

 $\mathbf{F} c$ jump to

prev occu

of c

line 9

jump to

matching

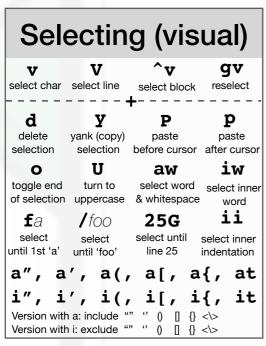
bracket

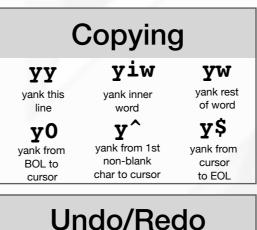
jump to

next occur

scroll current line to

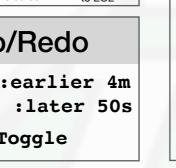
middle of the screen





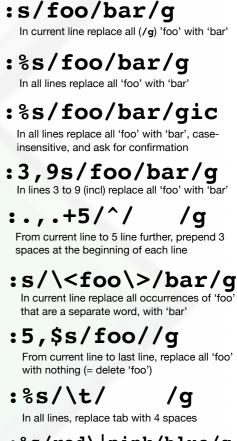
u undo

:UndotreeToggle





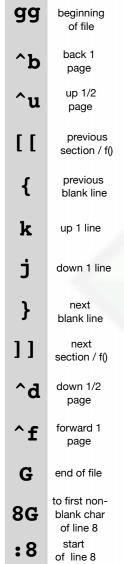




Substitute

/q

/q

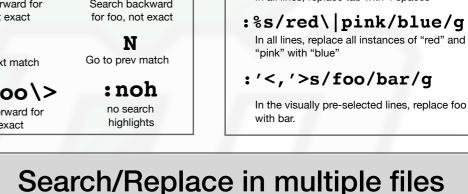


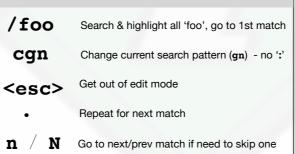
version with a exclude 0 1 8 <>				
Move				
m n my this line to after line n	: m . +2 mv this line 2 lines below	:2,4m \$ mv lines 2,3,4 to after last line		
:2,4m 9	:'<,'>m'>-5	:.,.+2m 9		
mv lines 2,3,4 to after	Move selection to 5 lines before end of	mv 3 lines from this line to		

selection

after line 9







:vimgrep /foo/g *.py	Search in all files. Use : copen to get a list of matches.	
:bufdo %s/foo/bar/ge update		
Apply to all buffers. Ignore pattern not found errors (e). Save file if changes were made (update).		

:arg *.cpp Create arglist with C++ files and load them into buffers :argadd *.h Add header files to arglist, and load into buffers :argdo %s/foo/bar/ge update Apply to every file in arglist

netrw & Ctrl-p

:edit|read|write sftp://user@hostname/path

:edit folder (folder can be.)

:Vexplore split left-right and show file browser

toggle between file browser view options/pathmove cursor to nearest match in file browser

p filename Fuzzy search file. Use ^f and ^b to toggle between files, buffers, and **m**ost **r**ecently

used. Can be used for "Open Recent..."

Global command

: [range]g/pattern/command

:q/foo/d Delete all lines that contain 'foo'

Move all lines from cursor to EOF, that do not contain 'foo', to EOF.

For all lines move them to the beginning of the file = reverse file.

:g/^foo/s/\$/bar For all lines starting with 'foo', substitute/append 'bar' to EOL.

:g/foo/# Show with line nrs. Then go to a line.

Case switching

Toggle case of character under cursor

g~iw Toggle case of the inner word

g~\$ Toggle case from cursor to EOL

gUw Convert rest of word to large caps

guu Convert current line to upper case

gu0 Convert from BOL to cursor to small caps.

Vimtex

\11 Start/stop continuous compiling mode

\1v Jump to PDF viewer at the same point of the cursor

\1c Clean auxiliary files

[Jump to the previous (sub)section

]] Jump to the next (sub)section

% Move between matching delimiters

Tags and YCM

: tag name Jump to (1st) match of the tag. :stag shows it in a new window. Name can be a regex (e.g. /Detect*).

:ts name List all tag matches, and lets you select one. If name is absent, use list of last :tag command.

Jump to tag List all matches of Go back In insert mode: show under cursor tag under cursor one jump window with f() def.

:Tagbar Toggle show/hide tagbar

Double-click or press enter on a tag to jump

• Press space on a function to display arg list.

:!ctags -R --extra=+f Create tags file, recursively, in current dir

Windows

:split :vsplit :only :close

split topbottom

split left-right

keep only current window close current

Indenting

CSV

:WhatColumn Show current column nr

:nrColumns Show total numbers of columns

:DeleteColumn 2 Delete column nr 2

:MoveColumn 2 4 Move col 2 to right next to col 4

:DupColumn Duplicate current column

: Header 2 Open separate fixed window with first 2 header lines. Close with :Header!

