LINUX 作業系統實務
04. Vim

2020 TKU

Sherry Yin

補充一些常識 - Grub

- Grub, LILO是linux內建的多重開機管理程式
- Grand Unified Bootloader由Erich Boleyn設計並實作 (www.gnu.org)
- 預設讀取 /boot/grub/menu.lst設定檔 (連接到 /boot/grub/grub.conf)

Menu.lst

default 0 - 指定逾時過期時要啟動哪個項目。

timeout 10 - 指定啟動預設項目之前等待使用者輸入的時間 (秒)。

Splashimage=(hd0, 1)/boot/grub/splash.xpm.gz – 開機背景圖片

Hiddenmenu – 隱藏多重開機選單 預設等待秒數後 自動啟動預設開機的作業系統

title Solaris – 選單上顯示的名稱

  root (hd0,0,a) - 存放系統核心的分割區 (第一顆硬碟的第一個分區, 磁碟片段)

  kernel /platform/i86pc/multiboot -B console=ttya - 指定要載入的核心及掛載系統的根目錄

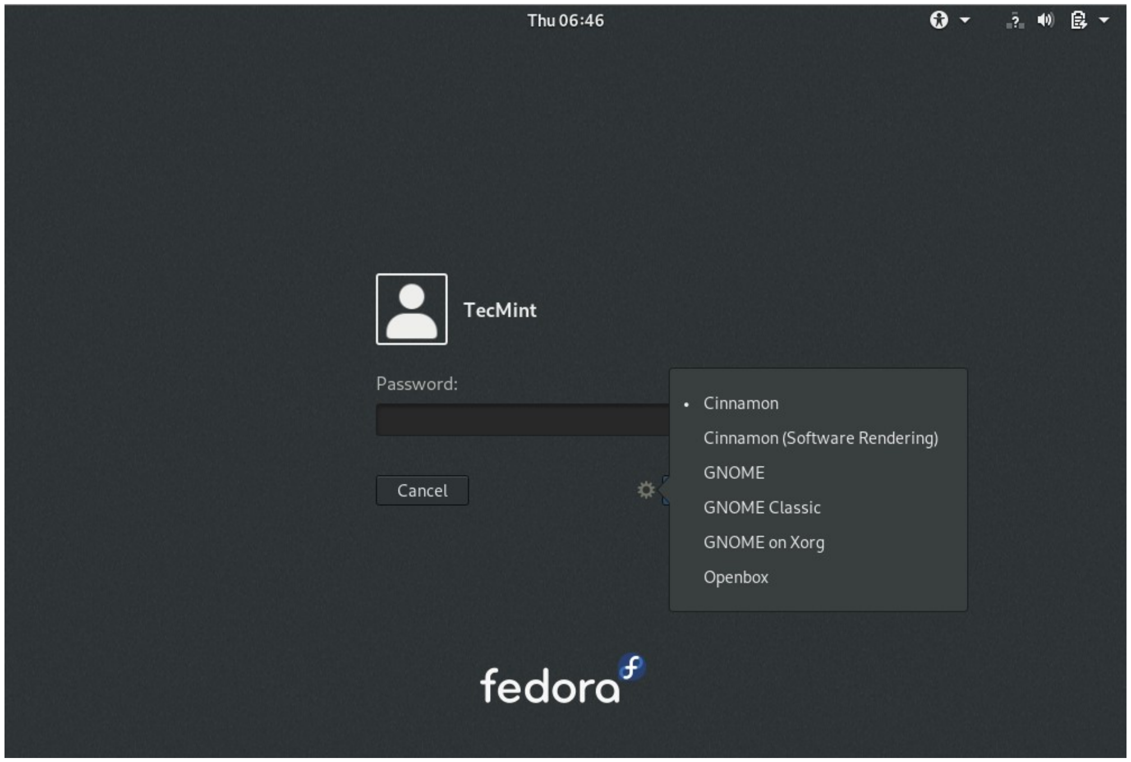
  module /platform/i86pc/boot_archive

#----- second_disk - ADDED BY LIVE UPGRADE - DO NOT EDIT -----

https://docs.oracle.com/cd/E19253-01/819-7792/gavhe/index.html

```
title second_disk
  root (hd0,1,a)
  kernel /platform/i86pc/multiboot
  module /platform/i86pc/boot_archive
title second_disk failsafe
  root (hd0,1,a)
  kernel /boot/multiboot kernel/unix -s
  module /boot/x86.miniroot-safe
#----- second_disk -------------- END LIVE UPGRADE ------------
title Windows
  root (hd0,0)
  chainloader -1 – 啟動該分割區中第一個磁區裡的開機程式
```

