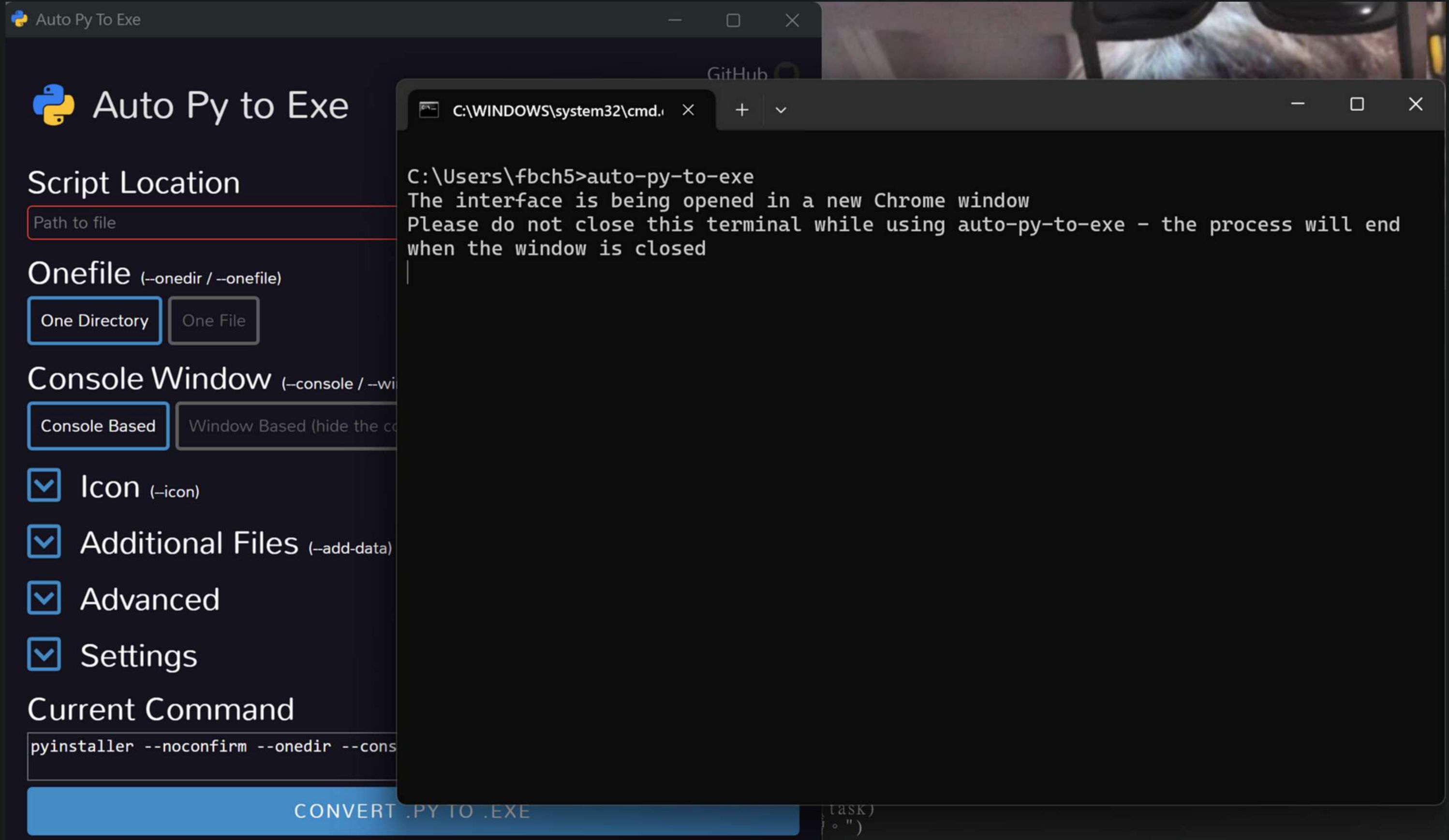
第 29 行 · 第 13 欄 | 1,242 個字元

100%

Windows (CRLF)

UTF-8

# 開啟auto-py-to-exe





# 配置如下

Auto Py To Exe

腳本位置

D:\Windows路徑用\桌面\新增資料夾 (3)\test.py

瀏覽

單檔案 (--onedir / --onefile)

單目錄

單檔案

控制台視窗 (--console / --windowed)

基於控制台

基於視窗 (隱藏控制台)

☒ 圖示 (--icon)

☒ 附加檔案 (--add-data)

☒ 進階

☒ 設定

auto-py-to-exe 特定的選項

輸出路徑 ● D:\Windows路徑用\桌面\新增資料夾 (3)

瀏覽

增加遞歸限制 ● 開啟

手動提供選項

手動參數輸入 ● 參數

配置

從 JSON 檔案導入配置

將配置導出到 JSON 檔案

當前命令

pyinstaller --noconfirm --onefile --console "D:\Windows路徑用\桌面\新增資料夾 (3)\test.py"

