VIM QUICK REFERENCE CARD

Basic movement
h l k j . . . . . . character left, right; line up, down
b w . . . . . . . . . word/token left, right
g e . . . . . . . . . beginning of next paragraph
} . . . . . . . . . . beginning of previous, next sentence
0 gm . . . . . . beginning, middle of line
$ . . . . . . . . . . first, last character of line
nG ngg . . . . . . line n, default the last, first
n%. . . . percentage of the file (n must be provided)
n] . . . . . . . . . . column n of current line
% . . . . . . . . . . match of next brace, bracket, comment, #define
nH nL . . . . . . line n from start, bottom of window
M . . . . . . . . . . middle line of window

Insertion & replace → insert mode
i a . . . . . . . insert before, after cursor
I A . . . . . . . insert at beginning, end of line
gI . . . . . . . insert in first column
o 0 . . . . open a new line below, above the current line
re . . . . . . . replace character under cursor with c
grc . . . . . . like r, but without affecting layout
R . . . . . . . replace characters starting at the cursor
gR . . . . . . . like R, but without affecting layout
cm . . . . . . . change text of movement command
cc=S . . . . . . change current line
C . . . . . . . . . . change to the end of line
s . . . . . . . change one character and insert
- . . . . . . . . . . . . . switch case and advance cursor
G . . . . . . . switch case of movement command m
gum gum . . . lowercase, uppercase text of movement m
<m >m . . . . shift left, right text of movement m
n<< n>> . . . shift n lines left, right

Deletion
x X . . . . . . . delete character under, before cursor
dm . . . . . . . delete text of movement command m
dd . . . . . . . delete current line, to the end of line
J gJ . . . . . . join current line with next, without space
:rd . . . . . . . delete range r lines
:rdr . . . . . . . delete range r lines into register x

Visual mode
v V . . . . . . start/stop highlighting characters, lines, block
o . . . . . . exchange cursor position with start of highlighting
gv . . . . . . . highlight starting on previous visual area
aw as ap . . . . . . select a word, a sentence, a paragraph
ab aB . . . . . . . select a block ( ), a block { }

Insert mode
"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 = Y . . . . . yank current line into register
p P . . . . . . . put register after, before cursor position
]p [p . . . . . . like p, P with indent adjusted
gp gP . . . . . . like p, P leaving cursor after new text

Advanced insertion
g?m . . . . . perform rot13 encoding on movement m
nA nX . . . . . . . . n, −n to number under cursor
gqm . . . . . format lines of movement m to fixed width
:rc e w= . . . . center lines in range r to width w
:rl i e= . . . . left align lines in range r with indent i
:ri w e= . . . right align lines in range r to width w
!mce=. . . . filter lines of movement m through command c
!n!ce= . . . . . . . filter n lines through command c
)r!ce= . . . . filter range r lines through command c

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

Undoing, repeating & registers
u U . . . . undo last command, restore last changed line
. . . . . . repeat last changes, redo last undo
. . . . . . repeat last changes with count replaced by n
qc qC . . . record, append typed characters in register c
q . . . . . . stop recording
:reg c . . . execute the content of register c
:reg= . . . . . . . . repeat previous c command
:reg c . . . . . . . . execute register c as an Ex command
:reg p/c ev= . . execute Ex command c on range r
  [ where pattern p matches

Complex movement
- + . . . . . line up, down on first non-blank character
B W . . . . . . space-separated word left, right
gE E . . . . . . end of space-separated word left, right
g0 . . . . . . beginning of screen line
g* gS . . . first, last character of screen line
gk g] . . . . . screen line up, down
tc Tc . . . . . . . . previous occurrence of character c
tc Tc . . . . . . . . before next, previous occurrence of c
; . . . . . . repeat last ftFT, in opposite direction
[[ ] ] . . . . . . start of section backward, forward
[] ] . . . . . . end of section backward, forward
[[ ] ] . . . . . . unclosed ( ) backward, forward
[] ] . . . . . . unclosed { } backward, forward
m m . . . . . . . . start of backward, forward Java method
[#] # . unclosed #, #else, #endif backward, forward
[* ] * . . . . . . start, end of /* */ backward, forward

Search & substitution
/s= ?s= . . . . search forward, backward for s
/s/o= ?s?o= . . . . search fwd, bwd for s with offset o
n w/o= . . . . repeat forward last search
N w/o= . . . . . 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
tra /t/x= . . substitute f by t in range r
  [ x: g—all occurrences, c—confirm changes
  ρa x= . . . . . . repeat substitution with new r & x
Special characters in search patterns

