novafacing / configuration.nix

Last active 2 years ago • Report abuse

Code   Revisions 7

configuration.nix

```nix
# sh <(curl -L https://nixos.org/nix/install)
# nix-build https://github.com/LnL7/nix-darwin/archive/master.tar.gz -A installer
# ./result/bin/darwin-installer
# Go through installer, put this file where you want it
# darwin-rebuild switch -I darwin-config=${HOME}/wherever/you/put/it/configuration.nix
# There will probably be errors :)

{ config, pkgs, lib, ... }:
let
  username = "novafacing";
  name = "chaos";
  fenix = import
    (fetchTarball "https://github.com/nix-community/fenix/archive/main.tar.gz")
    { };
  fisa-code = pkgs.callPackage ({ lib, stdenvNoCC, fetchFromGitHub }:
    stdenvNoCC.mkDerivation rec {
      pname = "fisa-code";
      version = "1.0";
      src = fetchFromGitHub {
        owner = "sainnhe";
        repo = "icursive-nerd-font";
        rev = "623feb6815753c5679ef4111fb137b8dae4fb983";
        hash = "sha256-NJjyOsDCQ+QmCMlP6ZwBOBdEcKqRQdevUIilWr21snU=";
      };

      installPhase = ''
        runHook preInstall
        install -m444 -Dt $out/share/fonts/truetype dist/*/*.ttf
        runHook postInstall
      '';

      meta = with lib; {
        homepage = "https://github.com/sainnhe/icursive-nerd-font";
        description = "Patched Nerd Fonts";
        longDescription = "Patched Nerd Fonts";
        platforms = platforms.all;
      };
    }) { };
  sold = pkgs.callPackage ({ lib, stdenv, fetchFromGitHub, cmake, mimalloc
    , ninja, openssl, zlib, testers }:

    stdenv.mkDerivation rec {
      name = "mold";
      pname = "mold";

      src = fetchFromGitHub {
        owner = "bluewhalesystems";
        repo = "sold";
        rev = "ab4245a0919c16775c9d497dfa3f03330bda733b";
        hash = "sha256-hAxxnXy7OodIu5rSGlZImktd8ucNLauDeSgGsbaOIps=";
      };

      nativeBuildInputs = [ cmake ninja ];

      buildInputs = [ openssl zlib ]
        ++ lib.optionals (!stdenv.isDarwin) [ mimalloc ];

      postPatch = ''
        sed -i CMakeLists.txt -e '/.*set(DEST\ .*/d'
      '';

      cmakeFlags = [ "-DMOLD_USE_SYSTEM_MIMALLOC:BOOL=ON" ];

      env.NIX_CFLAGS_COMPILE =
        toString (lib.optionals stdenv.isDarwin [ "-faligned-allocation" ]);

      meta = with lib; {
        description = "A faster drop-in replacement for existing Unix linkers";
        longDescription = ''
          mold is a faster drop-in replacement for existing Unix linkers. It is
          several times faster than the LLVM lld linker. mold is designed to
          increase developer productivity by reducing build time, especially in
          rapid debug-edit-rebuild cycles.
        '';
        homepage = "https://github.com/rui314/mold";
        changelog = "https://github.com/rui314/mold/releases/tag/v${version}";
        license = licenses.agpl3Plus;
        maintainers = with maintainers; [ azahi nitsky ];
        platforms = platforms.unix;
      };
    }) { };
  stateVersion = 4;
in let
  home-packages = [
    fenix.default.toolchain
    fenix.rust-analyzer-vscode-extension
    fenix.rust-analyzer
    sold
    pkgs.vscode
    pkgs.kitty
    pkgs.python311Full
    pkgs.llvmPackages_16.clang
    pkgs.llvmPackages_16.compiler-rt
    pkgs.llvmPackages_16.libclang
    pkgs.llvmPackages_16.libcxx
    pkgs.llvmPackages_16.libcxxabi
    pkgs.llvmPackages_16.libllvm
    pkgs.llvmPackages_16.libunwind
    pkgs.llvmPackages_16.lld
    pkgs.llvmPackages_16.lldb
    pkgs.llvmPackages_16.openmp
    pkgs.fd
    pkgs.silver-searcher
    pkgs.nil
    pkgs.nixfmt
  ];
in {
  imports = [ <home-manager/nix-darwin> ];
  documentation = {
    enable = false;
    doc = { enable = false; };
    info = { enable = false; };
    man = { enable = false; };
  };
  environment = {
    darwinConfig = "$HOME/config/configuration.nix";
    systemPackages = [
      pkgs.darwin.apple_sdk.frameworks.System
      pkgs.darwin.apple_sdk.frameworks.CoreFoundation
      pkgs.darwin.apple_sdk.frameworks.CoreServices
    ];
    systemPath = [ ];
    shellAliases = { };
    variables = { };
  };
  fonts = {
    fontDir = { enable = true; };
    fonts = [ pkgs.font-awesome fisa-code pkgs.nerdfonts ];
  };
  users = {
    users = {
      ${username} = {
        name = "${username}";
        home = "/Users/${username}";
      };
    };
  };
  home-manager = {
    useGlobalPkgs = true;
    users = {
      ${username} = { lib, pkgs, ... }: {
        manual = { manpages = { enable = false; }; };
        programs = {
          zsh = {
            enable = true;
            enableAutosuggestions = true;
            enableCompletion = true;
            enableSyntaxHighlighting = true;
            enableVteIntegration = true;
            envExtra = ''
                                          [[ -o login ]] && export PATH='/usr/local/bin
                                    if [ -e '/nix/var/nix/profiles/etc/profile.d/nix-da
                                        . '/nix/var/nix/profiles/etc/profile.d/nix-daem
                                    fi
                                    [[ -f "$HOME/.cargo/env" ]] && source "$HOME/.cargo
                                    export NIX_PATH="''${NIX_PATH}:darwin-config=/Users
                                          export PATH="''${HOME}/install/bin/:''${PATH}
