李松錡 Follow

Feb 21, 2018 · 5 min read

Python 呼叫其他程式

今天在寫一隻要用在 coursera 某個課程作業的 auto testing 的程式，因為作業裡面給的皮是一隻很單純的 py 程式，資料的讀入跟結果都用 stdin 跟 stdout，為了配合他資料傳遞的方式 (用 stdin 跟 stdout)，所以我也剛好研究了一下怎麼在 python 裡面呼叫新的 process，及怎麼與 stdin, stdout 溝通。

根據 python subprocess 的文件檔，大致上可以理解為 Popen 這個方法是比較低階的操作，而 run, 是基於 Popen 上面再包一層比較抽象的高階方法，此外大家也比較常用的可能還有這個，基於 run 上再包一層更高階的 check_output。

Popen

首先從最低階的 Popen 方法開始講起，Popen 會開啟一個 process (我猜才因此叫 P(rocess) open)。當我們直接呼叫 Popen 的時候，python 就會叫起一個新的 process，以下直接看這個 ipython 裡面執行的過程

```
In [1]: from subprocess import Popen

In [2]: Popen(['date', '-R'])
Wed, 21 Feb 2018 16:51:28 +0800
Out[2]: <subprocess.Popen at 0x10905a518>
```

在使用 Popen 的時候，類似於我們打在 shell 裡面的一行指令，但會用一個 list，把程式跟參數分別用 string 包起來，例如上面的範例，我們想執行 date 這支程式，同時希望給他 -R 的參數，則會像範例中把 date 和 -R 分別用以 string 表示，然後存成一個 list 作為第一個參數送進去，電腦就會創建一個新的 process 去執行 date -R。

更多時候我們其實不只希望可以讓電腦執行其他程式，還希望本來只會 print 在 shell 裡面的結果可以讓 python 收到，然後繼續做其他的處理，有些程式是有互動介面的，

正在等待使用者的 stdin 輸入東西按 enter 繼續執行，這個時候我們就需要更多的功能，而 subprocess 裡面還提供了 Popen.communicate 讓我們可以跟 stdin, stdout互動。我們可以在 Popen 的 keyword 中可以加入 stdin 或 stdout，且給定 subprocess.PIPE，以下直接以一個範例來解釋:

```
In [1]: from subprocess import Popen, PIPE

In [2]: p = Popen(['date', '-R'], stdout=PIPE)

In [3]: out = p.communicate()

In [4]: out
Out[4]: (b'Wed, 21 Feb 2018 17:22:50 +0800\n', None)
```

我們可以看到指定了 stdout 以後，會把本來輸出到 shell 的 stdout 和 stderr 組成一個新的 tuple return 回來，因此在 ipython 第四行的部分把 out 印出來就可以看到 return 的是一個 tuple，其第一個元素是以型態為 bytes 的 stdout，第二個元素則是 stderr，因為執行 date -R 並沒有得到 stderr，因此是 None。另外如果是要輸入 stdin 的話，則把要輸入的文字以 bytes 的型態作為 communicate 的第一個參數即可。

run 與 check_output

run 其實就是把 Popen 包起來用的一個方法，根據文檔，我們也可以設定 stdin 和 stdout 的參數，在 frequently used arguments 中就有提到 stdin, stdout, stderr 可以是 file handler，所以我們可以直接讀檔，給他 file object 作為 stdin 當作輸入, 給他 file object 作為 stdout 和 stderr 把得到的輸出存成檔案。

```
In [1]: from subprocess import run

In [2]: with open('output.txt', 'w') as f:
   ...:     run(['date', '-R'], stdout=f)
   ...:

In [3]: cat output.txt
Wed, 21 Feb 2018 18:00:21 +0800
```

可以看到輸出結果會直接被存成檔案。那如果我希望輸出的結果不要存檔案，而是拿來 python 裡面用怎麼辦呢? 只要指定輸出結果一樣輸出回 subprocess.PIPE 就可以了

```
In [1]: from subprocess import run, PIPE

In [2]: out = run(['date', '-R'], stdout=PIPE)

In [3]: out
Out[3]: CompletedProcess(args=['date', '-R'], returncode=0,
stdout=b'Wed, 21 Feb 2018 18:03:44 +0800\n')

In [4]: out.stdout
Out[4]: b'Wed, 21 Feb 2018 18:03:44 +0800\n'
```

而 check_output 在文件中也有提到，其實就等同於上面 ipython 第二行執行的部分，也就是我們可以用更高階的 check_output 更簡潔的得到結果:

```
In [1]: from subprocess import check_output

In [2]: out = check_output(['date', '-R'])

In [3]: out
Out[3]: b'Wed, 21 Feb 2018 18:24:53 +0800\n'
```

Python Subprocess