SwitchDesk

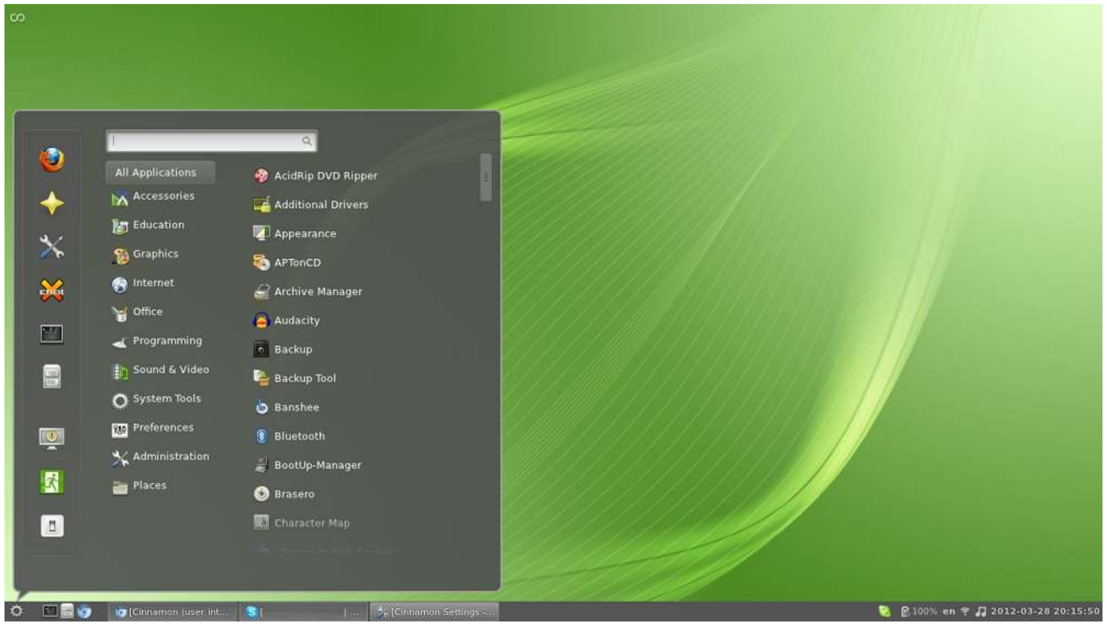

Thu 06:46
TecMint
Password:
Cancel
• Cinnamon
Cinnamon (Software Rendering)
GNOME
GNOME Classic
GNOME on Xorg
Openbox
fedora

switchdesk指令

- Home目錄下會產生.Xclients檔案, 該檔案室用來指定以startx指令進入X Window時要啟動的桌面環境或視窗管理程式
- Hint: The SDDM display manager has its own preferences file. It is under /etc/sddm.conf
- 如無法進入X Window, 可嘗試移除.Xclients檔案, 下次再進入視窗環境就會採用預設GNOME環境.

切換文字模式

- 在X Window中開啟文字模式視窗 – 應用程式鈕, 附屬應用程式/終端機



其他方法

- 切換虛擬主控台 – Ctrl + Alt + Fn (F1 – F7)
- 開機直接進入文字模式: 開啟 /etc/inittab檔案,

/etc/inittab

id : 3 : initdefault : - 如為3則是開機直接進入文字模式 action "initdefault"告訴 init 將系統開至Run Level 3

si : : sysinit: /etc/rc.d/rc.sysinit - runlevel 欄位空白告訴init不管系統開到哪一個Run Level都要執行這一項目(rc.sysinit)

action "sysinit"告訴init優先執行這個項目。

"sysinit"將忽略runlevel欄位。

l0 : 0 : wait: /etc/rc.d/rc 0

l1 : 1 : wait: /etc/rc.d/rc 1

l2 : 2 : wait: /etc/rc.d/rc 2

l3 : 3 : wait: /etc/rc.d/rc 3

l4 : 4 : wait: /etc/rc.d/rc 4

l5 : 5 : wait: /etc/rc.d/rc 5

l6 : 6 : wait: /etc/rc.d/rc 6

ud : : once : /sbin/update

ca : : ctrlaltdel : /sbin/shutdown -t3 -r now action "ctrlaltdel"告訴init當系統收到SIGINT(or User同時按下Alt+Ctrl+Del時)，執行這個項目(/sbin/shutdown -t3 -r now)。

pf : : powerfail :/sbin/shutdown -f -h +2 "Power Failure; System Shutting Down"

pr : 12345 : powerokwait:/sbin/shutdown -c "Power Restored; Shutdown Cancelled"

