Why bother with an oldish editor?...well it's

- Great & fast once used to it
  - Easy on man and machine
    - Home keys faster than arrows or mice
    - Less risk of repetitve strain injury (RSI)
    - Small memory & processing load
    - Works over ssh – no need for remote desktop
- So good it became great : fans added to it
  - Everything, everywhere for everyone
    - On all main platforms :
    - On all major editors and IDE's : VisualStudio, Eclipse
    - Even on browsers : Firefox, Chrome

Visual Editor (Improved) – vi(m)

- Vi(m) is fast for man and machine
  - for machine : small memory footprint, minimal graphics : fast
    - Emacs is slow to download, install, startup & load and run
  - And human : small hand movements, minimal delay and pain.
    - Emacs requires lots of keychords… ok if you have big flexible fingers…
- Vim : key idea => keyboard is fastest as you're keyed in
  - All movement & selection done by keys when in normal mode
  - Experienced users can fly compared even to GUI
    - GUI
      - Few seconds to get to mouse click and back
      - Even a second to get to arrow keys and back
    - Vim
      - fraction of a second for both
- Vim – should you learn it… depends on your needs…know it exists, and if you ever need a go anywhere, do anything editor, it's there !

Visual Editor (Improved) – vi(m)

- Less risk of RSI (repetitive strain injury) from
  - mouse & arrow keypad
  - hands are in natural position
  - Neck, shoulder & arm musculoskeletal system is not strained by reaching
  - (basically hanging your arm out all day stresses arm, shoulder & neck)
- As an aside : curse of the GUI : Grab a User Injury
  - Many (even mathematicians) have had to switch mouse hand
  - Or use a trackball : radiographers, submariners, even WebKiosks at transit stations :
- But the CLI guys Clatter Lithely Indefinitely
  - luddites won out in the end …this new-fangled ones were mangled

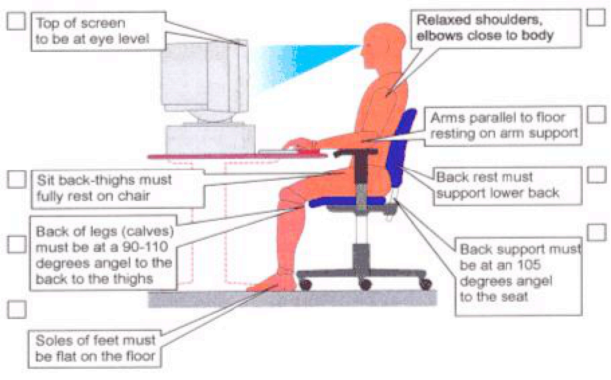
For more info and links : http://www.vim.org/

- Especially the Wiki
  - With tips
    - http://vim.wikia.com/wiki/Vim_Tips_Wiki
  - And documentation
    - http://vim.wikia.com/wiki/Vim_documentation
- Videos – had a brief glance…
  - Youtube
    - Vim for Windows series seems good.
    - 7 Habits for Effective TextEditing 2.0 – by the vim developer.
  - Vimeo : from vim.org (great if you're taking it easy!)
    - http://www.derekwyatt.org/vim/vim-tutorial-videos/

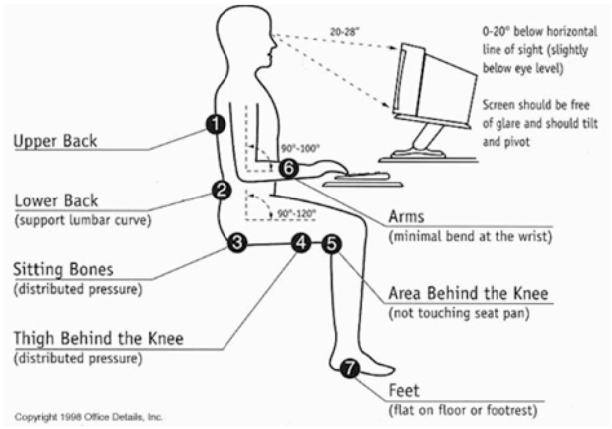
RSI – Recommended Seating Instructions !?

