ERIC BAILEY

DOTFILES

Contents

Machines

NiXPS

NixOS configuration

The NiXPS configuration is an expression that takes, at least, `config`, `lib`, and `pkgs`.

⟨*machines/nixps/configuration.nix* 1⟩≡

```
{ config, lib, pkgs, ... }:
let
  username = "yurrriq";
in
{
```

   ⟨*Set the location to MSP* 2⟩
   ⟨*Blacklist a few kernel modules* 3⟩
   ⟨*Configure the LUKS device* 4⟩
   ⟨*Configure the environment* 5⟩
   ⟨*Configure the file systems* 6⟩
   ⟨*Tweak hardware config* 7⟩
   ⟨*Configure Bluetooth* 8⟩
   ⟨*Configure PulseAudio and OpenGL* 9⟩
   ⟨*Configure home-manager* 10⟩
   ⟨*Configure networking* 11⟩
   ⟨*Configure Nix* 12⟩
   ⟨*Configure sudo* 13⟩
   ⟨*Configure X Server* 14⟩
   ⟨*Configure sops-nix* 15⟩
   ⟨*Configure user* 16⟩

```
}
```

Configure the *module*.   FIXME

⟨*Set the location to MSP* 2⟩≡   (1)

```
airportCode = "MSP";
```

⟨*Blacklist a few kernel modules* 3⟩≡   (1)

```
boot.blacklistedKernelModules = [ "nouveau" "nvidia" "psmouse" ];
```

Configure the *LUKS*[1] devices.   1

⟨*Configure the LUKS device* 4⟩≡   (1)

```
boot.initrd.luks.devices.root.device = "/dev/nvme0n1p2";
```

⟨*Configure the environment* 5⟩≡   (1)
```
environment.homeBinInPath = true;
environment.pathsToLink = [
  "/lib/aspell"
  "/share/emacs/site-lisp"
  "/share/fish"
];
```

⟨*Configure the file systems* 6⟩≡   (1)
```
fileSystems."/" = {
  device = "/dev/disk/by-uuid/024a1168-9949-4cb2-bbd1-4b19a9d49ef2";
  fsType = "ext4";
};

fileSystems."/boot" = {
  device = "/dev/disk/by-uuid/7574-B246";
  fsType = "vfat";
};

fileSystems."/var/lib/docker/plugins" = {
  device = "/var/lib/docker/plugins";
  fsType = "none";
  options = [ "bind" ];
};

fileSystems."/var/lib/docker/overlay2" = {
  device = "/var/lib/docker/overlay2";
  fsType = "none";
  options = [ "bind" ];
};
```

NixOS/nixpkgs#24711[2]

> Figure out wtf is going on here

⟨*Tweak hardware config* 7⟩≡   (1)
```
hardware.bumblebee.enable = false;
hardware.nvidiaOptimus.disable = false;
```

[2] https://github.com/NixOS/nixpkgs/issues/24711

```
hardware.bumblebee.enable = true;
hardware.bumblebee.connectDisplay = true;
hardware.bumblebee.pmMethod = "bbswitch";
hardware.opengl.driSupport32Bit = true;
hardware.nvidia = {
  modesetting.enable = true;
  optimus_prime = {
    enable = true;
    nvidiaBusId = "PCI:1:0:0";
    intelBusId = "PCI:0:2:0";
  };
};
```

⟨*Configure Bluetooth* 8⟩≡   (1)
```
  services.blueman.enable = true;
  hardware.bluetooth = {
    enable = true;
    config = {
      General = {
        Enable = "Source,Sink,Media,Socket";
      };
    };
  };
```

Tweak PulseAudio and OpenGL settings for Steam.

⟨*Configure PulseAudio and OpenGL* 9⟩≡   (1)
```
  hardware.pulseaudio.support32Bit = true;
  hardware.opengl.driSupport32Bit = true;
```

⟨*Configure home-manager* 10⟩≡   (1)
```
  home-manager.useUserPackages = true;
  home-manager.users."${username}" = args:
    import ./home.nix (args // { inherit pkgs; });
```

⟨*Configure networking* 11⟩≡   (1)
```
  networking.firewall = {
    enable = true;
    allowedTCPPortRanges = [
      # { from = 8000; to = 8000; }
    ];
  };

  networking.hostName = "nixps";
```

⟨*Configure Nix* 12⟩≡   (1)
```
  nix = {
    buildCores = 8;
    trustedUsers = [ "root" username ];
  };
```

⟨*Configure sudo* 13⟩≡   (1)
```
  security.sudo = {
    enable = true;
    extraConfig = "
      ${username} ALL=(ALL) NOPASSWD: ALL
    ";
  };
```

⟨*Configure X Server* 14⟩≡   (1)

```
services.xserver = {
  displayManager.autoLogin = {
    enable = true;
    user = username;
  };
  dpi = 180;
};
```

adopt sops-nix

⟨*Configure sops-nix* 15⟩≡   (1)

```
# FIXME
# sops = {
#   gnupgHome = "/home/yurrriq/.gnupg";
#   secrets."${username}.hashedPassword" = {
#     format = "binary";
#     sopsFile = ./. + "/secrets/${username}.hashedPassword.enc";
#   };
#   sshKeyPaths = [];
# };
```

⟨*Configure user* 16⟩≡   (1)

```
users.mutableUsers = false;
users.users."${username}" = {
  name = username;
  hashedPassword = lib.fileContents (./. + "/secrets/${username}.hashedPassword");
  # FIXME: hashedPassword = lib.fileContents config.sops.secrets."${username}.hashedPassword".path;
  group = "users";
  extraGroups = [
    "audio"
    "disk"
    "docker"
    "http"
    "networkmanager"
    "systemd-journal"
    "video"
    "wheel"
  ];
  createHome = true;
  uid = 1000;
  home = "/home/${username}";
  shell = "/etc/profiles/per-user/${username}/bin/fish";
};
```

home-manager configuration

⟨*machines/nixps/home.nix* 17⟩≡

```
{ lib, pkgs, ... }:
{
  ⟨Import some modules 18⟩
  ⟨Configure account 19⟩
  ⟨Add some more user packages 20⟩
  _module.args.pkgs = lib.mkForce pkgs;
}
```

⟨*Import some modules* 18⟩≡   (17)
```
  imports = [
    ../../config/bash.nix
    ../../config/bat.nix
    ../../config/browserpass.nix
    ../../config/direnv.nix
    ../../config/dunst.nix
    ../../config/emacs
    ../../config/firefox.nix
    ../../config/fish
    ../../config/fzf.nix
    ../../config/git
    ../../config/gpg.nix
    ../../config/htop.nix
    ../../config/jq.nix
    ../../config/kitty.nix
    ../../config/man.nix
    ../../config/nixpkgs
    ../../config/password-store.nix
    ../../config/rebar3.nix
    ../../config/starship.nix
    ../../config/taskwarrior
    ../../config/xmonad
  ];
```

⟨*Configure account* 19⟩≡   (17)
```
  accounts.email.accounts.primary = {
    address = "eric@ericb.me";
    gpg.key = "F88372B24A806FF23BCB3A4E2DDDF8606958B3F9";
    primary = true;
    realName = "Eric Bailey";
  };
```

⟨*Add some more user packages* 20⟩≡   (17)
```
  home.packages = with pkgs; [
    appimage-run
    calibre
    fd
    frescobaldi
    musescore
    openscad
    powertop
    reaper
    steam
    tellico
    zoom-us
  ];
```

SRUXPS

NixOS configuration

The SRUXPS configuration is an expression that takes, at least,
`config`, `lib`, and `pkgs`.

⟨*machines/sruxps/configuration.nix* 21⟩≡

```
{ config, lib, pkgs, ... }:
let
  username = "e.bailey";
in
{
  ⟨Set the location to MSP 22⟩
  ⟨Configure the LUKS devices 23⟩
  ⟨Configure vm.swappiness 24⟩
  ⟨Add the coretemp kernel module 25⟩
  ⟨Link some extra paths 26⟩
  ⟨Configure the file systems 27⟩
  ⟨Configure Bluetooth 29⟩
  ⟨Configure home-manager 30⟩
  ⟨Configure networking 31⟩
  ⟨Configure Nix 32⟩
  ⟨Configure sudo 33⟩
  ⟨Configure services 34⟩
  ⟨Configure user 35⟩
  virtualisation.docker.enable = true;
}
```

> Describe the location module

⟨*Set the location to MSP* 22⟩≡   (21)

```
  airportCode = "MSP";
```

Configure the *LUKS* devices.

