VIM QUICK REFERENCE CARD

Movements

h l k j . . . . character left, right; line up, down
b w . . . . word/token left, right
ge e . . . . end of word/token left, right
{ } . . . . beginning of previous, next paragraph
( ) . . . . beginning of previous, next sentence
0 ^ \$ . . . . beginning, first, last character of line
nG $n$gg . . . . line 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 W . . . . space-separated word left, right
gE E . . . . end of space-separated word left, right
g0 gm . . . . beginning, middle of *screen* line
g^ g\$ . . . . first, last character of *screen* line
fc Fc . . . . next, previous occurence of character c
tc Tc . . . . before next, previous occurence of c

Insertion & Replace → insert mode

i a . . . . insert before, after cursor
I A . . . . insert at beginning, end of line
gI . . . . insert text in first column
o O . . . . . . open a new line below, above the current line
rc . . . . replace character under cursor with c
gr$c$ . . . . 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 m
cc *or* S . . . . change current line
C . . . . change to the end of line

Deletion

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

Insert Mode

^Vc ^Vn . . . . insert char c literally, decimal value n
^A . . . . insert previously inserted text
^@ . . . . . . same as ^A and stop insert → command mode
^Rx ^R^Rx . . . . insert content of register x, literally
^N ^P . . . . text completion before, after cursor
^W . . . . delete word before cursor
^U . . . . delete all inserted character in current line
^D ^T . . . . shift left, right one shift width
^Kc_1c_2 *or* c_1←c_2 . . . . enter digraph $\{c_1, c_2\}$
^Oc . . . . execute c in temporary command mode
^X^E ^X^Y . . . . scroll up, down
⟨*esc*⟩ *or* ^[. . . . . abandon edition → command mode

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
]p [p . . . . like p, P with indent adjusted
gp gP . . . . like p, P leaving cursor after new text

Standard Mode Formatting/Filtering

Leave out m for visual mode commands
gq$m$ gqgq . . . . . format movement m/current paragraph
:$r$ce w↩ . . . . center lines in range r to width w
:$r$le i↩ . . . . left align lines in range r with indent i
:$r$ri 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!c↩ . . . . filter range r lines through command c
~ . . . . switch case and advance cursor
g~m . . . . switch case of movement command m
gu$m$ gUm . . . lowercase, uppercase text of movement m
<m >m . . . . shift left, right text of movement m
n≪ n≫ . . . . shift n lines left, right

Visual Mode

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 { }
n> n< = ... indent/unindent n levels, reindent

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
:rg/p/c↩ . . . . execute *Ex* command c on range r
⌊ where pattern p matches

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/x↩ . . . . substitute f by t in range r
⌊ x : g—all occurrences, c—confirm changes
:rs x↩ . . . . repeat substitution with new r & x

Patterns (differences to Perl)

:help pattern↩ . . . . . show complete help on patterns
:help perl-patterns↩ . . . show comparison with perl
\< \> . . . . 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
\{−} . . . . 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

Marks and Motions
mc . . . . mark current position with mark $c \in [a..Z]$
'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
: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

Tags
:ta t↩ . . . . jump to tag t
:$n$ta↩ . . . . jump to n^{th} newer tag in 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
:$n$po↩ :n^T↩ . . . . . . jump back from, to n^{th} older tag
:tl↩ . . . . jump to last matching tag
^W} :pt t↩ . . . . preview tag under cursor, tag t
^W] . . . . split window and show tag under cursor
^Wz or :pc↩ . . . . close tag preview window

Multiple Files / Buffers (↩)
: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 ^U . . . . scroll half a page up, down
^F ^B . . . . scroll page up, down
zt or z↩ . . . . set current line at top of window
zz or z. . . . . set current line at center of window
zb or 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 or :split↩ . . . . split window in two
^Wn or :new↩ . . . . create new empty window
^Wo or :on↩ . . . . . . make current window one on screen
^Wj ^Wk . . . . move to window below, above
^Ww ^W^W . . . . move to window below, above (wrap)
^Wn+ ^Wn- . . . Increase/decrease window size by n lines
^Wn > ^Wn < . . . . Increase/decrease window width

Misc Ex Commands (↩)
:help holy-grail . . . . show 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 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

Folding
:set fmt=indent↩ . . . . indent-foldmethod
zf$m$ . . . . create fold of movement m
:$r$fo . . . . 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

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

Miscellaneous
:sh↩ :!c↩ . . . start shell, execute command c in shell
K . . . . run keywordprg (man) on word under cursor
:make↩ . . . . run compiler, 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
^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
gf . . . . open file which filename is under cursor
:mkview [f] . . . . save view configuration [to file f]
:loadview [f] . . . . load view configuration [from file f]
:set ff=dos↩ . . . . convert file to dos eol format
:e ++ff=unix↩ . . . . reopen file in unix eol format