- Top of screen at eye level, or a little below
  - So your neck is straight
- Keyboard within easy reach
  - No stressful stretching
  - Arms close to body
  - Palms may rest during inactivity
    - Some fast typists require hands hovering over keyboard
- Sitting comfortably
  - All bent joints at right angles, Not just the right angle
    - Elbows, hip, knees, ankles.

RSI avoidance advice…



More RSI avoidance advice



Stuff the right angles – hang free on the knee!

Visual Editor (Improved) – vi(m) is

- a screen editor,
  - the unit of change is *character* rather than a line, but can switch to alternative modes, such as word, line & block
- so fast and flexible that it became famous
  - So it was extended, amended, and became feature rich
    - It may seem daunting to learn
    - But the basics are easy, and practice makes perfect, 'poetry in motion'!?
- found on **ALL** Unix/Linux/Mac installations
  - Some ICT multinationals use vi, (not vim even!) exclusively for just that reason!
  - ICT uses Linux : web, servers, routers, exchanges, supercomputers, so ICT companies use Linux and vim
  - Will have the edit done with vim by the time you figure out how to install another editor!
  - No royalties

Vim is everywhere

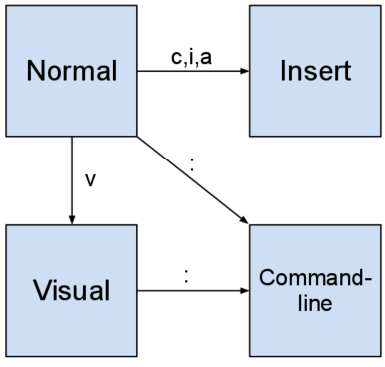
- Bundled
  - At the command line with all Unix, Linux, & Mac
- gvim supports graphical mode with mouse etc.
- also available as an app for
  - Mac
    - Both as Vim & MacVim (gVim on Mac)
    - And a special Mac adapted version MacVim, which has Mac specific features such as cmd-key bindings etc.
  - Windows
  - Mobiles : Android, iOS :
    - can edit big files others can't even open

Great editor This let's me edit large text files that other editors would not open.

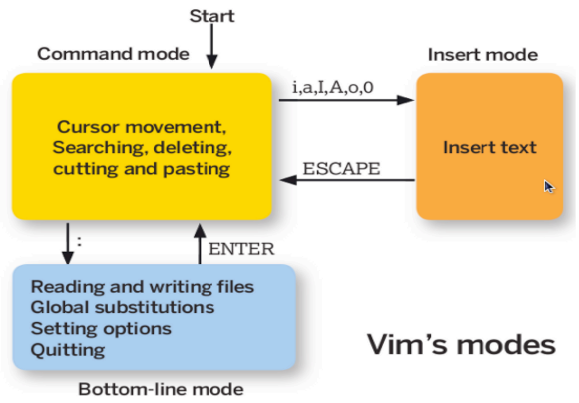
What you see is what you get – no hidden corruption

- Unlike MS-Word, WordPerfect, or other word processors, *vi* files consist of plain ASCII text
  - There are no special hidden formatting codes to corrupt your files on switching from one editor to a run time…
  - **THIS IS CRITICAL TO CORRECT CODE**

Modes



http://tuxradar.com/content/vim-master-basics



Starting vi

- Command line options

vi	invokes *vi* with a blank buffer
vi file	invokes *vi* on *file* or on a blank buffer creating *file* if it does not exist
vi file1 file2...	invokes *vi* on *files* sequentially
vi -R file	invokes *vi* on *file* in read-only mode
view file	invokes *vi* on *file* in read-only mode
vi -r file	recovers *file* and recent edits after a system crash (or at least tries!)
vi + file	opens *file* with cursor at last line
vi +n file	opens *file* with cursor at line n
vi +/expr file	opens *file* with cursor at *expr*

vi Architecture

- *vi* is based on ex
  - All *ex* commands are available from within *vi*
- Changing from *vi* to *ex* and back
  - From *vi* command mode, a *Q* will switch to *ex*
  - From *ex* command mode *:vi* will switch back to *vi*
- Like *ed* and *ex*, *vi* is modal
  - Command and text entry modes

vi Buffers

- *vi* is also based on memory buffers, like *ed*, changes are made to the buffer, not your file until a write is performed
- *vi* has three kinds of buffers
  - the normal default buffer your edit file resides in
  - the anonymous buffer, a special buffer used for copying text
  - named
    - 1-9 hold the last nine deletions
    - a-z for storing copied text (like the Windows clipboard)