                                          export PATH="''${HOME}/.nix-profile/bin/:''${
            '';
            history = {
              ignoreDups = true;
              extended = true;
              save = 1000000;
              share = true;
              size = 1000000;
            };
            initExtra = "";
            oh-my-zsh = {
              enable = true;
              extraConfig = "";
              plugins = [ "git" "python" "rust" "extract" "sudo" ];
            };
            plugins = [{
              name = "z";
              src = pkgs.fetchFromGitHub {
                owner = "rupa";
                repo = "z";
                rev = "master";
                sha256 = "sha256-4jMHh1GVRdFNjUjiPH94vewbfLcah7Agu153zjVNE14=";
              };
            }];
          };
          atuin = {
            enable = true;
            enableZshIntegration = true;
            flags = [ ];
            settings = {
              db_path = "~/.history.db";
              key_path = "~/.atuin-key";
              session_path = "~/.atuin-key";
              dialect = "us";
              auto_sync = false;
              sync_frequency = "5m";
              sync_address = "https://api.atuin.sh";
              search_mode = "fuzzy";
              inline_height = 20;
              style = "compact";
              show_preview = true;
              exit_mode = "return-query";
              history_filter = [ "chpasswd" ];
            };
          };
          bat = { enable = true; };
          bottom = { enable = true; };
          command-not-found = {

          };
          feh = { enable = true; };
          gh = {
            enable = true;
            settings = { git_protocol = "ssh"; };
          };
          git = {
            enable = true;
            aliases = {
              status = "status --sort --branch";
              grog =
                "log --graph --abbrev-commit --decorate --all --format=format:'%C(bold

            };
            delta = {
              enable = true;
              options = { };
            };
            extraConfig = {
              init = { defaultBranch = "main"; };

              "filter \"lfs\"" = {
                required = true;
                clean = "git-lfs clean -- %f";
                smudge = "git-lfs smudge -- %f";
                process = "git-lfs filter-process";
              };
            };
            ignores = [ ".DS_STORE" ];
            userName = "novafacing";
            userEmail = "rowanbhart@gmail.com";
          };
          go = {
            enable = true;

          };
          gpg = { enable = true; };
          jq = { enable = true; };
          kitty = {
            enable = true;
            darwinLaunchOptions = [ ];
            environment = { };
            extraConfig = "";
            font = {
              name = "Fisa Code";
              size = 12.0;
            };
            settings = {
              disable_ligatures = "tab cursor";
              scrollback_lines = 100000;
              enable_audio_bell = "no";
              term = "xterm";
              allow_remote_control = "yes";
              # Colorscheme
              background = "#32302f";
              foreground = "#d4be98";
              cursor = "#89b482";
              selection_foreground = "#32302f";
              selection_background = "#d4be98";
              color0 = "#32302f";
              color8 = "#d4be98";
              # red
              color1 = "#ea6962";
              # light red
              color9 = "#fb4934";
              # green
              color2 = "#a9b665";
              # light green
              color10 = "#6c782e";
              # yellow
              color3 = "#d8a657";
              # light yellow
              color11 = "#e78a4e";
              # blue
              color4 = "#7daea3";
              # light blue
              color12 = "#45707a";
              # magenta
              color5 = "#d3869b";
              # light magenta
              color13 = "#945e80";
              # cyan
              color6 = "#89b482";
              # lighy cyan
              color14 = "#4c7a5d";
              # light gray
              color7 = "#928474";
              # dark gray
              color15 = "#665c54";
            };
            shellIntegration = { enableZshIntegration = true; };
          };
          lsd = {
            enable = true;
            enableAliases = true;
            settings = {
              date = "relative";
              ignore-globs = [ ".git" ];
            };
          };
          neovim = {
            enable = true;
            defaultEditor = true;
            extraConfig = "";
            plugins = with pkgs.vimPlugins;
              [

              ];
            viAlias = true;
            vimAlias = true;
            vimdiffAlias = true;
            withNodeJs = true;
            withPython3 = true;
          };
          nix-index = {
            enable = true;
            enableZshIntegration = true;
          };
          pandoc = { enable = true; };
          ssh = {
            enable = true;
            compression = true;
            controlMaster = "auto";
            matchBlocks = {
              grobot = {