⟨*Configure the LUKS devices* 23⟩≡   (21)

```
  boot.initrd.luks.devices = {
    cryptkey.device = "/dev/disk/by-uuid/603b64c6-8544-4b43-9b6a-7d8a08091514";

    cryptroot = {
      device = "/dev/disk/by-uuid/c4e26a93-c0c9-4680-bb79-8c0ca47df96c";
      keyFile = "/dev/mapper/cryptkey";
    };

    cryptswap = {
      device = "/dev/disk/by-uuid/565c0358-110e-4279-ba59-619cb2cc1ebf";
      keyFile = "/dev/mapper/cryptkey";
    };
  };
```

https://docs.oracle.com/en/operating-systems/oracle-linux/
7/admin/section_h4v_3yt_tr.html

⟨*Configure vm.swappiness* 24⟩≡   (21)

```
  boot.kernel.sysctl = {
    "vm.swappiness" = 1;
  };
```

⟨*Add the coretemp kernel module* 25⟩≡   (21)
```
boot.kernelModules = [
  ”coretemp”
];
```

⟨*Link some extra paths* 26⟩≡   (21)
```
environment.homeBinInPath = true;
environment.pathsToLink = [
  ”/lib/aspell”
  ”/share/emacs/site-lisp”
  ”/share/fish”
  # FIXME: ”/share/icons”
];
```

> btrfs clean boot?

⟨*Configure the file systems* 27⟩≡   (21) 28 ▷
```
fileSystems.”/” = {
  device = ”/dev/disk/by-uuid/71cfcb8a-13a9-49f0-b034-a9c8841be07b”;
  fsType = ”btrfs”;
  options = [ ”subvol=@” ”rw” ”noatime” ”compress=zstd” ”ssd” ”space_cache” ];
};

fileSystems.”/home” = {
  device = ”/dev/disk/by-uuid/71cfcb8a-13a9-49f0-b034-a9c8841be07b”;
  fsType = ”btrfs”;
  options = [ ”subvol=@home” ”rw” ”noatime” ”compress=zstd” ”ssd” ”space_cache” ];
};

fileSystems.”/nix” = {
  device = ”/dev/disk/by-uuid/71cfcb8a-13a9-49f0-b034-a9c8841be07b”;
  fsType = ”btrfs”;
  options = [ ”subvol=@nix” ”rw” ”noatime” ”compress=zstd” ”ssd” ”space_cache” ];
};

fileSystems.”/var” = {
  device = ”/dev/disk/by-uuid/71cfcb8a-13a9-49f0-b034-a9c8841be07b”;
  fsType = ”btrfs”;
  options = [ ”subvol=@var” ”rw” ”noatime” ”compress=zstd” ”ssd” ”space_cache” ];
};

fileSystems.”/boot” = {
  device = ”/dev/disk/by-uuid/54D0-C859”;
  fsType = ”vfat”;
};
```

⟨*Configure the file systems* 27⟩+≡   (21) ◁27
```
fileSystems.”/mnt/music” = {
  device = ”192.168.1.147:/volume1/homes/eric/music”;
  fsType = ”nfs”;
  options = [ ”noatime” ”noauto” ”rw” ”x-systemd.automount” ];
};
```

⟨*Configure Bluetooth* 29⟩≡   (21)
```
hardware.bluetooth = {
  enable = true;
  config = {
    General = {
      Enable = "Source,Sink,Media,Socket";
    };
  };
};
```

⟨*Configure home-manager* 30⟩≡   (21)
```
home-manager.useUserPackages = true;
home-manager.users."${username}" = args:
  import ./home.nix (args // { inherit pkgs; });
```

⟨*Configure networking* 31⟩≡   (21)
```
networking.hostName = "MSP-EBAILEY01";

networking.interfaces.wlp1s0.useDHCP = true;
networking.useDHCP = false; # NOTE: Deprecated, so set it false.
```

⟨*Configure Nix* 32⟩≡   (21)
```
nix.trustedUsers = [ "root" username ];
```

⟨*Configure sudo* 33⟩≡   (21)
```
security.sudo = {
  enable = true;
  extraConfig = "
    ${username} ALL=(ALL) NOPASSWD: ALL
  ";
};
```

⟨*Configure services* 34⟩≡   (21)
```
services.blueman.enable = true;

services.fwupd.enable = true;

services.lorri.enable = false;

services.tlp.enable = true;

services.xserver.displayManager.autoLogin = {
  enable = true;
  user = username;
};

services.xserver.monitorSection = "
  DisplaySize 406 228
";
```

⟨*Configure user* 35⟩≡   (21)
```
  users.mutableUsers = false;
  users.users."${username}" = {
    name = username;
    hashedPassword = lib.fileContents (./. + "/secrets/${username}.hashedPassword");
    group = "users";
    extraGroups = [
      "audio"
      "disk"
      "docker"
      "http"
      "networkmanager"
      "systemd-journal"
      "video"
      "wheel"
    ];
    createHome = true;
    uid = 1001;
    home = "/home/${username}";
    shell = "/etc/profiles/per-user/${username}/bin/fish";
  };
```

home-manager configuration

⟨*machines/sruxps/home.nix* 36⟩≡
```
  { lib, pkgs, ... }:
  {
```

　　⟨*Import various program configs* 37⟩

　　⟨*Configure the primary email account* 38⟩

　　⟨*Install some user packages* 39⟩

```
    _module.args.pkgs = lib.mkForce pkgs;
```

```
  }
```

⟨*Import various program configs* 37⟩≡   (36)
```
  imports = [
    ../../config/bash.nix
    ../../config/bat.nix
    ../../config/browserpass.nix
    ../../config/bugwarrior.nix
    ../../config/direnv.nix
    ../../config/dunst.nix
    ../../config/emacs
    ../../config/firefox.nix
    ../../config/fish
    ../../config/fzf.nix
    ../../config/git
    ../../config/gpg.nix
    ../../config/htop.nix
    ../../config/jq.nix
    ../../config/kitty.nix
    ../../config/man.nix
    ../../config/nixpkgs
    ../../config/password-store.nix
    ../../config/rebar3.nix
    ../../config/starship.nix
    ../../config/taskwarrior
    ../../config/xmonad
  ];
```

Configure Vim

⟨*Configure the primary email account* 38⟩≡   (36)
```
  accounts.email.accounts.primary = {
    address = "e.bailey@sportradar.com";
    gpg.key = "EFD6F1EDC84D2FA935E38570462054AB8B682702";
    primary = true;
    realName = "Eric Bailey";
  };
```

⟨*Install some user packages* 39⟩≡   (36) 40 ▷
```
  home.packages = with pkgs; (
    [
      aws-iam-authenticator
      ec2instanceconnectcli
```

⟨*Install some user packages* 39⟩+≡   (36) ◁39 42 ▷
```
      progress
```

Clementine[3] is a great music player.

I haven't used Dhall in a while, but it's still interesting.

⟨ * 41⟩≡   46 ▷
```
      dhall
      dhall-json
```

I might need Docker Compose[4] again someday, but not lately.

⟨*Install some user packages* 39⟩+≡   (36) ◁40 43 ▷
```
      docker-compose
```

iw[5] is useful for troubleshooting wireless issues, like in Trondheim.

⟨*Install some user packages* 39⟩+≡   (36) ◁42 44 ▷
```
      iw
      jdk
```

[3] https://www.clementine-player.org/

[4] https://docs.docker.com/compose/

[5] https://wireless.wiki.kernel.org/en/users/documentation/iw

```
https://github.com/Azure/kubelogin
```

⟨*Install some user packages* 39⟩+≡   (36) ◁43 45▷
```
    kubectx
    kubelogin
    k9s
    stern
```

naal[6] is my tool for non-interactive AWS Azure login.

