

neovim

- [NvChad Shortcuts & Hotkeys](#)
- [NvChad Installation](#)
- [nvim cheatsheet](#)
- [neovim + lazyvim Installation](#)

NvChad Shortcuts & Hotkeys

https://www.reddit.com/r/neovim/comments/12qku4w/nvchad_cheatsheet/

blankline Jump to current_context <leader> + cc	comment toggle comment <leader> + /	comment (v) toggle comment <leader> + /	general toggle relative number <leader> + rn window right <C-l> move down <C-j> window down <C-j> Mapping cheatsheet <leader> + ch new buffer <leader> + b save file <C-s> move down <Down> window up <C-k> copy whole file <C-c> move up <Up> clear highlights <Esc> toggle line number <leader> + n window left <C-h> move up k
general (l) beginning of line <C-b> move right <C-l> end of line <C-e> move down <C-j> move left <C-h> move up <C-k>	gitsigns Reset hunk <leader> + rh Jump to next hunk jc Preview hunk <leader> + ph Toggle deleted <leader> + td Jump to prev hunk lc Blame line <leader> + gb	nvimtree focus nvimtree <leader> + e toggle nvimtree <C-n>	general (t) escape terminal mode <C-x>
general (v) move up <Up> move down <Down>	lspconfig lsp references gr floating diagnostic <leader> + f add workspace folder <leader> + wa lsp declaration gD remove workspace folder <leader> + wr goto_next jd diagnostic setloclist <leader> + q lsp hover K lsp formatting <leader> + fm lsp implementation gi lsp signature_help <leader> + ls lsp definition type <leader> + D list workspace folders <leader> + wl lsp rename <leader> + ra lsp definition gd goto prev [d lsp code_action <leader> + ca	nvterm (t) toggle vertical term <A-v> toggle horizontal term <A-h> toggle floating term <A-i>	whichkey which-key query lookup <leader> + wk which-key all keymaps <leader> + wK
general (x) dont copy replaced text p move up k move down j		tabuline close buffer <leader> + x goto next buffer <tab> goto prev buffer <S-tab>	
		telescope git status <leader> + gt find buffers <leader> + fb pick hidden term <leader> + pt help page <leader> + fh nvchad themes <leader> + th find oldfiles <leader> + fo find files <leader> + ff find in current buffer <leader> + fz find all <leader> + fa git commits <leader> + cm live grep <leader> + fw	

space+c+h CHEATSHEET

space+t+h THEMES

ctrl+n nvimtree (m: mark, a: add file, c: copy, p: paste, r: rename)

ctrl+hjkl fenster switchen

space+f+f find file

space+f+b find buffered file

:vsp vertical split

:sp split

space+n line numbers

space+r+n relative line numbers

space+x close buffer tab

tab / shift+tab switch buffer tab

space+h terminal horizontal split

space+v terminal vertical split

NvChad Installation

The screenshot shows an IDE with several files open. The left sidebar shows a file explorer with folders like 'assets', 'config', 'src', and 'static'. The main editor area shows code in 'utils.js' and 'api.js'. A function autocomplete menu is open over the 'function' keyword in 'utils.js', listing options like 'function', 'Function', 'wrap selection in arrow function', etc. The bottom right shows a terminal window with the output of 'astro dev', indicating that the development server is running on 'http://localhost:3000/'.

The screenshot shows an IDE with several files open. The left sidebar shows a file explorer with folders like 'benchmarks', 'examples', 'lib', 'middleware', 'router', 'spec.js', and 'view.js'. The main editor area shows code in 'spec.js' and 'route.js'. A function autocomplete menu is open over the 'function' keyword in 'spec.js', listing options like 'function', 'Function', 'wrap selection in arrow function', etc. The bottom right shows a terminal window with the output of 'astro dev', indicating that the development server is running on 'http://localhost:3000/'.