將.PY 轉換為 .EXE



# 執行後即在選定位置生成exe可執行檔

Auto Py To Exe

136053 INFO: Initializing module graph hook caches...

136093 INFO: Analyzing modules for base\_library.zip ...

137944 INFO: Processing standard module hook 'hor...

138114 INFO: Processing standard module hook 'ho...

141132 INFO: Processing standard module hook 'ho...

145467 INFO: Caching module dependency graph...

145518 INFO: Looking for Python shared library...

145533 INFO: Using Python shared library: C:\Use...

145551 INFO: Analyzing D:\Windows路徑用\桌面\新增...

145569 INFO: Processing module hooks (post-graph...

145589 INFO: Performing binary vs. data reclassi...

145600 INFO: Looking for ctypes DLLs

145631 INFO: Analyzing run-time hooks ...

145635 INFO: Including run-time hook 'pyi\_rth\_in...

145648 INFO: Creating base\_library.zip...

145727 INFO: Looking for dynamic libraries

145887 INFO: Extra DLL search directories (AddDL...

145894 INFO: Extra DLL search directories (PATH)

146201 INFO: Warnings written to C:\Users\fbch5\...

146244 INFO: Graph cross-reference written to C:

146273 INFO: checking PYZ

146281 INFO: Building PYZ because PYZ-00.toc is

146296 INFO: Building PYZ (ZlibArchive) C:\Users...

146590 INFO: Building PYZ (ZlibArchive) C:\Users...

146613 INFO: checking PKG

146623 INFO: Building PKG because PKG-00.toc is

146639 INFO: Building PKG (CArchive) test.pkg

149103 INFO: Building PKG (CArchive) test.pkg co

149126 INFO: Bootloader C:\Users\fbch5\AppData\L

149130 INFO: checking EXE

149136 INFO: Building EXE because EXE-00.toc is

149151 INFO: Building EXE from EXE-00.toc

149167 INFO: Copying bootloader EXE to C:\Users\

149324 INFO: Copying icon to EXE

149417 INFO: Copying 0 resources to EXE

149432 INFO: Embedding manifest in EXE

149532 INFO: Appending PKG archive to EXE

149628 INFO: Fixing EXE headers

152171 INFO: Building EXE from EXE-00.toc comple

152180 INFO: Build complete! The results are ava

Moving project to: D:\Windows路徑用\桌面\新增資料夾(3)  
Complete.

新增資料夾 (3)

開始備份

Windows路徑用 > 桌面 > 新增資料夾 (3)

搜尋 新增資料夾 (3)

新增

剪貼

複製

貼上

刪除

排序

檢視

詳細資料

名稱

修改日期

類型

test.exe

2025/5/14 上午 10:44

應用程式

test.py

2025/5/14 上午 10:39

Python 來源

test.exe

共用

詳細資料

類型

應用程式

大小

7.66 MB

檔案位置

D:\Windows路徑用\桌面\新增...

修改日期

2025/5/14 上午 10:44

內容

2 個項目 | 已選取 1 個項目 7.66 MB |

您的 exe 有問題? 請閱讀 這篇文章是關於如何修復常見的問題 尋找可能的解決方案。

清除輸出

打開輸出目錄



# 圖形化介面應用程式範例

## 1.複製CharGPT 生成的code

幫我用Python寫一個簡單的圖形畫介面的應用程式

當然可以！以下是一個使用 Python 的 `Tkinter` 套件所寫的簡單圖形化應用程式範例。這個應用程式是一個基本的視窗，裡面有一個輸入框、一個按鈕，以及一個用來顯示輸入內容的標籤。

### ✓ 功能：

- 使用者輸入文字後按下按鈕
- 螢幕上會顯示輸入的文字

### ✓ Python 程式碼 ( `simple_gui.py` )：

python

✓ 已複製

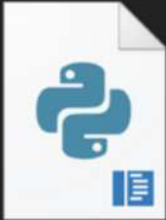
✎ 編輯

```
import tkinter as tk
from tkinter import messagebox

def on_button_click():
    user_input = entry.get()
    if user_input:
        label_result.config(text=f"你輸入的是：{user_input}")
    else:
        messagebox.showwarning("輸入錯誤", "請輸入一些文字。")
```

在桌面建立一個新的資料夾並把code貼到記事本存成副檔名為.py

名稱	修改日期	類型	大小
test.py	2025/5/14 上午 10:39	Python 來源檔案	2 KB



test.py

共用

詳細資料

類型

Python 來源檔案

大小

1.48 KB

檔案位置

D:\Windows路徑用\桌面\新增...

