Setting Up Windsurf Editor with Neovim Integration

This guide walks you through the process of setting up Windsurf Editor (a VSCode fork by Codeium) with Neovim integration on Linux. Learn how to install necessary components, configure essential extensions, and customise your development environment.

Prerequisites

- Linux operating system
- Basic familiarity with command-line operations
- Understanding of Vim/Neovim concepts

Installing Neovim

1. Download the latest Neovim AppImage from the official repository:

   Download from https://github.com/neovim/neovim/releases

2. Move the AppImage to your preferred location:

   ```
   mv ~/Downloads/nvim-linux-x86_64.appimage /mnt/hdd2/PortablePrograms/
   ```

3. Make the AppImage executable:

   ```
   chmod u+x /mnt/hdd2/PortablePrograms/nvim-linux-x86_64.appimage
   ```

4. Create a symbolic link in your local bin directory:

   ```
   mkdir -p $HOME/.local/bin
   ln -s /mnt/hdd2/PortablePrograms/nvim-linux-x86_64.appimage
   ~/.local/bin/nvim
   ```

5. Install the Python Neovim package:

   ```
   sudo apt install python3-neovim
   ```

Essential Extensions Setup

Windsurf uses Open-VSX as its marketplace instead of the Microsoft Visual Studio Code Marketplace. Here are the must-have extensions for an optimal development experience:

Development Tools

1. C/C++ Language Support
    - ccls (ccls-project.ccls)
    - clangd (llvm-vs-code-extensions.vscode-clangd)
2. Neovim Integration
    - VSCode Neovim (asvetliakov.vscode-neovim)
3. CMake Support
    - CMake (twxs.cmake)
    - CMake Tools (ms-vscode.cmake-tools)
4. Code Analysis
    - C/C++ Advanced Lint (jbenden.c-cpp-flylint)
    - Install dependency: `pipx install lizard`

Version Control and Comparison

1. Meld Diff (danielroedl.meld-diff)
2. GitLens (eamodio.gitlens)

Language Enhancement

1. Better C++ Syntax (jeff-hykin.better-cpp-syntax)
2. Cpp Reference (Guyutongxue.cpp-reference)
3. Rust Analyzer (rust-lang.rust-analyzer)

UI and Productivity

1. GitHub Theme (GitHub.github-vscode-theme)
2. VSCode Icons (vscode-icons-team.vscode-icons)
3. Snippet IDE (solomonkinard.snippet-ide)

4. Markdown Lint (DavidAnson.vscode-markdownlint)

AI Integration

1. Windsurf Editor is powered by Codeium AI, already. Log in to your Codeium account from the editor.
2. Hugging Face VSCode Extension (HuggingFace.huggingface-vscode)

Neovim Configuration

1. Create a custom directory for your Neovim configuration:

   ```
   mkdir -p ~/.vimdotcommon/
   ```

2. Configure the Neovim executable path in Windsurf:
   - Open Settings
   - Search for `vscode-neovim.neovimExecutablePaths.linux`
   - Set the path to your Neovim AppImage location (e.g., `/mnt/hdd2/PortablePrograms/nvim-linux-x86_64.appimage`)
3. Set the custom init file location:
   - Search for `vscode-neovim.neovimInitVimPaths.linux`
   - Set the path to your custom configuration directory (e.g., `/home/YOURUSERNAME/.vimdotcommon/`)
4. Create the `init.lua` configuration file in your custom directory with the provided configuration (see the full configuration provided at the end of this document)

Key Features of the Neovim Configuration

The provided `init.lua` includes several important features:

- Hybrid line numbering (absolute + relative)
- Smart indentation settings
- Enhanced search functionality
- Custom key mappings
- Automatic cursor position restoration

- Window management optimizations

Usage Tips

1. Use the built-in web browser with `CTRL+SHIFT+p` and type "Simple Browser"
2. The configuration uses `\` as the leader key
3. Quick escape from insert mode using either `jk` or `kj`
4. Many standard VSCode keybindings are preserved to maintain familiarity

