

Good and Bad User Interface Design

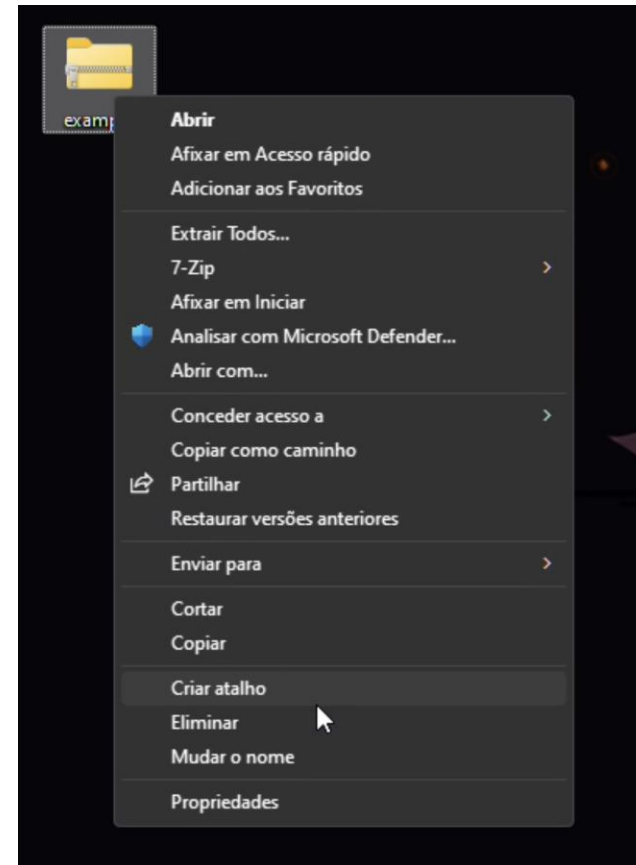
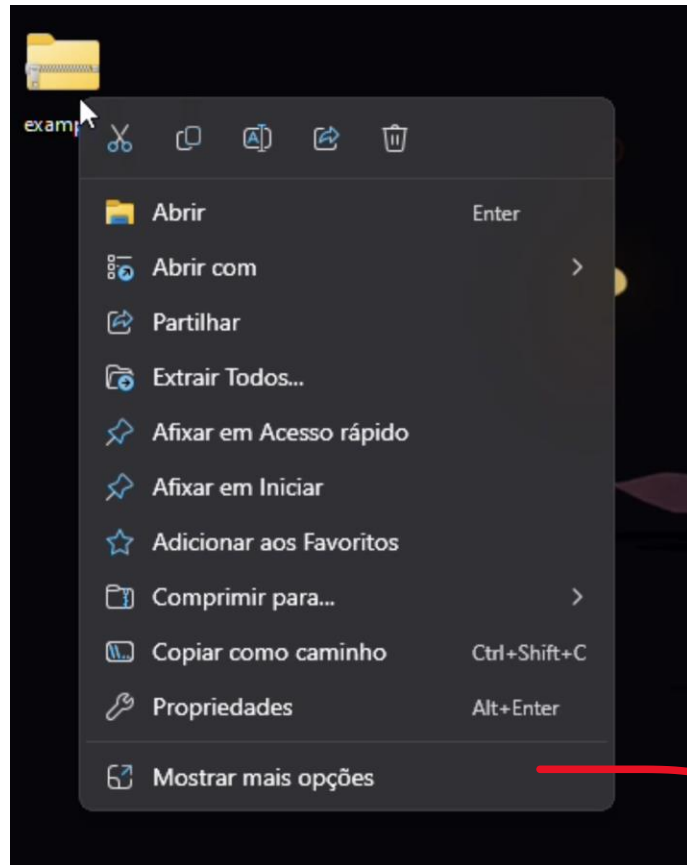
Diogo Pinto 67535