Cursor Movement

- By Character

  h, j, k, l => quicker with your fingers on the buttons!
  (←, ↓ ,↑,→)

- By Text Objects

w or W	Forward a word
b or B	Backwards a word
e or E	End of word
) and (	Beginning of next, previous paragraph

Movement by Lines

0, $	Beginning, end of current line
^	First character of current line (ignore spaces)
+, -	Next, previous line
n	Column *n* of current line
H	Top line of screen
M	Middle line of screen
L	Bottom line of screen
*n*H	*n* (number) of lines after top line
*n*L	*n* (number) of lines before bottom

Command Multipliers

- Almost any command can be repeated multiple times by using a command multiplier
- Examples:
  - 4j moves down 4 lines
  - 6w or 6W moves forward 6 words
  - 10dd deletes 10 lines, starting with the current line
  - 5x deletes 5 characters
  - y4w copies 4 words to the anonymous buffer

Screen Movement

CTRL-F , CTRL-B	Scroll forward/backwards one screen
CTRL-D , CTRL-U	Scroll down/up one-half screen
CTRL-E , CTRL-Y	Show one more line at bottom/top of window
z RETURN	Reposition current line to top of screen
z.	Reposition current line to middle of screen
z-	Reposition current line to bottom of screen

Searches

The search syntax in *vi* is similar to *ed* …
…but only uses one search indicator, / ? etc

/expr	searches forward for expr
?expr	searches backwards for expr
n, N	repeat last search in same, opposite direction
/, ?	repeat previous search forward, backward
fx	search forward for x on current line
Fx	search backwards for x on current line
;	repeat previous current-line search
,	repeat previous current line search in other direction

Using Line Numbers

CTRL-G	displays current line number
*n*G	Move to line number *n*
G	Move to last line in file
:*n*	Move to line number *n*

Marking Lines

As in ed, lines can be marked so you can refer to them in commands other than by line number

m*x*	Mark current **cursor position** as *x*
:ma *x*	Mark current **line** as *x*
`*x*	Move cursor to *x*
‘*x*	Move to beginning of line containing *x*
``	Return to previous mark
‘’	Return to beginning of line containing previous mark

Inserting Text

- vi is more accommodating about inserting text in an empty file than ed
- All the insert commands work whether text exists or not

- i, a Insert text before/after cursor
- I, A Insert text at beginning/end of line
- o, O Open a new line for text below/above cursor

Changing Text

r	replace character at cursor with next typed char
~	Change case of character under cursor
cm	Change text block defined by movement command *m* (cw or cL for word or line)
cc	Change current line
C	Change to end of line
R	Type over characters
s	Delete character and substitute typed text
S	Delete current line and substitute typed text

Note: r and ~ leave you in command mode

Deletions and Moves

x	Delete character under cursor
X	Delete character before cursor
d*m*	Delete text block defined by movement command *m* (dw or dL for delete word or line)
dd	Delete current line
D	Delete from cursor to end of line
p	Put deleted text **before** cursor
P	Put deleted text **after** cursor

Yank and Put (Copy & Paste)

- Copying (Yanking) Text
  - yy,Y Copy current line
  - y*m* Copy text block defined by movement command *m* (yw or yL for word or line)

    Note: multipliers can be used,

    ie; y3w for yank 3 words

    or y10L for yank 10 lines
- Pasting (Put'ing) Text
  - p, P Put yanked text after, before cursor

Other Commands

u, U	Undo last edit, restore current line
J	Join current and next lines Note case of command (J vs j)
. (period)	Repeat last edit
CTRL-L , CTRL-R	Redraw Screen