Troubleshooting

- If keybindings conflict, prefer Windsurf's native bindings
- Some features like code folding are intentionally disabled to prevent conflicts
- The trailing whitespace removal feature may need manual configuration

---

1. Extension ID: `ccls-project.ccls`

Info:

```
Name: ccls
Id: ccls-project.ccls
Description: C/C++/ObjC language server supporting cross references,
hierarchies, completion and semantic highlight
Version: 0.1.31
Publisher: ccls-project
VS Marketplace Link: https://open-vsx.org/vscode/item?itemName=ccls-
project.ccls
```

2. Extension ID: `llvm-vs-code-extensions.vscode-clangd`

Info:

```
Name: clangd
Id: llvm-vs-code-extensions.vscode-clangd
Description: C/C++ completion, navigation, and insights
Version: 0.1.33
Publisher: llvm-vs-code-extensions
VS Marketplace Link: https://open-vsx.org/vscode/item?itemName=llvm-vs-code-
extensions.vscode-clangd
```

3. Extension ID: `asvetliakov.vscode-neovim`

Info:

```
Name: VSCode Neovim
Id: asvetliakov.vscode-neovim
Description: Vim mode for VSCode, powered by Neovim
Version: 1.18.17
Publisher: asvetliakov
VS Marketplace Link: https://open-vsx.org/vscode/item?
itemName=asvetliakov.vscode-neovim
```

Extension Setting icon -> Type `vscode-neovim.neovimExecutablePaths.linux`

Set the path to the exact location of Neovim.

Example:

```
/mnt/hdd2/PortablePrograms/nvim-linux-x86_64.appimage
```

Set the custom init file for Neovim.

`vscode-neovim.neovimInitVimPaths.linux` (Where Windsurf looks for `init.vim`)

Set the path to your custom Neovim configuration file.

Example:

```
/home/YOURUSERNAME/.vimdotcommon/init.lua
```

4. Extension ID: `twxs.cmake`

Info:

```
Name: CMake
Id: twxs.cmake
Description: CMake language support for Visual Studio Code
Version: 0.0.17
Publisher: twxs
VS Marketplace Link: https://open-vsx.org/vscode/item?itemName=twxs.cmake
```

5. Extension ID: `ms-vscode.cmake-tools`

Info:

```
Name: CMake Tools
Id: ms-vscode.cmake-tools
Description: Extended CMake support in Visual Studio Code
Version: 1.20.51
Publisher: ms-vscode
VS Marketplace Link: https://open-vsx.org/vscode/item?itemName=ms-vscode.cmake-
tools
```

6. Dependency:

```
pipx install lizard
```

Extension ID: `jbenden.c-cpp-flylint`

Info:

```
Name: C/C++ Advanced Lint
Id: jbenden.c-cpp-flylint
Description: An advanced, modern, static analysis extension for C/C++ that
supports a number of back-end analyzer programs.
Version: 1.15.0
Publisher: jbenden
VS Marketplace Link: https://open-vsx.org/vscode/item?itemName=jbenden.c-cpp-
flylint
```

7. Extension ID: `danielroedl.meld-diff`

Info:

```
Name: Meld Diff
Id: danielroedl.meld-diff
Description: Use meld (or other tools like WinMerge, Beyond Compare, ...) to
compare files, folders, clipboard or git changes from visual studio code
directly.
Version: 1.5.1
Publisher: danielroedl
VS Marketplace Link: https://open-vsx.org/vscode/item?
itemName=danielroedl.meld-diff
```

8. Extension ID: `HuggingFace.huggingface-vscode`

Info:

```
Name: llm-vscode
Id: HuggingFace.huggingface-vscode
Description: LLM powered development for VS Code
Version: 0.2.2
Publisher: Hugging Face
VS Marketplace Link: https://open-vsx.org/vscode/item?
itemName=huggingface.huggingface-vscode
```