                hostname = "192.168.0.2";
                user = "grobot";

              };
            };
          };
          starship = {
            enable = true;
            enableZshIntegration = true;
            settings = {

              "$schema" = "https://starship.rs/config-schema.json";
              format = ''
                [╭─](fg:#5a524c)$directory$git_branch$git_status$git_metrics$time$userr
                [╰─ϕ ](fg:#5a524c)'';

              command_timeout = 500;

              add_newline = false;

              python = { format = "[ 🐍$virtualenv $version](bg:#282828)"; };

              git_metrics = {
                format =
                  "([+$added](fg:#89b482 bg:#7c6f64)[-$deleted](fg:#ea6962 bg:#7c6f64))
                disabled = false;
              };

              git_branch = {
                format =
                  "[ on $symbol$branch(:$remote_branch)](fg:#ddc7a1 bg:#7c6f64)";
              };
              git_status = {
                format = "[ $all_status$ahead_behind ](fg:#ddc7a1 bg:#7c6f64)";
                ahead = "⇡\${count}";
                diverged = "⇕⇡\${ahead_count}⇣\${behind_count}";
                behind = "⇣\${count}";
              };

              directory = {
                format =
                  "[◀](fg:#89b482)[ $path ](fg:#282828 bg:#89b482)[▶](bg:#7c6f64 fg:#89
              };
              fill = {
                symbol = "·";
                style = "fg:#46413e";
              };

              status = {
                format =
                  "[ ](bg:#282828)[$status ](bg:#282828)[❯](fg:#89b482 bg:#282828)[▶](f
                disabled = false;
                success_symbol = "✓";
              };

              hostname = {
                format = "[@$hostname](bg:#504945)[▶](fg:#504945 bg:#282828)";
                ssh_only = false;
              };

              username = {
                show_always = true;
                format = "[$user](bg:#504945)";
                disabled = false;
              };

              time = {
                format =
                  "[ $time ](bg:#7c6f64 fg:#45403d)[▶](fg:#7c6f64 bg:#504945)";
                disabled = false;
              };
            };
          };
          vscode = {
            enable = true;
            enableExtensionUpdateCheck = false;
            enableUpdateCheck = false;
            package = pkgs.vscode;
            extensions = [
              pkgs.vscode-extensions.bierner.markdown-mermaid
              pkgs.vscode-extensions.christian-kohler.path-intellisense
              pkgs.vscode-extensions.dotjoshjohnson.xml
              pkgs.vscode-extensions.eamodio.gitlens
              # pkgs.vscode-extensions.edwinkofler.vscode-assorted-languages
              pkgs.vscode-extensions.esbenp.prettier-vscode
              # pkgs.vscode-extensions.exodiusstudios.comment-anchors
              pkgs.vscode-extensions.formulahendry.auto-rename-tag
              # pkgs.vscode-extensions.github.copilot
              pkgs.vscode-extensions.github.vscode-pull-request-github
              # pkgs.vscode-extensions.inferrinizzard.prettier-sql-vscode
              pkgs.vscode-extensions.james-yu.latex-workshop
              # pkgs.vscode-extensions.janisdd.vscode-edit-csv
              pkgs.vscode-extensions.jebbs.plantuml
              # pkgs.vscode-extensions.karunamurti.tera
              pkgs.vscode-extensions.llvm-vs-code-extensions.vscode-clangd
              # pkgs.vscode-extensions.maelvalais.autoconf
              # pkgs.vscode-extensions.mesonbuild.mesonbuild
              pkgs.vscode-extensions.ms-azuretools.vscode-docker
              pkgs.vscode-extensions.ms-python.python
              pkgs.vscode-extensions.ms-python.vscode-pylance
              pkgs.vscode-extensions.ms-toolsai.jupyter
              pkgs.vscode-extensions.ms-toolsai.jupyter-keymap
              pkgs.vscode-extensions.ms-toolsai.jupyter-renderers
              pkgs.vscode-extensions.ms-toolsai.vscode-jupyter-cell-tags
              pkgs.vscode-extensions.ms-toolsai.vscode-jupyter-slideshow
              # pkgs.vscode-extensions.ms-vscode-remote.remote-containers
              pkgs.vscode-extensions.ms-vscode-remote.remote-ssh
              # pkgs.vscode-extensions.ms-vscode-remote.remote-ssh-edit
              # pkgs.vscode-extensions.ms-vscode-remote.remote-wsl
              # pkgs.vscode-extensions.ms-vscode-remote.vscode-remote-extensionpack
              pkgs.vscode-extensions.ms-vscode.cmake-tools
              # pkgs.vscode-extensions.ms-vscode.cpptools
              pkgs.vscode-extensions.ms-vscode.hexeditor
              # pkgs.vscode-extensions.ms-vscode.remote-explorer
              # pkgs.vscode-extensions.ms-vscode.remote-server
              pkgs.vscode-extensions.naumovs.color-highlight
              # pkgs.vscode-extensions.npclaudiu.vscode-gn
              # pkgs.vscode-extensions.perkovec.emoji
              pkgs.vscode-extensions.pkief.material-icon-theme
              # pkgs.vscode-extensions.pnp.polacode
              pkgs.vscode-extensions.redhat.java
              pkgs.vscode-extensions.redhat.vscode-yaml
              # pkgs.vscode-extensions.richie5um2.vscode-sort-json
              pkgs.vscode-extensions.rust-lang.rust-analyzer
