Vim Cheat Sheet

Global

`:help keyword` - open help for keyword

`:saveas file` - save file as

`:close` - close current pane

`K` - open man page for word under the cursor

Cursor movement

`h` - move cursor left

`j` - move cursor down

`k` - move cursor up

`l` - move cursor right

`H` - move to top of screen

`M` - move to middle of screen

`L` - move to bottom of screen

`w` - jump forwards to the start of a word

`W` - jump forwards to the start of a word (words can contain punctuation)

`e` - jump forwards to the end of a word

`E` - jump forwards to the end of a word (words can contain punctuation)

`b` - jump backwards to the start of a word

`B` - jump backwards to the start of a word (words can contain punctuation)

`%` - move to matching character (default supported pairs: '()', '{}', '[]' - use <code>:h matchpairs</code> in vim for more info)

`0` - jump to the start of the line

`^` - jump to the first non-blank character of the line

`$` - jump to the end of the line

`g_` - jump to the last non-blank character of the line

`gg` - go to the first line of the document

`G` - go to the last line of the document

`5G` - go to line 5

`fx` - jump to next occurrence of character x

`tx` - jump to before next occurrence of character x

`Fx` - jump to previous occurence of character x

`Tx` - jump to after previous occurence of character x

`;` - repeat previous f, t, F or T movement

`,` - repeat previous f, t, F or T movement, backwards

`}` - jump to next paragraph (or function/block, when editing code)

`{` - jump to previous paragraph (or function/block, when editing code)

`zz` - center cursor on screen

`Ctrl` + `e` - move screen down one line (without moving cursor)

`Ctrl` + `y` - move screen up one line (without moving cursor)

`Ctrl` + `b` - move back one full screen

`Ctrl` + `f` - move forward one full screen

`Ctrl` + `d` - move forward 1/2 a screen

`Ctrl` + `u` - move back 1/2 a screen

Tip Prefix a cursor movement command with a number to repeat it. For example, `4j` moves down 4 lines.

Insert mode - inserting/appending text

`i` - insert before the cursor

`I` - insert at the beginning of the line

`a` - insert (append) after the cursor

`A` - insert (append) at the end of the line

`o` - append (open) a new line below the current line

`O` - append (open) a new line above the current line

`ea` - insert (append) at the end of the word

`Esc` - exit insert mode

Editing

`r` - replace a single character

`J` - join line below to the current one with one space in between

`gJ` - join line below to the current one without space in between

`gwip` - reflow paragraph

`cc` - change (replace) entire line

`C` - change (replace) to the end of the line

`c$` - change (replace) to the end of the line

`ciw` - change (replace) entire word

`cw` - change (replace) to the end of the word

`s` - delete character and substitute text

`S` - delete line and substitute text (same as cc)

`xp` - transpose two letters (delete and paste)

`u` - undo

`Ctrl` + `r` - redo

`.` - repeat last command

Marking text (visual mode)

`v` - start visual mode, mark lines, then do a command (like y-yank)

`V` - start linewise visual mode

`o` - move to other end of marked area

`Ctrl` + `v` - start visual block mode

`O` - move to other corner of block

`aw` - mark a word

`ab` - a block with ()

`aB` - a block with {}

`ib` - inner block with ()

`iB` - inner block with {}

`Esc` - exit visual mode

Visual commands

`>` - shift text right

`<` - shift text left

`y` - yank (copy) marked text

`d` - delete marked text

`~` - switch case

Registers

`:reg` - show registers content

`"xy` - yank into register x

`"xp` - paste contents of register x

> **Tip** Registers are being stored in ~/.viminfo, and will be loaded again on next restart of vim.

> **Tip** Register 0 contains always the value of the last yank command.

