Vim cheatsheet

Limited time offer: Get 10 free Adobe Stock images.

ads via Carbon

Vim is a very efficient text editor. This reference was made for Vim 8.0.
For shortcut notation, see `:help key-notation`.

Exiting

`:qa`	Close all files
`:qa!`	Close all files, abandon changes
`:w`	Save
`:wq` / `:x`	Save and close file
`:q`	Close file
`:q!`	Close file, abandon changes
`ZZ`	Save and quit
`ZQ`	Quit without checking changes

Navigating

`h` `j` `k` `l`	Arrow keys
`<C-U>` / `<C-D>`	Half-page up/down
`<C-B>` / `<C-F>`	Page up/down
Words	
`b` / `w`	Previous/next word
`ge` / `e`	Previous/next end of word

Line	
`0` (zero)	Start of line
`^`	Start of line (after whitespace)
`$`	End of line
Character	
`fc`	Go forward to character c
`Fc`	Go backward to character c
Document	
`gg`	First line
`G`	Last line
`:n`	Go to line n
`nG`	Go to line n
Window	
`zz`	Center this line
`zt`	Top this line
`zb`	Bottom this line
`H`	Move to top of screen
`M`	Move to middle of screen
`L`	Move to bottom of screen
Search	
`n`	Next matching search pattern
`N`	Previous match
`*`	Next whole word under cursor
`#`	Previous whole word under cursor
Tab pages	
`:tabedit [file]`	Edit file in a new tab
`:tabfind [file]`	Open file if exists in new tab

`:tabclose`	Close current tab
`:tabs`	List all tabs
`:tabfirst`	Go to first tab
`:tablast`	Go to last tab
`:tabn`	Go to next tab
`:tabp`	Go to previous tab

Editing

`a`	Append
`A`	Append from end of line
`i`	Insert
`o`	Next line
`O`	Previous line
`s`	Delete char and insert
`S`	Delete line and insert
`C`	Delete until end of line and insert
`r`	Replace one character
`R`	Enter Replace mode
`u`	Undo changes
`<C-R>`	Redo changes

Exiting insert mode

`Esc` / `<C-[>`	Exit insert mode
`<C-C>`	Exit insert mode, and abort current command

Clipboard

`x`	Delete character
`dd`	Delete line (Cut)
`yy`	Yank line (Copy)
`p`	Paste
`P`	Paste before
`"*p` / `"+p`	Paste from system clipboard
`"*y` / `"+y`	Paste to system clipboard

Visual mode

`v`	Enter visual mode
`V`	Enter visual line mode
`<C-V>`	Enter visual block mode
In visual mode	
`d` / `x`	Delete selection
`s`	Replace selection
`y`	Yank selection (Copy)

See Operators for other things you can do.

Operators

Usage

Operators let you operate in a range of text (defined by motion). These are performed in normal mode.

`d`	`w`
Operator	Motion

Operators list

`d`	Delete
`y`	Yank (copy)
`c`	Change (delete then insert)
`>`	Indent right
`<`	Indent left
`=`	Autoindent
`g~`	Swap case
`gU`	Uppercase
`gu`	Lowercase
`!`	Filter through external program

See `:help operator`

Examples

Combine operators with motions to use them.

dd	(repeat the letter) Delete current line
dw	Delete to next word
db	Delete to beginning of word
2dd	Delete 2 lines
dip	Delete a text object (inside paragraph)
(in visual mode) d	Delete selection

See: `:help motion.txt`

Text objects

Usage

Text objects let you operate (with an operator) in or around text blocks (objects).

`v`	`i`	`p`
Operator	[i]nside or [a]round	Text object

Text objects

`p`	Paragraph
`w`	Word
`s`	Sentence
`[` `(` `{` `<`	A [], (), or {} block
`'` `"` `` ` ``	A quoted string
`b`	A block [(
`B`	A block in [{
`t`	A XML tag block

Examples

`vip`	Select paragraph
`vipipipip`	Select more
`yip`	Yank inner paragraph
`yap`	Yank paragraph (including newline)
`dip`	Delete inner paragraph
`cip`	Change inner paragraph

See Operators for other things you can do.