              # pkgs.vscode-extensions.sainnhe.gruvbox-material
              pkgs.vscode-extensions.serayuzgur.crates
              pkgs.vscode-extensions.tamasfe.even-better-toml
              pkgs.vscode-extensions.twxs.cmake
              # pkgs.vscode-extensions.vadimcn.vscode-lldb
              # pkgs.vscode-extensions.vgalaktionov.moonscript
              pkgs.vscode-extensions.vscodevim.vim
              pkgs.vscode-extensions.yzhang.markdown-all-in-one
            ];
            keybindings = [
              {
                key = "ctrl+l";
                command = "workbench.action.focusRightGroup";
              }
              {
                key = "ctrl+k ctrl+right";
                command = "-workbench.action.focusRightGroup";
              }
              {
                key = "ctrl+h";
                command = "workbench.action.focusLeftGroup";
              }
              {
                key = "ctrl+k ctrl+left";
                command = "-workbench.action.focusLeftGroup";
              }
              {
                key = "ctrl+shift+\\";
                command = "-workbench.action.terminal.focusTabs";
                when =
                  "terminalFocus && terminalHasBeenCreated || terminalFocus && terminal
              }
              {
                key = "ctrl+k";
                command = "workbench.action.focusActiveEditorGroup";
              }
              {
                key = "ctrl+o";
                command = "workbench.action.files.openFolder";
                when = "true";
              }
              {
                key = "ctrl+j";
                command = "-workbench.action.togglePanel";
              }
              {
                key = "ctrl+j";
                command = "-extension.vim_ctrl+j";
                when =
                  "editorTextFocus && vim.active && vim.use<C-j> && !inDebugRepl";
              }
              {
                key = "ctrl+j";
                command = "workbench.action.terminal.focus";
              }
              {
                key = "ctrl+shift+k";
                command = "workbench.action.files.showOpenedFileInNewWindow";
                when = "emptyWorkspaceSupport";
              }
              {
                key = "ctrl+k o";
                command = "-workbench.action.files.showOpenedFileInNewWindow";
                when = "emptyWorkspaceSupport";
              }
              {
                key = "ctrl+l";
                command = "workbench.action.terminal.focusNextPane";
                when =
                  "terminalFocus && terminalHasBeenCreated || terminalFocus && terminal
              }
              {
                key = "alt+down";
                command = "-workbench.action.terminal.focusNextPane";
                when =
                  "terminalFocus && terminalHasBeenCreated || terminalFocus && terminal
              }
              {
                key = "ctrl+h";
                command = "workbench.action.terminal.focusPreviousPane";
                when =
                  "terminalFocus && terminalHasBeenCreated || terminalFocus && terminal
              }
              {
                key = "alt+left";
                command = "-workbench.action.terminal.focusPreviousPane";
                when =
                  "terminalFocus && terminalHasBeenCreated || terminalFocus && terminal
              }
              {
                key = "ctrl+o";
                command = "-workbench.action.files.openFile";
                when = "true";
              }
            ];
            userSettings = {
              "[c]" = {
                "editor.defaultFormatter" =
                  "llvm-vs-code-extensions.vscode-clangd";
              };
              "[cpp]" = {
                "editor.defaultFormatter" =
                  "llvm-vs-code-extensions.vscode-clangd";
              };
              "[html]" = {
                "editor.defaultFormatter" = "vscode.html-language-features";
              };
              "[java]" = { "editor.defaultFormatter" = "redhat.java"; };
              "[javascript]" = {
                "editor.defaultFormatter" = "esbenp.prettier-vscode";
              };
              "[json]" = {
                "editor.defaultFormatter" = "vscode.json-language-features";
              };
              "[jsonc]" = {
                "editor.defaultFormatter" = "esbenp.prettier-vscode";
              };
              "[markdown]" = {
                "editor.defaultFormatter" = "yzhang.markdown-all-in-one";
              };
              "[nix]" = { "editor.defaultFormatter" = "jnoortheen.nix-ide"; };
              "[plaintext]" = {
                "editor.unicodeHighlight.ambiguousCharacters" = false;
                "editor.unicodeHighlight.invisibleCharacters" = false;
              };
              "[rust]" = {
                "editor.defaultFormatter" = "rust-lang.rust-analyzer";
              };
              "[sql]" = { "editor.defaultFormatter" = "mtxr.sqltools"; };
              "[toml]" = {
                "editor.defaultFormatter" = "tamasfe.even-better-toml";
              };
              "[typescript]" = {
                "editor.defaultFormatter" = "esbenp.prettier-vscode";
              };
