

常用到的 Linux 命令 (群暉)

指定目前的工作目錄

```
OpenSSH SSH client
C:\Users\Atrisk>ssh -p 5358 administrator@60.249.144.60 ← 用 windows 10 的 ssh 命令登入群暉的主機
administrator@60.249.144.60's password:
Administrator@WTGroup:~$ cd bash_scripts/ ← 移動目錄 (cd, change directory)
Administrator@WTGroup:~/bash_scripts$ ll ← 列出目前路徑下的所有的檔案和目錄 (ll, ls -al)
total 8
drwxrwxrwx+ 1 Administrator users 92 Nov  5 16:38 .
drwxrwxrwx+ 1 Administrator users 214 Nov  6 08:31 ..
-rwxrwxrwx+ 1 Administrator users 84 Nov  5 16:38 chk_windy_failure.sh
drwxrwxrwx+ 1 Administrator users  0 Nov  5 16:31 test
-rwxrwxrwx+ 1 Administrator users  0 Nov  5 16:28 test_1.txt
-rwxrwxrwx  1 Administrator users 71 Nov  5 16:31 test_if.sh
Administrator@WTGroup:~/bash_scripts$ chk_windy_failure.sh
-sh: chk_windy_failure.sh: command not found
Administrator@WTGroup:~/bash_scripts$ ./chk windy failure.sh
/volume1/WTGroup_Data/tmp_python/synoscheduler/7/1604376003/output.log:523:抓取失敗: 大潭 B, 25.048646, 121.032604
/volume1/WTGroup_Data/tmp_python/synoscheduler/7/1604408402/output.log:128:抓取失敗: 林口 A1, 25.13562, 121.29696
/volume1/WTGroup_Data/tmp_python/synoscheduler/7/1604530803/output.log:10:抓取失敗: 協和 A1, 25.16100, 121.74080
/volume1/WTGroup_Data/tmp_python/synoscheduler/7/1604570402/output.log:273:抓取失敗: 通霄 5號號機出水口, 24.498489, 121.660134 2020/11/05 18:05:34
/volume1/WTGroup_Data/tmp_python/synoscheduler/7/1604588402/output.log:535:抓取失敗: 台中 B, 24.227458, 120.452595
/volume1/WTGroup_Data/tmp_python/synoscheduler/7/1604631602/output.log:344:抓取失敗: 興達 A1 2020/11/06 11:06:27
/volume1/WTGroup_Data/tmp_python/synoscheduler/7/1604653202/output.log:542:抓取失敗: 台中 放流口 2020/11/06 17:10:10
/volume1/WTGroup_Data/tmp_python/synoscheduler/7/1604743202/output.log:235:抓取失敗: 通霄 A2 2020/11/07 18:04:30
/volume1/WTGroup_Data/tmp_python/synoscheduler/7/1604743202/output.log:240:抓取失敗: 通霄 A3 2020/11/07 18:04:45
/volume1/WTGroup_Data/tmp_python/synoscheduler/7/1604743202/output.log:278:抓取失敗: 塔山 1A 2020/11/07 18:05:34
Administrator@WTGroup:~/bash_scripts$ echo $PATH ← 用 echo 顯示 PATH 變數 (系統搜尋檔案用) · 記得加 $
/sbin:/bin:/usr/sbin:/usr/bin:/usr/syno/sbin:/usr/syno/bin:/usr/local/sbin:/usr/local/bin
Administrator@WTGroup:~/bash_scripts$ _
```

