

VIM Cheat Sheets

:viusageShow a summary of all commands
:h keyword.....Open help for keyword
:ter.....Open a terminal window
:sav file..... Save file as
k..... Open man page for word under the cursor

Movements

h l k jcharacter left, right; line up, down
b wstart of word/token left, right
B Wspace-separated word left, right
ge eend of word/token left, right
gE Eend of space-separated word left, right
{ }.....beginning of previous, next paragraph
().....beginning of previous, next sentence
O ^ g_ \$ beginning, first character, last character and end of line
nG ngg Go to line *n*, when *N* is no indicated go to last line and first line
:n.....Go to line *n*
n|Go to column *n* of current line
%match of next brace, bracket, comment, :h matchpairs
H M L..... beginning, middle, botton of *screen*
ngj ngkmove cursor down, up *n* lines
fc Fcnext, previous occurence of character *c*
tc Tcbefore next, previous occurence of *c*
; , .Repeat last f,t,F,T movement backward, forward
..Repeat las change

Editing

i ainsert before, after cursor
I Ainsert at beginning, end of line
gIinsert text in first column
o O ... open a new line below, above the current line
ea..... Insert at the end of the word
rcreplace character under cursor with *c*
grclike *r*, but without affecting layout
Rreplace characters starting at the cursor
gRlike *R*, but without affecting layout
x Xdelete character under, before cursor
cmchange text of movement command *m*

Cchange to the end of line
^nautocomplete next match before the cursor
^pautocomplete next match after the cursor
^rx Insert the contents of register *x*
dmdelete text of movement command *m*
J gJjoin current line with next, without space
sdelete a character and substitute text
Sdelete a line and substitute text
xptanspose two letters ab
uUndo
^rredo

Search & Substitution

/pattern ?pattern search forward, backward for pattern
/s/o↔ ?s?o↔.....search fwd, bwd for *s* with offset *o*
n_{or} /↔repeat forward last search
N_{or} ?↔repeat backward last search
* ..search backward, forward for word under cursor
g# g*same, but also find partial matches
gd gD ..local, global definition of symbol under cursor
:rs/old/new/xsubstitute old by new in range *r*
[*x* : g—all occurrences, c—confirm changes
:rs *x*repeat substitution with new *r* & *x*
:nohremove highlight matches

Formatting/Filtering

m is movement; leave out for visual mode commands
gqmformat *m* (*formatprg*)
gqgq_{or} gqip.....format current paragraph
gwipjoin current line with next, without space
g~*m* gum gUm ...switch case, lc, uc on movement *m*
~switch case and advance cursor
= *m*.....reindent movement *m* (*equalprg*)
:rce *w*center lines in range *r* to width *w*
:rle *i*left align lines in range *r* with indent *i*
:rri *w*right align lines in range *r* to width *w*
!mc↔ ..filter lines of movement *m* through command *c*
n!|c↔filter *n* lines through command *c*
:r|cfilter range *r* lines through command *c*
<*m* >*m*shift left, right text of movement *m*
n<< n>>shift *n* lines left, right
>%indent inner a block with () or -"

^A ^X increment/decrement number under cursor

Visual Mode & Text Objects

v V ^V highlight characters, lines, block
oexchange cursor pos with start of highlighting
gvstart highlighting on previous visual area
aw as apselect a word, a sentence, a paragraph
ab aB a] a>.....select a block (), { }, [] < >
at.....select an html tag
a" a' a'.....select a quotet string
iw is ip ib iB i] i> it i" i' i' ..select "inner"
n> n< =indent/unindent *n* levels, reindent

Undoing, Repeating & Registers

u Uundo last command, restore last changed line
. ^Rrepeat last changes, redo last undo
n.repeat last changes with count replaced by *n*
qc qC ..record, append typed characters in register *c*
qstop recording
@cexecute the content of register *c*
@@repeat previous @ command
:@cexecute register *c* as an *Ex* command

Copying (Yanking)

"*x*use register *x* for next delete, yank, put
:reg [*x*]show content registers/*x* reg
ymyank the text of movement command *m*
yy_{or} Yyank current line into register
p Pput register after, before cursor position
]p [plike p, P with indent adjusted
gp gPlike p, P leaving cursor after new text

Patterns (differences to Perl) :help pattern

\< \>start, end of word
\i \k \I \Kan identifier, keyword; excl. digits
\f \p \F \P ..a file name, printable char.; excl. digits
\e \t \r \b*<esc>*, *<tab>*, *<↵>*, *<←>*
\= * \+ ...match 0..1, 0..∞, 1..∞ of preceding atoms
\{*n,m*}match *n* to *m* occurrences
\{-}non-greedy match
\|separate two branches (*≡ or*)
\(\)group patterns into an atom
\& \1the whole matched pattern, 1st () group