[6] `https://github.com/yurrriq/naal`

⟨*Install some user packages* 39⟩+≡   (36) ◁44 47▷
```
    # FIXME: naal
```

It would be cool to play with Next[7] some lazy day.

[7] `https://next.atlas.engineer/`

⟨* 41⟩+≡   ◁41
```
    next
```

PulseAudio Volume Control[8] is useful for configuring Bluetooth audio.

[8] `https://freedesktop.org/software/pulseaudio/pavucontrol/`

⟨*Install some user packages* 39⟩+≡   (36) ◁45 48▷
```
    renderizer
    scc
```

Try to improve battery life.

⟨*Install some user packages* 39⟩+≡   (36) ◁47 49▷
```
    powertop
```

⟨*Install some user packages* 39⟩+≡   (36) ◁48
```
    zoom-us
  ] ++ (
  with nodePackages; [
    aws-azure-login
  ]
  )
);
```

Modules

Common

This module is a lambda that takes at least `pkgs` in its arguments.

⟨*modules/common.nix* 50⟩≡   51 ▷
```
{ pkgs, ... }:

{
```

Configure garbage collection (`gc`) and my personal Cachix[9]-based   [9] https://cachix.org
`binaryCaches`.

- https://yurrriq.cachix.org

- https://yurrriq-nur-packages.cachix.org

⟨*modules/common.nix* 50⟩+≡   ◁50 52▷
```
  nix = {
    binaryCaches = [
      "https://yurrriq.cachix.org"
      "https://yurrriq-nur-packages.cachix.org"
    ];
    binaryCachePublicKeys = [
      "yurrriq.cachix.org-1:evpJ5wKluf7QOCcv69VkIxCOtHgubrqXlZpp3JAXLBE="
      "yurrriq-nur-packages.cachix.org-1:7kbjuGBUZcWf876g2cdelmIQXrXzOhpMVBqYOyyAv70="
    ];
    # TODO: daemonNiceLevel = 19;
    extraOptions = "
      experimental-features = flakes nix-command
    ";
    gc = {
      automatic = true;
      options = "--delete-older-than 30d";
    };
    optimise.automatic = true;
    package = pkgs.nixUnstable;
  };
```

Install the Nerd Fonts[10] patched Iosevka[11].

[10] https://www.nerdfonts.com

[11] http://be5invis.github.io/Iosevka

⟨*modules/common.nix* 50⟩+≡   ◁51 53▷

```
fonts = {
  enableFontDir = true;
  fonts = with pkgs; [
    (nerdfonts.override { fonts = [ "Iosevka" ]; })
  ];
};
```

Enable system-wide Bash completion.

⟨*modules/common.nix* 50⟩+≡   ◁52 54▷

```
programs.bash.enableCompletion = true;
```

⟨*modules/common.nix* 50⟩+≡   ◁53

```
}
```

Packages

⟨*modules/packages.nix* 55⟩≡   56▷

```
{ lib, pkgs, ... }:

{
```

⟨*modules/packages.nix* 55⟩+≡   ◁55 57▷

```
environment.systemPackages = with pkgs; (
  [
    aspell
    aspellDicts.en
    awscli
    bind
    cachix
    clac
    clementine
    coreutils
    # TODO: cquery
    curl
    expect
    gawk
    gcc
    gnumake
    gnused
    gnutar
    gzip
    httpie
    # TODO: http-prompt
    # TODO: idris
    lorri
    moreutils
    neofetch
    nixUnstable
    noweb
    pandoc
    prettyping
    ripgrep
    shellcheck
    shfmt
    silver-searcher # TODO: find helm-rg solution
    slack
    spotify
    stow
```

⟨*modules/packages.nix* 55⟩+≡   ◁56 58▷
```
      (
        texlive.combine {
          inherit (texlive) scheme-small
            datetime
            dirtytalk
            fmtcount
            framed
            fvextra
            hardwrap
            ifplatform
            latexmk
            minted
            titlesec
            todonotes
            tufte-latex
            xetex
            xstring
            ;
          inherit noweb;
        }
      )
```

⟨*modules/packages.nix* 55⟩+≡   ◁57 59▷
```
      tree
      unzip
      vim
      watch
      # wipe
      yq
    ]
```

⟨*modules/packages.nix* 55⟩+≡   ◁58 60▷
```
    ) ++ (
    with python3Packages; [
      pygments
    ]
```

⟨*modules/packages.nix* 55⟩+≡   ◁59 61▷
```
    ) ++ lib.optionals stdenv.isDarwin (
    [
      m-cli
      sourcetree
      onyx
    ] ++ (
    with chunkwm; [
        core
        ffm
        tiling
      ]
    )
```

⟨*modules/packages.nix* 55⟩+≡   ◁60 62▷

```
  ) ++ lib.optionals stdenv.isLinux [
    dolphin
    libnotify
    libreoffice
    lm_sensors
    mtr
    networkmanager-openconnect
    nfs-utils
    openconnect
    paper-icon-theme
    pavucontrol
    playerctl
    # psmisc
    # pv
    qpdfview
    wirelesstools
```

Install Tomb[12] and related packages.   [12] https://www.dyne.org/software/
                                                              tomb/
⟨*modules/packages.nix* 55⟩+≡   ◁61 63▷

```
    file
    lsof
    tomb
```

⟨*modules/packages.nix* 55⟩+≡   ◁62 64▷

```
    (signal-desktop.override { spellcheckerLanguage = "en_US"; })
```

⟨*modules/packages.nix* 55⟩+≡   ◁63 65▷

```
    thunderbird
    xclip
    xorg.xbacklight
  ];
```

⟨*modules/packages.nix* 55⟩+≡   ◁64

```
  }
```

Location

⟨*modules/location.nix* 66⟩≡   67▷

```
  { config, lib, ... }:
```

⟨*modules/location.nix* 66⟩+≡   ◁66 68▷

```
  let
    airportCode = config.airportCode;
  in
  {
```

⟨*modules/location.nix* 66⟩+≡   ◁67 69▷

```
    options.airportCode = lib.mkOption {
      default = "MSP";
      type = lib.types.enum [ "ATL" /* "LHR" */ "LJU" "MSP" "TRD" ];
    };
```

⟨*modules/location.nix* 66⟩+≡   ◁68 70▷
```
config = lib.mkMerge [
```

The Sportradar office in Minneapolis.

⟨*modules/location.nix* 66⟩+≡   ◁69 71▷
```
  (
    lib.mkIf (airportCode == "MSP") {
      location = {
        latitude = 44.98;
        longitude = -93.27;
      };
      time.timeZone = "America/Chicago";
    }
  )
```

Cook Out on Ponce de Leon Ave.

⟨*modules/location.nix* 66⟩+≡   ◁70 72▷
```
  (
    lib.mkIf (airportCode == "ATL") {
      location = {
        latitude = 33.77;
        longitude = -84.37;
      };
      time.timeZone = "America/New_York";
    }
  )
```

The Sportradar office in London.

⟨*modules/location.nix* 66⟩+≡   ◁71 73▷
```
  (
    lib.mkIf (airportCode == "LHR") {
      location = {
        latitude = 51.51;
        longitude = -0.09;
      };
      time.timeZone = "Europe/London";
    }
  )
```

The Sportradar office in Ljubljana.

⟨*modules/location.nix* 66⟩+≡   ◁72 74▷
```
  (
    lib.mkIf (airportCode == "LJU") {
      location = {
        latitude = 46.09;
        longitude = 14.55;
      };
      time.timeZone = "Europe/Ljubljana";
    }
  )
```

The Sportradar office in Trondheim.