<https://nvchad.com/docs/quickstart/install>

```
# essentials für zb C-Compiler
apt install build-essential

# Nerdfont installieren
wget -P ~/.local/share/fonts https://github.com/ryanoasis/nerd-
fonts/releases/download/v3.0.2/JetBrainsMono.zip \
&& cd ~/.local/share/fonts \
&& unzip JetBrainsMono.zip \
&& rm JetBrainsMono.zip \
&& fc-cache -fv

# Nvim installieren
snap install nvim --classic

# NvChad installieren
git clone https://github.com/NvChad/starter ~/.config/nvim && nvim

# in nvim:
:MasonInstallAll
```

Falls es Probleme auf einem Hetzner Server mit dem Nutzerverzeichnis unter root gibt:

(in nvim mit `:echo stdpath("config")` prüfen was nvim als home-Verzeichnis nimmt)

```
mkdir -p /home/root/.config
mv /root/.config/nvim /home/root/.config/
```

nvim cheatsheet

https://www.reddit.com/r/vim/comments/18b1ngj/neovim_cheat_sheet/

EDITING

- r replace single character
- R enter replace mode
- J join line below to current one with one space (or between lines)
- kJ join line below to current one with no spaces
- cc change entire line
- C change to the end of line
- o change with (out)liner, which inserts a line
- cw change inner word
- s delete character and substitute text
- S delete line and substitute text
- xp transpose two letters (a->-a)
- v swap
- vy swap
- yy yank
- repeat last command

CUT & PASTE

- p/p/P yank single line
- P/P/P yank # of lines
- y yank from cursor to start of next word
- Y yank # of lines with a movement (Dy|l)
- "yy yank current line into named buffer a
- P put (paste) after the cursor
- P put (paste) before the cursor
- "p put text from buffer a before cursor
- dd delete current line
- DD delete # of lines downward from cursor
- ddm delete # of lines with movement
- D delete to end of line
- ddw delete until word
- x delete character at cursor
- X delete backwards from cursor

SCREEN MOVEMENT

- zz cursor line to center of screen
- zt cursor line to top of screen
- zb cursor line to bottom of screen
- "u move screen forward one line
- "y move screen backward one line
- "u move forward half a screen
- "u move backward half a screen
- "f move forward a full screen
- "b move backward a full screen

VISUAL MODE

- v start visual mode, mark lines, then perform a command (like yank)
- V start line-wise visual mode
- o move to other end of marked area
- "v start visual-block mode
- O move to other corner of a block
- hl highlight/match a word
- da a block with (V|v)
- lB/lb lesser block with (V|v)
- Esc: exit visual mode

MARK POSITION

- MC mark current position as c
- c move cursor to c
- ' return to previous context
- 'c move to the beginning of the line containing mark c
- marks list of marks

REGISTER

- reg show registers content
- "y yank into register x
- "p paste contents of register x

HOW TO THINK IN VIM

To be efficient in Vim, learning commands is not enough. To accomplish editing tasks efficiently, you must learn to think in Vim. **REGISTER MORE SENTENCES**. Because Vim is a symbolic language with a formal grammatical structure, these Vim sentences describe what you need to do in a somewhat natural way and put the heart of Vim editing.

VBMS are the thing you want to do - (C)hange, (D)elate, (Y)ank, (V)isual etc.

REGISTER change the marks are abbreviated - (C)hange, (D)elate, (Y)ank, (V)isual, etc.

MARKS are what you are editing - (C)hange, (D)elate, (Y)ank, (V)isual, etc.

See examples

- aw delete inside word to delete word under cursor
- cw change the inner word, respecting delimitation
- c1* change inside quotation marks to change string inside next marks
- c1x change everything from the cursor till the letter x
- vap visually select (around) this paragraph
- yi(copy everything inside the parentheses
- di/fo delete from here until the second occurrence of 'fo'
- f/bo find the next occurrence of 'o' - move forward 2 words, and change word
- TM hit is the the thing. If you can't figure out more than 2-3 lines, there's probably a more efficient way to move or accomplish what you're doing. Stop and think it over!

