

vim 程式編輯器

陳建良



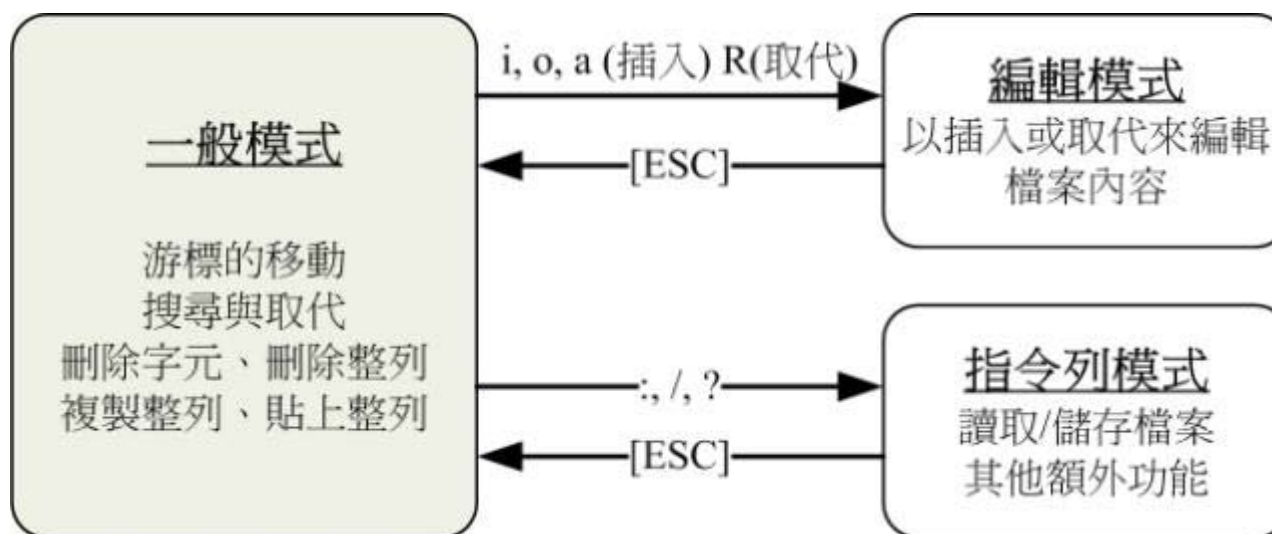
內容

- vi 與 vim
- vi 的使用
- vim 的額外功能
- 其他 vim 使用注意事項

vi 與 vim

- 所有的 **Unix Like** 系統都會內建 **vi** 文書編輯器，其他的文書編輯器則不一定會存在；
- 很多個別軟體的編輯介面都會主動呼叫 **vi** (例如未來會談到的 **crontab**, **visudo**, **edquota** 等指令)；
- **vim** 具有程式編輯的能力，可以主動的以字體顏色辨別語法的正確性，方便程式設計；
- 因為程式簡單，編輯速度相當快速。

vi 的使用

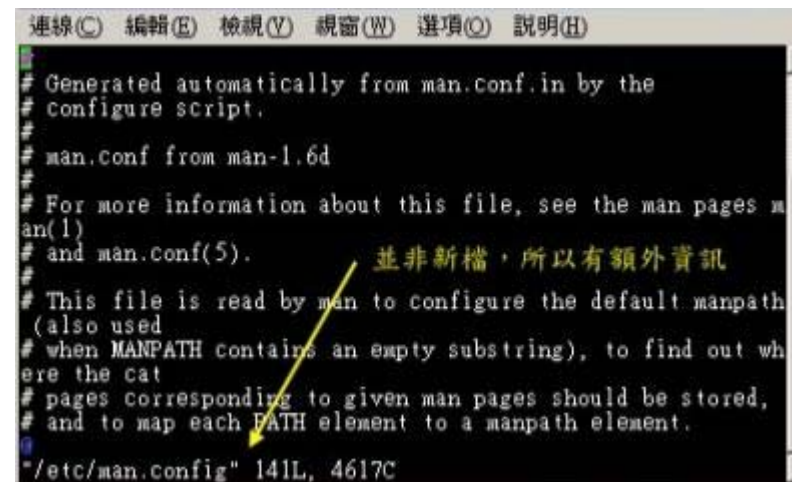


vi 三種模式的相互關係

簡易執行範例

■ 使用 vi 進入一般模式

```
[root@www ~]# vi test.txt
```





按下 [ESC] 按鈕回到一般模式

- 假設我已經按照上面的樣式給他編輯完畢了，那麼應該要如何退出呢？是的！沒錯！就是給他按下 [Esc] 這個按鈕即可！馬上你就會發現畫面左下角的 – INSERT – 不見了！

