UNIX 基礎命令

2017-10-16

虛擬主控台與虛擬終端機

- 在一些常見的 UNIX-like 系統（GNU/Linux, *BSD）上，如果沒有預設啟動圖形化界面，則會進入指令登入提示的文字界面。
- 一般提供六到八個系統主控台（console），透過 **Alt** + **F?** 按鍵切換，其中 **F?** 對應 **F1** 至 **F6** 或 **F8** 之間。若進入圖形化界面可透過 **Ctrl** + **Alt** + **F?** 切換至文字界面的主控台。
- 一般而言第一個主控台（通常為 `ttyv0` 或 `tty0`）會用來顯示系統的訊息。其他切換到的稱為虛擬主控台（virtual console）。
- 在圖形使用界面下，可透過某些通稱為虛擬終端機 （terminal emulator）的應用程式，模擬主控台環境。
- 當使用者以登入主控台，或是執行虛擬終端機後，執行一種稱為 UNIX shell 的命令列介面程式。

UNIX shell

- UNIX shell 或簡稱 Shell（殼層？），是傳統上的使用者與計算機互動介面，處理輸入命令或喚起執行程式。
- 在 Shell 環境下，UNIX 的操作命令可分為 Shell 提供命令與單獨的程式。
- 常見的 Shell 分類：
  - Bourne shell (sh) 早期 UNIX 上常見的 Shell 替代 Thompson shell（第一個 UNIX Shell）。
  - Bourne again shell (bash) 是 GNU 計劃所發展的，用來相容 sh 與開放原碼的版本，常見在各種 Linux 發行版。
  - C shell (csh) 由 *BSD 系統上發展的類似於 C 語言語法的 shell。
  - tcsh 是 C shell 後繼改良版。

- 命令列介面呈現的方式，由左至右依序為：提示（prompt）、命令 (command)、參數列（parmaters）。
- Shell 顯示提示時，通常表示之前工作已完成，並等候接受使用者輸入命令。通常用下列字元結尾，'$'，'%'，'#'，':'，'>'。
- Shell script（Shell 腳本）內容由一連串的 shell 命令或程式呼叫所組成的程式，透過 Shell 直譯去執行。
- UNIX 上的 shell script 同常是寫成與 Bourne shell 相容的腳本；通常第一行會寫明所使用的 shell 直譯器。例如

```
001  #!/bin/sh
002
003  /bin/date # Show today
004  #/usr/bin/cal
```

  通常以 '#' 字元之後的內容為註解（comment）。

環境變數

- 環境變數（Enviroment Variable），提供使用者登入系統之後，預設的功能與操作環境的設定值，或是狀態的資訊。
- HOME－使用者家目錄。
- SHELL－預設的 Shell 路徑。
- PATH－可執行程式的檔案尋找路徑。
- USER－使用者名稱。
- 用 env 命令顯示所有設定的 Shell 環境變數。
- 登入後 Shell 先讀取系統預設的環境變數設定檔，然後在讀取使用者自訂的設定檔，設定環境變數。位於家目錄下的 .profile 檔，.bashrc 檔或.cshrc，依照 Shell 而定。
- 取得某環境變數的值，在其變數名稱前加上「$」。

  ```
  # echo $USER
  ```

  echo 是用來顯示一行文字的命令。

線上手冊查詢

- `man`
  提供使用者查詢某命令的線上手冊說明。
  簡易語法：`man [章節] 頁面（欲查詢的命令）`

  ```
  # man man
  # man 7 man
  ```

- `whatis`
  顯示關於某命令的描述與所屬的章節。

  ```
  # whatis man
  ```

- `whereis`
  顯示命令的檔案位置、手冊檔案位置或相關資訊。

  ```
  # whereis man
  # whereis export
  ```

使用者權限

- 傳統 UNIX 將檔案區分為三種權限，分別為讀取（read）、寫入（write）與執行（execute）。以「rwx」字母分別表示有權限操作，若表為「-」則為無操作權限。
- 檔案權限對使用者區分成三種類別；分別是所有者（owner）、群組（group）、其他用戶（other）。