1 : 2345 : respawn : /sbin/mingetty tty1

2 : 2345 : respawn : /sbin/mingetty tty2

3 : 2345 : respawn : /sbin/mingetty tty3

4 : 2345 : respawn : /sbin/mingetty tty4

5 : 2345 : respawn : /sbin/mingetty tty5 表示在進入RunLevel 2/3/4/5/6時執行 "/sbin/mingetty tty5 "，action "respawn"表示在一個session結束後(User由logout Console tty5)，在執行一次"/sbin/mingetty tty5 "。

6 : 2345 : respawn : /sbin/mingetty tty6

x : 5 : respawn : /etc/X11/prefdm -nodaemon

如/etc/inittab設定錯誤

- 無法開機
- 當使用Grub為開機管理程式, 可在開機顯示選單畫面時按任意鍵, 再按[a]鍵, 並在指令列輸入以下參數進入單人模式:
  - grub append> ro root=LABEL=/1 rhgb quiet **s**

關機選項

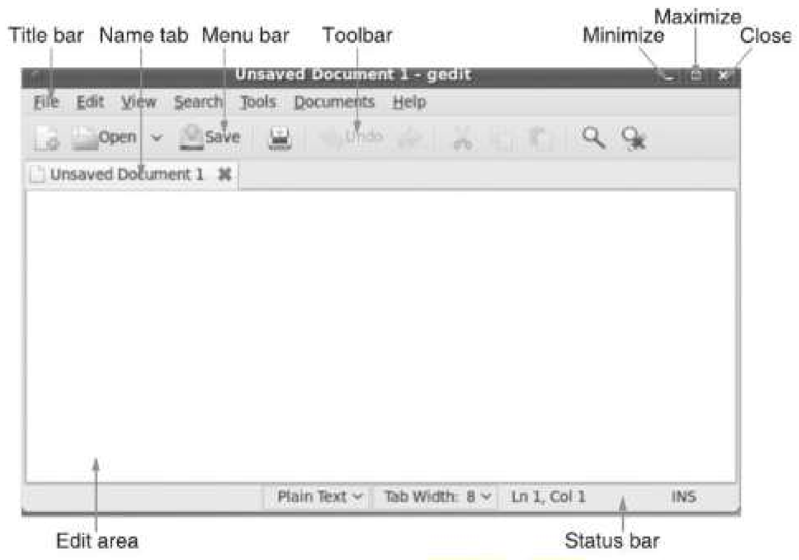
- 立即關機: shutdown –h now
- 指定時間關機: shutdown -h +5   (5分鐘後關機)
- 指定關機前通知訊息: shutdown –h +5 “system will shutdown after 5 minutes”
- 關機後重新開機: shutdown –r now (立即關機並重開機)
  - Shutdown –r 23:59 & 指定 23:59重開機
- 取消關機: shutdown –c
- 嚇一嚇使用者: shutdown –k 18:30
- 查詢關機紀錄: last –x shutdown

Disable Ctrl+Alt+Del重開機

- 修改/etc/inittab檔案:
  - #ca::ctrlaltdel:/sbin/shutdown –t3 –r now
  - (把此行註釋起來)