在一般模式中按下 :wq 儲存後離開 vi

[illegible]

儲存並離開 vi 環境

按鍵說明

- ✓ 第一部份：一般模式可用的按鈕說明，游標移動、複製貼上、搜尋取代等

移動游標的方法	
h 或 向左方向鍵(←)	游標向左移動一個字元
j 或 向下方向鍵(↓)	游標向下移動一個字元
k 或 向上方向鍵(↑)	游標向上移動一個字元
l 或 向右方向鍵(→)	游標向右移動一個字元
如果你將右手放在鍵盤上的話，你會發現 hjkl 是排列在一起的，因此可以使用這四個按鈕來移動游標。如果想要進行多次移動的話，例如向下移動 30 行，可以使用 "30j" 或 "30↓" 的組合按鍵，亦即加上想要進行的次數(數字)後，按下動作即可！	
[Ctrl] + [f]	螢幕『向下』移動一頁，相當於 [Page Down]按鍵 (常用)
[Ctrl] + [b]	螢幕『向上』移動一頁，相當於 [Page Up] 按鍵 (常用)
[Ctrl] + [d]	螢幕『向下』移動半頁
[Ctrl] + [u]	螢幕『向上』移動半頁

+	游標移動到非空白字元的下一列
-	游標移動到非空白字元的上一列
n<space>	那個 n 表示『數字』，例如 20 。按下數字後再按空白鍵，游標會向右移動這一行的 n 個字元。例如 20<space> 則游標會向後面移動 20 個字元距離。
0 或功能鍵[Home]	這是數字『0』：移動到這一行的最前面字元處 (常用)
\$ 或功能鍵[End]	移動到這一行的最後面字元處(常用)
H	游標移動到這個螢幕的最上方那一行的第一個字元
M	游標移動到這個螢幕的中央那一行的第一個字元
L	游標移動到這個螢幕的最下方那一行的第一個字元
G	移動到這個檔案的最後一行(常用)
nG	n 為數字。移動到這個檔案的第 n 行。例如 20G 則會移動到這個檔案的第 20 行(可配合 :set nu)
gg	移動到這個檔案的第一行，相當於 1G 啊！(常用)
n<Enter>	n 為數字。游標向下移動 n 行(常用)

搜尋與取代

/word	向游標之下尋找一個名稱為 word 的字串。例如要在檔案內搜尋 vbird 這個字串，就輸入 /vbird 即可！(常用)
?word	向游標之上尋找一個字串名稱為 word 的字串。
n	這個 n 是英文按鍵。代表『重複前一個搜尋的動作』。舉例來說，如果剛剛我們執行 /vbird 去向下搜尋 vbird 這個字串，則按下 n 後，會向下繼續搜尋下一個名稱為 vbird 的字串。如果是執行 ?vbird 的話，那麼按下 n 則會向上繼續搜尋名稱為 vbird 的字串！
N	這個 N 是英文按鍵。與 n 剛好相反，為『反向』進行前一個搜尋動作。例如 /vbird 後，按下 N 則表示『向上』搜尋 vbird 。
使用 /word 配合 n 及 N 是非常有幫助的！可以讓你重複的找到一些你搜尋的關鍵字！	
:n1,n2s/word1/word2/g	n1 與 n2 為數字。在第 n1 與 n2 行之間尋找 word1 這個字串，並將該字串取代為 word2 ！舉例來說，在 100 到 200 行之間搜尋 vbird 並取代為 VBIRD 則：『:100,200s/vbird/VBIRD/g』。(常用)
:1,\$s/word1/word2/g	從第一行到最後一行尋找 word1 字串，並將該字串取代為 word2 ！(常用)
:1,\$s/word1/word2/gc	從第一行到最後一行尋找 word1 字串，並將該字串取代為 word2 ！且在取代前顯示提示字元給使用者確認 (confirm) 是否需要取代！(常用)