Reading

:r *newfile*	Read *newfile* into current edit file
:r !*command*	Read the output of Unix *command* into current edit file
:e!	Return to version of current file at time of last write (discarding all changes since)
:e *file2*	Edit *file2* without leaving vi
:e #	Edit alternate file

Writing

:w	Write (save) file
:w!	Write file (overriding file protection)
:3,9w *newfile*	Write lines 3 through 9 to *newfile*
:3,9w>>file	Append lines 3 through 9 to file
:w%.new	

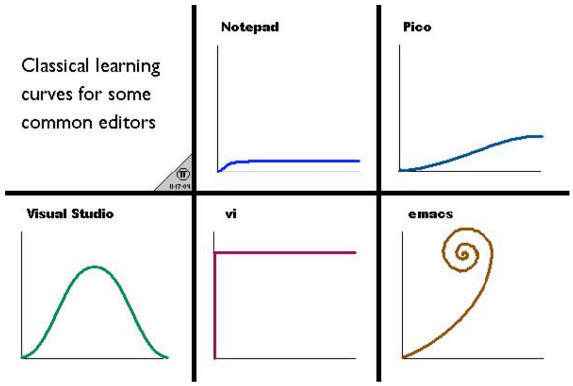
Exiting

ZZ	Write (save) and quit
:x	Write (save) and quit
:wq	Write (save) and quit
:q	Quit
:q!	Quit discarding any unsaved changes
Q	Quit *vi* and invoke *ex*
:n	Edit next file
	(if the vi file1 file2 ... variant was used)

Why VIM?

- Lightweight
- Everywhere – Win, *nix, Mac, even has a Visual Studio emulator
- Hones important skills (like regex) ***
- More productive (quicker) in longer term…
  - Stick to the Keyboard
    - faster than catching a mouse!
  - Stick to the home row h, j, k, l
    - Faster than catching an arrow … keys
- Remote coding via ssh
- Pair programming (ssh + tmux)
- Extensive range of plugins
  - Language specific : syntax colouring, autocompletion etc.
  - File system display e.g. NerdTree
- Active developer community

Learning curve ! – throwing a curve?



Configurable

- .vim file
- .vim directory
- Vim scripts and plugins
- Google “vim Janus”
- Github astrails dotvimhttp ://www.ibm.com/developerworks/aix/library/au-customize_vi/

Favorite Plugins

- Surround plugin
- Rails
- Nerdtree
- Command-t (fuzzy finding)

Favorite keys

- cw and ciw – change word,
- S) – replace (delete) current line with ‘)’
- cc to change line
- ndw, ndd to delete ‘n’ words, lines
- D – delete to end of line

Practice cw

- Load up piece of text
- Use / to find instance of ‘the’
- Use ‘cw’ to change ‘the’ to ‘your’
- Use ‘esc’ and then ‘n’ to move to next find
- Use ‘.’ to repeat
- Use ‘n’ and ‘.’ to repeat as many times as possible

Practice ciw

- Load up piece of text
- Use / to find instance of ‘he’
- Use ‘**ciw**’ to change ‘the’ to ‘your’
- Use ‘esc’ and then ‘n’ to move to next find
- Use ‘.’ to repeat
- Use ‘n’ and ‘.’ to repeat as many times as possible

Practice o and O

- Load up piece of text
- Use **j** to go to 2nd line
- For each line use shift O to enter a line above and o to insert a line below
- Use . To repeat actions