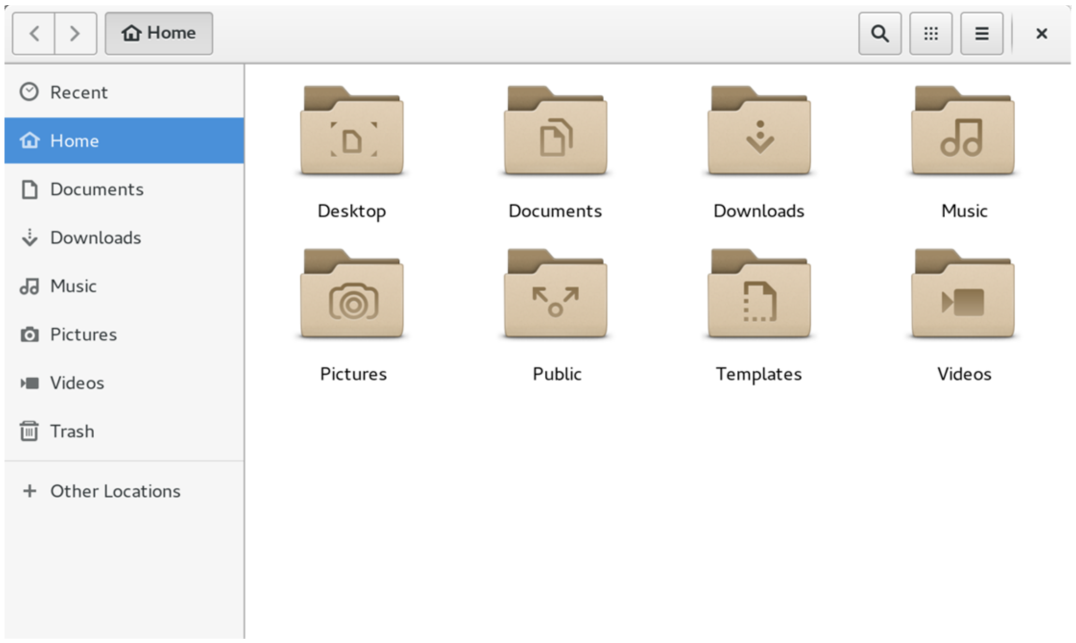
X window中的Gedit

- gedit是一個GNOME桌面環境下相容UTF-8的文字編輯器。它簡單易用，有良好的語法突顯，對中文支援很好，支援包括GB2312、GBK在內的多種字元編碼。gedit是一款自由軟體。



GNOME Files (Nautilus)瀏覽器開啟Gedit

- 直接雙擊檔案
- 可勾選顯示隱藏檔案
- 可勾選顯示行號



最容易上手的nano (pico類似)

- nano是Unix和類Unix系統中的一個文字編輯器，是Pico的複製品（clone）。
- nano最早在1999年由Chris Allegretta釋出，名字叫TIP（TIP isn't Pico）。2000年1月10日，此軟體正式改名為nano。
- nano這個名字來自於國際單位制詞頭nano，意思是nano是pico的一千倍。
- 後來，nano增加了一些Pico所沒有的功能，比如支援彩色的文字（語法突顯）、正規表示式搜尋和替換、平滑捲動、支援多個緩衝區。
- nano，像Pico一樣，是面向鍵盤的，它通過Control鍵來控制。

vim的歷史

- vi是一種電腦純文字編輯器，由美國計算機科學家比爾·喬伊（Bill Joy）完成編寫，並於1976年以BSD協定授權發布。
- vi是「Visual」的不正規的縮寫。
- 從2006年開始，作為「單一UNIX規範」（Single UNIX Specification）的一部分，vi或vi的一種變形版本一定會在UNIX中找到。
- 直到現在，vi仍然被廣泛的使用，並且贏得1991年在USENET的票選；vi比Emacs的Bulkier版本啟動的更快，並且占記憶體更少。
- 當救急軟碟作為恢復硬碟崩潰的媒介以來，vi通常被用戶選擇，因為一張軟碟正好儲存下vi，並且幾乎所有人都可以很輕鬆的使用vi。
- Vim（Vi IMproved）是一種升級版，類似nvi。在大多數Linux系統中都安裝了Vim。

3 modes in vi

vi sometext

    ~: means nonexistent lines

Command Mode: move cursor, can’t enter or replace text. h (left) and l(right), how about 5h and 3l?

