Contents

[Chapter 1 入門篇 1](#_Toc2291242)

[Chapter 3: 終端機/命令提示字元 1](#_Toc2291243)

[Chapter 5: 開始使用Git 4](#_Toc2291244)

# Chapter 1 入門篇

**為什麼要學習Git?**

透過這樣的系統，可以清楚地記錄每一個檔案是**誰**在**什麼時候**加進來、**什麼時候被修改或刪除**

**Git的優缺點**

|  |  |
| --- | --- |
| **優點** | **缺點** |
| 免費、開源 | 指令非常多 |
| 速度快、體積小 |  |
| 分散式系統 |  |

**Git v.s. Github**

|  |  |
| --- | --- |
| Git | Github |
| * 版本控制軟體 | * 商業網站，其本體是一個Git伺服器 |

**Git與其他版本控制系統的差異**

1. 分散式系統: 就算沒有網路也可以在本地操作，待有網路後再與其他人同步即可
2. 處理檔案的方式: 用拍照(snapshot)的方式，來記錄每一次不一樣的地方，在每一次版本變化的時候，會更新並記錄整個目錄跟檔案的樹狀結構

# Chapter 3: 終端機/命令提示字元

**什麼事終端機?**

用來顯示及輸入資料的東西，終端機本身通常不是一部電腦

**常用命令**



使用Git時，**指令要在正確的目錄下才能正常運作**，所以學會目錄的切換很重要

**目錄切換即顯示**

# 往上一層目錄移動

$ cd ..

# 這個 “~” 符號表示home目錄

$ cd ~/projects/namecards

# 顯示所在目錄

$ pwd

**檔案列表**

# a 是指連小數點開頭的檔案(例如 .gitignore)也會顯示，l則是完整檔案的# 權限、擁有者以及建立、修改時間

$ ls -al

**建立檔案、目錄**

# 建立檔案

$ touch index.html

# 建立目錄

$ mkdir demo

**檔案操作**

# 把檔案index.html複製一份成 about.html

$ cp index.html about.html

# 把檔案更名為 about.html

$ mv index.html about.html

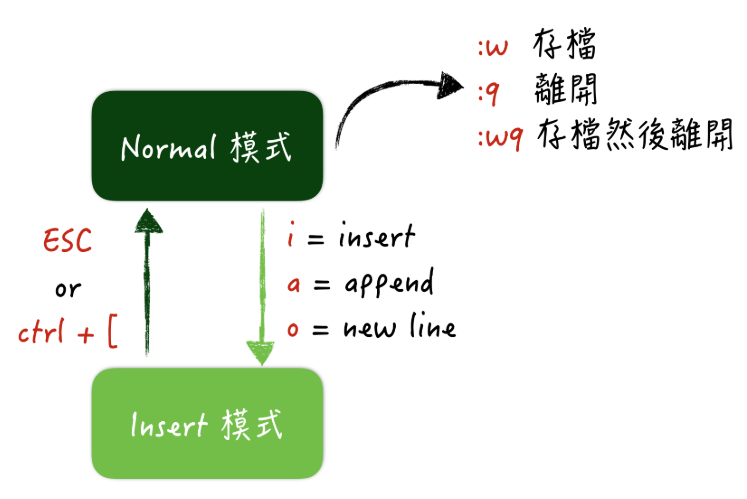
# 刪除檔案

$ rm index.html

# 刪除這個目錄裡所有的html檔案

$ rm \*.html

**簡易vim操作**



說明如下：

1. **Normal 模式，又稱命令模式，在這個模式下，無法輸入文字，僅能進行複製、貼上、存檔或離開動作**。
2. **要開始輸入文字，需要先按下 i、a 或 o 這三個鍵其中一個進入 Insert 模式，便能開始打字。**其中，i 表示 insert，a 表示 append，而 o 則是表示會新增一行並開始輸入。
3. 在 Insert 模式下，按下 ESC 鍵或是 Ctrl + [ 組合鍵，可退回至 Normal 模式。
4. 在 Normal 模式下，按下 :w 會進行存檔，按下 :q 會關閉這個檔案（但若未存檔會提示先存檔再離開），而 :wq 則是存檔完成後直接關閉這個檔案。

# Chapter 5: 開始使用Git

**新增、初始repository**

# 想針對本來就存在的目錄進行版控，到那個目錄執行git init

$ git init

#如果目錄不想再被Git控制，把.git目錄移除即可

**把檔案交給Git**

# 查詢現在目錄的狀態

$ git status

# 建立一個內容為”hello git”並命名為welcome.html的檔案

$ echo “hello git” > welcome.html

Untracked files 意思是這個檔案尚未被家道Git版本控制系統裡，還沒開始正式被Git追蹤

# 把welcome.html交給Git，讓Git開始追蹤它

$ git add welcome.html

現在welcome.html從untracked file變成new file狀態，表示這個檔案已經被放到暫存區(staging area) aka 索引(index)

# 一次把特定類型的檔案加到暫存區

$ git add \*.html

# 一次把**所有檔案**加到暫存區，在Git2.x後，這兩個指令功能變一樣

$ git add --all

$ git add .

**【狀況題】如果在 git add 之後又修改了那個檔案的內容？**

想像一下這個情境：

* 你新增了一個檔案叫做 abc.txt。
* 然後，執行 git add abc.txt 把檔案加至暫存區。
* 接著編輯 abc.txt 檔案。
* 完成編輯後，接著你可能會想要進行 Commit，把剛剛修改的內容存下來。這是新手可能會犯的錯誤之一，以為 Commit 指令就會把所有的異動都存下來，事實上這樣的想法是不太正確的。執行一下 git status 指令，看一下目前的狀態：

如果確定這個修改是你要的，就請再次使用 git add abc.txt 指令，再把 abc.txt 加至暫存區。

**把暫存區的內容提交到倉庫裡存檔**

如果僅是透過 git add 指令把異動加到暫存區是不夠的，這樣還不算是完成整個流程。要讓暫存區的內容永久的存下來的話，使用的是 git commit 指令：

$ git commit -m "init commit"

記得說明你這次commit什麼，若沒有加入訊息參數則會跳出vim

**Git每次只會處理暫存區裡的內容**