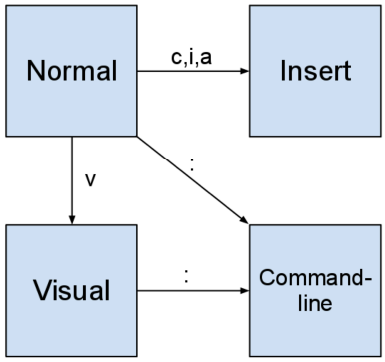
Vim Everywhere

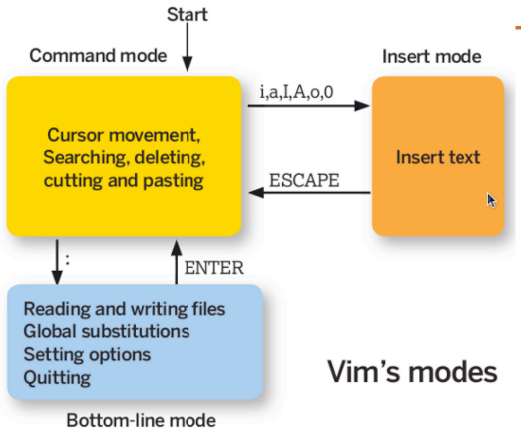
- Sublime Text (vintage mode)
- Cloud9 IDE
- Eclim
- Firefox : Vimium
- Chrome : vimperator
- Gmail
- Google Reader
- vsVim for visual studio
- ViEmu for Word
- Vi-like command line in zsh (configure yourself)

Resources

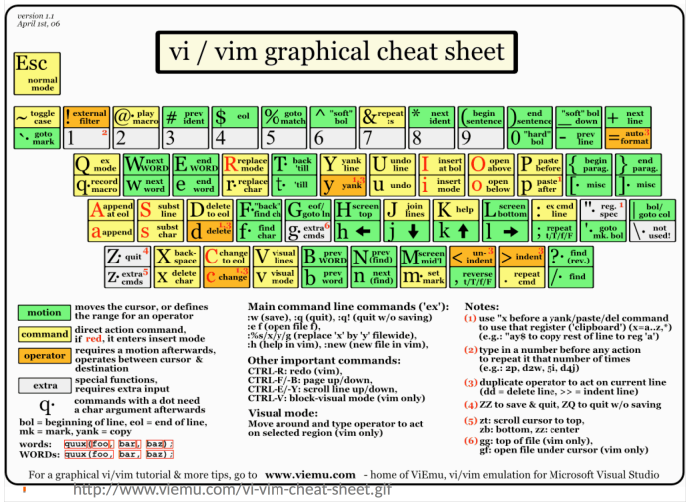
- Vim Revisited - Mislav Marohnić
  - http://mislav.uniqpath.com/2011/12/vim-revisited/
- Drew Neil
  - Vim – Walking without Crutches
    - https://github.com/alkesh/vim-config
  - Practical Vim – edit text at the speed of thought!
    - http://pragprog.com/book/dnvim/practical-vim
- Graphical Cheat Sheet and Tutorial
  - http://www.viemu.com/a_vi_vim_graphical_cheat_sheet_tutorial.html
- A Byte of Vim :
  - http://swaroopch.com/notes/vim/

Modes



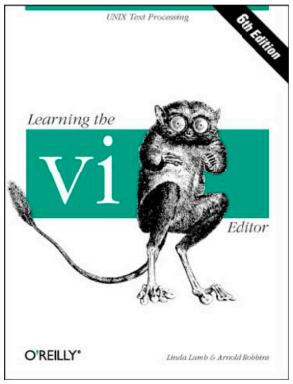


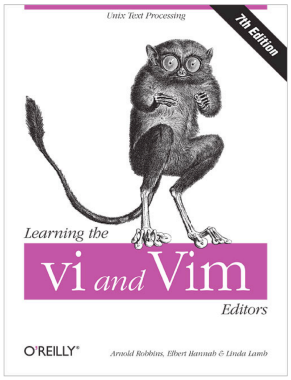
http://tuxradar.com/content/vim-master-basics



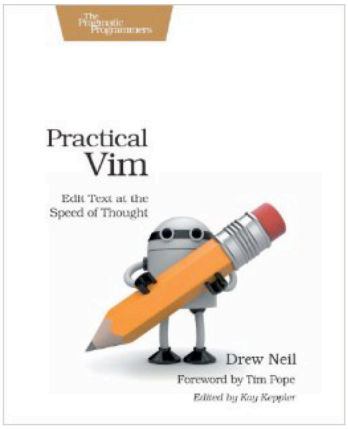
http://www.viemu.com/vi-vim-cheat-sheet.gif