刪除、複製與貼上

x, X	在一行字當中， x 為向後刪除一個字元 (相當於 [del] 按鍵)， X 為向前刪除一個字元(相當於 [backspace] 亦即是倒退鍵) (常用)
nx	n 為數字，連續向後刪除 n 個字元。舉例來說，我要連續刪除 10 個字元，『10x』。
dd	刪除游標所在的那一整列(常用)
ndd	n 為數字。刪除游標所在的向下 n 列，例如 20dd 則是刪除 20 列 (常用)
d1G	刪除游標所在到第一行的所有資料
dG	刪除游標所在到最後一行的所有資料
d\$	刪除游標所在處，到該行的最後一個字元
d0	那個是數字的 0，刪除游標所在處，到該行的最前面一個字元
yy	複製游標所在的那一行(常用)
nyy	n 為數字。複製游標所在的向下 n 列，例如 20yy 則是複製 20 列(常用)
y1G	複製游標所在列到第一列的所有資料
yG	複製游標所在列到最後一列的所有資料
y0	複製游標所在的那個字元到該行行首的所有資料
y\$	複製游標所在的那個字元到該行行尾的所有資料

p, P	p 為將已複製的資料在游標下一行貼上，P 則為貼在游標上一行！舉例來說，我目前游標在第 20 行，且已經複製了 10 行資料。則按下 p 後，那 10 行資料會貼在原本的 20 行之後，亦即由 21 行開始貼。但如果是按下 P 呢？那麼原本的第 20 行會被推到變成 30 行。(常用)
J	將游標所在列與下一列的資料結合成同一列
c	重複刪除多個資料，例如向下刪除 10 行，[10cj]
u	復原前一個動作。(常用)
[Ctrl]+r	重做上一個動作。(常用)
這個 u 與 [Ctrl]+r 是很常用的指令！一個是復原，另一個則是重做一次～ 利用這兩個功能按鍵，你的編輯，嘿嘿！很快樂的啦！	
.	不要懷疑！這就是小數點！意思是重複前一個動作的意思。如果你想要重複刪除、重複貼上等等動作，按下小數點『.』就好了！(常用)

一般模式切換到編輯模式的可用的按鈕說明

進入插入或取代的編輯模式	
i, I	進入插入模式(Insert mode)： i 為『從目前游標所在處插入』，I 為『在目前所在行的第一個非空白字元處開始插入』。(常用)
a, A	進入插入模式(Insert mode)： a 為『從目前游標所在的下一個字元處開始插入』，A 為『從游標所在行的最後一個字元處開始插入』。(常用)
o, O	進入插入模式(Insert mode)： 這是英文字母 o 的大小寫。o 為『在目前游標所在的下一行處插入新的一行』；O 為在目前游標所在處的上一行插入新的一行！(常用)
r, R	進入取代模式(Replace mode)： r 只會取代游標所在的那一個字元一次；R會一直取代游標所在的文字，直到按下 ESC 為止；(常用)
上面這些按鍵中，在 vi 畫面的左下角處會出現『--INSERT--』或『--REPLACE--』的字樣。由名稱就知道該動作了吧！！特別注意的是，我們上面也提過了，你想要在檔案裡面輸入字元時，一定要在左下角處看到 INSERT 或 REPLACE 才能輸入喔！	
[Esc]	退出編輯模式，回到一般模式中(常用)