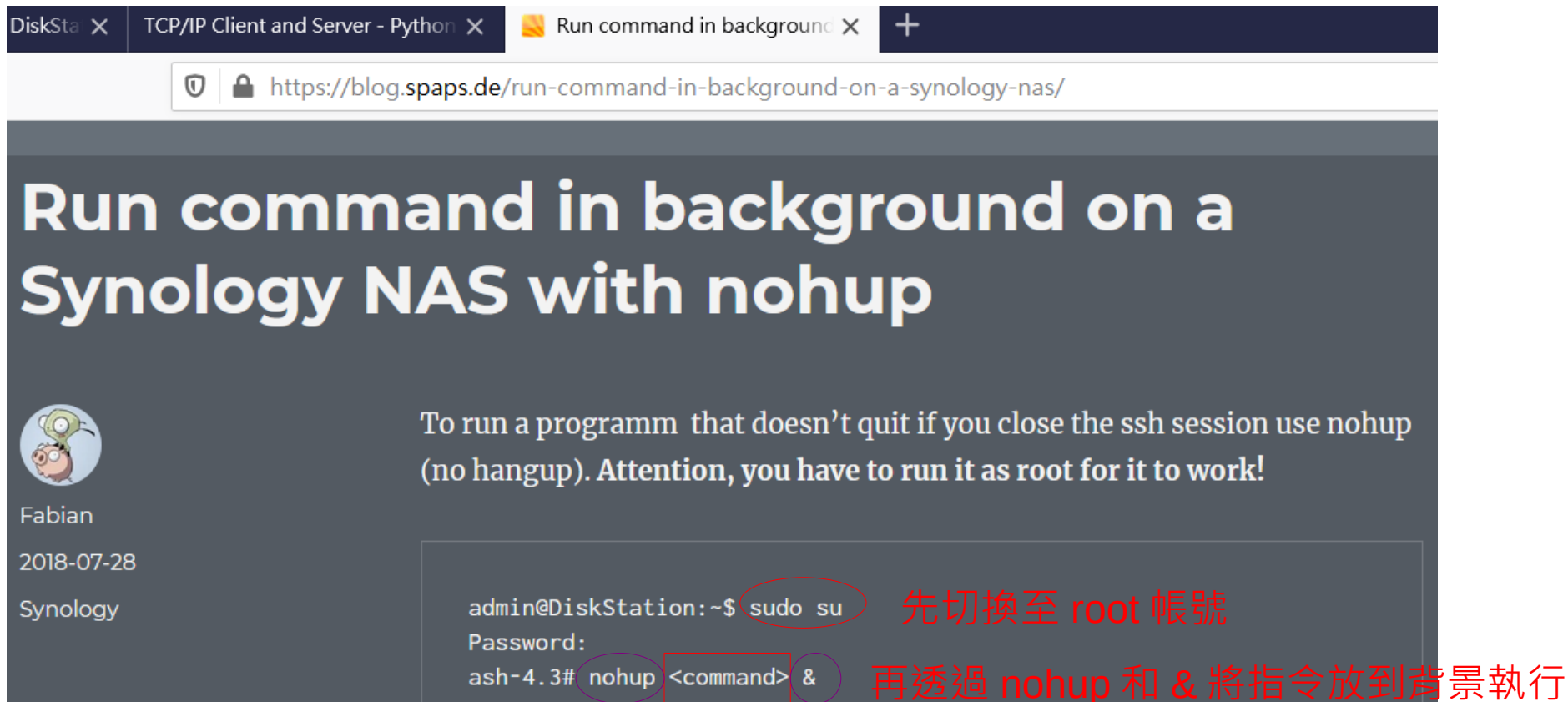
即使執行檔和目前工作目錄位在同一路徑，還是得指定路徑 `./` (不同於 windows 的使用經驗)