標示	數值	權限		
		讀取	寫入	執行
`---`	0	否	否	否
`--x`	1	否	否	是
`-w-`	2	否	是	否
`-wx`	3	否	是	是
`r--`	4	是	否	否
`r-x`	5	是	否	是
`rw-`	6	是	是	否
`rwx`	7	是	是	是

- 表示上從左至右以三個字元為一組，表示所有者、群組與其他用戶對於此檔案的操作權限。例如所有者只可讀取與寫入，群組只可讀，其他用戶皆不可，則表示為

  `rw-r-----` 或者是 640

  以數字方式表示。

檔案系統（filesystem）

- 不同的 UNIX-like 作業系統使用或支援數種不同的類型檔案系統，都支援基本的使用者存取權限管理，彼此之間有某一種程度上的相容性。
- 不同於 Windows 系統的習慣，檔案與目錄的名稱是會區分大小寫字母，「file」與「File」是不同的檔案。鍵盤上可打出的字元除了「\」與「/」 外，都可作為名稱；此外依照不同系統對於多國語言處理能力情況，也可以使用非英文字或母字命名。
- 不使用副檔名（filename extension）區分檔案型態。
- 在 UNIX 作業系統中，將裝置視為檔案系統中的檔案。
- UNIX-like 的檔案系統是以樹狀結構來表示與管理，依照功能與用途分類至於不同的目錄（directory）。
- 「.」表示目前的目錄，「..」代表上層目錄。

常見的目錄結構與功能分類

目錄	用途
/	檔案系統的最上層。
/bin/	基本程式工具。
/dev/	裝置節點（Device node）。
/etc/	系統設定檔。
/home/	擺放使用者資料（家目錄），有時候會是 /usr/home/。
/proc/	行程（Process）檔案系統。
/sbin/	系統程式及管理工具。
/usr/	使用者安裝的工具與應用程式。
/tmp/	擺放暫存檔案用。
/var/	存放各種用途的日誌紀錄檔。

檔案管理的命令或程式

- `pwd`
  顯示使用者現在所在的目錄。
- `ls`
  顯示目錄中內容，預設不顯示「.」開頭的檔案。
  簡易語法：`ls` [選擇] [目錄／檔案路徑]
  - 清單顯示

    ```
    # ls -l
    ```

    依欄位顯示：權限、擁有者、群組、檔案大小、日期、名稱。其中權限欄的最左字元表示檔案的類型，標準檔案（-）、目錄（d）、特殊字元裝置（c）、Socket（c）、捷徑（l）、管線（p）之後為權限狀態。
  - 詳細的內容

    ```
    # ls -a
    ```

- `cd`

  改變目錄位置。

  簡易語法：`cd` [路徑]