Diff

`gvimdiff file1 file2 [file3]`	See differences between files, in HMI

Misc

Folds

`zo` / `zO`	Open
`zc` / `zC`	Close
`za` / `zA`	Toggle
`zv`	Open folds for this line
`zM`	Close all
`zR`	Open all
`zm`	Fold more (foldlevel += 1)
`zr`	Fold less (foldlevel -= 1)
`zx`	Update folds

Uppercase ones are recursive (eg, `zO` is open recursively).

Navigation

`%`	Nearest/matching `{[()]}`
`[(` `[{` `[<`	Previous `(` or `{` or `<`
`])`	Next
`[m`	Previous method start
`[M`	Previous method end

Jumping

`<C-O>`	Go back to previous location
`<C-I>`	Go forward

`gf`	Go to file in cursor

Counters

`<C-A>`	Increment number
`<C-X>`	Decrement

Windows

`z{height}<Cr>`	Resize pane to `{height}` lines tall

Tags

`:tag Classname`	Jump to first definition of Classname
`<C-]>`	Jump to definition
`g]`	See all definitions
`<C-T>`	Go back to last tag
`<C-O> <C-I>`	Back/forward
`:tselect Classname`	Find definitions of Classname
`:tjump Classname`	Find definitions of Classname (auto-select 1st)

Case

`~`	Toggle case (Case => cASE)
`gU`	Uppercase
`gu`	Lowercase
`gUU`	Uppercase current line (also `gUgU`)
`guu`	Lowercase current line (also `gugu`)

Do these in visual or normal mode.

Marks

`` `^ ``	Last position of cursor in insert mode
`` `. ``	Last change in current buffer
`` `" ``	Last exited current buffer
`` `0 ``	In last file edited
`''`	Back to line in current buffer where jumped from
``` `` ```	Back to position in current buffer where jumped from
`` `[ ``	To beginning of previously changed or yanked text
`` `] ``	To end of previously changed or yanked text
`` `< ``	To beginning of last visual selection
`` `> ``	To end of last visual selection
`ma`	Mark this cursor position as a
`` `a ``	Jump to the cursor position a
`'a`	Jump to the beginning of the line with position a
`d'a`	Delete from current line to line of mark a
`` d`a ``	Delete from current position to position of mark a
`c'a`	Change text from current line to line of a
`` y`a ``	Yank text from current position to position of a
`:marks`	List all current marks
`:delm a`	Delete mark a
`:delm a-d`	Delete marks a, b, c, d
`:delm abc`	Delete marks a, b, c

Misc

`.`	Repeat last command
`]p`	Paste under the current indentation level

`:set ff=unix`	Convert Windows line endings to Unix line endings

Command line

`<C-R><C-W>`	Insert current word into the command line
`<C-R>"`	Paste from " register
`<C-X><C-F>`	Auto-completion of path in insert mode

Text alignment

```
:center [width]
:right [width]
:left
```

See `:help formatting`

Calculator

`<C-R>=128/2`	Shows the result of the division : '64'

Do this in insert mode.

Exiting with an error

```
:cq
:cquit
```

Works like `:qa`, but throws an error. Great for aborting Git commands.

Spell checking

`:set spell spelllang=en_us`	Turn on US English spell checking
`]s`	Move to next misspelled word after the cursor
`[s`	Move to previous misspelled word before the cursor
`z=`	Suggest spellings for the word under/after the cursor

`zg`	Add word to spell list
`zw`	Mark word as bad/mispelling
`zu` / `C-X (Insert Mode)`	Suggest words for bad word under cursor from spellfile

See `:help spell`

Also see

- Vim cheatsheet (vim.rotrr.com)
- Vim documentation (vimdoc.sourceforge.net)
- Interactive Vim tutorial (openvim.com)

0 Comments for this cheatsheet. Write yours!

Search 356+ cheatsheets

Over 356 curated cheatsheets, by developers for developers.

Devhints home

Other Vim cheatsheets

Vimdiff
cheatsheet ●

Vim scripting
cheatsheet ●

Tabular
cheatsheet ●

Projectionist
cheatsheet

Vim digraphs
cheatsheet

Vim Easyalign
cheatsheet

Top cheatsheets

Elixir
cheatsheet ●

ES2015+
cheatsheet ●

React.js
cheatsheet ●

Vimdiff
cheatsheet ●

Vim scripting
cheatsheet ●

Vue.js
cheatsheet ●