1.48 KB |

編輯

```
tasks.append(task)
print("已新增。")

def delete_task(tasks):
    show_tasks(tasks)
    try:
        index = int(input("請輸入要刪除的編號："))
        if 1 <= index <= len(tasks):
            removed = tasks.pop(index - 1)
            print(f"已刪除：「{removed}」")
        else:
            pass
    except:
        pass
```

第 29 行 · 第 13 欄

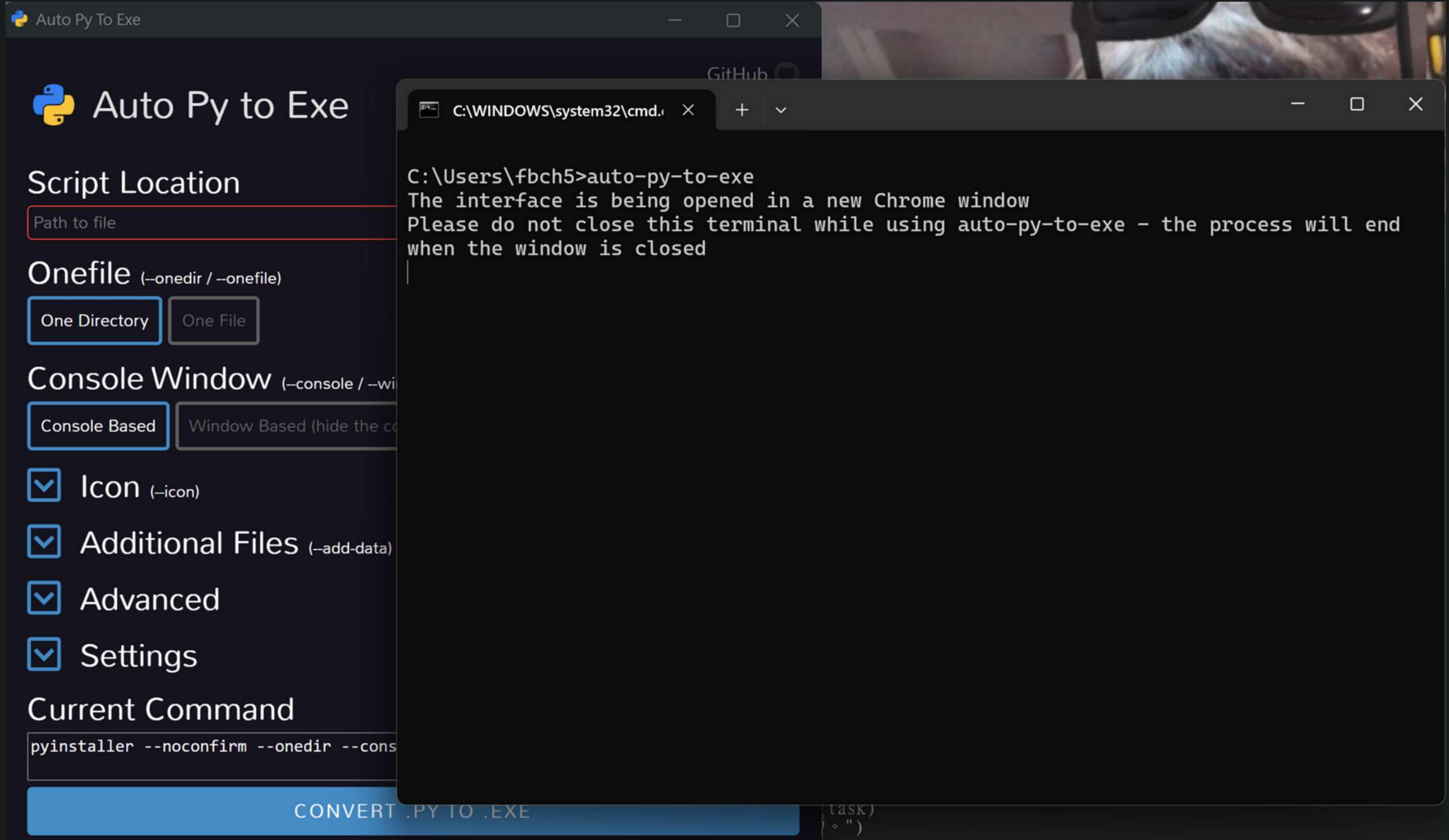
1,242 個字元

100%

Windows (CRLF)

UTF-8

# 開啟auto-py-to-exe



# 配置如下

Auto Py To Exe

單檔案 (–onedir / –onefile)

D:\Windows路徑用\桌面\新增資料夾 (3)\test.py

瀏覽

單目錄

單檔案

控制台視窗 (–console / –windowed)

基於控制台

基於視窗 (隱藏控制台)

☒ 圖示 (–icon)

☒ 附加檔案 (–add-data)

☒ 進階

☒ 設定

auto-py-to-exe 特定的選項

輸出路徑 ⓘ

D:\Windows路徑用\桌面\新增資料夾 (3)

瀏覽

增加遞歸限制 ⓘ

開啟

手動提供選項

手動參數輸入 ⓘ

參數

配置

從 JSON 檔案導入配置

將配置導出到 JSON 檔案

當前命令

pyinstaller --noconfirm --onefile --windowed "D:\Windows路徑用\桌面\新增資料夾 (3)\test.py"

將.PY 轉換為 .EXE



# 執行後即在選定位置生成exe可執行檔