自訂的指令 (chk_windy_failure.sh)

```
OpenSSH SSH client 為了知道 2020/11/07 電廠下載遺失幾筆資料，敲了以下這麼多來字口
Administrator@WTGroup:~$ grep -rn /volumel/WTGroup_Data/tmp_python/synoscheduler/7 -e "2020/11/07"
/volumel/WTGroup_Data/tmp_python/synoscheduler/7/1604743202/output.log:235:抓取失敗: 通霄 A2 2020/11/07 18:04:30
/volumel/WTGroup_Data/tmp_python/synoscheduler/7/1604743202/output.log:240:抓取失敗: 通霄 A3 2020/11/07 18:04:45
/volumel/WTGroup_Data/tmp_python/synoscheduler/7/1604743202/output.log:278:抓取失敗: 塔山 1A 2020/11/07 18:05:34
Administrator@WTGroup:~$ cat bash_scripts/chk_windy_failure.sh
#!/bin/bash
grep -rn /volumel/WTGroup_Data/tmp_python/synoscheduler/7 -e "失敗"
Administrator@WTGroup:~$ chmod a+x bash_scripts/chk_windy_failure.sh
Administrator@WTGroup:~$ ./bash_scripts/chk_windy_failure.sh
/volumel/WTGroup_Data/tmp_python/synoscheduler/7/1604376003/output.log:523:抓取失敗: 大潭 B, 25.048646, 121.032604
/volumel/WTGroup_Data/tmp_python/synoscheduler/7/1604408402/output.log:128:抓取失敗: 林口 A1, 25.13562, 121.29696
/volumel/WTGroup_Data/tmp_python/synoscheduler/7/1604530803/output.log:10:抓取失敗: 協和 A1, 25.16100, 121.74080
/volumel/WTGroup_Data/tmp_python/synoscheduler/7/1604570402/output.log:273:抓取失敗: 通霄 5號號機出水口, 24.498489,
660134 2020/11/05 18:05:34
/volumel/WTGroup_Data/tmp_python/synoscheduler/7/1604588402/output.log:535:抓取失敗: 台中 B, 24.227458, 120.452595
/volumel/WTGroup_Data/tmp_python/synoscheduler/7/1604631602/output.log:344:抓取失敗: 興達 A1 2020/11/06 11:06:27
/volumel/WTGroup_Data/tmp_python/synoscheduler/7/1604653202/output.log:542:抓取失敗: 台中 放流口 2020/11/06 17:10:10
/volumel/WTGroup_Data/tmp_python/synoscheduler/7/1604743202/output.log:235:抓取失敗: 通霄 A2 2020/11/07 18:04:30
/volumel/WTGroup_Data/tmp_python/synoscheduler/7/1604743202/output.log:240:抓取失敗: 通霄 A3 2020/11/07 18:04:45
/volumel/WTGroup_Data/tmp_python/synoscheduler/7/1604743202/output.log:278:抓取失敗: 塔山 1A 2020/11/07 18:05:34
/volumel/WTGroup_Data/tmp_python/synoscheduler/7/1604894402/output.log:10:抓取失敗: 協和 A1 2020/11/09 12:00:27
/volumel/WTGroup_Data/tmp_python/synoscheduler/7/1604894402/output.log:25:抓取失敗: 協和 A3 2020/11/09 12:00:54
```

和直接敲指令的方式相比，直接執行檔案後的結果不再顯示顏色了

在背景下執行指令



The image shows a browser window with the following elements:

- Browser tabs: DiskSta x, TCP/IP Client and Server - Python x, Run command in background x, +
- Address bar: <https://blog.spaps.de/run-command-in-background-on-a-synology-nas/>
- Article title: **Run command in background on a Synology NAS with nohup**
- Author: Fabian (with a cartoon pig avatar), 2018-07-28, Synology
- Text: To run a program that doesn't quit if you close the ssh session use nohup (no hangup). Attention, you have to run it as root for it to work!
- Terminal screenshot:

```
admin@DiskStation:~$ sudo su
Password:
ash-4.3# nohup <command> &
```

Red annotations on the terminal screenshot:

- A red circle around `sudo su` with the text "先切換至 root 帳號" (First switch to root account).
- A red circle around `nohup` and another around `&` with the text "再透過 nohup 和 & 將指令放到背景執行" (Then use nohup and & to put the command in the background).

將 TCP Server 丟到背景執行

```
Administrator@WTGroup:~$ sudo su
Password:
ash-4.3# nohup python3 ./python3_scripts/tcp_server_client/tcp_server.py &
[1] 30497
ash-4.3# nohup: ignoring input and appending output to 'nohup.out'
```

```
import socket
import sys

# Create a TCP/IP socket
sock = socket.socket(socket.AF_INET, socket.SOCK_STREAM)

# Connect the socket to the port where the server is listening
#server address = ('localhost', 10000)
#server address = ('168.95.1.1', 10000)
server_address = ('60.249.144.60', 10000)

print('connecting to %s port %s' % server_address)
sock.connect(server_address)

try:
    # Send data
    #message = 'This is the message. It will be repeated.'
    #message = b'This is the message. It will be repeated.'
    message = b'123456789.abcdefghi from pc'
    client 傳給 server 的資料
    print('sending "%s"' % message)
    sock.sendall(message)

    # Look for the response
    amount_received = 0
    amount_expected = len(message)

    while amount_received < amount_expected:
        data = sock.recv(16)
        amount_received += len(data)
        print('received "%s"' % data)

finally:
    print('closing socket')
    sock.close()
```

```
Python 3.8.5 Shell
File Edit Shell Debug Options Window Help
Python 3.8.5 (tags/v3.8.5:580fbb0, Jul 20 2020, 15:57:54) [MSC v.1924 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license()" for more information.
>>>
===== RESTART: D:\Python_projects\tcp_server_client\tcp_client_2.py =====
connecting to 60.249.144.60 port 10000
sending "b'123456789.abcdefghi from pc'"
received "b'123456789.abcde'"
received "b'ghi from pc'"
closing socket
>>> |
```