Input Mode: press ‘i’, enter text, use [Ctrl-w] to erase the entire word

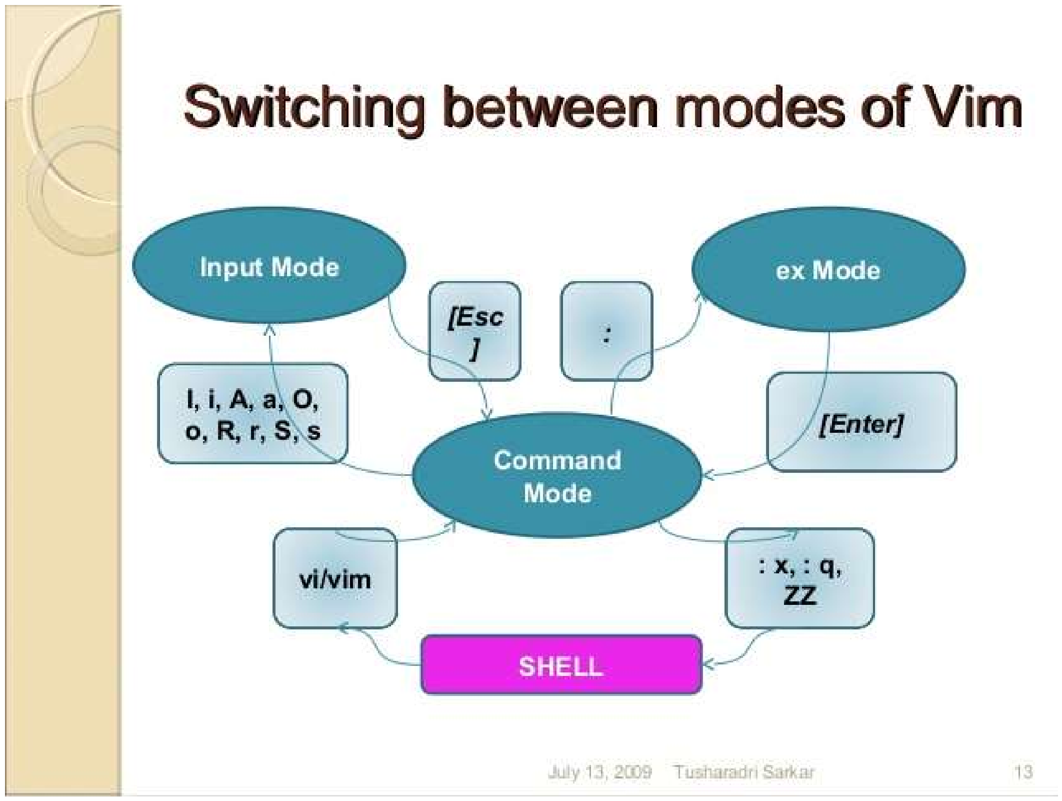
    press [Esc] key to return to Command Mode.

ex Mode or Last Line Mode: enter a ‘:’ under Command Mode. To save the entered text from the buffer to a file.

Usage of each mode

- Command mode: the default mode. Every key pressed is a command to run on text. Navigation, copying, and deleting.
- Input mode:
  - i: inserts text to left of cursor
  - a: appends text to right of cursor
  - I: inserts text at beginning of line
  - A: appends text at end of line
  - o: opens line below
  - O: opens line above
  - rch: replaces single character under cursor with ch
  - R: replaces text from cursor to right
  - s: replaces single character under cursor with any number of characters
  - S: replaces entire line

- ex Mode or Last Line Mode: file handling and performing substitution. (vi was originally hard-linked to a line editor named ex)


Switching between modes of Vim
Input Mode
[Esc]
:
ex Mode
I, i, A, a, O, o, R, r, S, s
Command Mode
[Enter]
vi/vim
: x, : q, ZZ
SHELL
July 13, 2009 Tusharadri Sarkar 13

A few tips

- Repeat factor is a command prefix to repeat the command as many times as the prefix.
  - E.g. k moves the cursor up one line, 10k moves 10 lines.
- Undo: press [Esc] and then u to undo the last action. In linux use [Ctrl-r].
- [Ctrl-l]: clear the screen.
- Better avoid using [CapsLock].
- Don’t use the PC navigation keys. Up, down, left, right, [page up] and [page down].
- Check the TERM variable to confirm the control sequences.