31 de Outubro 2024

Windows 11 Right-Click Context Menu

- The new right-click context menu in Windows 11 is bad for several reasons:
 - **Hidden Options:** The new right-click menu has been streamlined to show fewer options by default. To access the full set of context menu options, users often need to click on "Show more options," which adds an unnecessary step and can slow down workflows, especially for power users.
 - **Reduced Efficiency:** Frequent users of the classic Windows context menu may find the extra click disruptive when trying to perform tasks that were previously more straightforward.
 - **Inconsistent Layout:** Some users have noted that the new design is not consistent with older applications or third-party software, leading to a mix of the new and old context menus that disrupts the user experience.
 - **Learning Curve:** While the new design aims to simplify the interface for casual users, it introduces a learning curve for those accustomed to the older, more detailed menu.
 - **Visual Clarity:** The minimalistic approach may lead to less visual distinction between items, making it harder to quickly identify or select options.

Windows 11 Right-Click Context Menu Example:



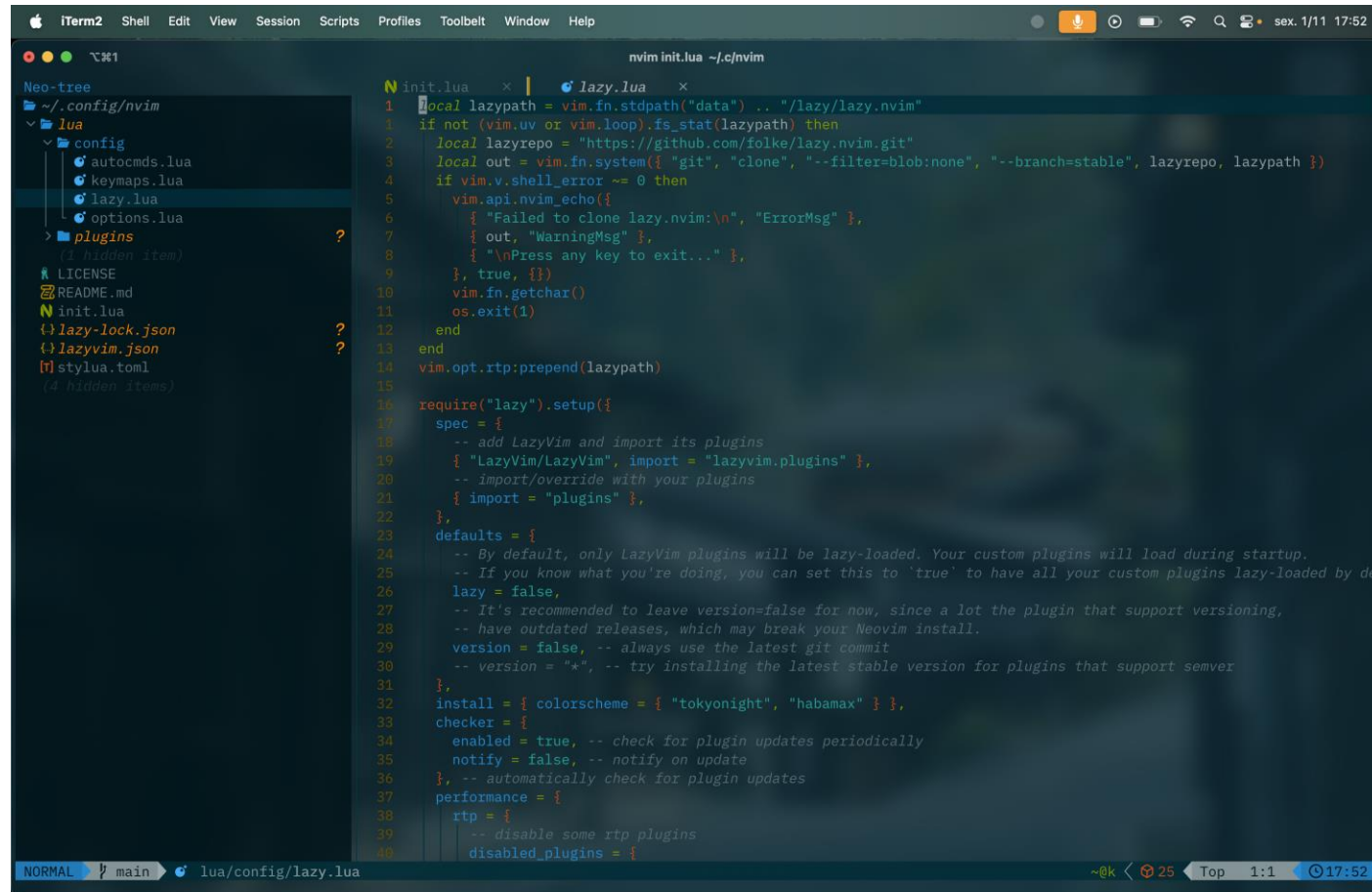
Windows 11 Right-Click Context Menu

- A very quick fix is to just remove the first menu.

Neovim: A Good User Interface for Developers

- Neovim is often praised for its user interface design:
 - **User-Centric Design:** Neovim allows users to fully customize their editing environment through configuration files.
 - **Reduced Clutter:** The interface is clean and minimalistic, which helps reduce distractions.
 - **Efficiency:** Neovim emphasizes keyboard shortcuts, which allows for rapid navigation and editing without the need to rely on a mouse.
 - **User specific:** Although it has a good minimal design it is intended for its target user, for the casual user it might be very confusing.

Neovim: A Good User Interface for Developers