⟨*modules/location.nix* 66⟩+≡   ◁73 75▷

```
    (
      lib.mkIf (airportCode == "TRD") {
        location = {
          latitude = 63.43;
          longitude = 10.40;
        };
        time.timeZone = "Europe/Oslo";
      }
    )
```

⟨*modules/location.nix* 66⟩+≡   ◁74

```
    ];

  }
```

NixOS

⟨*modules/nixos.nix* 76⟩≡

```
  { config, lib, pkgs, ... }:

  {
```
    ⟨*Configure boot loader* 77⟩
    ⟨*Configure fonts, keyboard, and locale* 78⟩
    ⟨*Configure location* 79⟩
    ⟨*Enable networkmanager* 80⟩
    ⟨*Allow unfree packages* 81⟩
    ⟨*Configure services* 82⟩
    ⟨*Configure audio* 84⟩
```
    system.stateVersion = "20.09";
```
    ⟨*Configure virtualisation* 85⟩
```
  }
```

⟨*Configure boot loader* 77⟩≡   (76)

```
  boot = {
    cleanTmpDir = true;
    loader = {
      systemd-boot.enable = true;
      efi.canTouchEfiVariables = true;
    };
  };
```

⟨*Configure fonts, keyboard, and locale* 78⟩≡   (76)

```
  console.font = "Lat2-Terminus16";
  console.keyMap = "us";
  i18n.defaultLocale = "en_US.UTF-8";
```

> Link/document this

⟨*Configure location* 79⟩≡   (76)

```
  location.provider = "manual";
```

⟨*Enable networkmanager* 80⟩≡   (76)

```
  networking.networkmanager.enable = true;
```

⟨*Allow unfree packages* 81⟩≡   (76)

```
  nixpkgs.config.allowUnfree = true;
```

⟨*Configure services* 82⟩≡   (76)
```
  services = {
    fstrim.enable = true;

    logind.lidSwitch = "hibernate";

    kbfs.enable = true;

    redshift = {
      enable = true;
      temperature.night = 2300;
    };

    unclutter = {
      enable = false; # FIXME
      extraOptions = [ "exclude-root" "ignore-scrolling" ];
      threshold = 1;
      timeout = 1;
    };
```
   ⟨*Configure Xserver* 83⟩
```
  };
```

⟨*Configure Xserver* 83⟩≡   (82)

```
xserver = {
  enable = true;

  autorun = true;

  displayManager = {
    lightdm.enable = true;
    session = [
      {
        name = "home-manager";
        manage = "window";
        start = "
          ${pkgs.runtimeShell} $HOME/.hm-xsession &
          waitPID=$!
        ";
      }
    ];
  };

  layout = "us";

  libinput = {
    enable = true;
    accelSpeed = "1.0";
    disableWhileTyping = true;
    naturalScrolling = false;
    tapping = true;
  };

  videoDrivers = [
    "intel"
  ];

  xkbOptions = "ctrl:nocaps,compose:ralt";
};
```

⟨*Configure audio* 84⟩≡   (76)

```
hardware.pulseaudio = {
  enable = true;
  extraModules = [ pkgs.pulseaudio-modules-bt ];
  package = pkgs.pulseaudioFull;
};
sound.enable = true;
```

> Document podman vs dockerd

⟨*Configure virtualisation* 85⟩≡   (76)

> Fix rootless podman

```
virtualisation = {
  docker = {
    enable = lib.mkDefault false;
    liveRestore = lib.mkDefault false;
  };
  podman = {
    enable = lib.mkDefault (!config.virtualisation.docker.enable);
    dockerCompat = lib.mkDefault true;
  };
};
```

Darwin

⟨*modules/darwin.nix* 86⟩≡   87 ▷
```
{ pkgs, ... }:

{
```

⟨*modules/darwin.nix* 86⟩+≡   ◁86 88 ▷
```
  nixpkgs.config.allowUnfree = true;
```

⟨*modules/darwin.nix* 86⟩+≡   ◁87 89 ▷
```
  services.chunkwm = {
    enable = true;
    extraConfig = "
      chunkc core::log_file /usr/local/var/log/chunkwm.log
      chunkc core::log_level info
      chunkc tiling::rule -owner "Cisco AnyConnect Secure Mobility Client" -state float &
      chunkc tiling::rule -owner Emacs -except "^$" -state tile &
    ";
    package = pkgs.chunkwm.core;
    plugins = {
      dir = "/run/current-system/sw/bin/chunkwm-plugins/";
      list = [ "ffm" "tiling" ];
      "tiling".config = "
        chunkc set desktop_padding_step_size    0
        chunkc set desktop_gap_step_size        0
        chunkc set global_desktop_mode          bsp
        chunkc set global_desktop_offset_top    0
        chunkc set global_desktop_offset_bottom 0
        chunkc set global_desktop_offset_left   0
        chunkc set global_desktop_offset_right  0
        chunkc set global_desktop_offset_gap    0
        chunkc set bsp_spawn_left               1
        chunkc set bsp_optimal_ratio            1.618
        chunkc set bsp_split_mode               optimal
        chunkc set bsp_split_ratio              0.66
        chunkc set window_focus_cycle           all
        chunkc set mouse_follows_focus          1
        chunkc set window_region_locked         1
      ";
    };
  };
```

⟨*modules/darwin.nix* 86⟩+≡   ◁88 90▷

```
services.skhd = {
  enable = true;
  package = pkgs.skhd;
  skhdConfig = "
    alt - space : /Applications/kitty.app/Contents/MacOS/kitty –single-instance -d ~

    ctrl + alt + cmd - up    : chunkc tiling::window –warp north
    ctrl + alt + cmd - right : chunkc tiling::window –warp east
    ctrl + alt + cmd - down  : chunkc tiling::window –warp south
    ctrl + alt + cmd - left  : chunkc tiling::window –warp west

    shift + alt + cmd - up    : chunkc tiling::window –focus north
    shift + alt + cmd - right : chunkc tiling::window –focus east
    shift + alt + cmd - down  : chunkc tiling::window –focus south
    shift + alt + cmd - left  : chunkc tiling::window –focus west

    ctrl + alt + cmd - e : chunkc tiling::desktop –equalize
    ctrl + alt + cmd - f : chunkc tiling::window –toggle float
    ctrl + alt + cmd - m : chunkc tiling::window –toggle fullscreen

    shift + cmd - 1 : chunkc tiling::window –send-to-desktop 1
    shift + cmd - 2 : chunkc tiling::window –send-to-desktop 2
    shift + cmd - 3 : chunkc tiling::window –send-to-desktop 3
    shift + cmd - 4 : chunkc tiling::window –send-to-desktop 4
    shift + cmd - 5 : chunkc tiling::window –send-to-desktop 5
    shift + cmd - 6 : chunkc tiling::window –send-to-desktop 6
  ";
};
```

> Dig through https://github.com/mathiasbynens/dotfiles/blob/master/.macos

Set `AppleKeyboardUIMode` = **3** to enable full keyboard access, e.g. tab in dialogs. Disable press-and-hold, in favor of key repeat.

⟨*modules/darwin.nix* 86⟩+≡   ◁89 91▷

```
system.defaults.NSGlobalDomain = {
  AppleKeyboardUIMode = 3;

  ApplePressAndHoldEnabled = false;
  InitialKeyRepeat = 10;
  KeyRepeat = 1;

  NSAutomaticDashSubstitutionEnabled = false;
  NSAutomaticQuoteSubstitutionEnabled = false;
  NSAutomaticSpellingCorrectionEnabled = true;

  NSNavPanelExpandedStateForSaveMode = true;
  NSNavPanelExpandedStateForSaveMode2 = true;
};
```

⟨*modules/darwin.nix* 86⟩+≡   ◁90 92▷

```
system.defaults.dock = {
  autohide = true;
  mru-spaces = false;
  orientation = "right";
  showhidden = true;
};
```

⟨*modules/darwin.nix* 86⟩+≡   ◁91 93▷
```
    system.defaults.finder = {
      AppleShowAllExtensions = true;
      FXEnableExtensionChangeWarning = false;
      QuitMenuItem = true;
    };
```

⟨*modules/darwin.nix* 86⟩+≡   ◁92 94▷
```
    system.defaults.trackpad.Clicking = true;
```

⟨*modules/darwin.nix* 86⟩+≡   ◁93
```
  }
```

Software configuration

Bash

Enable the Bash module, as well as autojump.

⟨*config/bash.nix* 95⟩≡
```
{ ... }:

{

  programs.autojump.enable = true;

  programs.bash.enable = true;

}
```

bat

Enable the bat[13] module.