Books 1998 – 2008 – Vi/m since 1976/1992

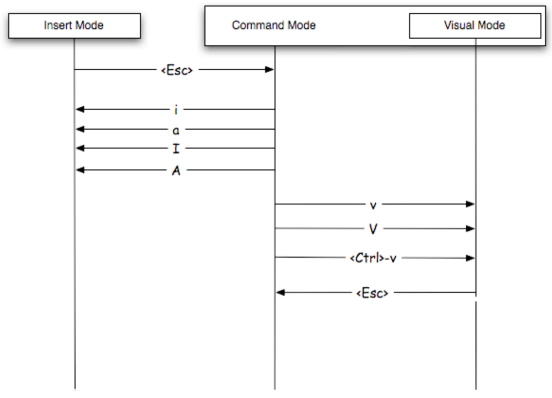




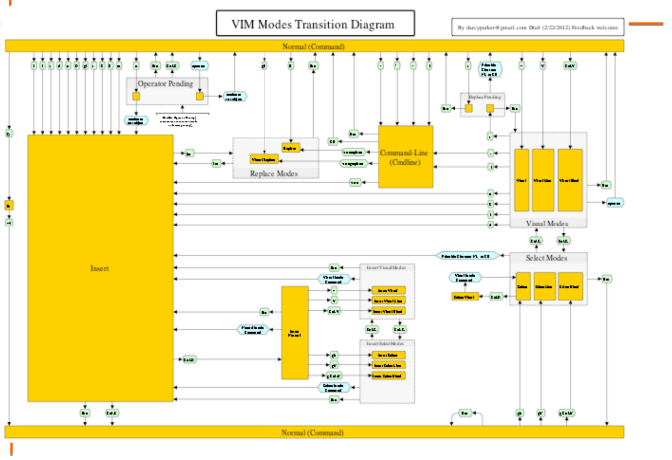
Drew Neil...

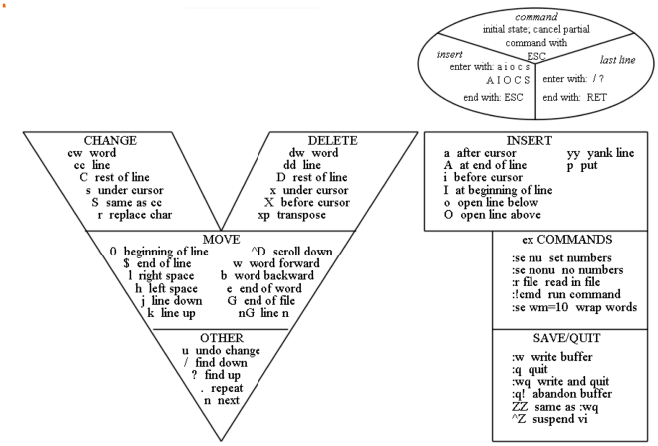


Rave reviews...



http://blog.interlinked.org/tutorials/vim_tutorial.html





http://amalloy.hubpages.com/hub/Why-old-text-editors-are-still-great

Learning Vim; the Vim Way

VimGolf

http://vimgolf.com/

… any developers ideal help..

Bubble, bubble, toil and spelling trouble

Vim can spell!

Enter

:echo &spelllang

And it will show the language… en for English

Or you can set the language

:set spelllang=en_GB.UTF-8

UCS Transformation Format (8-bit), variable format to represent all Unicode characters backwardly compatible with ASCII

Recommended for and used in over half the WEB,

And all mail programs should handle it.

To check spelling

: set spell OR : set nospell

Will turn (on / off) spelling error highlighting

]s OR [s

will jump to next/previous error from cursor

]S OR [S

will jump to words not in the dictionary.

To correct spelling

Placing the cursor over a highlighted word and entering

'z='

will give a list of numbered suggestions, with the correct one inserted on entering it's number

zg

marks a highlighted error as good & acceptable

zw

(the word need not be highlighted in this case)

Adds it to the bad word list, so will be highlighted as an error.