The screenshot shows a Neovim editor window with a dark theme. On the left, a Neo-tree file explorer displays the directory structure of the configuration folder, including files like `autocmds.lua`, `keymaps.lua`, `lazy.lua`, `options.lua`, `plugins`, `LICENSE`, `README.md`, `init.lua`, `lazy-lock.json`, `lazyvim.json`, and `stylua.toml`. The main editor area displays the contents of `init.lua`, which is a Lua script for configuring Neovim. The script includes comments and code for setting up lazy loading of plugins, defining a color scheme, and configuring the plugin checker and performance settings. The status bar at the bottom indicates the current file is `lua/config/lazy.lua` and shows various icons for navigation and search.

```
1 local lazypath = vim.fn.stdpath("data") .. "/lazy/lazy.nvim"
2 if not (vim.uv or vim.loop).fs_stat(lazypath) then
3   local lazyrepo = "https://github.com/folke/lazy.nvim.git"
4   local out = vim.fn.system({ "git", "clone", "--filter=blob:none", "--branch=stable", lazyrepo, lazypath })
5   if vim.v.shell_error ~= 0 then
6     vim.api.nvim_echo({
7       { "Failed to clone lazy.nvim:\n", "ErrorMsg" },
8       { out, "WarningMsg" },
9       { "\nPress any key to exit..." },
10    }, true, {})
11    vim.fn.getchar()
12    os.exit(1)
13  end
14  vim.opt.rtp:prepend(lazypath)
15
16  require("lazy").setup({
17    spec = {
18      -- add LazyVim and import its plugins
19      { "LazyVim/LazyVim", import = "lazyvim.plugins" },
20      -- import/override with your plugins
21      { import = "plugins" },
22    },
23    defaults = {
24      -- By default, only lazyvim plugins will be lazy-loaded. Your custom plugins will load during startup.
25      -- If you know what you're doing, you can set this to 'true' to have all your custom plugins lazy-loaded by default.
26      lazy = false,
27      -- It's recommended to leave version=false for now, since a lot of the plugin that support versioning,
28      -- have outdated releases, which may break your Neovim install.
29      version = false, -- always use the latest git commit
30      -- version = "*", -- try installing the latest stable version for plugins that support semver
31    },
32    install = { colorscheme = { "tokyonight", "habamax" } },
33    checker = {
34      enabled = true, -- check for plugin updates periodically
35      notify = false, -- notify on update
36    }, -- automatically check for plugin updates
37    performance = {
38      rtp = {
39        -- disable some rtp plugins
40        disabled_plugins = {
```