[13] https://github.com/sharkdp/bat

⟨*config/bat.nix* 96⟩≡   97 ▷
```
{ ... }:

{

  programs.bat = {
    enable = true;
    config = {
```

For the pager, use `less`.

⟨*config/bat.nix* 96⟩+≡   ◁ 96 98 ▷
```
      pager = "less -FR";
```

Set the style to `changes`.

⟨*config/bat.nix* 96⟩+≡   ◁ 97 99 ▷
```
      style = "changes";
```

Use the Monokai Extended theme.

⟨*config/bat.nix* 96⟩+≡   ◁ 98 100 ▷
```
      theme = "Monokai Extended";
```

⟨*config/bat.nix* 96⟩+≡   ◁ 99
```
    };
  };

}
```

Browserpass

⟨*config/browserpass.nix* 101⟩≡

```
{ pkgs, ... }:

{

  home.packages = with pkgs; [
    browserpass
  ];

  programs.browserpass = {
    enable = true;
    browsers = [ "firefox" ];
  };

}
```

bugwarrior

⟨*config/bugwarrior.nix* 102⟩≡   103 ▷

```
{ config, pkgs, ... }:

{
```

⟨*config/bugwarrior.nix* 102⟩+≡   ◁102 104 ▷

```
  home.packages = with pkgs; [
    bugwarrior
  ];
```

⟨*config/bugwarrior.nix* 102⟩+≡   ◁103 105 ▷

```
  imports = [
    ./password-store.nix
    ./taskwarrior
    ./password-store.nix
  ];
```

⟨*config/bugwarrior.nix* 102⟩+≡   ◁104 106▷

```
xdg.configFile."bugwarrior/bugwarriorrc".text = "
  [general]
  targets = sportradar_jira
  taskrc = ${config.home.homeDirectory}/.taskrc
  inline_links = False
  annotation_links = True
  annotation_comments = True
  legacy_matching = False
  log.level = DEBUG
  log.file = ${config.home.homeDirectory}/log/bugwarrior.log
  annotation_length = 80

  [sportradar_jira]
  service = jira
  jira.base_uri = https://jira.sportradar.ag
  jira.username = e.bailey
  jira.password = @oracle:eval:pass jira.sportradar.ag
  jira.query = ((assignee = e.bailey OR reporter = e.bailey) OR (summary ~ e.baley OR description ~ e.bailey OR c
  jira.version = 8
  jira.add_tags = work
  jira.description_template = {{jiraid}}: {{jirasummary}}
";
```

⟨*config/bugwarrior.nix* 102⟩+≡   ◁105

```
}
```

direnv

⟨*config/direnv.nix* 107⟩≡

```
{ ... }:

{

  programs.direnv = {
    enable = true;
    enableZshIntegration = false;
  };

}
```

Dunst

⟨*config/dunst.nix* 108⟩≡

```
{ pkgs, ... }:

{
  services.dunst = {
    enable = true;
```
⟨*Configure Dunst icon theme* 109⟩
```
    settings = {
      global = {
```
⟨*Configure Dunst global settings* 110⟩
```
      };
```
⟨*Configure Dunst shortcuts* 111⟩
⟨*Configure Dunst urgency styling* 112⟩
⟨*Configure critical Slack notifications* 113⟩
```
    };
  };
}
```

⟨*Configure Dunst icon theme* 109⟩≡   (108)

```
iconTheme = {
  package = pkgs.paper-icon-theme;
  name = "Paper";
  size = "48x48";
};
```

⟨*Configure Dunst global settings* 110⟩≡   (108)

```
font = "Iosevka Term 10";
markup = "yes";
plain_text = "no";
format = "<b>%s</b>\n%b";
sort = "yes";
indicate_hidden = "yes";
alignment = "center";
bounce_freq = 0;
show_age_threshold = 30;
word_wrap = "yes";
ignore_newline = "no";
stack_duplicates = "yes";
hide_duplicates_count = "yes";
geometry = "700x5-0-70";
shrink = "no";
transparency = 15;
idle_threshold = 0;
follow = "keyboard";
sticky_history = "yes";
history_length = 15;
show_indicators = "no";
startup_notification = false;
dmenu = "/run/current-system/sw/bin/dmenu -p dunst:";
browser = "/etc/profiles/per-user/e.bailey/bin/firefox -new-tab";
icon_position = "left";
max_icon_size = 80;
frame_width = 0;
frame_color = "#8EC07C";
```

⟨*Configure Dunst shortcuts* 111⟩≡   (108)

```
shortcuts = {
  close = "mod1+space";
  close_all = "mod4+mod1+space";
  history = "ctrl+grave";
};
```

⟨*Configure Dunst urgency styling* 112⟩≡   (108)

```
urgency_low = {
  frame_color = "#3B7C87";
  foreground = "#3B7C87";
  background = "#2B313C";
  timeout = 4;
};
urgency_normal = {
  frame_color = "#5B8234";
  foreground = "#5B8234";
  background = "#2B313C";
  timeout = 6;
};
urgency_critical = {
  frame_color = "#B7472A";
  foreground = "#B7472A";
  background = "#191311";
  timeout = 8;
};
```

⟨*Configure critical Slack notifications* 113⟩≡ (108)

```
  slack = {
    appname = "Slack";
    body = "*critical*";
    frame_color = "#B7472A";
    foreground = "#B7472A";
    background = "#191311";
    urgency = "critical";
  };
```

Emacs

⟨*config/emacs/default.nix* 114⟩≡

```
  { config, lib, pkgs, ... }:

  {

    imports = [
      ./packages.nix
    ];

    home.file.".emacs.d/init.el".source = ./init.el;

    home.sessionVariables = rec {
      EDITOR = "emacsclient -nw -a \"\"";
      GIT_EDITOR = EDITOR;
      VISUAL = "emacsclient -cna \"\"";
    };

    programs.emacs.enable = true;

    programs.fish.shellAliases = lib.mkIf (config.programs.fish.enable) rec {
      e = "emacsclient -na \"\"";
      ec = e + " -c";
      et = "emacsclient -nw -a \"\"";
    };

    services.emacs.enable = ! pkgs.stdenv.isDarwin;

  }
```

INSTALL some Emacs packages.

⟨*config/emacs/packages.nix* 115⟩≡

```
{ pkgs, ... }:

{

  home.packages = with pkgs; [
    graphviz
    sqlite
  ];

  programs.emacs.extraPackages = epkgs: (
    with epkgs.elpaPackages; [
      ⟨elpa packages 116⟩
    ]
  ) ++ (
    with epkgs.melpaPackages; [
      ⟨melpa packages 117⟩
    ]
  );

}
```

ELPA Packages

⟨*elpa packages* 116⟩≡   (115)

```
mmm-mode
rainbow-mode
```

MELPA Packages

⟨*melpa packages* 117⟩≡   (115)

```
avy
better-defaults
clj-refactor
clojure-mode
# TODO: company-lsp
# TODO: cquery
crux
# dhall-mode
direnv
dockerfile-mode
editorconfig
elixir-mode
emojify
# enh-ruby-mode
erlang
fill-column-indicator
fish-mode
frames-only-mode
gap-mode
go-mode
graphviz-dot-mode
haskell-mode
helm-ag
# TODO: hindent
hl-todo
htmlize
idris-mode
j-mode
lfe-mode
kubernetes-tramp
# TODO: lsp-haskell
# TODO: lsp-mode
# TODO: lsp-ui
magit
markdown-mode
multiple-cursors
nix-mode
nyan-mode
org-roam
dash
f
s
emacsql
emacsql-sqlite3
paredit
rainbow-delimiters
# robe
rust-mode
# rvm
smex
terraform-mode
tuareg
use-package
whitespace-cleanup-mode
```

```
yaml-mode
```

Init File

⟨*config/emacs/init.el* 118⟩≡   119 ▷
```
(server-start)
```

⟨*config/emacs/init.el* 118⟩+≡   ◁ 118 120 ▷
```
(setq server-window 'pop-to-buffer-same-window)
```

⟨*config/emacs/init.el* 118⟩+≡   ◁ 119 121 ▷
```
(column-number-mode 1)
```

⟨*config/emacs/init.el* 118⟩+≡   ◁ 120 122 ▷
```
(add-to-list 'exec-path "/run/current-system/sw/bin")
```