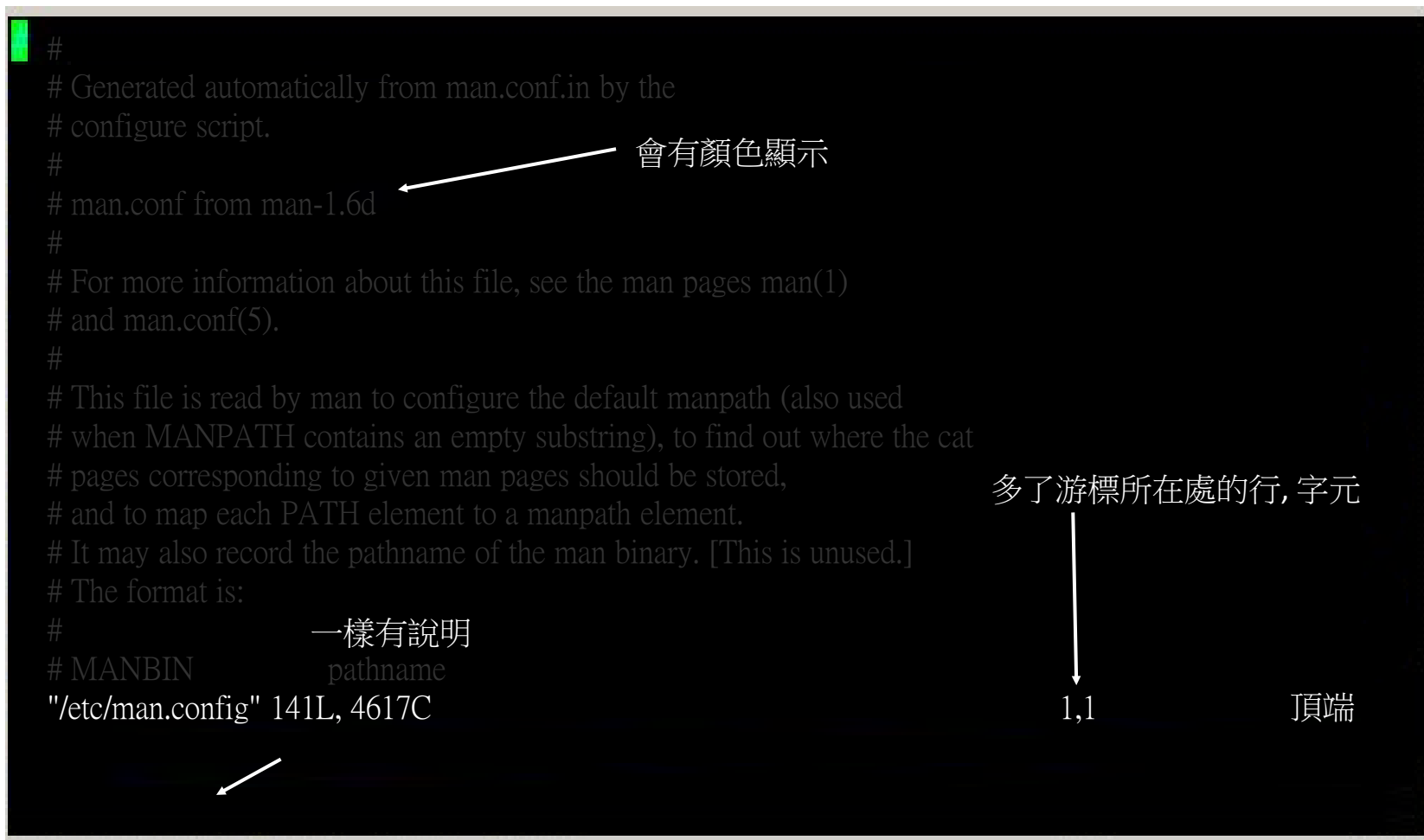
一般模式切換到指令列模式的可用的按鈕說明

指令列的儲存、離開等指令	
<code>:w</code>	將編輯的資料寫入硬碟檔案中(常用)
<code>:w!</code>	若檔案屬性為『唯讀』時，強制寫入該檔案。不過，到底能不能寫入，還是跟你對該檔案的檔案權限有關啊！
<code>:q</code>	離開 vi (常用)
<code>:q!</code>	若曾修改過檔案，又不想儲存，使用！為強制離開不儲存檔案。
注意一下啊，那個驚嘆號 (!) 在 vi 當中，常常具有『強制』的意思～	
<code>:wq</code>	儲存後離開，若為 <code>:wq!</code> 則為強制儲存後離開 (常用)
<code>ZZ</code>	這是大寫的 Z 喔！若檔案沒有更動，則不儲存離開，若檔案已經被更動過，則儲存後離開！
<code>:w [filename]</code>	將編輯的資料儲存成另一個檔案（類似另存新檔）
<code>:r [filename]</code>	在編輯的資料中，讀入另一個檔案的資料。亦即將『filename』這個檔案內容加到游標所在行後面
<code>:n1,n2 w [filename]</code>	將 n1 到 n2 的內容儲存成 filename 這個檔案。
<code>:! command</code>	暫時離開 vi 到指令列模式下執行 command 的顯示結果！例如『:! ls /home』即可在 vi 當中察看 /home 底下以 ls 輸出的檔案資訊！
vim 環境的變更	
<code>:set nu</code>	顯示行號，設定之後，會在每一行的字首顯示該行的行號
<code>:set nonu</code>	與 set nu 相反，為取消行號！