- `.` any single character, start, end of line
- `\` \ start, end of word
- `[c1-c2]` a single character in range `c1`-`c2`
- `[c1-c2]` a single character not in range
- `\i` `\v` `\K` ... an identifier, keyword; excl. digits
- `\f` `\p` `\F` ... a file name, printable char.; excl. digits
- `\s` `\S` ... a white space, a non-white space
- `\[`... match 0..1, 0..\infty, 1..\infty of preceding atoms
- `\]` ... separate two branches (\equiv `or`)
- `{}` ... group patterns into an atom
- `\&` `\n` ... the whole matched pattern, `n`\textsuperscript{th} `O` group
- `\u` `\l` ... next character made upper, lower case

Offsets in search commands

- `n \plus \n` ... `n` line downward in column 1
- `n \minus \n` ... `n` line upward in column 1
- `e\n` `e\minus \n` ... `n` characters right, left to end of match
- `s\n` `s\minus \n` ... `n` characters right, left to start of match
- `;c` ... execute search command `sc` next

Marks and motions

- `m` ... mark current position with mark `c` \in `{a..Z}` ... go to mark `c` in current, `C` in any file
- `e` ... go to position before jump, at last edit
- `m` ... go to start, end of previously operated text
- `:marks` ... print the active marks list
- `:jumps` ... print the jump list
- `n\prime` ... go to `n`\textsuperscript{th} older position in jump list
- `n\uparrow` ... go to `n`\textsuperscript{th} newer position in jump list

Key mapping \& abbreviations

- `:map` `c` \to `e` ... in normal \& visual mode
- `:map!` `c` \to `e` ... in insert \& cmd-line mode
- `:unmap` `c` \to `e` ... remove mapping `c`
- `:mk` `f` ... write current mappings, settings... to file `f`
- `:ab` `c` \to `e` ... add abbreviation for `c` \to `e`
- `:ab` `c` \to `e` ... show abbreviations starting with `c`
- `:una` `c` \to `e` ... remove abbreviation `c`

Tags

- `:ta` `t` ... jump to `t`\textsuperscript{th} newer tag in list
- `:nta` `t` ... jump to `t`\textsuperscript{th} newer tag in list
- `:i` `T` ... jump to the tag under cursor, return from tag
- `:ts` `t` ... list matching tags and select one for jump
- `:tg` `t` ... jump to tag or select one if multiple matches
- `:npo` `n` `T` ... jump back from, to `n`\textsuperscript{th} older tag
- `:tl` ... jump to last matching tag
- `:Wj` `p` ... preview tag under cursor, tag `p`
- `:Wk` ... split window and show tag under cursor
- `:Wz` `p` ... close tag preview window

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
- `\i` ... position of mark `i`

Folding

- `zfm` ... create fold of movement `m`
- `:rf` ... create fold for range `r`
- `zd` `zE` ... delete fold at cursor, all in window `z`
- `zo` `z0` `zC` ... open, close one fold; recursively
- `:z` `z` ... move to start, end of current open fold
- `:z` `zk` ... move down, up to start, end of next fold

Scrolling \& multi-windowing

- `E` `Y` ... scroll line up, down
- `D` `U` ... scroll half a page up, down
- `F` `B` ... scroll page up, down
- `zt` `z` ... set current line at top of window
- `zz` ... set current line at center of window
- `zp` `z` ... set current line at bottom of window
- `zh` `zl` ... scroll one character to the right, left
- `zH` `zL` ... scroll half a screen to the right, left
- `Ws` `Wj` ... split window in two
- `Wn` ... create new empty window
- `Wv` `Wk` ... move to window below, above
- `Ww` `Ww` ... move to window below, above

Miscellaneous

- `:sh` ... start shell, execute command `c` in shell
- `:mkview` ... lookup keyword under cursor with `man`
- `:cp` ... display the next, previous error
- `:cl` ... list all errors, read errors from file
- `zG` ... redraw screen, show filename and position
- `gG` ... show cursor column, line, and character position
- `ga` ... show ASCII value of character under cursor
- `gf` ... open file which filename is under cursor
- `:cp` ... display output to file `f`
- `:mkview` `/f` ... save view configuration [to file `f`]
- `:loadview` `/f` ... load view configuration [from file `f`]
- `@` `K` \_ `Ft` `Fn` ... unmapped keys

This card may be freely distributed under the terms of the GNU general public licence — Copyright © 2003 by Laurent Grégoire (laurent.gregoire@icam.fr) — v1.5 — The author assumes no responsibility for any errors on this card. The latest version can be found at http://tnerual.eriogerg.free.fr/