  路徑（path）可以是絕對路徑或相對路徑，絕對路徑以根目錄開始，相對路徑從所在目錄開始，無給定切換至預設家目錄。

  絕對路徑： `/usr/bin/cc`

  相對路徑： `../`

- `cp` 複製檔案。

  簡易語法：`cp` 來源路徑 目標路徑

  `cp -r` 來源目錄路徑 目標路徑

- `mv`

  檔案或目錄的搬移或修改名稱。

  簡易語法：`mv` 來源 目標

- mkdir

  建立目錄。

  簡易語法：mkdir [選擇] 目錄名稱

  ```
  # mkdir newtestdir
  ```

- rmdir

  移除空目錄。

  簡易語法：rmdir [選擇] 目錄名稱

- touch

  修改檔案時間戳記。

  簡易語法：touch [選擇] 檔案

  - 產生空檔

    ```
    # touch testfile
    ```

  - 修改時間

    ```
    # touch -t [[CC]YY]MMDDhhmm[.ss] testfile
    ```

    格式為 [[CC]YY]MMDDhhmm[.ss] 以 201710021500.30 為例表示 2017年10月02日15時00分30秒。

- `chmod`

  修改權限。

  簡易語法：`chmod [選擇] 權限 路徑`

  - 修改成擁有者唯讀，其餘皆不可。

    ```
    # chmod 400 testfile
    ```

  - 目錄套用權限（唯讀）。

    ```
    # chmod 555 testdir
    ```

    目錄權限移除可執行，則不能改變路徑至此目錄。

- `rm`

  移除檔案或目錄。

  簡易語法：`rm [選擇] 檔案 ...`

  - 忽略唯讀檔案刪除詢問訊息，強制移除檔案。

    `rm -f 檔案`

  - 遞迴刪除目錄與檔案。

    `rm -r 目錄`

- `file`
  檢查檔案類型。
- `df`
  顯示磁碟未使用量。
- `du`
  顯示檔案在磁碟中使用量，此值會大於或等於檔案實際大小。
  簡易語法：`du` [選擇] [目錄／檔案路徑]
  ```
  # du /bin/ls
  ```

簡易的檔案搜尋

- `find` 程式是 UNIX 上功能強大的的搜尋檔案工具。
- 簡易語法：`find` [搜尋路徑] [目標的表達式]
  例如找尋 `/usr` 目錄下名稱為 `gcc` 的檔案或目錄。

  ```
  # find /usr -name gcc
  ```

  名稱為 `gcc` 在加上任意長度字元的檔案或目錄。

  ```
  # find /usr -name 'gcc*'
  ```

- 目標的表達式
  - 不區分大小寫：`-iname 'pattern'`
  - 檔案類型：`-type c`，其中 `c` 為 `d` 表示目錄，`f` 表示一般的檔案。
  - 檔案大小：`-size ±n`，「+」為大於的檔案，「-」為小於的檔案。

系統與使用者資訊

- `hostname`
  電腦的網路名稱。
- `uname`
  顯示系統資訊。
  簡易語法：`uname [選擇]`
  - 系統指令架構
    ```
    # uname -m
    ```
  - 所有資訊
    ```
    # uname -a
    ```
- `uptime`
  系統已啟動時間。
- `date`
  系統的時間日期，若沒有校時有可能不準確。

- `cal`
  顯示月曆。
- 顯示登入系統的使用者
  - `who`
    顯示登入的使用者。
  - `w`
    顯示登入的使用者與其正在執行的程式。
- `# id`
  使用者識別碼與所屬群組識別碼資訊。
- `# whoami`
  使用者帳號名稱。
- `clear`
  清除終端機畫面。

行程管理

- sleep
  延遲一給定時間以秒為單位。
  簡易語法：sleep 時間（秒）
  ```
  # sleep 5
  ```
- 在命令最後方加上「&」可以背景執行
  ```
  # sleep 30 &
  ```
- ps
  顯示被執行程式的識別碼（PID）與行程狀態。
- top
  監視系統執行的行程，依照系統資源使用率的高低排列，按 **Q** 鍵結束 top 監視程式。

- 在執行中的行程按下 **Ctrl** + **c** 按鍵會終止執行。
- `kill`
  終止或送訊號給行程。
  簡易語法：`kill` [選擇] `PID`
- 按下 **Ctrl** + **z** 暫停行程。
- 在 Shell 中有暫停行程可用命令 `fg` 恢復前景執行，或是以命令 `bg` 讓其背景執行。

基本文字工具

- `cat`
  檢視檔案內容或是連續檢視檔案內容。
  簡易語法：`cat [選擇] [檔案 ...]`
  - 從檔案輸入

    ```
    # cat testfile1 testfile2
    ```
  - 加入行號顯示

    ```
    # cat -n testfile
    ```
  - 若無給予檔案則由標準輸入讀入，用 **Ctrl** + **d** 送出檔案結尾（EOF）結束輸入。