⟨*config/emacs/init.el* 118⟩+≡   ◁ 121 123 ▷
```
(menu-bar-mode 0)
(scroll-bar-mode 0)
(tool-bar-mode 0)
```

⟨*config/emacs/init.el* 118⟩+≡   ◁ 122 124 ▷
```
(let ((font "Iosevka-10"))
  (set-face-attribute 'default t :font font)
  (set-frame-font font nil t))
```

⟨*config/emacs/init.el* 118⟩+≡   ◁ 123 125 ▷
```
(require 'package)
```

⟨*config/emacs/init.el* 118⟩+≡   ◁ 124 126 ▷
```
(setq-default frames-only-mode t
              indent-tabs-mode nil
              package-archives nil
              package-enable-at-startup nil)
```

⟨*config/emacs/init.el* 118⟩+≡   ◁ 125 127 ▷
```
(package-initialize)

(eval-when-compile
  (require 'use-package))

(setq-default use-package-always-defer t
              use-package-always-ensure t)
```

https://stackoverflow.com/a/18330742

⟨*config/emacs/init.el* 118⟩+≡   ◁126 128▷
```
(let ((backup-directory (concat user-emacs-directory "backup")))
  (unless (file-exists-p backup-directory)
    (make-directory backup-directory t))
  (setq backup-directory-alist '(("" . ,backup-directory))))

(setq auto-save-default        t
      auto-save-interval       200
      auto-save-timeout        20
      backup-by-copying        t
      delete-by-moving-to-trash t
      delete-old-versions      t
      kept-new-versions        6
      kept-old-versions        2
      make-backup-files        t
      vc-make-backup-files     t
      version-control          t)
```

https://anirudhsasikumar.net/blog/2005.01.21.html Copyright
©2004-2011 Anirudh Sasikumar. All rights reserved.

⟨*config/emacs/init.el* 118⟩+≡   ◁127 129▷
```
(define-minor-mode sensitive-mode
  "For sensitive files like password lists.
It disables backup creation and auto saving.

With no argument, this command toggles the mode.
Non-null prefix argument turns on the mode.
Null prefix argument turns off the mode."
  ;; The initial value.
  nil
  ;; The indicator for the mode line.
  " ▨"
  ;; The minor mode bindings.
  nil
  (if (symbol-value sensitive-mode)
      (progn
        ;; disable backups
        (set (make-local-variable 'backup-inhibited) t)
        ;; disable auto-save
        (if auto-save-default
            (auto-save-mode -1)))
    ;; resort to default value of backup-inhibited
    (kill-local-variable 'backup-inhibited)
    ;; resort to default auto save setting
    (if auto-save-default
        (auto-save-mode 1))))
```

⟨*config/emacs/init.el* 118⟩+≡   ◁128 130▷
```
(dolist (pattern '("^\\(/dev/shm/\\|/tmp/\\)"
                   "\\.\\(enc\\|gpg\\|hashedPassword\\)$"))
  (add-to-list 'auto-mode-alist (cons pattern 'sensitive-mode)))
```

⟨*config/emacs/init.el* 118⟩+≡   ◁129 131▷
```
(setq custom-file "~/.emacs.d/private/local/custom.el")
```

⟨*config/emacs/init.el* 118⟩+≡   ◁130 132▷
```
(load-theme 'wombat)
```

⟨*config/emacs/init.el* 118⟩+≡   ◁131 133▷
```
(global-set-key (kbd "C-x C-k") 'kill-this-buffer)
(global-set-key (kbd "s-u") 'revert-buffer)
```

⟨*config/emacs/init.el* 118⟩+≡   ◁132 134▷
```
(setq c-default-style     "k&r"
      c-basic-offset       4
      emacs-lisp-mode-hook '(fci-mode paredit-mode
                             rainbow-delimiters-mode)
      js-indent-level      2
      text-mode-hook       '(text-mode-hook-identify))
```

⟨*config/emacs/init.el* 118⟩+≡   ◁133 135▷
```
(setq-default fill-column 80)
```

⟨*config/emacs/init.el* 118⟩+≡   ◁134 136▷
```
(org-babel-do-load-languages
 'org-babel-load-languages
 '((shell . t)))
```

⟨*config/emacs/init.el* 118⟩+≡   ◁135 137▷
```
(use-package avy
  :demand
  :config
  (global-set-key (kbd "C-;") 'avy-goto-char)
  (global-set-key (kbd "C-'") 'avy-goto-char-2)
  (global-set-key (kbd "M-g f") 'avy-goto-line))
```

⟨*config/emacs/init.el* 118⟩+≡   ◁136 138▷
```
(use-package crux
  :demand
  :config (global-set-key (kbd "C-a") 'crux-move-beginning-of-line))
```

⟨*config/emacs/init.el* 118⟩+≡   ◁137 139▷
```
(use-package editorconfig
  :ensure t
  :config
  (editorconfig-mode 1))
```

⟨*config/emacs/init.el* 118⟩+≡   ◁138 140▷
```
(use-package emojify)
```

⟨*config/emacs/init.el* 118⟩+≡   ◁139 141▷
```
(use-package helm
  :demand
  :config
  (global-set-key (kbd "M-s-f") 'helm-do-grep-ag)
  (global-set-key (kbd "M-s-f") 'helm-do-grep-ag))
```

⟨*config/emacs/init.el* 118⟩+≡   ◁140 142▷
```
(use-package hl-todo
  :demand
  :config (global-hl-todo-mode t))
```

⟨*config/emacs/init.el* 118⟩+≡   ◁141 143▷
```
(use-package magit
  :demand
  :config (global-magit-file-mode t))
```

⟨*config/emacs/init.el* 118⟩+≡   ◁142 144▷
```
(use-package multiple-cursors
  :demand
  :config (global-set-key (kbd "C-S-c C-S-c") 'mc/edit-lines))
```

⟨*config/emacs/init.el* 118⟩+≡   ◁143 145▷
```
(use-package nix-mode
  :mode ("\\.nix\\'"))
```

⟨*config/emacs/init.el* 118⟩+≡   ◁144 146▷
```
(use-package noweb-mode
  :load-path "/run/current-system/sw/share/emacs/site-lisp"
  :mode ("\\.nw\\'")
  :demand)
```

⟨*config/emacs/init.el* 118⟩+≡   ◁145 147▷
```
(use-package nyan-mode
  :demand
  :config (nyan-mode 1))
```

⟨*config/emacs/init.el* 118⟩+≡   ◁146 148▷
```
(use-package smex
  :demand
  :config
  (global-set-key (kbd "M-x") 'smex)
  (global-set-key (kbd "M-X") 'smex-major-mode-commands)
  (global-set-key (kbd "C-c C-c M-x") 'execute-extended-command))
```

⟨*config/emacs/init.el* 118⟩+≡   ◁147 149▷
```
(use-package tuareg
  :mode ("\\.ml\\'" "\\.mli\\'"))
```

⟨*config/emacs/init.el* 118⟩+≡   ◁148
```
(use-package whitespace-cleanup-mode
  :demand
  :config (global-whitespace-cleanup-mode t))
```