`\u \l` upper, lowercase character
`\c \C` ignore, match case on next pattern
`\%x` match hex character
`\@= \@!` (?=pattern) (!pattern)
`\@<= \@<!` (?<=pattern) (?<!pattern)
`\@>` (?>pattern)
`_^ _` start-of-line/end-of-line, anywhere in pattern
`_.` any single char, including end-of-line
`\zs \ze` set start/end of pattern
`\%^ \%$` match start/end of file
`\%V` match inside visual area
`\'m` match with position of mark m
`\%(\)` unnamed grouping
`\[\]` collection with end-of-line included
`\%[\]` sequence of optionally matched atoms
`\v` very magic: patterns almost like perl

Spell Checking

`:set spell spelllang=de_20` activate spellcheck
`]s` next misspelled word
`zg zG` add good word (to internal word list)
`zw zW` mark bad word (to internal word list)
`z=` suggest corrections

Marks, Motions, and Tags

`mc` mark current position with mark $c \in [a..Z]$
`'c 'C` go to mark c in current, C in any file
`'' ''` go to position before jump, at last edit
`'[']` go to start, end of previously operated text
`:marks` print the active marks list
`:jumps` print the jump list
`n^O` go to n^{th} older position in jump list
`n^I` go to n^{th} newer position in jump list
`] ^T` ...jump to the tag under cursor, return from tag
`:ts t`list matching tags and select one for jump
`:tj t` ...jump to tag or select one if multiple matches
`gf` open file which filename is under cursor
`:tags` print tag list

Multiple Files / Buffers (\leftrightarrow)

`:tab ball`show buffer tablist
`:buffers`show list of buffers
`:buffer n`switch to buffer n

`:badd f.txt` load file into new buffer
`:bdelete n` delete buffer n (also with filename)
`:bnext :bprev :bfirst :blast` ... buffer movement

Scrolling & Multi-Windowing

`^d` Move forward 1/2 page (^= Ctrl)
`^u` Move backward 1/2 page (^= Ctrl)
`^D ^U ^F ^B` scroll half / full page page down, up
`zt zz zb` ...current line to top, center, bottom of win
`:[v]split ^Ws ^Wv` split window (horiz., vert.)
`:vert help` open help vertically
`^Wf ^W` open file, tag in split
`:[v]new ^W^n` new empty window (horiz., vert.)
`:on ^Wo` make current win the only one
`^Wj ^Wk ^Wh ^Wl`move down, up, left, right to win
`^WJ ^WK ^WH ^WL` move win down, up, left, right
`^Wn+ ^Wn-` increase/decrease win size by n lines
`^Wn > ^Wn <` increase/decrease window width
`^Wn_ ^Wn|` set height/width (max if no n)
`^W=` Make win size equal
`^Wr ^Wx` Rotate, exchange windows
`^Wc` close window

Misc Ex Commands (\leftrightarrow) :help holy-grail

`:e f` edit file f , reload current file if no f
`:r w f`write range r to file f (current file if no f)
`:r w >> f` append range r to file f
`:r g[!]/p/cmd` ..ex cmd on range r [not] matching p
`:q :q!` ...quit and confirm, quit and discard changes
`:wq or :x or ZZ` write to current file and exit
`:r f` insert content of file f below cursor
`:r! c` insert output of command c below cursor
`:rc a :rm a` copy, move range r below line a

Ex Ranges

`,` ; separates two lines numbers, set to first line
`n` an absolute line number n
`.` \$ the current line, the last line in file
`% *` entire file, visual area
`'t` position of mark t
`/p/ ?p?` the next, previous line where p matches
`+n -n` $+n, -n$ to the preceding line number

Completion

`^X^L` whole lines
`^X^N ^X^I` .. keywords in current file, plus included files
`^X^K ^X^N` keywords in dictionary, thesaurus
`^X^]` ^X^F ^X^D tags, filenames, defs/macros
`^X^V` vim command line
`^X^U ^X^O` user defined, omni- completion

Folding

`:set fdm=indent` indent-foldmethod
`zfm` create fold of movement m
`:rfo` create fold for range r
`zd zE` delete fold at cursor, all in window
`zo zc zO zC` open, close one fold; recursively
`[z]z` move to start, end of current open fold
`zj zk` move down, up to start, end of next fold
`zm zM` fold more, close all folds
`zr zR` fold less, open all folds
`zn zN zi` fold non, fold normal, invert folding
`:set foldcolumn=4` show foldcolumn

Compiling

`:compiler c` set/show compiler plugins
`:make` run `makeprg`, jump to first error
`:cope` navigate errors from make
`:cn :cp` display the next, previous error
`:cl :cf` list all errors, read errors from file

Miscellaneous

`:sh :!c` start shell, execute command c in shell
`K` run `keywordprg` (man) on word under cursor
`^L` redraw screen
`^G` ...show cursor column, line, and character position
`:set cuc` show cursor column visually
`ga` show ASCII value of character under cursor
`:mkview [f] :loadview [f]` .. save/load configuration
`:set ff=dos` convert file to dos eol format
`:e ++ff=unix` reopen file in unix eol format
`:write ++enc=utf-8` Write file in utf-8
`:set hlsearch` highlight searches
`:rhardcopy > file.ps` print range to ps file
`:set list` show listchar characters (tabs etc.)