vim 的暫存檔、救援回復與開啟時的警告訊息

- **[O]pen Read-Only**：打開此檔案成為唯讀檔，可以用在你只是想要查閱該檔案內容並不要進行編輯行為時。一般來說，在上課時，如果你是登入到同學的電腦去看他的設定檔，結果發現其實同學他自己也在編輯時，可以使用這個模式；
- **(E)dit anyway**：還是用正常的方式打開你要編輯的那個檔案，並不會載入暫存檔的內容。不過很容易出現兩個使用者互相改變對方的檔案等問題！不好不好！
- **(R)ecover**：就是載入暫存檔的內容，用在你要救回之前未儲存的工作。不過當你救回來並且儲存離開 vim 後，還是要手動自行刪除那個暫存檔喔！
- **(D)elete it**：你確定那個暫存檔是無用的！那麼開啟檔案前會先將這個暫存檔刪除！這個動作其實是比較常做的！因為你可能不確定這個暫存檔是怎麼來的，所以就刪除掉它吧！哈哈！
- **(Q)uit**：按下 q 就離開 vim，不會進行任何動作回到命令提示字元。
- **(A)bort**：忽略這個編輯行為，感覺上與 quit 非常類似！也會送你回到命令提示字元就是囉！

vim 的額外功能



```
#  
# Generated automatically from man.conf.in by the  
# configure script.  
#  
# man.conf from man-1.6d  
#  
# For more information about this file, see the man pages man(1)  
# and man.conf(5).  
#  
# This file is read by man to configure the default manpath (also used  
# when MANPATH contains an empty substring), to find out where the cat  
# pages corresponding to given man pages should be stored,  
# and to map each PATH element to a manpath element.  
# It may also record the pathname of the man binary. [This is unused.]  
# The format is:  
#  
# MANBIN      pathname  
"/etc/man.config" 141L, 4617C
```

會有顏色顯示

多了游標所在處的行, 字元

1,1

頂端

一樣有說明

vim 的圖示示意

vim 的額外功能

```
# Every automatically generated MANPATH includes these fields
#
MANPATH /usr/man
MANPATH /usr/share/man
MANPATH /usr/local/man
MANPATH /usr/local/share/man
MANPATH /usr/X11R6/man
#
# Uncomment if you want to include one of these by default
```

43,2 30%

vim 的圖示示意

區塊選擇(visual Block)

```
192.168.1.1    host1.class.net
192.168.1.2    host2.class.net
192.168.1.3    host3.class.net
192.168.1.4    host4.class.net
.....中間省略.....
```

區塊選擇的按鍵意義	
v	字元選擇，會將游標經過的地方反白選擇！
V	行選擇，會將游標經過的行反白選擇！
[Ctrl]+v	區塊選擇，可以用長方形的方式選擇資料
y	將反白的地方複製起來
d	將反白的地方刪除掉

```
192.168.1.1  host1.class.net
192.168.1.2  host2.class.net
192.168.1.3  host3.class.net
192.168.1.4  host4.class.net
192.168.1.5  host5.class.net
192.168.1.6  host6.class.net
192.168.1.7  host7.class.net
192.168.1.8  host8.class.net
192.168.1.9  host9.class.net
~
-- VISUAL BLOCK --          1,16      All
```

游標移動到此再按下
[ctrl]+v

會出現這個訊息

進入區塊功能的示意圖

```
192.168.1.1  host1.class.net
192.168.1.2  host2.class.net
192.168.1.3  host3.class.net
192.168.1.4  host4.class.net
192.168.1.5  host5.class.net
192.168.1.6  host6.class.net
192.168.1.7  host7.class.net
192.168.1.8  host8.class.net
192.168.1.9  host9.class.net
~
-- VISUAL BLOCK --          9,20      All
```