- `grep`
  從輸入的文件檔案或串流中，抓取符合給定字串的行。
  簡易語法：`grep [選擇] 樣式 [檔案 ...]`

- `head`

  將檔案的前面部份顯示至標準輸出（終端機），預設前10行。

  簡易語法：`head [選擇] [檔案 ...]`

  - 自訂輸出 20 行數。

    ```
    # head -n 20 testfile
    ```

  - 若無給予檔案則由標準輸入讀入。

- `tail`

  概念類似 `head` 將檔案的後面部份顯示至標準輸出（終端機），預設倒數10行。

  簡易語法：`tail [選擇] [檔案 ...]`

- `more`

  一個分頁顯示文字檔工具，以鍵盤命令操作，**b** 鍵為前一個頁面（螢幕行數），**d** 鍵為後一個頁面，**q** 鍵離開並結束程式。

  簡易語法：`more [選擇] [檔案 ...]`

- `less`
  類似 `more`，部份操作命令也相同，但可以一次只捲動一行，用 **j** 鍵往下一行，**k** 鍵往回前一行。
- 環境變數的 `PAGER` 用做為設定預設分頁檢視文件程式的設定，一般會設為用 `more` 或 `less` 。
- `wc` 計算檔案長度相關資訊（行數、字數與位元數）。
  簡易語法：`wc [選擇] [檔案 ...]`
  - 只取得行數
    ```
    # wc -l testfile
    ```
  - 只取得字數（以空白（whitespace）區隔）
    ```
    # wc -w testfile
    ```
  - 只取得位元數
    ```
    # wc -c testfile
    ```

導管與輸入／輸出重導向

- 標準輸入（stardard input），是指輸入至程式的串流資料，一般預設都是以鍵盤輸入，其檔案描述子為 0。
- 標準輸出（standard output），是指程式輸出的串流資料，一般預設都是以終端機為輸出，其檔案描述子為 1。
- 標準錯誤輸出（standard error），是指程式執行時錯誤訊息的輸出，也是預設以終端機為輸出，其檔案描述子為 2。
- 檔案描述子（file descriptor）是用抽象化代號的方式，去代表指向某檔案。

- 導管（pipe）或稱管線，是 UNIX-like 系統中用來將一個行程執行的標準輸出，連結到另一個程式行程的標準輸入。

  例１、`# ls /usr/bin | more`

  例２、`# ls /usr/bin | grep gcc | more`
- 輸出重導向覆寫檔案
  - 標準輸出：命令 `>` 檔案。

    例、`# ls > listfiles`
  - 標準錯誤輸出：命令 `2>` 檔案（sh/bash）。
- 輸出重導向從檔案最後附加
  - 標準輸出：命令 `>>` 檔案。
  - 標準錯誤輸出：命令 `2>>` 檔案（sh/bash）。

- 標準輸出與標準錯誤輸出的分離：

  用法：(命令 > 存正常的檔案) >& 存錯誤的檔案（csh/tcsh）

  命令 > 存正常的檔案 2> 存錯誤的檔案（sh/bash）

  例：比較下列結果。

```
# find /home -name '.cshrc' > list.out
# (find /home -name '.cshrc' > list.out) >& list.err
```

- 標準錯誤輸出重導向到標準輸出

  命令 &> 檔案

  命令 >& 檔案

  命令 > 檔案 2>&1

  例：`# find /home -name '.cshrc' &> list.all`

- `tee`

  同時將資料串流送到標準輸出與檔案。

  簡易語法：`tee [選擇] [檔案 ...]`

  ```
  # ls -lh / | tee filelist
  ```

- 輸入重導向：命令 < 檔案
- 字元重導向：命令 << 關鍵字

  例１、(csh/tcsh)
  ```
  # cat > testfile << EOF!
  Another redirection trying
  EOF!
  ```

  例２、(sh/bash)
  ```
  # cat > testfile << EOF
  Another redirection trying again
  EOF
  ```