9. Built-in Web Browser CTRL+SHIFT+p -> Type `Simple Browser`

10. Extension ID: `jeff-hykin.better-cpp-syntax`

Info:

```
Name: Better C++ Syntax
Id: jeff-hykin.better-cpp-syntax
Description: The bleeding edge of the C++ syntax
Version: 1.27.1
Publisher: jeff-hykin
VS Marketplace Link: https://open-vsx.org/vscode/item?itemName=jeff-
hykin.better-cpp-syntax
```

11. Extension ID: `Guyutongxue.cpp-reference`

Info:

```
Name: Cpp Reference
Id: Guyutongxue.cpp-reference
Description: View CppReference.com in VS Code
Version: 0.2.3
Publisher: Guyutongxue
VS Marketplace Link: https://open-vsx.org/vscode/item?itemName=Guyutongxue.cpp-
reference
```

12. Extension ID: `GitHub.github-vscode-theme`

Info:

```
Name: GitHub Theme
Id: GitHub.github-vscode-theme
Description: GitHub theme for VS Code
Version: 6.3.5
Publisher: GitHub
VS Marketplace Link: https://open-vsx.org/vscode/item?itemName=GitHub.github-
vscode-theme
```

13. Extension ID: `eamodio.gitlens`

Info:

```
Name: GitHub Theme
Id: GitHub.github-vscode-theme
Description: GitHub theme for VS Code
Version: 6.3.5
Publisher: GitHub
VS Marketplace Link: https://open-vsx.org/vscode/item?itemName=GitHub.github-
vscode-theme
```

14. Extension ID: `DavidAnson.vscode-markdownlint`

Info:

```
Name: markdownlint
Id: DavidAnson.vscode-markdownlint
Description: Markdown linting and style checking for Visual Studio Code
Version: 0.59.0
Publisher: DavidAnson
VS Marketplace Link: https://open-vsx.org/vscode/item?
itemName=DavidAnson.vscode-markdownlint
```

15. Extension ID: `solomonkinard.snippet-ide`

Info:

```
Name: Snippet IDE
Id: solomonkinard.snippet-ide
Description: 10x productivity boost
Version: 0.0.2
Publisher: solomonkinard
VS Marketplace Link: https://open-vsx.org/vscode/item?
itemName=solomonkinard.snippet-ide
```

16. Extension ID: `vscode-icons-team.vscode-icons`

Info:

```
Name: vscode-icons
Id: vscode-icons-team.vscode-icons
Description: Icons for Visual Studio Code
Version: 12.11.0
Publisher: vscode-icons-team
VS Marketplace Link: https://open-vsx.org/vscode/item?itemName=vscode-icons-
team.vscode-icons
```

17. Extension ID: `rust-lang.rust-analyzer`

Info:

```
Name: rust-analyzer
Id: rust-lang.rust-analyzer
Description: Rust language support for Visual Studio Code
Version: 0.4.2312
Publisher: rust-lang
VS Marketplace Link: https://open-vsx.org/vscode/item?itemName=rust-lang.rust-
analyzer
```

The Neovim config file for Windsurf:

File: `init.lua`

```
-- init.lua

--
================================================================================
-- Neovim Configuration for Windsurf Editor
--
================================================================================
-- This configuration is designed to work with the Neovim plugin for the
Windsurf
-- editor, a fork of VSCode. It aims to provide a seamless integration
experience
-- for users familiar with Vim and Neovim.
--
-- Windsurf Version: 1.3.4
-- Codeium Version: 1.36.1
-- Codeium Commit: ff5014a12e72ceb812f9e7f61876befac66725e5
-- VSCode OSS Version: 1.94.0
-- Commit: ff5014a12e72ceb812f9e7f61876befac66725e5
-- Date: 2025-02-14T20:42:19.835Z
-- Electron: 30.5.1
-- ElectronBuildId: undefined
-- Chromium: 124.0.6367.243
-- Node.js: 20.16.0
-- V8: 12.4.254.20-electron.0
-- OS: Linux x64 6.1.0-31-amd64
--
-- Plugin Information:
-- Name: VSCode Neovim
-- Id: asvetliakov.vscode-neovim
-- Description: Vim mode for VSCode, powered by Neovim
-- Version: 1.18.17
-- Publisher: asvetliakov
-- VS Marketplace Link: https://open-vsx.org/vscode/item?
itemName=asvetliakov.vscode-neovim
--
-- Configuration Details:
-- This configuration is written by Kimi (Kimi AI Assistant,
https://kimi.moonshot.cn)
--- as a reference for
-- users who are integrating Neovim with the Windsurf editor. It is intended to
-- provide a basic setup that can be customized further based on individual
needs.
--
-- Note:
-- This configuration should be loaded from a custom directory by the Neovim
-- plugin of the Windsurf editor. It is not intended to replace the users'
-- existing GVim/Vim/Neovim configuration. Some features may not work properly.
```

```lua
--
================================================================================

-- Set leader key to \
vim.g.mapleader = "\\"
vim.g.maplocalleader = "\\"

-- Disable Vi compatibility mode
vim.opt.compatible = false

-- Enable filetype detection and plugins
vim.cmd([[filetype plugin indent on]])
vim.cmd([[syntax enable]])

-- Line numbers (hybrid mode)
vim.opt.number = true                -- Show absolute line numbers
vim.opt.relativenumber = true        -- Show relative line numbers

-- Automatic number toggle
vim.api.nvim_create_augroup("numbertoggle", { clear = true })
vim.api.nvim_create_autocmd(
    { "BufEnter", "FocusGained", "InsertLeave", "WinEnter" },
    {
        group = "numbertoggle",
        callback = function()
            if vim.wo.number and vim.fn.mode() ~= "i" then
                vim.wo.relativenumber = true
            end
        end,
    }
)
vim.api.nvim_create_autocmd(
    { "BufLeave", "FocusLost", "InsertEnter", "WinLeave" },
    {
        group = "numbertoggle",
        callback = function()
            if vim.wo.number then
                vim.wo.relativenumber = false
            end
        end,
    }
)

-- UI configuration
vim.opt.laststatus = 3               -- Always show status line
vim.opt.colorcolumn = "80"           -- Safe margin at column 80
vim.opt.scrolloff = 8                -- Scroll offset
vim.opt.cursorline = true            -- Highlight current line
```

```lua
vim.opt.cursorcolumn = true          -- Highlight current column

-- Editing settings
vim.opt.tabstop = 2                  -- Tab width
vim.opt.shiftwidth = 2               -- Indentation width
vim.opt.softtabstop = 2              -- Soft tab width
vim.opt.expandtab = true             -- Use spaces instead of tabs
vim.opt.smarttab = true              -- Smart tab behavior
vim.opt.autoindent = true            -- Auto indentation
vim.opt.smartindent = true           -- Smart indentation
vim.opt.cindent = true               -- C-style indentation
vim.opt.preserveindent = true        -- Preserve indentation structure
vim.opt.shiftround = true            -- Round indent to shiftwidth
vim.opt.undolevels = 1000            -- Undo levels
vim.opt.history = 1000               -- Command & search history length

-- Search settings
vim.opt.ignorecase = true            -- Ignore case in search
vim.opt.smartcase = true             -- Case-sensitive if uppercase is present