用鍵盤將游標移動
到此處，畫面會跟
著反白喔！

區塊選擇的結果示意圖



```
192.168.1.1    host1.class.net  host1
192.168.1.2    host2.class.net  host2
192.168.1.3    host3.class.net  host3
192.168.1.4    host4.class.net  host4
192.168.1.5    host5.class.net  host5
192.168.1.6    host6.class.net  host6
192.168.1.7    host7.class.net  host7
192.168.1.8    host8.class.net  host8
192.168.1.9    host9.class.net  host9
~
1,33          All
```

將區塊的資料貼上後的結果

多檔案編輯

多檔案編輯的按鍵	
:n	編輯下一個檔案
:N	編輯上一個檔案
:files	列出目前這個 vim 的開啟的所有檔案

```
192.168.1.4    host4.class.net  host4
192.168.1.5    host5.class.net  host5
192.168.1.6    host6.class.net  host6
192.168.1.7    host7.class.net  host7
192.168.1.8    host8.class.net  host8
192.168.1.9    host9.class.net  host9
~
:files
 1 %a    "hosts"          line 1
 2      "/etc/hosts"      line 0
Press ENTER or type Command to continue
```

多檔案編輯示意圖

多視窗功能

```
# Decompress with given decompressor when input file has given ex
# version.
# The command given must act as a filter.
#
.gz          /usr/bin/gunzip -c
.bz2        /usr/bin/bzip2 -c -d
.z          /bin/zcat
.Z
.F
.Y
/etc/man.config 141,1 Bot
#
# Generated automatically from man.config.in by the
# configure script
#
man.config from man-1.6d
#
# For more information about this file, see the man pages man(1)
# and man.config(5)
#
@
/etc/man.config 1,1 Top
:sp
```

游標依舊在上面的視窗

其實是同一個檔案啦

游標位置並不相同

視窗分割的示意圖

多視窗功能



```
Do not remove the following line, or various programs
# that require network functionality will fail.
127.0.0.1          www.vbird.tsai www localhost.localdomain
localhost
::1               localhost6.localdomain6 localhost6
~
~
/etc/hosts          1,1          All
.gz                /usr/bin/gunzip -c
.bz2               /usr/bin/bzip2 -c -d
.z
.Z                /bin/zcat
.F
.Y
/etc/man.config     141,1       Bot
#
# Generated automatically from man.conf.in by the
# configure script.
# man.Conf from man-1.6d
/etc/man.config     1,1          Top
```

看吧！檔案不見得是相同的！

視窗分割的示意圖

多視窗功能

多視窗情況下的按鍵功能	
<code>:sp [filename]</code>	開啟一個新視窗，如果有加 filename ，表示在新視窗開啟一個新檔案，否則表示兩個視窗為同一個檔案內容(同步顯示)。
<code>[ctrl]+w+ j</code> <code>[ctrl]+w+↓</code>	按鍵的按法是：先按下 [ctrl] 不放，再按下 w 後放開所有的按鍵，然後再按下 j (或向下方向鍵)，則游標可移動到下方的視窗。
<code>[ctrl]+w+ k</code> <code>[ctrl]+w+↑</code>	同上，不過游標移動到上面的視窗。
<code>[ctrl]+w+ q</code>	其實就是 :q 結束離開啦！舉例來說，如果我想要結束下方的視窗，那麼利用 <code>[ctrl]+w+↓</code> 移動到下方視窗後，按下 :q 即可離開，也可以按下 <code>[ctrl]+w+q</code> 啊！