INSERT MODE

- I insert before the cursor
- i insert at the beginning of the line
- A append after the cursor
- a append at the end of the line
- O open new line below the current one
- O open new line above the current one
- oa append at the end of the word
- o#i insert text # times
- Esc exit insert mode

MACROS

- qa record macro at register "a"
- q stop recording macro
- Ma run macro at register "a"
- @@ repeat last macro

SEARCH

- /pattern search for pattern
- ?pattern search for pattern backwards
- Search Pattern: Finds...
 - a line beginning with char a
 - [abc] string containing any of the letters in the brackets; john or joan
 - \. disable char special meaning
 - A, M, AAA
 - * a or y
 - r-d rad or rod or rid period matches any single character
 - [:cm] the character or then
 - [:cm:] the or between
 - \< the^ the

REPLACE

- s/old/new replace next old with new
- s/old/new/g replace instances of old with new in current line
- %s/old/new/g replace all old with new throughout file
- :%s/old/new/g replace all old with new in file with combinations
- 2,3h/r/j/g replace instances of r with y between lines 2 to 3 inclusive. Ignore case
- {r/y}/g delete lines containing/ not containing instances of string x

MISCELLANEOUS COMMANDS

- !q or ZZ write (save) and quit file
- !f write file overriding protection
- !qf quit file (disregarding any changes)
- ! command give this command
- r file read file contents into cur. file
- map x <C> define keystroke x as a command sequence
- unmap x disable the map for x
- !ab a abbreviate p as a, when a is typed in insert mode, it expands to full words or phrases (S)
- unab a disable abbreviation for a
- set [all] display options set by user, if all is included, these are default
- set o activate option o
- set ovv assign value v to option o
- set no deactivate option o

NAVIGATION BAR

- 0 first line of line
- ^ first character of line
- / search
- < previous
- F find
- T toggle
- ge/gE word
- b/B backward
- w/W word
- e/E end
- t toggle
- f find
- > next
- % percent
- \$ end of line

CENTRAL COLUMN

- gG first line
- H cursor to top of screen
- H registered position
- C beginning of sentence
- # half screen
- N next screen
- M cursor to middle of screen
- g_ next/downward

Text Manipulation

i	insert before cursor	r	replace single character
I	insert at start of line	cc	replace line
a	insert after cursor	cw	replace to end of word
A	insert at end of line	c\$	replace to end of line
o	add new line below cursor	s	substitute character
O	add new line above cursor	S	substitute line
ea	insert at end of line	u	undo
Esc	exit	Ctrl	redo

Visual Mode

v	enter visual mode	~	change case
V	enter linewise visual mode	Esc	exit visual mode
Ctrl	start visual block mode		
>	shift text left		
<	shift text right		
>>	shift left by shiftwidth		
<<	shift right by shiftwidth		
==	auto-indent line		

h

move cursor left

b

move to start of previous word

B

move to start of previous word (inc. punctuation)

0

move to start of line

Vim

l

move cursor right

w

move to start of next word

W

move to start of next word (inc. punctuation)

\$

move to end of line

y

yank/copy

yy

yank a line

yw

yank a word

y\$

yank to end of line

p

paste after cursor

P

paste before cursor

dd

delete/cut a line

dw

delete a word

D

delete to end of line

X

delete character

/string

search for "string"

j

move cursor down

G

jump to last line in document

ZZ

save and quit

ZQ

quit without saving

:w

write/save

:q

quit (fails if there are changes)

:wq

write and quit

:X

write and quit

:q!

force quit without saving

:qa

quit all vim buffers

Text Manipulation Cont.

Save & Exit

LINUXACADEMY.COM



neovim + lazyvim Installation

neovim

<https://github.com/neovim/neovim/blob/master/INSTALL.md>

```
sudo apt install software-properties-common
# sudo add-apt-repository ppa:neovim-ppa/unstable
sudo add-apt-repository ppa:neovim-ppa/stable
sudo apt update
sudo apt install neovim
```

lazyvim

<https://www.lazyvim.org/installation>

```
# required
mv ~/.config/nvim{,.bak}
# optional but recommended
mv ~/.local/share/nvim{,.bak}
mv ~/.local/state/nvim{,.bak}
mv ~/.cache/nvim{,.bak}

git clone https://github.com/LazyVim/starter ~/.config/nvim

rm -rf ~/.config/nvim/.git
```

```
nvim
```

It is recommended to run

```
:LazyHealth
```

after installation. This will load all plugins and check if everything is working correctly.

```
sudo apt-get install xclip
```