(1) TCP server 程式放在背景執行
該程式會將 client 端丟來的資料，
切成每 16 bytes 為一組，再一組
一組回傳給 client 端

(2) pc 端的 TCP/IP client 程式 (測試)

(3) client 收到的資料

使用 docker 或任務排程都得使用到 Linux 指令

控制台

所有我新增的任務都用到 linux 指令

已啟動	擁有者	應用程式	任務名稱	動作	下次執行時間
<input checked="" type="checkbox"/>	root	使用者定義指令碼	Windy crawler - ...	執行: docker co...	2020-11-09 1...
<input type="checkbox"/>	root	使用者定義指令碼	windy_gps_Upd...	執行: docker co...	2020-11-09 1...
<input checked="" type="checkbox"/>	root	使用者定義指令碼	windy_webscrap...	執行: docker co...	2020-11-09 1...
<input checked="" type="checkbox"/>	root	使用者定義指令碼	Test docker pyth...	執行: docker co...	2020-11-10 0...
<input checked="" type="checkbox"/>	root	DSM 自動更新	DSM Auto Update	發送 DSM 更新通知	2020-11-14 0...
<input checked="" type="checkbox"/>	Administrator	使用者定義指令碼	webscraping_do...	執行: export PYT...	2020-11-14 1...
<input checked="" type="checkbox"/>	root	S.M.A.R.T. 檢測	Auto S.M.A.R.T. ...	對所有支援快速檢...	2020-11-18 0...

編輯任務

一般 排程 任務設定

通知設定

透過電子郵件傳送執行細節

電子郵件: admin@example.com

僅在指令碼異常終止時傳送執行細節

執行命令

使用者定義指令碼

```
docker container start windy_powerplants
docker container exec windy_powerplants sh -c "cd /root ; python3
windy_chrome_PowerPlants_docker.py ; exit"
docker container stop windy_powerplants
```