Insert, append and replace

- i: to the left of the cursor
- a: to the right of the cursor
- o: opens a new line below the current line
- O: opens a new line above the current line
- r: replace current cursor with a single character, no [Esc] required
- s: replaces one character with many
- R: replaces all text on the right of the cursor
- S: replaces the entire line

Text completion

- For example, to enter printf, enter pr, then press
  - [Ctrl-p]
- If there are other words, keep pressing [Ctrl-f] shows all matching words, use [Ctrl-n] to see the list backwards.

ex mode

There are three operations that we perform with the file in buffer:

1. Save and continue editing (:w)
2. Save and exit (:x and :wq)
3. Abandon all changes and quit (:q and :q!)

:w n2w.pl – Save As…

:n1, n2w build.sql – Writes lines n1 to n2 to file build.sql

:.w build.sql – writes current line to file build.sql ($is the last line)

:e note1 – stops editing current file and edits file note1

:e! – loads last saved edition of current file (like revert)

:e# - returns to editing most recently edited file

Navigation

- h (or [Backspace], l (or [spacebar]), k, j
- Left, right, up, down
- [Ctrl-f]: scrolls full page forward, e.g. 5[Ctrl-f]
- [Ctrl-b]: scrolls full page back
- [Ctrl-d]: scrolls half page forward
- [Ctrl-u]: scrolls half page back
- [Ctrl-l]: redraws the screen (no repeat factor)

Word and line navigation

- Word:
  - b: moves back to beginning of word, e.g. 4b – moves back four words to beginning of word
  - w: moves forward to beginning of word
- Line:
  - 0 or |: moves to beginning of line
  - 30|: moves to column 30
  - ^: moves to first word in line
  - $: moves to end of line
  - 1G: moves to beginning of buffer
  - 40G: moves to line 40
  - G: moves to end of buffer

Simple editing functions

- x: deletes a single character
- X: erases all text to the beginning of the line
- dd: deletes a line
- p: put the deleted or copied text on right of the cursor
- P: put the deleted or copied text on left of the cursor
- J: 4J joins following 3 lines with current one
- ~: the tilde, upper case becomes lower, vise versa.

delete, copy and move

- d: delete
- d$ or D: deletes from cursor to end of line
- 5dd: deletes five lines
- d/}: deletes from cursor up to first occurrence of }
- d30G: deletes from cursor up to line number 30
- y: yank (copy), 3yw or y3w – yanks three words from cursor position
- 5yy: yanks five lines
- yG: yanks from cursor to end of file
- y?case: yanks from cursor up to first occurrence of string case in reverse direction
- c: change, text within the changing range will be displayed in $s.

From one file to another

- :e foo – if the current file is saved, switch to another file.
- “a: access the buffer, e.g. “ayy: copy a line to this buffer
- “ap: place the copied text from buffer a below the current line

Undo and repeat

- u: single-level undoing
- U: reverses all changes made to the current line, note that you have to stay on the current line to invoke this command.

Search

- /: searches forward
- ?: searches backward
- n: repeats search in same direction of original search
- N: repeats search in direction opposite to that along which previous search was made
- :n1,n2s/s1/s2 – replaces first occurrence of string or regular expression s1 with string s2 in lines n1 to n2
- :1,10s/find/look/g – replaces all occurrences of find with look in lines 1 to 10
- :.,$s/find/look/gc – interactively replaces find with look from current line throught end
- :s – repeats last substitution on current line

Repeat

- . – the dot command is used for repeating both Input and Command mode commands. E.g. when you have deleted two lines of text with 2dd, then position the cursor at the desired location and press '.'
- Use u to undo this repeat.

Substitution – search and replace

:address/source_pattern/target_pattern/flags

:1,$s/double/float/g – double is replaced with float globally throughout the file.

:3,10s/msg/message/g – all occurrences in lines 3 through 10

:$s/msg/message/g – all occurrences in last line

:.s/echo/printf/ - only first occurrence in current line

:1,$s/message/msg/gc – interactively replace a string by adding the c (confirmatory) parameter

Customize vi

- :set – set vi variables
  - :set autoindent or :set ai – places the cursor in the next line at the current indentation
  - :set nonumber – hides all line numbers
  - :set ignorecase – the search commands pursue a case-insensitive search
  - :set showmatch – when enter a) or }, the cursor will jump to its matching counterpart and stay there for a second then return to current location.

Settings

- The settings are permanent only when they are placed in ~/.exrc or ~/.vimrc in linux.
  - aw: autowrite, write current file automatically whenever switching files with :e
  - ic: ignore case
  - magic: treats regular expression characters as special when search
  - nu: displays line numbers
  - showmode: displays a message when vi is in input mode
  - ts: tabstop, sets tabs for display (default: eight spaces)
  - ws: wrapscan, continues pattern search by moving to other end of a file so that entire file is scanned

Map

- :map – lets you assign a set of keystrokes to a key. E.g. :w saves the buffer
  - :map g :w^M (^M means the [Enter] key, it is [Ctrl-m]. This character can be entered by first pressing [Ctrl-v] and then [Ctrl-m]. This means you can press g in command mode to save the buffer.
  - #2 means the function key [F2]
  - :map! #2 ^[:w^M - ^[is the [Esc] character which switches to Command mode before :w saves the buffer.
  - :unmap cancels a Command mode map
  - :unmap! Cancels an Input Mode map

Define abbreviations

:ab – the abbreviate command, is used to expand short strings to long words.