:%ArrangeColumn! Prettify the columns by aligning them

:NewDelimiter; Change the delimiter to;

:1,10Sort 3 Sort lines 1-10 of column 3

:SearchInColumn 1 /foo/

Search for patter foo in column 1

:%Substitute 1,4/foo/bar/gc

Subst in cols 1-4 foo with bar, globally + ask for confirmation

Fugitive

:Gblame Show git blame info for an indexed file.

:Gedit SHA Show info for blob, tree, commits, and tags

:Gedit master:python/transits.py Read-only!

: Gdiff Shows differences between working copy of the current file and the index, in a split window. **:q** to exit.

:Gdiffsplit! In case of merge conflicts: open 3-way diffs

E.g. if you have conflicts after:

(feature) \$ git checkout master
(master) \$ git merge feature

Left window: version of the master branch Right window: version of the feature branch

Center window: current working copy

Keep your cursor in the center window (working copy).

:diffget //2 include left version in working copy
:diffget //3 include right version in working copy

:diffupdate update the diff highlighting

fzf

:Files Fuzzy search files

:Lines Fuzzy grep in all open buffers

:BLines Fuzzy grep in current buffer

: Tags Fuzzy search all tags

:BTags Fuzzy search tags in current buffer

:GFiles? Git status

: Commits Fuzzy search in all commits

:BCommits Fuzzy search in commits of this buffer

Special Text Objects

gn Current search pattern

ii Inner indentation level, no line above

aa Function <u>arg</u> under cursor, with comma

a<u>i</u> Indentation level + one line above (*Python*)

ia Function **a**rg under cursor, without comma (*inner*)

a<u>I</u>ndentation level + line above + line below (C++)

Example in C++: caI: change current indentation level incl. {} above/below

Easymotion

\\w Word motion, then select appropriate letter. Forward.

\\b Idem but backward

\\s char Trigger character motion, then select letter.

Miscellaneous

^a / **^x** Incr/decr the nr under cursor. **J** Join this line with the next one

:set scrollbind Scroll synchronously. Set in all windows that need to be in-sync. Add! to to undo.

:ab fe for example Replace abbreviation on the fly

:set list Show all the invisible characters

:set nolist Stop showing all the invisible characters

^oo After startup: load last edited file ^n In insert mode: word completion

:Matrix Follow the white rabbit

:ToggleWhitespace (Un)highlight trailing whitespace

:StripWhitespace Strip trailing whitespace

:set wrap linebreak Wrap text around

:%s/ctrl)v(ctrl)m/\r/g Convert dos ^M to unix carriage return

gv Reselect last selection

Vr- Select line, replace all chars with `-`

Tabularize

:Tab/= Align selection on the '=' char

:Tab/,/r1c110

First part right aligned, then 1 space, then comma center aligned, then 1 space, then part left aligned, no extra space, then rest

Repeating

Repeat search - forward

; Rpt last f c - forward

N Repeat search - backward

Rpt last f c - backward

Rpt last ':' command

Rpt last :s/x/y/ subst

Rpt last change in Normal mode

Swapping

CX {motion} Select the first part of text to be swapped...

... go to the 2nd part of text. Repeat cmd: swaps

CXX Like cx for the current line
CXC Clear selection and memory

Folds

zo/zc
Open/close

1 fold under

cursor

:c `a

:]'/['

zO/zC

Open/close all folds under cursor

zm/zr

Increase /reduce fold level

Go to start of next/previous fold

zj/zk

Old School:

^v

qcc

Block select

select Select the block

With tcomment:

gc{motion}

Insert mode at beginningWrite comment symbols

<esc> Exit & apply to all lines

| ^

(Un)commenting

Block select

Select comment symbols

d delete

Toggle comment. E.g gc4j or gcaB

Markers

m a
m a!
'a
`a
:marks
:delm ab /!

Set marker named 'a' under cursor

Same but override previous marker 'a' Go to start of line containing marker 'a' Jump to marker 'a' (back quote)

Show a list of markers

Delete marker 'a' and 'b' / all markers

Change from cursor to marker 'a'

Jump to next/prev line with marker Special marker: last change in buffer

MacVim

Toggle comment for this line.

%= / **%-**

Increase/decrease font size

Macros

qa Start recording to register 'a'... Fancy commands

Stop recording

ter 'a' @a

Execute macro

5@a Execute

Execute 5 times

:reg List all macros

Clipboard

:w !pbcopy

Write selected lines to clipboard (visual mode)

:r !pbpaste

Insert contents of clipboard at current position

Insert Mode

^o In INSERT mode: go back to normal mode for 1 command

^r=<expr> In Insert mode: compute and insert result of expression