Firefox

⟨*config/firefox.nix* 150⟩≡

```
{ pkgs, ... }:

{

  programs.firefox = {
    enable = true;
    extensions = with pkgs.nur.repos.rycee.firefox-addons; [
      browserpass
      darkreader
      https-everywhere
      lastpass-password-manager
      privacy-badger
    ];
    profiles = {
      default = {
        settings = {
          # http://kb.mozillazine.org/About:config_entries
          "browser.ctrlTab.recentlyUsedOrder" = false;
          "browser.newtabpage.activity-stream.asrouter.userprefs.cfr.addons" = false;
          "browser.newtabpage.activity-stream.asrouter.userprefs.cfr.features" = false;
          "browser.newtabpage.activity-stream.feeds.section.topstories" = false;
          "browser.newtabpage.activity-stream.feeds.snippets" = false;
          # FIXME: "browser.newtabpage.activity-stream.improvesearch.topSiteSearchShortcuts.havePinned" = null;
          "browser.newtabpage.activity-stream.section.highlights.includePocket" = false;
          "browser.newtabpage.activity-stream.showSearch" = false;
          # FIXME: "browser.newtabpage.pinned" = "[]";
          # FIXME: "browser.search.defaultenginename" = "DuckDuckGo";
          # FIXME: "browser.search.defaulturl" = "https://duckduckgo.com/?q=";
          "browser.search.hiddenOneOffs" = "Google,Bing,Amazon.com,eBay,Twitter,Wikipedia (en)";
          # FIXME: git "browser.search.selectedEngine" = "DuckDuckGo";
          "browser.search.suggest.enabled" = false;
          "browser.startup.page" = 3;
          "browser.tabs.unloadOnLowMemory" = true;
          "browser.urlbar.placeholderName" = "DuckDuckGo";
          "extensions.activeThemeID" = "firefox-compact-dark@mozilla.org";
          "font.size.variable.x-western" = 12;
          "signon.rememberSignons" = false;
        };
      };
    };
  };

}
```

fish

⟨*config/fish/default.nix* 151⟩≡

```
{ config, lib, pkgs, ... }:

{

  imports = [
    ./abbrs.nix
    ./aliases.nix
  ];

  home = {
    packages = with pkgs; [
      exa
    ];
    sessionVariables = {
      SHELL = "fish";
      TERMINAL = "kitty";
    };
  };

  programs.autojump.enable = true;

  programs.fish =
    let
      inherit (lib.strings) fileContents;
    in
    {
      enable = true;
      interactiveShellInit = fileContents ./interactiveShellInit.fish;
      promptInit = lib.mkIf (!(config.programs.starship.enable)) "
        ${fileContents ./sushi/fish_prompt.fish}
        ${fileContents ./sushi/fish_right_prompt.fish}
      ";
      shellInit = fileContents ./shellInit.fish;
    };

}
```

Fish abbreviations

"`abbr` manages abbreviations - user-defined words that are replaced with longer phrases after they are entered.

 For example, a frequently-run command like `git checkout` can be abbreviated to `gco`. After entering `gco` and pressing `Space` or `Enter`, the full text `git checkout` will appear in the command line."[14]

⟨*config/fish/abbrs.nix* 152⟩≡
```
{ ... }:

{

  programs.fish.shellAbbrs = {
    ⟨AWS Azure login abbreviation 153⟩
    ⟨Kubernetes fish abbreviations 154⟩
    ⟨Nix fish abbreviations 155⟩
    ⟨ripgrep fish abbreviations 156⟩
  };

}
```

[14] abbr - manage fish abbreviations. URL https://fishshell.com/docs/current/cmds/abbr.html

NAAL[15] IS COOL AND ALL, BUT...

 ... it's a bad habit too. the `aal` abbeviation makes being not so sneaky a bit less annoying.

[15] https://github.com/yurrriq/naal

⟨*AWS Azure login abbreviation* 153⟩≡   (152)
```
  aal = "aws-azure-login –no-prompt";
```

`kubectl` IS VERY CUMBERSOME to type all the time. Using fish abbrevations, save some precious keystrokes. It's definitely worth the time.

⟨*Kubernetes fish abbreviations* 154⟩≡   (152)
```
  kc = "kubectl";
  kcd = "kubectl drain –delete-local-data –ignore-daemonsets";
  kcnp = "kubectl get pods –field-selector=spec.nodeName=";
  kcx = "kubectl –context";
  kgno = "kubectl get nodes";
  kn = "kubens";
  kns = "kubens";
  kt = "stern";
  kx = "kubectx";
```

IT'S USEFUL TO RUN `nix build` with various flags. Why not save some keystrokes for those tasks, too?

⟨*Nix fish abbreviations* 155⟩≡   (152)
```
  nb = "nix build";
  nbd = "nix build –dry-run";
  nbn = "nix build –no-link";
  nbo = "nix build -o";
```

I'M NOT QUITE USED TO all the `ripgrep` flags yet. I'm also a lazy typist, so define some memorable abbreviations.

"Searches case insensitively if the pattern is all lowercase. Search case sensitively otherwise."[16]

⟨*ripgrep fish abbreviations* 156⟩≡   (152)   157 ▷
```
rg = "rg -S";
```

⟨*ripgrep fish abbreviations* 156⟩+≡   (152)   ◁156   158 ▷
```
rga = "rg –hidden –iglob !.git";
rgf = "rg -F";
```

"Never print the file path with the matched lines", and "[s]uppress line numbers."[17]

⟨*ripgrep fish abbreviations* 156⟩+≡   (152)   ◁157   159 ▷
```
rgin = "rg -IN";
```

```
λ rg -IN name: helmfile.d/*.yaml | sort -u
- name: cert-manager
- name: cluster-autoscaler
- name: elastalert
- name: elasticsearch-client
- name: elasticsearch-curator
- name: elasticsearch-exporter
- name: elasticsearch-master
- name: external-dns
- name: fluentd-elasticsearch
- name: hubble
- name: kibana
- name: kube-resource-report
- name: metrics-server
- name: nginx-ingress-core
- name: prometheus-operator
```

⟨*ripgrep fish abbreviations* 156⟩+≡   (152)   ◁158
```
rgn = "rg –no-heading";
```

⟨*config/fish/aliases.nix* 160⟩≡
```
{ ... }:

{

  programs.fish.shellAliases = rec {
    gpg = "gpg2";

    k = "clear";

    l = "exa –color=auto -G";
    ll = "exa –color=auto -Gla";

    # Old Darwin habits
    pbcopy = "xclip -sel clipboard";
    pbpaste = "${pbcopy} -o";
  };

}
```

[16] Andrew Gallant. rg - man page. URL https://www.mankier.com/1/rg

[17] Andrew Gallant. rg - man page. URL https://www.mankier.com/1/rg

⟨*config/fish/shellInit.fish* 161⟩≡

```
for p in /run/current-system/sw/bin ~/bin
    if not contains $p $fish_user_paths
        set -U fish_user_paths $p $fish_user_paths
    end
end

set -U fish_user_paths /run/wrappers/bin $fish_user_paths


function fish_title
    echo "$PWD | $_" | sed "s|$HOME|~|g"
end
```

⟨*config/fish/interactiveShellInit.fish* 162⟩≡   163 ▷

```
function clone
    function __update
        test -d $argv[1]; and cd $argv[1]; and git fetch -all; and git pull
    end

    function __usage
        echo "Usage: clone [username] [repository] [[destination]]"
    end

    set -local num_args (count $argv)

    if test $num_args -ge 2
        set -local user $argv[1]
        set -local repo $argv[2]

        if test $num_args -eq 2
            set dest ~/src/$user/$repo
        else if test $num_args -eq 3
            set dest $argv[3]/$user/$repo
        else
            __usage
        end

        echo $dest

        git clone git@github.com:$user/$repo.git $dest; or __update $dest; or __usage
    else
        __usage
    end
end
```

⟨*config/fish/interactiveShellInit.fish* 162⟩+≡   ◁162 164▷

```
function latest -d 'Print the latest tag (on GitHub) for a given user and repo.'
  # TODO: __usage

  set -local num_args (count $argv)

  if test $num_args -eq 2
    set -local user $argv[1]
    set -local repo $argv[2]
    http https://api.github.com/repos/$user/$repo/tags | jq '.[0].name'
  end
end
```

⟨*config/fish/interactiveShellInit.fish* 162⟩+≡   ◁163 165▷

```
command -sq aws; and command -sq jq; and \
function describe-cert -d 'List the domains for a given ACM certificate'
    test (count $argv) -ne 1; and return
    aws acm describe-certificate --certificate-arn $argv[1] |
    jq -r '.Certificate | .SubjectAlternativeNames[]'
end
```

⟨*config/fish/interactiveShellInit.fish* 162⟩+≡   ◁164 166▷

```
command -sq fluidsynth; and function playmidi
    fluidsynth -i ~/lib/arachno-soundfont/Arachno\ SoundFont\ -\ Version\ 1.0.sf2 $argv
end
```