              "C_Cpp.clang_format_fallbackStyle" =
                "{Language =        Cpp, BasedOnStyle =  LLVM, AccessModifierOffset = -
              "C_Cpp.clang_format_sortIncludes" = true;
              "C_Cpp.default.cppStandard" = "c++20";
              "C_Cpp.default.cStandard" = "c17";
              "C_Cpp.dimInactiveRegions" = false;
              "C_Cpp.formatting" = "clangFormat";
              "C_Cpp.inactiveRegionBackgroundColor" = "#3c3836";
              "C_Cpp.intelliSenseEngine" = "disabled";
              "clangd.arguments" = [ "--enable-config" ];
              "clangd.checkUpdates" = true;
              "clangd.path" = "/usr/bin/clangd";
              "cmake.configureOnOpen" = true;
              "color-highlight.languages" = [ "*" ];
              "color-highlight.markerType" = "dot-after";
              "debug.internalConsoleOptions" = "neverOpen";
              "debug.onTaskErrors" = "debugAnyway";
              "editor.acceptSuggestionOnEnter" = "off";
              "editor.autoClosingQuotes" = "never";
              "editor.bracketPairColorization.enabled" = true;
              "editor.cursorBlinking" = "smooth";
              "editor.cursorStyle" = "block";
              "editor.defaultFormatter" = "ms-python.python";
              "editor.fontFamily" = "Fisa Code";
              "editor.fontLigatures" = true;
              "editor.fontSize" = 16;
              "editor.formatOnPaste" = true;
              "editor.formatOnSave" = true;
              "editor.guides.bracketPairs" = true;
              "editor.guides.highlightActiveBracketPair" = true;
              "editor.guides.highlightActiveIndentation" = "always";
              "editor.inlineSuggest.enabled" = true;
              "editor.renderWhitespace" = "all";
              "editor.rulers" = [ 88 ];
              "editor.stickyScroll.enabled" = true;
              "editor.suggestSelection" = "first";
              "editor.tokenColorCustomizations" = {
                "textMateRules" = [
                  {
                    "name" = "comment";
                    "scope" = [ "comment" ];
                    "settings" = { "fontStyle" = "italic"; };
                  }
                  {
                    "name" = "Keyword Storage";
                    "scope" = [ "keyword" "keyword.control" "storage" ];
                    "settings" = { "fontStyle" = "italic"; };
                  }
                ];
              };
              "explorer.confirmDelete" = false;
              "explorer.confirmDragAndDrop" = false;
              "explorer.fileNesting.enabled" = true;
              "explorer.fileNesting.expand" = false;
              "extensions.ignoreRecommendations" = true;
              "files.exclude" = {
                "**/.classpath" = true;
                "**/.factorypath" = true;
                "**/.project" = true;
                "**/.settings" = true;
              };
              "files.maxMemoryForLargeFilesMB" = 8192;
              "files.refactoring.autoSave" = true;
              "github.copilot.enable" = {
                "*" = true;
                "markdown" = false;
                "plaintext" = false;
                "yaml" = false;
              };
              "githubPullRequests.createOnPublishBranch" = "never";
              "githubPullRequests.remotes" = [ "origin" ];
              "gruvboxMaterial.colorfulSyntax" = true;
              "gruvboxMaterial.italicKeywords" = true;
              "gruvboxMaterial.lightPalette" = "material";
              "gruvboxMaterial.lightContrast" = "soft";
              "gruvboxMaterial.lightSelection" = "yellow";
              "gruvboxMaterial.darkPalette" = "material";
              "gruvboxMaterial.darkContrast" = "soft";
              "gruvboxMaterial.darkCursor" = "yellow";
              "gruvboxMaterial.darkSelection" = "green";
              "java.codeGeneration.generateComments" = true;
              "java.completion.guessMethodArguments" = true;
              "java.jdt.ls.java.home" = "/usr/lib/jvm/java-17-openjdk-amd64";
              "java.jdt.ls.vmargs" = ''
                -XX:+UseParallelGC -XX:GCTimeRatio=4 -XX:AdaptiveSizePolicyWeight=90 -D
              "javascript.updateImportsOnFileMove.enabled" = "always";
              "keyboard.dispatch" = "keyCode";
              "latex-workshop.latex.recipe.default" = "lastUsed";
              "latex-workshop.latex.recipes" = [
                {
                  "name" = "lualatex ➞ bibtex ➞ lualatex×2";
                  "tools" = [ "lualatex" "bibtex" "lualatex" "lualatex" ];
                }
                {
                  "name" = "Just pdflatex";
                  "tools" = [ "pdflatex" ];
                }
                {
                  "name" = "Just lualatex";
                  "tools" = [ "lualatex" ];
                }
                {
                  "name" = "latexmk 🔃";
                  "tools" = [ "latexmk" ];
                }
                {
                  "name" = "latexmk (latexmkrc)";
                  "tools" = [ "latexmk_rconly" ];
                }
                {
                  "name" = "latexmk (lualatex)";
                  "tools" = [ "lualatexmk" ];
                }
                {
                  "name" = "pdflatex ➞ bibtex ➞ pdflatex×2";
