Vim Cheat Sheet

Global

`:h[elp] keyword` - open help for keyword

`:sav[eas] file` - save file as

`:clo[se]` - close current pane

`:ter[minal]` - open a terminal window

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

> **Tip** Run `vimtutor` in a terminal to learn the first Vim commands.

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 `:h matchpairs` 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

`5gg` or `5G` - go to line 5

`gd` - move to local declaration

`gD` - move to global declaration

`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 movemen 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

`Ctrl` + `h` - delete the character before the cursor during insert mode

`Ctrl` + `w` - delete word before the cursor during insert mode

`Ctrl` + `j` - begin new line during insert mode

`Ctrl` + `t` - indent (move right) line one shiftwidth during insert mode

`Ctrl` + `d` - de-indent (move left) line one shiftwidth during insert mode

`Ctrl` + `n` - insert (auto-complete) next match before the cursor during insert mode

`Ctrl` + `p` - insert (auto-complete) previous match before the cursor during insert mode

`Ctrl` + `rx` - insert the contents of register x

`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

`g~` - switch case up to motion

`gu` - change to lowercase up to motion

`gU` - change to uppercase up to motion

`cc` - change (replace) entire lin

`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

`U` - restore (undo) last changed line

`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 {}

`at` - a block with <> tags

`ib` - inner block with ()

`iB` - inner block with {}

`it` - inner block with <> tags

`Esc` - exit visual mode

Tip Instead of `b` or `B` one c

Visual commands

`>` - shift text right

`<` - shift text left

`y` - yank (copy) marked text

`d` - delete marked text

`~` - switch case

`u` - change marked text to lowercase

`U` - change marked text to uppercase

Registers

`:reg[isters]` - show registers content

`"xy` - yank into register x

`"xp` - paste contents of register x

`"+y` - yank into the system clipboard register

`"+p` - paste from the system clipboard register

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

Tip Special registers:

- `0` - last yank
- `"` - unnamed register, last delete or yank
- `%` - current file name
- `#` - alternate file name
- `*` - clipboard contents (X11 primary)
- `+` - clipboard contents (X11 clipboard)
- `/` - last search pattern
- `:` - last command-line
- `.` - last inserted text
- `-` - last small (less than a line) delete
- `=` - expression register
- `_` - black hole register

Marks and positions

`: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

`` `0 `` - go to the position where Vim was previously exited

`` `" `` - go to the position when last editing this file

`` `. `` - go to the position of the last change in this file

`` `` `` - go to the position before the last jump

`:ju[mps]` - list of jumps

`Ctrl` + `i` - go to newer position in jump list

`Ctrl` + `o` - go to older position in jump list

`:changes` - list of changes

`g,` - go to newer position in change list

`g;` - go to older position in change list

`Ctrl` + `]` - jump to the tag under cursor

Tip To jump to a mark you can either use a backtick (`` ` ``) or an apostrophe (`'`). Using an apostrophe jumps to the beginning (first non-black) of the line holding the mark.

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) 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

Indent text

`>>` - indent (move right) line one shiftwidth

`<<` - de-indent (move left) line one shiftwidth

`>%` - indent a block with () or {} (cursor on brace)

`>ib` - indent inner block with ()

`>at` - indent a block with <> tags

`3==` - re-indent 3 lines

`=%` - re-indent a block with () or {} (cursor on brace)

`=iB` - re-indent inner block with {}

`gg=G` - re-indent entire buffer

`]p` - paste and adjust indent to current line

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 aw

`: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[lsearch]` - remove highlighting of search matches

Search in multiple files

`:vim[grep] /pattern/ {`{file}`}` - search for pattern in multiple files

e.g. `:vim[grep] /foo/ **/*`

`:cn[ext]` - jump to the next match

`:cp[revious]` - jump to the previous match

`:cope[n]` - open a window containing the list of matches

`:ccl[ose]` - close the quickfix window

Tabs

`:tabnew` or `:tabnew {page.words.file}` - open a file in a new tab

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

`gt` or `:tabn[ext]` - move to the next tab

`gT` or `:tabp[revious]` - move to the previous tab

`#gt` - move to tab number #

`:tabm[ove] #` - move current t

`:tabc[lose]` - close the current tab and all its windows

`:tabo[nly]` - close all tabs except for the current one

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

Working with multiple files

`:e[dit] file` - edit a file in a new buffer

`:bn[ext]` - go to the next buffer

`:bp[revious]` - go to the previous buffer

`:bd[elete]` - delete a buffer (close a file)

`:b[uffer]#` - go to a buffer by index #

`:b[uffer] file` - go to a buffer by file

`:ls` or `:buffers` - list all open buffers

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

`:vs[plit] file` - open a file in a new buffer and vertically split window

`:vert[ical] ba[ll]` - edit all buffers as vertical windows

`:tab ba[ll]` - edit all buffers as tabs

`Ctrl` + `ws` - split window

`Ctrl` + `wv` - split window vertically

`Ctrl` + `ww` - switch windows

`Ctrl` + `wq` - quit a window

`Ctrl` + `wx` - exchange current window with next one

`Ctrl` + `w=` - make all windows equal height & width

`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)

Diff

`zf` - manually define a fold up to motion

`zd` - delete fold under the cursor

`za` - toggle fold under the cursor

`zo` - open fold under the cursor

`zc` - close fold under the cursor

`zr` - reduce (open) all folds by one level

`zm` - fold more (close) all folds by one level

`zi` - toggle folding functionality

`]c` - jump to start of next change

`[c` - jump to start of previous change

`do` or `:diffg[et]` - obtain (get) difference (from other buffer)

`dp` or `:diffpu[t]` - put difference (to other buffer)

`:diffthis` - make current window part of diff

`:dif[fupdate]` - update differences

`:diffo[ff]` - switch off diff mode for current window

Tip The commands for folding (e.g. `za`) operate on one level. To operate on all levels, use uppercase letters (e.g. `zA`).

Tip To view the differences of files, one can directly start Vim in diff mode by running `vimdiff` in a terminal. One can even set this as `git difftool`.

Additional Resources

Languages

العربية

Deutsch

English

Español

Persian

Français

עברית

Bahasa Indonesia

Italiano

日本語

한국어

□□□□ ဖ ာ

Nederlands

Polski

Português - Brasil

Português - Portugal

Romana

Русский

සිංහල

Slovenčina

Svenska

ภาษาไทย

Türkçe

Українська

简体中文

中文 (台灣)

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.

❤ Please consider sponsoring this project ❤!

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 .

More resources

Interactive Vim tutorial: Open Vim

Vim quick reference from Vim help pages: quickref.txt

List of all Vim ex (`:`) commands:

Checkout the source on Github

version: 3.1.1