vim.opt.hlsearch = true              -- Highlight search results
vim.opt.incsearch = true             -- Incremental search

-- Window management
vim.opt.hidden = false               -- Remove buffer when closing
vim.opt.mouse = "a"                  -- Enable mouse support
vim.opt.wrap = false                 -- Disable line wrapping

-- Completion & wildmenu
vim.opt.complete = ".,w,b,u,U,k,t,i,d" -- Tab completion options
vim.opt.wildmenu = true              -- Enable wildmenu
vim.opt.wildmode = "longest:full"    -- Wildmenu behavior

-- Folding
--- Conflicts with Windsurf keybindings. Windsurf manages key combinations.
-- -- -- vim.opt.foldmethod = "syntax"       -- Use syntax-based folding
-- -- -- vim.opt.foldcolumn = "5"            -- Fold column width

-- Key mappings
-- Swap ; and : for command mode
-- https://vi.stackexchange.com/questions/44653/swap-k-and-t-not-working
vim.keymap.set('n', ';', ':', { remap = false, desc = 'Help' })
vim.keymap.set('n', ';', ':', { remap = false, desc = 'Till (backwards)' })

-- jk/kj to escape insert mode
vim.cmd([[imap jk <ESC>]])
vim.cmd([[imap kj <ESC>]])

-- Windows management
```

```lua
vim.keymap.set("n", "<C-F4>", "<cmd>lua CloseFile()<CR>", { noremap = true,
desc = "Close current buffer" })
vim.keymap.set("n", "<C-D>", ":bdelete<CR>", { noremap = true, desc = "Delete
current buffer" })

-- Clipboard mappings (uses system clipboard)
--- Conflicts with Windsurf keybindings. Windsurf manages key combinations.
-- -- -- vim.keymap.set("v", "<C-C>", '"+y', { noremap = true, desc = "Copy" })
-- -- -- vim.keymap.set("v", "<C-X>", '"+x', { noremap = true, desc = "Cut" })
-- -- -- vim.keymap.set("n", "<C-V>", '"+p', { noremap = true, desc = "Paste in
normal mode" })
-- -- -- vim.keymap.set("v", "<C-V>", '"+p', { noremap = true, desc = "Paste in
visual mode" })
-- -- -- vim.keymap.set("i", "<C-V>", '<C-R>+', { noremap = true, desc = "Paste
in insert mode" })

-- Save file
--- Conflicts with Windsurf keybindings. Windsurf manages key combinations.
-- -- -- vim.keymap.set("n", "<C-S>", '<Esc>:w<CR>', { noremap = true, desc =
"Save file" })
vim.keymap.set("i", "<C-S>", '<C-O>:w<CR>', { noremap = true, desc = "Save file
in insert mode" })

-- Undo and redo mappings
--- Conflicts with Windsurf keybindings. Windsurf manages key combinations.
-- -- -- vim.keymap.set("n", "<C-Z>", "u", { noremap = true, desc = "Undo" })
-- -- -- vim.keymap.set("i", "<C-Z>", "<C-O>u", { noremap = true, desc = "Undo
in insert mode" })
-- -- -- vim.keymap.set("n", "<C-Y>", "u", { noremap = true, desc = "Redo" })

-- Code folding
--- Conflicts with Windsurf keybindings. Windsurf manages key combinations.
-- -- -- vim.keymap.set("n", "<F8>", "za", { noremap = true, desc = "Toggle
fold" })
-- -- -- vim.keymap.set("n", "<S-F8>", "zR", { noremap = true, desc = "Open all
folds" })
-- -- -- vim.keymap.set("n", "<Space>", "za", { noremap = true, desc = "Toggle
fold under cursor" })

-- Custom functions
function CloseFile()
  if vim.opt.previewwindow:get() then
    vim.cmd("close")
  elseif vim.fn.winnr("$") > 1 then
    vim.cmd("bdelete")
  else
    print("Cannot delete last buffer")
  end
```

```
end

-- Restore cursor position
vim.api.nvim_create_autocmd("BufReadPost", {
  callback = function()
    local last_pos = vim.fn.line("'\"")
    if last_pos > 1 and last_pos <= vim.fn.line("$") then
      vim.cmd("normal " .. last_pos .. "G")
    end
  end,
})

-- GUI window position
vim.api.nvim_create_autocmd("GUIEnter", {
  callback = function()
    vim.cmd("winpos 0 0")
  end,
})

-- Strip trailing whitespace on save
--- Does not work.
vim.api.nvim_create_autocmd("BufWritePre", {
  callback = function()
    vim.cmd([[%s/\s\+$//e]])
    vim.cmd([[%s/^\t* //e]])
  end,
})
```

Conclusion

This setup provides a powerful development environment that combines the efficiency of Neovim with the modern features of Windsurf Editor. The configuration can be further customised based on your specific needs and preferences.

Remember to check the official documentation for both Windsurf and Neovim for more advanced configurations and features.