                  "tools" = [ "pdflatex" "bibtex" "pdflatex" "pdflatex" ];
                }
                {
                  "name" = "Compile Rnw files";
                  "tools" = [ "rnw2tex" "latexmk" ];
                }
                {
                  "name" = "Compile Jnw files";
                  "tools" = [ "jnw2tex" "latexmk" ];
                }
                {
                  "name" = "tectonic";
                  "tools" = [ "tectonic" ];
                }
              ];
              "latex-workshop.latex.tools" = [
                {
                  "args" = [
                    "-synctex=1"
                    "-shell-escape"
                    "-interaction=nonstopmode"
                    "-file-line-error"
                    "%DOCFILE%"
                  ];
                  "command" = "lualatex";
                  "name" = "lualatex";
                }
                {
                  "args" = [
                    "-synctex=1"
                    "-shell-escape"
                    "-interaction=nonstopmode"
                    "-file-line-error"
                    "%DOCFILE%"
                  ];
                  "command" = "pdflatex";
                  "name" = "pdflatex";
                }
                {
                  "args" = [
                    "-synctex=1"
                    "-interaction=nonstopmode"
                    "-file-line-error"
                    "%DOCFILE%"
                  ];
                  "command" = "xelatex";
                  "name" = "xelatex";
                }
                {
                  "args" = [ "%DOCFILE%" ];
                  "command" = "bibtex";
                  "name" = "bibtex";
                }
              ];
              "latex-workshop.view.pdf.invertMode.enabled" = "always";
              "nix.serverPath" = "nil";
              "nix.formatterPath" = "nixfmt";
              "notebook.cellToolbarLocation" = {
                "default" = "right";
                "jupyter-notebook" = "left";
              };
              "prettier.documentSelectors" = [ "*.js" "*.ts" "*.html" "*.css" ];
              "prettier.printWidth" = 88;
              "prettier.tabWidth" = 4;
              "prettier.trailingComma" = "all";
              "prettier.vueIndentScriptAndStyle" = true;
              "python.defaultInterpreterPath" = "python3";
              "python.formatting.provider" = "black";
              "python.languageServer" = "Pylance";
              "python.linting.mypyEnabled" = true;
              "python.linting.pylintEnabled" = true;
              "redhat.telemetry.enabled" = false;
              "rust-analyzer.check.command" = "clippy";
              "rust-analyzer.cargo.extraEnv" = {
                "CARGO_PROFILE_RUST_ANALYZER_INHERITS" = "dev";
              };
              "rust-analyzer.cargo.extraArgs" = [ "--profile" "rust-analyzer" ];
              "security.workspace.trust.untrustedFiles" = "open";
              "settingsSync.ignoredSettings" =
                [ "-python.formatting.blackPath" ];
              "sqltools.completionLanguages" = [ ];
              "sqltools.highlightQuery" = false;
              "sqltools.useNodeRuntime" = true;
              "telemetry.telemetryLevel" = "off";
              "terminal.external.linuxExec" = "kitty";
              "terminal.integrated.cursorBlinking" = true;
              "terminal.integrated.customGlyphs" = false;
              "terminal.integrated.defaultProfile.linux" = "zsh";
              "terminal.integrated.detectLocale" = "on";
              "terminal.integrated.drawBoldTextInBrightColors" = false;
              "terminal.integrated.enableMultiLinePasteWarning" = false;
              "terminal.integrated.fontFamily" = "Fisa Code";
              "terminal.integrated.fontSize" = 16;
              "terminal.integrated.gpuAcceleration" = "on";
              "terminal.integrated.inheritEnv" = true;
              "terminal.integrated.scrollback" = 100000;
              "terminal.integrated.shellIntegration.enabled" = true;
              "typescript.updateImportsOnFileMove.enabled" = "always";
              "vim.easymotion" = true;
              "vim.easymotionDimBackground" = true;
              "vim.easymotionDimColor" = "#7c6f64";
              "vim.easymotionMarkerBackgroundColor" = "#45403d";
              "vim.easymotionMarkerFontWeight" = "bold";
              "vim.easymotionMarkerForegroundColorOneChar" = "#d3869b";
              "vim.easymotionMarkerForegroundColorTwoCharFirst" = "#d8a657";
              "vim.easymotionMarkerForegroundColorTwoCharSecond" = "#e78a4e";
              "vim.hlsearch" = true;
              "vim.normalModeKeyBindingsNonRecursive" = [
                {
                  "after" = [ "<leader>" "<leader>" "<leader>" "b" "d" "w" ];
                  "before" = [ "t" ];
                }
                {
                  "before" = [ "g" "n" ];
                  "after" = [ ];
                  "commands" = [{ "command" = "editor.action.marker.next"; }];
                }
```

```
          ];
          "vim.smartRelativeLine" = true;
          "vim.statusBarColorControl" = false;
          "vim.statusBarColors.commandlineinprogress" =