Marks

`:marks` - list of marks

`ma` - set current position for mark A

`` `a `` - jump to position of mark A

`` y`a `` - yank text to position of mark A

Macros

`qa` - record macro a

`q` - stop recording macro

`@a` - run macro a

`@@` - rerun last run macro

Cut and paste

`yy` - yank (copy) a line

`2yy` - yank (copy) 2 lines

`yw` - yank (copy) the characters of the word from the cursor position to the start of the next word

`y$` - yank (copy) to end of line

`p` - put (paste) the clipboard after cursor

`P` - put (paste) before cursor

`dd` - delete (cut) a line

`2dd` - delete (cut) 2 lines

`dw` - delete (cut) the characters of the word from the cursor position to the start of the next word

`D` - delete (cut) to the end of the line

`d$` - delete (cut) to the end of the line

`x` - delete (cut) character

Exiting

`:w` - write (save) the file, but don't exit

`:w !sudo tee %` - write out the current file using sudo

`:wq` or `:x` or `ZZ` - write (save) and quit

`:q` - quit (fails if there are unsaved changes)

`:q!` or `ZQ` - quit and throw away unsaved changes

`:wqa` - write (save) and quit on all tabs

Search and replace

`/pattern` - search for pattern

`?pattern` - search backward for pattern

`\vpattern` - 'very magic' pattern: non-alphanumeric characters are interpreted as special regex symbols (no escaping needed)

`n` - repeat search in same direction

`N` - repeat search in opposite direction

`:%s/old/new/g` - replace all old with new throughout file

`:%s/old/new/gc` - replace all old with new throughout file with confirmations

`:noh` - remove highlighting of search matches

Search in multiple files

`:vimgrep /pattern/ {file}` - search for pattern in multiple files

e.g. `:vimgrep /foo/ **/*`

`:cn` - jump to the next match

`:cp` - jump to the previous match

`:copen` - open a window containing the list of matches

Working with multiple files

`:e file` - edit a file in a new buffer

`:bnext` or `:bn` - go to the next buffer

`:bprev` or `:bp` - go to the previous buffer

`:bd` - delete a buffer (close a file)

`:ls` - list all open buffers

`:sp file` - open a file in a new buffer and split window

`:vsp file` - open a file in a new buffer and vertically split window

`Ctrl` + `ws` - split window

`Ctrl` + `ww` - switch windows

`Ctrl` + `wq` - quit a window

`Ctrl` + `wv` - split window vertically

`Ctrl` + `wh` - move cursor to the left window (vertical split)

`Ctrl` + `wl` - move cursor to the right window (vertical split)

`Ctrl` + `wj` - move cursor to the window below (horizontal split)

`Ctrl` + `wk` - move cursor to the window above (horizontal split)

Tabs

`:tabnew` or `:tabnew file` - open a file in a new tab

`Ctrl` + `wT` - move the current split window into its own tab

`gt` or `:tabnext` or `:tabn` - move to the next tab

`gT` or `:tabprev` or `:tabp` - move to the previous tab

`#gt` - move to tab number #

`:tabmove #` - move current tab to the #th position (indexed from 0)

`:tabclose` or `:tabc` - close the current tab and all its windows

`:tabonly` or `:tabo` - close all tabs except for the current one

`:tabdo` command - run the `command` on all tabs (e.g. `:tabdo q` - closes all opened tabs)

Additional Resources

Languages

العربية

Deutsch

English

Español

Persian

Français

Bahasa Indonesia

日本語

한국어

Nederlands

Italiano

Polski

Português - Brasil

Português - Portugal

Romana

Русский

Slovenčina

සිංහල

Svenska

ภาษาไทย

Türkçe

Українська

简体中文

繁體中文

About the vim cheat sheet

This project aims to be one of the most accessible vim guides available. We made sure to support mobile, desktop, and other languages.

You can read about how to contribute (and help improve) by viewing our README. There you can see how to set up this project, or how to contribute a new language. Here is a big thank you to our contributors!

Other places to find this document

This document was embedded in DuckDuckGo.

Checkout the source on Github