連 docker 指令下，都包含了 linux 指令，exit 是離開 bash shell 的指令

過濾上一頁抓取失敗的紀錄

```
Administrator@WTGroup:~$ grep -rn /volumel/WTGroup_Data/tmp_python/synoscheduler/7 -e "失敗(2)"
/volumel/WTGroup_Data/tmp_python/synoscheduler/7/1604907782/output.log:326:抓取失敗(2): 通霄 5號號機出水口 2020/11/09 15:51:20
/volumel/WTGroup_Data/tmp_python/synoscheduler/7/1604908802/output.log:142:抓取失敗(2): 林口 A2 2020/11/09 16:03:33
/volumel/WTGroup_Data/tmp_python/synoscheduler/7/1604908802/output.log:231:抓取失敗(2): 南部 B 2020/11/09 16:06:16
/volumel/WTGroup_Data/tmp_python/synoscheduler/7/1604908802/output.log:308:抓取失敗(2): 通霄 5號號機出水口 2020/11/09 16:08:18
/volumel/WTGroup_Data/tmp_python/synoscheduler/7/1604908802/output.log:347:抓取失敗(2): 塔山 2A 2020/11/09 16:09:34
/volumel/WTGroup_Data/tmp_python/synoscheduler/7/1604916002/output.log:130:抓取失敗(2): 協和馬祖珠山 放流口 2020/11/09 18:03:50
/volumel/WTGroup_Data/tmp_python/synoscheduler/7/1604916002/output.log:378:抓取失敗(2): 塔山 背景站 2020/11/09 18:09:53
/volumel/WTGroup_Data/tmp_python/synoscheduler/7/1604916002/output.log:402:抓取失敗(2): 興達 A2 2020/11/09 18:10:38
/volumel/WTGroup_Data/tmp_python/synoscheduler/7/1604916002/output.log:451:抓取失敗(2): 大林 A1 2020/11/09 18:12:06
/volumel/WTGroup_Data/tmp_python/synoscheduler/7/1604923202/output.log:79:抓取失敗(2): 協和馬祖珠山 A1 2020/11/09 20:02:23
/volumel/WTGroup_Data/tmp_python/synoscheduler/7/1604923202/output.log:154:抓取失敗(2): 林口 A2 2020/11/09 20:04:35
/volumel/WTGroup_Data/tmp_python/synoscheduler/7/1604923202/output.log:318:抓取失敗(2): 通霄 5號號機出水口 2020/11/09 20:08:41
/volumel/WTGroup_Data/tmp_python/synoscheduler/7/1604923202/output.log:347:抓取失敗(2): 塔山 1B 2020/11/09 20:09:45
/volumel/WTGroup_Data/tmp_python/synoscheduler/7/1604923202/output.log:693:抓取失敗(2): 尖山 放流口 2020/11/09 20:18:51
/volumel/WTGroup_Data/tmp_python/synoscheduler/7/1604926802/output.log:262:抓取失敗(2): 南部 B 2020/11/09 21:07:34
/volumel/WTGroup_Data/tmp_python/synoscheduler/7/1604934002/output.log:392:抓取失敗(2): 興達 A2 2020/11/09 23:09:16
/volumel/WTGroup_Data/tmp_python/synoscheduler/7/1604934002/output.log:573:抓取失敗(2): 台中 B 2020/11/09 23:12:57
/volumel/WTGroup_Data/tmp_python/synoscheduler/7/1604934002/output.log:632:抓取失敗(2): 尖山 B 2020/11/09 23:14:36
/volumel/WTGroup_Data/tmp_python/synoscheduler/7/1604937602/output.log:142:抓取失敗(2): 林口 A3 2020/11/10 00:03:07
/volumel/WTGroup_Data/tmp_python/synoscheduler/7/1604941202/output.log:99:抓取失敗(2): 協和馬祖珠山 B 2020/11/10 01:02:32
/volumel/WTGroup_Data/tmp_python/synoscheduler/7/1604941202/output.log:122:抓取失敗(2): 林口 A1 2020/11/10 01:03:19
/volumel/WTGroup_Data/tmp_python/synoscheduler/7/1604941202/output.log:142:抓取失敗(2): 林口 A3 2020/11/10 01:04:04
/volumel/WTGroup_Data/tmp_python/synoscheduler/7/1604941202/output.log:152:抓取失敗(2): 林口 B 2020/11/10 01:04:37
/volumel/WTGroup_Data/tmp_python/synoscheduler/7/1604955602/output.log:25:抓取失敗(2): 協和 A2 2020/11/10 05:00:59
/volumel/WTGroup_Data/tmp_python/synoscheduler/7/1604955602/output.log:132:抓取失敗(2): 林口 A1 2020/11/10 05:03:43
/volumel/WTGroup_Data/tmp_python/synoscheduler/7/1604959202/output.log:216:抓取失敗(2): 南部 A2 2020/11/10 06:05:48
/volumel/WTGroup_Data/tmp_python/synoscheduler/7/1604966402/output.log:35:抓取失敗(2): 協和 A3 2020/11/10 08:01:10
/volumel/WTGroup_Data/tmp_python/synoscheduler/7/1604966402/output.log:112:抓取失敗(2): 協和馬祖珠山 放流口 2020/11/10 08:03:12
/volumel/WTGroup_Data/tmp_python/synoscheduler/7/1604966402/output.log:156:抓取失敗(2): 林口 A3 2020/11/10 08:04:47
/volumel/WTGroup_Data/tmp_python/synoscheduler/7/1604966402/output.log:299:抓取失敗(2): 通霄 B 2020/11/10 08:08:54
/volumel/WTGroup_Data/tmp_python/synoscheduler/7/1604966402/output.log:322:抓取失敗(2): 通霄 5號號機出水口 2020/11/10 08:09:59
/volumel/WTGroup_Data/tmp_python/synoscheduler/7/1604966402/output.log:351:抓取失敗(2): 塔山 1B 2020/11/10 08:11:03
/volumel/WTGroup_Data/tmp_python/synoscheduler/7/1604966402/output.log:409:抓取失敗(2): 興達 A2 2020/11/10 08:12:43
/volumel/WTGroup_Data/tmp_python/synoscheduler/7/1604966402/output.log:439:抓取失敗(2): 興達 放流口 2020/11/10 08:13:39
/volumel/WTGroup_Data/tmp_python/synoscheduler/7/1604966402/output.log:574:抓取失敗(2): 台中 A1 2020/11/10 08:17:08
/volumel/WTGroup_Data/tmp_python/synoscheduler/7/1604966402/output.log:638:抓取失敗(2): 尖山 A2 2020/11/10 08:18:40
Administrator@WTGroup:~$ grep -rn /volumel/WTGroup_Data/tmp_python/synoscheduler/7 -e "失敗(2)" | wc -l
36
Administrator@WTGroup:~$ grep -rn /volumel/WTGroup_Data/tmp_python/synoscheduler/7 -e "失敗" | wc -l
359
Administrator@WTGroup:~$
```

