

VIM QUICK REFERENCE CARD

:viusageShow a summary of all commands

Movements

h l k jcharacter left, right; line up, down
b wword/token left, right
ge eend of word/token left, right
{ }beginning of previous, next paragraph
()beginning of previous, next sentence
O ^ \$beginning, first, last character of line
nG nggline *n*, default the last, first
n|column *n* of current line
% ... match of next brace, bracket, comment, #define
- +line up, down on first non-blank character
B Wspace-separated word left, right
gE Eend of space-separated word left, right
gO gmbeginning, middle of *screen* line
g^ g\$first, last character of *screen* line
fC Fcnext, previous occurrence of character *c*
tC Tcbefore next, previous occurrence of *c*

Jumps

:jumps print the jump list
:ta ^[..... jump to tag (under cursor)
^O ^Ijump to older/newer location btw. buffers
ng; ng,go to or edit *n* older/newer position in
change list
' jump back on last edited line
' 'toggle back/forward to previous/next position

Insertion & Replace → insert mode

i ainsert before, after cursor
I Ainsert at beginning, end of line
gi gIinsert text on last edited line / first column
o O open a new line below, above the current line
rcreplace character under cursor with *c*
grclike *r*, but without affecting layout
Rreplace characters starting at the cursor
gRlike *R*, but without affecting layout
scsubstitute char. *c* under cursor
cmchange text of movement command *m*
cc or Schange current line

Cchange to the end of line

Deletion

x Xdelete character under, before cursor
dmdelete text of movement command *m*
dd Ddelete current line, to the end of line
dgndelete the next search pattern match
J gJjoin current line with next, without space
:rddelete range *r* lines
:rdxdelete range *r* lines into register *x*
qxqempty register *x*
:g/pattern/d *x*
:let @+ = @a.delete lines with pattern *c* into Register
x and copy to clipboard (:let ...@a)

Insert/ Command Mode

^Vc ^Vn insert char *c* literally, decimal value
n
^Vninsert decimal value of character
^Ainsert previously inserted text
^@ .same as ^A and stop insert → command mode
^Rx ^R^Rxinsert content of register *x*,
literally
^R^Wcontent under cursor to command mode
^N ^Ptext completion before, after cursor
^Wdelete word before cursor
^Udelete to start of current line
^D ^Tshift left, right one shift width
^Ocexecute *c* in temporary command mode
^X^E ^X^Yscroll up, down
<esc> or ^]abandon edition → command mode

Search & Substitution

/s↔ ?s↔search forward, backward for *s*
/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/f/t/xsubstitute *f* by *t* in range *r*
[*x* : g---all occurrences, c---confirm changes
:rs *x*repeat substitution with new *r* & *x*
:rg/p/c :rv/p/c .execute *Ex c* on range *r* where
p matches/does not match

Misc Ex Commands (↔)

:help holy-grailshow all Ex commands
:e *f* edit file *f*, reload current file if no *f*
:rw *f* ...write range *r* to file *f* (current file
if no *f*)
:rw>>*f*append range *r* to file *f*
:q :q!quit and confirm, quit and discard
changes
:wq or :x or ZZwrite 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*
:rg/p/-1jjoin all lines in *r* containing
pattern *p* with prev. line
:rg/p/s/q/y/g ..for every line in *r* containing
p, substitute *q* with *y*
:'a,'bg/p/jjoin any line containing the string
p to its subsequent line, if it lies between the
a and *b* marks

Ex Ranges

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

Standard Mode Formatting/ Filtering

Leave out *m* for visual mode commands
gqm gqgq format movement *m*/current paragraph
:rce *w*center lines in range *r* to width *w*
:rri *w* ..right align lines in range *r* to width
w

```
:rle i left align lines in range r with indent
i
!mc↔.....filter lines of movement m through
command c
n!!c↔.....filter n lines through command c
:r!c ....filter range r lines through command c
~.....switch case and advance cursor
g~m gum gUm .switch case, lc, uc on movement
m
guu gUU.....lower-/uppercase line
<m >m .shift left, right text of movement m
n<< n>> .....shift n lines left, right
^A ^X...increment/decrement number under cursor
```

Visual Mode

```
:h object-select.....Object selecting patterns
v V ^V .....start/stop highlighting characters,
lines, block
o .....exchange cursor position with start of
highlighting
gv .start highlighting on previous visual area
aw as ap .....select a word, a sentence, a
paragraph
ab aB .....select a block ( ), a block { }
```

Undoing, Repeating & Registers

```
u U ...undo last command, restore last changed
line
. ^R .....repeat last changes, redo last undo
n. .repeat last changes with count replaced by
n
qc qC .....record, append typed characters in
register c
q .....stop recording
@c .....execute the content of register c
@@ .....repeat previous @ command
:@c .....execute register c as an Ex command
```

Copying

```
"x ..use register x for next delete, yank, put
:reg .....show the content of all registers
:reg x .....show the content of registers x
ym .....yank the text of movement command m
```

```
yy or Y .....yank current line into register
p P put register after, before cursor position
lp [p .....like p, P with indent adjusted
gp gP .like p, P leaving cursor after new text
```