            [ "#32302f" "#89b482" ];
          "vim.statusBarColors.normal" = [ "#32302f" "#d8a657" ];
          "vim.statusBarColors.insert" = [ "#32302f" "#d3869b" ];
          "vim.statusBarColors.replace" = [ "#32302f" "#e78a4e" ];
          "vim.statusBarColors.searchinprogressmode" =
            [ "#32302f" "#89b482" ];
          "vim.statusBarColors.visual" = [ "#32302f" "#7daea3" ];
          "vim.statusBarColors.visualblock" = [ "#32302f" "#7daea3" ];
          "vim.statusBarColors.visualline" = [ "#32302f" "#7daea3" ];
          "vim.statusBarColors.easymotioninputmode" =
            [ "#32302f" "#e78a4e" ];
          "vim.statusBarColors.easymotionmode" = [ "#32302f" "#e78a4e" ];
          "vim.textwidth" = 88;
          "window.menuBarVisibility" = "toggle";
          "window.restoreWindows" = "none";
          "window.zoomLevel" = -2;
          "workbench.colorCustomizations" = {
            "editorBracketHighlight.foreground1" = "#ea6962";
            "editorBracketHighlight.foreground2" = "#d3869b";
            "editorBracketHighlight.foreground3" = "#e78a4e";
            "editorBracketHighlight.foreground4" = "#a9b665";
            "editorBracketHighlight.foreground5" = "#bd6f3e";
            "editorBracketHighlight.foreground6" = "#89b482";
            "editorStickyScroll.background" = "#3c3836";
            "editorStickyScrollHover.background" = "#46413e";
            "statusBar.background" = "#32302f";
            "statusBar.debuggingBackground" = "#32302f";
            "statusBar.debuggingForeground" = "#d8a657";
            "statusBar.foreground" = "#d8a657";
            "statusBar.noFolderBackground" = "#32302f";
          };
          "workbench.colorTheme" = "Gruvbox Material Dark";
          "workbench.editorAssociations" = {
            "*.ipynb" = "jupyter-notebook";
          };
          "workbench.editor.tabSizing" = "fixed";
          "workbench.iconTheme" = "material-icon-theme";
          "workbench.sideBar.location" = "right";
          "workbench.startupEditor" = "none";
          "color-highlight.enable" = true;
          "color-highlight.matchWords" = true;
          "editor.minimap.enabled" = false;
        };
      };
    };

    home = {
      file = {
            cargo-config-toml = {
              target = ".cargo/config.toml";
              enable = true;
              text = ''
                [target.aarch64-apple-darwin]
                linker = "clang"
                rustflags = [
                    "-C",
                    "link-arg=--ld-path=${sold}/bin/ld64.mold",
                    #"-L",
                    #"${pkgs.darwin.apple_sdk.frameworks.CoreFoundation}/Library/Framew
                    #"-L",
                    #"${pkgs.darwin.apple_sdk.frameworks.System}/Library/Frameworks/",
                    #"-L",
                    #"${pkgs.darwin.apple_sdk.frameworks.CoreFoundation}/lib/swift",
                    #"-L",
                    #"${pkgs.darwin.apple_sdk.frameworks.System}/lib/swift/",
                ]
              '';
            };
          };
          stateVersion = "23.05";
          packages = home-packages;
          activation = {
            unaliasApplications = lib.mkIf pkgs.stdenv.hostPlatform.isDarwin
              (lib.hm.dag.entryBefore [ "checkLinkTargets" ] ''
                echo "Un-Linking Home Manager applications..." 2>&1
                app_path="$HOME/Applications/Home Manager Apps"
                rm -rf "$app_path"
              '');
            aliasApplications = lib.mkIf pkgs.stdenv.hostPlatform.isDarwin (let
              apps = pkgs.buildEnv {
                name = "home-manager-applications";
                paths = home-packages;
                pathsToLink = "/Applications";
              };
            in lib.hm.dag.entryAfter [ "linkGeneration" ] ''
              echo "Linking Home Manager applications..." 2>&1
              app_path="$HOME/Applications/Home Manager Apps"
              tmp_path="$(mktemp -dt "home-manager-applications.XXXXXXXXXX")" || exit 1
              ${pkgs.fd}/bin/fd \
                  -t l -d 1 . ${apps}/Applications \
                  -x $DRY_RUN_CMD /Users/${username}/install/bin/mkalias -L {} "$tmp_pa
              $DRY_RUN_CMD rm -rf "$app_path"
              $DRY_RUN_CMD mv "$tmp_path" "$app_path"
            '');
          };
        };
        services = { };
      };
    };
  };
  networking = {
    computerName = "${name}";
    hostName = "${name}";
  };
  nix = {
    nixPath = [
      { darwin-config = "$HOME/config/configuration.nix"; }
      "/nix/var/nix/profiles/per-user/root/channels"
    ];
    extraOptions = ''
      auto-optimise-store = true
      experimental-features = nix-command flakes
      system = aarch64-darwin
      extra-platforms = x86_64-darwin aarch64-darwin
    '';
  };
  nixpkgs = {
    config = {
      allowUnfree = true;
      allowUnsupportedSystem = false;
    };
    overlays = [ ];
  };
  programs = { };
  services = {
    nix-daemon = {
      enable = true;
      logFile = "/var/log/nix-daemon.log";
    };
    skhd = {
      enable = true;
      skhdConfig = "";
    };
    /* spacebar = {
           enable = true;
           package = pkgs.spacebar;
           config = {
               position                   = "top";
