Neovim

Lua-scriptable modal editor

CONCEPTS

Neovim keeps Vim's modal editing (see the vim sheet). This sheet covers what Neovim adds: Lua config, built-in LSP, terminal, health checks.

Config lives in ~/.config/nvim/init.lua (init.vim also works). Lua modules go in lua/ next to it and load with require('name').

DIFFERENCES FROM VIM

g › **c** › **c**
Toggle line comment

g › **c** › *<motion>*
Toggle comment over motion

g › **O**
Outline of symbols / headings

] › **d** / **[** › **d**
Next / previous diagnostic

LUA CONFIG

:lua › *<code>*
Run a Lua expression

:= › *<expr>*
Print a Lua value

:source › **%**
Reload the current file

:messages
Show earlier messages and errors

:help › *<topic>*
Open help for a topic

Lua equivalents

Vimscript	Lua
`set number`	`vim.o.number = true`
`let g:x = 1`	`vim.g.x = 1`
`nnoremap a b`	`vim.keymap.set()`
`autocmd ...`	`nvim_create_autocmd()`
`command! ...`	`nvim_create_user_command()`

LSP

K
Hover documentation

LSP (cont.)

g › **r** › **n**
Rename symbol

g › **r** › **a**
Code action

g › **r** › **r**
List references

Ctrl + **]**
Go to definition

Ctrl + **S**
Signature help (insert mode)

:checkhealth › **vim.lsp**
Check LSP clients and logs

:lua › **vim.lsp.buf.format()**
Format the buffer

DIAGNOSTICS

Ctrl + **W** › **d**
Show diagnostic in float

TERMINAL

:terminal
Open a shell in a buffer

i / **a**
Enter terminal mode from normal

Ctrl + **** › **Ctrl** + **N**
Back to normal mode

:tnoremap › *<Esc>* › *<C-\><C-n>*
Make Esc leave terminal mode

WINDOWS & BUFFERS

Ctrl + **W** › **s** / **Ctrl** + **W** › **v**
Split horizontally / vertically

Ctrl + **W** › **h** / **j** / **k** / **l**
Move between windows

PLUGINS & HEALTH

:checkhealth
Run all health checks

nvim › **--clean**
Start without config or plugins