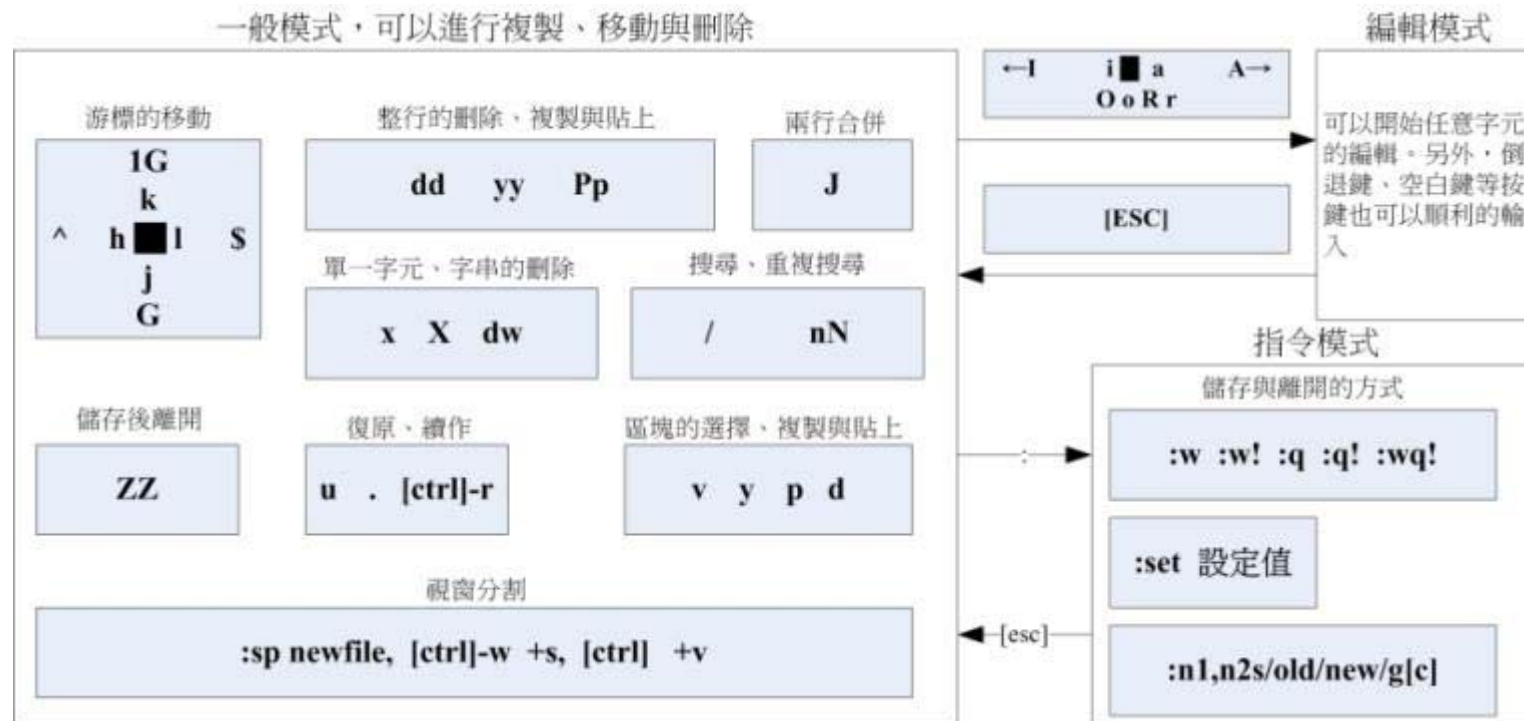
vim 環境設定與記錄： ~/.vimrc, ~/.viminfo

vim 的環境設定參數	
<code>:set nu</code> <code>:set nonu</code>	就是設定與取消行號啊！
<code>:set hlsearch</code> <code>:set nohlsearch</code>	hlsearch 就是 high light search (高亮度搜尋)。這個就是設定是否將搜尋的字串反白的設定值。預設值是 hlsearch
<code>:set autoindent</code> <code>:set noautoindent</code>	是否自動縮排？ autoindent 就是自動縮排。
<code>:set backup</code>	是否自動儲存備份檔？一般是 nobackup 的，如果設定 backup 的話，那麼當你更動任何一個檔案時，則原始檔案會被另存成一個檔名為 filename~ 的檔案。舉例來說，我們編輯 hosts ，設定 :set backup ，那麼當更動 hosts 時，在同目錄下，就會產生 hosts~ 檔名的檔案，記錄原始的 hosts 檔案內容
<code>:set ruler</code>	還記得我們提到的右下角的一些狀態列說明嗎？這個 ruler 就是在顯示或不顯示該設定值的啦！
<code>:set showmode</code>	這個則是，是否要顯示 --INSERT-- 之類的字眼在左下角的狀態列。
<code>:set backspace=(012)</code>	一般來說，如果我們按下 i 進入編輯模式後，可以利用倒退鍵 (backspace) 來刪除任意字元的。但是，某些 distribution 則不許如此。此時，我們就可以透過 backspace 來設定囉～當 backspace 為 2 時，就是可以刪除任意值； 0 或 1 時，僅可刪除剛剛輸入的字元，而無法刪除原本就已經存在的文字了！