Patterns (differences to Perl)

```
:help pattern ..show complete help on patterns
\< \> .....start, end of word
\i \k \I \K .....an 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)
\(< \>) .....group patterns into an atom
\& \1 ..the whole matched pattern, 1st () group
\u \l .....upper, lowercase character
\c \C .....ignore, match case on next pattern
\%x .....match hex character
\@= \@! .....(?=pattern) (!pattern)
\@<= \@<! .....(?<=pattern) (?<!pattern)
\@> .....(?>pattern)
\_ ^ \_ $ .start-of-line/end-of-line, anywhere in pattern
\_ . .....any single char, including end-of-line
\zs \ze .....set start/end of pattern
\%~ \%$ .....match start/end of file
\%V .....match inside visual area
\'m .....match with position of mark m
\%( \) .....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)
zug zuG.undo the addition of a word to the dictionary
zw zW .....mark bad word (to internal word list)
z= .....suggest corrections
```

Marks, Motions, and Tags

```
mc .....mark current position with mark c ∈ [a..Z]
'c 'C .....go to mark c in current, C in any file
'0..9 .....go to last exit position
'' '' .....go to position before jump, at last edit
'[ ']' ....go to start, end of previously operated text
:marks .....print the active marks list
n^0 .....go to nth older position in jump list
n^I .....go to nth 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
:tags .....print tag list
```

Multiple Files / Buffers (↔)

```
:h buffers.....get help for buffer-management
:tab ball .....show buffer tablist
:buffers .....show list of buffers
:on .....make current window one on screen
:new :vnew .....create new empty window (vert.)
:bn .....switch to buffer n
:bnext :bprev :bfirst :blast ...buffer movement
:bdn .....delete buffer n (also with filename)
:badd f.txt .....load file into new buffer
:sbn ..Split window and edit buffer n from the buffer
list
:verticaln :horizontalc ..execute c on vert./horiz.
split
:rightabovec :leftbelowc .....execute c (eg. sbn)
right/left above/below window
```

Buffer Shortcuts

```
:h ctrl-w .....complete list of all buffer commands
^~ ....toggle between the current and the last window
^Wf gf .open file under cursor in new/current window
^Ww ^W^W .....move to window below, above (wrap)
^Wj ^Wk .....move to window below, above
^Wt ^Wb .....move to top/bottom window
^Wo ^Wc .....close all other/current window(s)
^Ws ^Wv .....split window in two (vert.)
^Wx .....swap open buffer windows
^Wn+ ^Wn- ..increase/decrease window size by n lines
^Wn > ^Wn < .....increase/decrease window width
```

`^W` = Make all windows equally high and wide
`^W n_`set window height to n (default: very high)
`^W n|` set current window width to n

Tab Management

`:tabs` ..list all tabs including their displayed windows
`:tabfirst` go to first tab
`:tablast` go to last tab
`:tabnew` open a new empty tab page
`:tabc` close current tab page
`:wqa :qa` (save and) quit all tabs
`:tabo` close all other tabs
`gt gT`go to next/previous Tab
`ngt` goto tab in position n

Advanced Scrolling

`^D ^U`scroll half a page up, down
`^F ^B` scroll page up, down
`zt zz zb` ..current line to top, center, bottom of win.
`zh zl`scroll one character to the right, left
`zH zL`scroll half a screen to the right, left

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^]` tags
`^X^F`..... file names
`^X^D`..... definitions or macros
`^X^V`..... vim command line
`^X^U`..... user defined completion
`^X^O`..... omni completion

Folding

`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

Indentation

`:set fdm=indent`indent-foldmethod
`n> n< =` indent/unindent n levels, reindent
`G=gg`.....auto (re)indent entire document

Miscellaneous

`:rhardcopy rw!lp`sending r to printer (printout)
`:rhardcopy > file.ps` print range to ps file
`:e ++ff=unix`reopen file in unix eol format
`:mks name` save session $name$
`:mkview [f] :loadview [f]` ..save/load configuration
`:set cuc`show cursor column visually
`:set ff=dos` convert file to dos eol format
`:set hlsearch`highlight searches
`:set ic/noic` (un)ignore cases in substitute-command
`:set list`show listchar characters (tabs etc.)
`:sh :!c` start shell, execute command c in shell
`K` ...run `keywordprg` (manpage) on word under cursor
`^L` redraw screen
`^R = 5*5`.....insert 25 into text
`g^G`...show cursor column, line, and character position
`ga`show ASCII value of character under cursor
`gf` open file which filename is under cursor
`vim -S name`reload vim-session $name$

Most Common Digraphs

`:dig`complete list of all digraphs
`^K c_1c_2 or $c_1\leftarrow c_2$` enter digraph $\{c_1, c_2\}$
`Co` Copyright
`Rg` Registered Trademark
`!I` ¡: Inverteted Excl. Mark
`?I` ¿: Inverteted Quest. Mark
`Eu` Euro Currency

`Li` Britain Pound Currency
`14 12 34` $1/4, 1/2, 3/4$ etc.

More examples for how to use Exec-Commands see:
<http://stackoverflow.com/a/1220118>

This card may be freely distributed under the terms of the GNU general public licence — Copyright © 2009 by Michael Goerz.
<http://www.physik.fu-berlin.de/~goerz/>. Based on original by Laurent Grégoire (<http://tnerual.eriogerg.free.fr/>)