               display                    = "main";
               height                     = 34;
               title                      = "off";
               spaces                     = "off";
               clock                      = "on";
               power                      = "on";
               padding_left               = 20;
               padding_right              = 20;
               spacing_left               = 25;
               spacing_right              = 15;
               text_font                  = ''"Fisa Code:Regular:12.0"'';
               icon_font                  = ''"Font Awesome 6 Free:Solid:12.0"'';
               background_color           = "0xff252423";
               foreground_color           = "0xffd4be98";
               power_icon_color           = "0xffa9b665";
               battery_icon_color         = "0xffa9b665";
               dnd_icon_color             = "0xffe78a4e";
               clock_icon_color           = "0xff7daea3";
               power_icon_strip           = "□ □";
               spaces_for_all_displays    = "on";
               display_separator          = "on";
               display_separator_icon     = "□";
               clock_icon                 = "□";
               dnd_icon                   = "□";
               clock_format               = ''"%d/%m/%y %R"'';
               right_shell                = "on";
               right_shell_icon           = "□";
               right_shell_command        = "whoami";
           };
           extraConfig = "";
       };
    */
    yabai = {
      enable = true;
      config = { };
      extraConfig = "";
    };
  };
  # Nix-darwin does not link installed applications to the user environment. This means
  # in spotlight; and when launched through the dock they come with a terminal window.
  # Upstream issue = https://github.com/LnL7/nix-darwin/issues/214
  system = {
    activationScripts = {
      applications = {
        text = lib.mkForce ''
          echo "setting up ~/Applications..." >&2
          applications="$HOME/Applications"
          nix_apps="$applications/Nix Apps"

          # Needs to be writable by the user so that home-manager can symlink into it
          if ! test -d "$applications"; then
              mkdir -p "$applications"
              chown ${username} = "$applications"
              chmod u+w "$applications"
          fi

          # Delete the directory to remove old links
          rm -rf "$nix_apps"
          mkdir -p "$nix_apps"
          find ${config.system.build.applications}/Applications -maxdepth 1 -type l -ex
              while read src; do
                  # Spotlight does not recognize symlinks; it will ignore directory we
                  # It does understand MacOS aliases though; a unique filesystem featur
                  # from bash (as far as I know); so we use the oh-so-great Apple Scrip
                  /usr/bin/osascript -e "
                      set fileToAlias to POSIX file \"$src\"
                      set applicationsFolder to POSIX file \"$nix_apps\"
                      tell application \"Finder\"
                          make alias file to fileToAlias at applicationsFolder
                          # This renames the alias; 'mpv.app alias' -> 'mpv.app'
                          set name of result to \"$(rev <<< "$src" | cut -d'/' -f1 | re
                      end tell
                  " 1>/dev/null
              done
        '';
      };
    };
    stateVersion = stateVersion;
    defaults = {
      NSGlobalDomain = {
        AppleEnableMouseSwipeNavigateWithScrolls = false;
        AppleEnableSwipeNavigateWithScrolls = false;
        AppleICUForce24HourTime = false;
        AppleInterfaceStyle = "Dark";
        AppleInterfaceStyleSwitchesAutomatically = false;
        AppleScrollerPagingBehavior = true;
        AppleShowAllExtensions = true;
        AppleShowAllFiles = true;
        AppleShowScrollBars = "Automatic";
        AppleWindowTabbingMode = "always";
        NSAutomaticCapitalizationEnabled = false;
        NSAutomaticDashSubstitutionEnabled = false;
        NSAutomaticPeriodSubstitutionEnabled = false;
        NSAutomaticQuoteSubstitutionEnabled = false;
        NSAutomaticSpellingCorrectionEnabled = false;
        NSAutomaticWindowAnimationsEnabled = false;
        NSDocumentSaveNewDocumentsToCloud = false;
        NSNavPanelExpandedStateForSaveMode = true;
        NSScrollAnimationEnabled = true;
        NSTableViewDefaultSizeMode = 1;
        NSTextShowsControlCharacters = true;
        NSUseAnimatedFocusRing = false;
        NSWindowResizeTime = null;
        PMPrintingExpandedStateForPrint = false;
        _HIHideMenuBar = false;
        "com.apple.keyboard.fnState" = false;
        "com.apple.mouse.tapBehavior" = null;
        "com.apple.sound.beep.feedback" = 1;
        "com.apple.springing.delay" = null;
        "com.apple.springing.enabled" = true;
        "com.apple.swipescrolldirection" = true;
        "com.apple.trackpad.enableSecondaryClick" = true;
        "com.apple.trackpad.scaling" = 1.0;
        "com.apple.trackpad.trackpadCornerClickBehavior" = 1;
      };
    };
    keyboard = {
      enableKeyMapping = true;
      remapCapsLockToEscape = true;
    };
  };
}
```