<code>:set all</code>	顯示目前所有的環境參數設定值。
<code>:set</code>	顯示與系統預設值不同的設定參數，一般來說就是你有自行變動過的設定參數啦！
<code>:syntax on</code> <code>:syntax off</code>	是否依據程式相關語法顯示不同顏色？舉例來說，在編輯一個純文字檔時，如果開頭是以 <code>#</code> 開始，那麼該行就會變成藍色。如果你懂得寫程式，那麼這個 <code>:syntax on</code> 還會主動的幫你除錯呢！但是，如果你僅是編寫純文字檔案，要避免顏色對你的螢幕產生的干擾，則可以取消這個設定。
<code>:set bg=dark</code> <code>:set bg=light</code>	可用以顯示不同的顏色色調，預設是『light』。如果你常常發現註解的字體深藍色實在很不容易看，那麼這裡可以設定為 <code>dark</code> 喔！試看看，會有不同的樣式呢！

```
[root@www ~]# vim ~/.vimrc
"這個檔案的雙引號 (") 是註解
set hlsearch           "高亮度反白
set backspace=2        "可隨時用倒退鍵刪除
set autoindent         "自動縮排
set ruler              "可顯示最後一行的狀態
set showmode           "左下角那一行的狀態
set nu                 "可以在每一行的最前面顯示行號啦！
set bg=dark            "顯示不同的底色色調
syntax on              "進行語法檢驗，顏色顯示。
```

vim 常用指令示意圖



vim 常用指令示意圖

中文編碼的問題

- 你的 Linux 系統預設支援的語系資料：這與 `/etc/sysconfig/i18n` 有關；
- 你的終端介面 (`bash`) 的語系：這與 `LANG` 這個變數有關；
- 你的檔案原本的編碼；
- 開啟終端機的軟體，例如在 `GNOME` 底下的視窗介面。

DOS 與 Linux 的斷行字元

```
[root@www ~]# dos2unix [-kn] file [newfile]
[root@www ~]# unix2dos [-kn] file [newfile]
```

選項與參數：

- k : 保留該檔案原本的 mtime 時間格式 (不更新檔案上次內容經過修訂的時間)
- n : 保留原本的舊檔，將轉換後的內容輸出到新檔案，如： `dos2unix -n old new`

範例一：將剛剛上述練習的 `/tmp/vitest/man.config` 修改成為 dos 斷行

```
[root@www ~]# cd /tmp/vitest
[root@www vitest]# cp -a /etc/man.config .
[root@www vitest]# ll man.config
-rw-r--r-- 1 root root 4617 Jan  6  2007 man.config
[root@www vitest]# unix2dos -k man.config
unix2dos: converting file man.config to DOS format ...
# 螢幕會顯示上述的訊息，說明斷行轉為 DOS 格式了！
[root@www vitest]# ll man.config
-rw-r--r-- 1 root root 4758 Jan  6  2007 man.config
# 斷行字元多了 ^M，所以容量增加了！
```

範例二：將上述的 `man.config` 轉成 `man.config.linux` 的 Linux 斷行字元

```
[root@www vitest]# dos2unix -k -n man.config man.config.linux
dos2unix: converting file man.config to file man.config.linux in UNIX
format ...
[root@www vitest]# ll man.config*
-rw-r--r-- 1 root root 4758 Jan  6  2007 man.config
-rw----- 1 root root 4617 Jan  6  2007 man.config.linux
```

語系編碼轉換

```
[root@www ~]# iconv --list
[root@www ~]# iconv -f 原本編碼 -t 新編碼 filename [-o newfile]
```

選項與參數：

- list : 列出 iconv 支援的語系資料
- f : from , 亦即來源之意，後接原本的編碼格式；
- t : to , 亦即後來的新編碼要是什麼格式；
- o file : 如果要保留原本的檔案，那麼使用 -o 新檔名，可以建立新編碼檔案。

範例一：將 /tmp/vitest/vi.big5 轉成 utf8 編碼吧！

```
[root@www ~]# cd /tmp/vitest
[root@www vitest]# iconv -f big5 -t utf8 vi.big5 -o vi.utf8
[root@www vitest]# file vi*
vi.big5: ISO-8859 text, with CRLF line terminators
vi.utf8: UTF-8 Unicode text, with CRLF line terminators
# 是吧！有明顯的不同吧！^_^
```