兩道指令間隔著管線 (|) ,
wc (word count), -l (lines)

Linux pipes

In Linux, you can use a pipe to send the output of one command to be the input (argument) of another command:

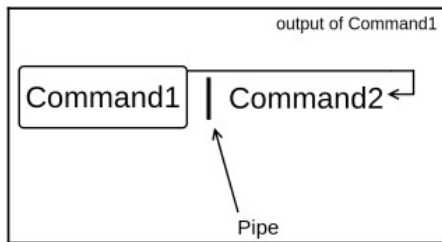


Figure 1 – A Linux pipe

A pipe is represented by the vertical bar character on your keyboard. Linux pipes are very useful as they allow you to accomplish a relatively complex task in an easy way, and throughout the book, you will see that they come in handy very often.

Let's do another example. If you want to display the seventh line of the file facts.txt, then you will show the first seven lines using the `head` command, then use a pipe to `tail` the last line:

```
elliott@ubuntu-linux:~$ head -n 7 facts.txt | tail -n 1
Linux is awesome
```

You can also use more than one pipe at a time as demonstrated in the following diagram:

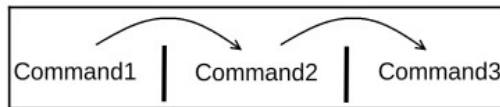


Figure 2: Two pipes

Input and output redirection

In this section, you will get to learn one of the coolest Linux features, which is I/O (input/output) redirection. Most Linux commands work with three different streams of data:

- Standard input (also referred to as `stdin`)
- Standard output (also referred to as `stdout`)
- Standard error (also referred to as `stderr`)

Most of the commands we have discussed so far produce some output. This output is sent to a special file called standard output (also referred to as `stdout`). By default, the standard output file is linked to the terminal, and that's why every time you run a command, you see the output on your terminal. Also, sometimes commands will produce error messages. These error messages are sent to another special file called standard error (also referred to as `stderr`), and it's also linked to the terminal by default.

Redirecting standard output

You know that running the `date` command will display the current date on your terminal:

```
elliott@ubuntu-linux:~$ date
Sat May 11 06:02:44 CST 2019
```

Now by using the greater than sign `>`, you can redirect the output of the `date` command to a file instead of your terminal! Have a look:

```
elliott@ubuntu-linux:~$ date > mydate.txt
```

As you can see, there is no output displayed on your screen! That's because the output got redirected to the file `mydate.txt`:

```
elliott@ubuntu-linux:~$ cat mydate.txt
Sat May 11 06:04:49 CST 2019
```

強大的 grep 指令

(Global Regular Expression Print)

- `grep -rn /volume1/WTGroup_Data/tmp_python/synoscheduler/7 -e "失敗"`
 - `/volume1/WTGroup_Data/tmp_python/synoscheduler/7`
 - 在群暉主機上，任務排程表內設定的除錯路徑，用來顯示程式執行過程中的資訊
 - `grep` 指令用到的選項 (指令的說明可以敲 `grep --help`)
 - `-r, --recursive` like `--directories=recurse`
 - `-n, --line-number` `print line number` with output lines (顯示在檔案內的行數)
 - `-e, --regexp=PATTERN` use `PATTERN` for matching (正規表示式)
- 從 `/volume1/.../7` 目錄開始 (含子目錄)，套用正規表示式搜尋檔案內包含「失敗」的字眼，並將所有檔案內包含這兩字的該行和其行數顯示出來