:ab pf printf – when enter pf, followed by a key which is neither alphanumeric or _, pf gets expanded to printf.

```
:ab incstd #include <stdio.h>
:ab sopl System.out.println
:ab psvm public static void main (String args[])
```

- All sets, maps and abbreviations are stored in $HOME/.exrc (.vimrc for vim), carry this file with you.

編輯軟體的編碼設定

- X window: 終端機 /設定字元編碼/中文
- LANG=zh_TW.Big5
- Putty: Window – translation – received data assumed to be in which character set → UTF-8 (如編輯Big5編碼的檔案 則須改為 Use font encoding)

Fun time!

- https://www.openvim.com/
- https://vim-adventures.com/
- http://www.vimgenius.com/
- https://www.vim.org/download.php
- https://www.tecmint.com/learn-vi-commands-with-pacvim-game/

一個案例練習

- 請在 /tmp 這個目錄下建立一個名為 vitest 的目錄；
- 進入 vitest 這個目錄當中；
- 將 /etc/man_db.conf 複製到本目錄底下(或由上述的連結下載 man_db.conf 檔案)；
- 使用 vi 開啟本目錄下的 man_db.conf 這個檔案；
- 在 vi 中設定一下行號；
- 移動到第 43 列，向右移動 59 個字元，請問你看到的小括號內是哪個文字？
- 移動到第一列，並且向下搜尋一下『 gzip 』這個字串，請問他在第幾列？
- 接著下來，我要將 29 到 41 列之間的『小寫 man 字串』改為『大寫 MAN 字串』，並且一個一個挑選是否需要修改，如何下達指令？如果在挑選過程中一直按『y』， 結果會在最後一列出現改變了幾個 man 呢？
- 修改完之後，突然反悔了，要全部復原，有哪些方法？
- 我要複製 66 到 71 這 6 列的內容(含有MANDB_MAP)，並且貼到最後一列之後；
- 113 到 128 列之間的開頭為 # 符號的註解資料我不要了，要如何刪除？
- 將這個檔案另存成一個 man.test.config 的檔名；
- 去到第 25 列，並且刪除 15 個字元，結果出現的第一個單字是什麼？
- 在第一列新增一列，該列內容輸入『I am a student...』；
- 儲存後離開吧！

步驟

- 『mkdir /tmp/vitest』
- 『cd /tmp/vitest』
- 『cp /etc/man_db.conf .』
- 『/bin/vi man_db.conf』
- 『:set nu』然後你會在畫面中看到左側出現數字即為行號。
- 先按下『43G』再按下『59→』會看到『 as 』這個單字在小括號內；
- 先執行『1G』或『gg』後，直接輸入『/gzip』，則會去到第 93 列才對！
- 直接下達『 :29,41s/man/MAN/gc 』即可！若一直按『y』最終會出現『在 13 列內置換 13 個字串』的說明。
- (1)簡單的方法可以一直按『 u 』回復到原始狀態，(2)使用不儲存離開『 :q! 』之後，再重新讀取一次該檔案；
- 『66G』 然後再『 6yy 』之後最後一列會出現『複製6列』之類的說明字樣。 按下『 G 』到最後一列，再給他『 p 』貼上6列！
- 因為 113~128 共 16 列，因此『 113G 』→『 16dd 』就能刪除 16 列，此時你會發現游標所在 113 列的地方變成 『 # Flags. 』 開頭囉
- 『 :w man.test.config 』，你會發現最後一列出現 "man.test.config" [New].. 的字樣。
- 『25G』 之後，再給他『 15x 』即可刪除 15 個字元，出現『 tree 』的字樣；
- 先『 1G 』去到第一列，然後按下大寫的『 O 』便新增一列且在插入模式；開始輸入『I am a student...』後， 按下[Esc]回到一般指令模式等待後續工作；
- 『:wq』