⟨*config/fish/interactiveShellInit.fish* 162⟩+≡   ◁165 167▷

```
command -sq kitty; and function icat
    kitty +kitten icat $argv
end
```

⟨*config/fish/interactiveShellInit.fish* 162⟩+≡   ◁166 168▷

```
command -sq kubectl; and begin
    # TODO: Add option to print server versions too.
    function k8s::versions
        printf "kubectl %s\n" (command kubectl version --client --short)
        printf "helm %s\n" (command helm version --client --short)
        command helmfile --version
        printf "kops %s\n" (command kops version)
    end

    function kcterm -d 'Terminate a Kubernetes node'
        test (count $argv) -ne 1; and return
        kubectl get node -o jsonpath='{.spec.providerID}' $argv[1] |
        cut -d'/' -f5 |
        xargs aws ec2 terminate-instances --instance-ids
    end
end
```

⟨*config/fish/interactiveShellInit.fish* 162⟩+≡   ◁167 169▷

```
# FIXME: functions rvm >/dev/null 2>&1; and rvm default
```

⟨*config/fish/interactiveShellInit.fish* 162⟩+≡   ◁168 170▷

```
set fish_greeting
```

⟨*config/fish/interactiveShellInit.fish* 162⟩+≡   ◁169

```
command -sq task; and command -sq jq; and function tj \
  -d 'Open the Jira ticket associated with a Taskwarrior task'
  test (count $argv) -ne 1; and return
  open (task $argv[1] export | jq -r '.[0].jiraurl')
end
```

⟨*config/fish/sushi/fish_prompt.fish* 171⟩≡

```fish
function fish_prompt
    set -l code $status

    if set -q IN_NIX_SHELL
        set symbol "λΠ "
    else
        set symbol "λ "
    end

    if git::is_repo
        set -l branch (git::branch_name ^/dev/null)
        set -l ref (git show-ref --head --abbrev | awk '{print substr($0,0,7)}' | sed -n 1p)

        if git::is_stashed
            echo -n -s (white)"^"(off)
        end

        echo -n -s (red)"("(off)

        if git::is_dirty
            printf (white)"*"(off)
        end

        if command git symbolic-ref HEAD > /dev/null ^/dev/null
            if git::is_staged
                printf (cyan)"$branch"(off)
            else
                printf (yellow)"$branch"(off)
            end
        else
            printf (dim)"$ref"(off)
        end

        for remote in (git remote)
            set -l behind_count (echo (command git rev-list $branch..$remote/$branch ^/dev/null | wc -l | tr -d " "
            set -l ahead_count (echo (command git rev-list $remote/$branch..$branch ^/dev/null | wc -l | tr -d " ")

            if test $ahead_count -ne 0; or test $behind_count -ne 0; and test (git remote | wc -l) -gt 1
                echo -n -s " "(orange)$remote(off)
            end

            if test $ahead_count -ne 0
                echo -n -s (white)" +"$ahead_count(off)
            end

            if test $behind_count -ne 0
                echo -n -s (white)" -"$behind_count(off)
            end
        end

        echo -n -s (red)") "(off)
    end

    if test "$code" = 0
        echo -n -s (red)"$symbol"(off)
```

```
    else
        echo -n -s (dim)"$symbol"(off)
    end
end
```

⟨*config/fish/sushi/fish__right__prompt.fish* 172⟩≡   173 ▷

```
# ——————————————————————————————— [ Colors ]

function orange
    set_color -o ee5819
end

function yellow
    set_color -o b58900
end

function red
    set_color -o d30102
end

function cyan
    set_color -o 2aa198
end

function white
    set_color -o fdf6e3
end

function dim
    set_color -o 4f4f4f
end

function off
    set_color -o normal
end
```

⟨*config/fish/sushi/fish_right_prompt.fish* 172⟩+≡   ◁172 174▷

```fish
# —————————————————————————————————————————- [ Git ]

function git::is_repo
    test -d .git; or command git rev-parse —git-dir >/dev/null ^/dev/null
end

function git::branch_name
    git::is_repo; and begin
        command git symbolic-ref —short HEAD ^/dev/null;
        or command git show-ref —head -s —abbrev | head -n1 ^/dev/null
    end
end

function git::is_dirty
    git::is_repo; and not command git diff —no-ext-diff —quiet —exit-code
end

function git::is_staged
    git::is_repo; and begin
        not command git diff —cached —no-ext-diff —quiet —exit-code
    end
end

function git::is_stashed
    git::is_repo; and begin
        command git rev-parse —verify —quiet refs/stash >/dev/null
    end
end
```

⟨*config/fish/sushi/fish_right_prompt.fish* 172⟩+≡   ◁173 175▷

```fish
# —————————————————————————————————————————- [ AWS ]

function aws::current_profile
    if set -q AWS_PROFILE
        printf "$AWS_PROFILE"
    else if set -q AWS_DEFAULT_PROFILE
        printf "$AWS_DEFAULT_PROFILE"
    end
end
```

⟨*config/fish/sushi/fish_right_prompt.fish* 172⟩+≡   ◁174 176▷

```fish
# ————————————————————————————————————— [ Kubernetes ]

function k8s::current_context
    command kubectl config current-context
end

function k8s::current_namespace
    command kubectl config view —minify -o jsonpath='{.contexts[0].context.namespace}'
end

function k8s::current_user
    command kubectl config view —minify -o jsonpath='{.users[0].name}' \
        | string replace "@sportradar.com" ""
end
```

⟨*config/fish/sushi/fish__right__prompt.fish* 172⟩+≡   ◁175

```
  # ————————————————————————————————- [ Right Prompt ]
function fish_right_prompt
    set -l cwd (basename (prompt_pwd))

    if git::is_repo
        set root_folder (command git rev-parse –show-toplevel ^/dev/null)
        set parent_root_folder (dirname $root_folder)
        set cwd (echo $PWD | sed -e "s|$parent_root_folder/||")
    end

    printf (yellow)"("(dim)$cwd(yellow)") "(off)

    set -l aws_profile (aws::current_profile)
    test -n "$aws_profile";
    and printf (dim)"{"(yellow)(aws::current_profile)(dim)"} "(off)

    command -sq kubectl; and begin
        if [ "e.bailey" = (k8s::current_user) ]
            printf (dim)"["(off)
        else
            printf (red)"["(off)
        end

        printf (yellow)(k8s::current_context)(off)

        set -l k8s_namespace (k8s::current_namespace)
        test -n "$k8s_namespace";
        and printf (dim)"/"(yellow)"$k8s_namespace"(off)

        if [ "e.bailey" = (k8s::current_user) ]
            printf (dim)"] "(off)
        else
            printf (red)"] "(off)
        end
    end

    printf (dim)(date +%H(yellow):(dim)%M(yellow):(dim)%S)(off)
end
```

fzf

⟨*config/fzf.nix* 177⟩≡

```
  { ... }:

  {

    programs.fzf = {
      enable = true;
      enableZshIntegration = false;
    };

  }
```

Git

⟨*config/git/default.nix* 178⟩≡
```
{ config, lib, pkgs, ... }:

{
  imports = [
    ./aliases.nix
    ./config.nix
    ./packages.nix
  ];

  programs.git = {
    enable = true;
    ignores = [
      "*~"
      ".DS_Store"
    ];
    lfs.enable = true;
  } // (
    with config.accounts.email.accounts.primary; {
      signing.key = gpg.key;
      userEmail = address;
      userName = realName;
    }
  );

}
```

⟨*config/git/packages.nix* 179⟩≡

```
{ lib, pkgs, ... }:

{
  home.packages = with pkgs; (
    [
      diff-pdf
      git
      git-lfs
      github-cli
      kdiff3
      nix-prefetch-git
      nix-prefetch-github
      sops
    ] ++ (
      lib.optionals pkgs.stdenv.isLinux [
        # git-cola
      ]
    ) ++ (
      with gitAndTools; [
        git-crypt
        # gitflow
        hub
        lab
      ]
    ) ++ (
      with nodePackages; [
        codeowners
        diff-so-fancy
      ]
    )
  );

}
```

⟨*config/git/config.nix* 180⟩≡

```nix
{ config, pkgs, ... }:

{

  programs.git.extraConfig = {

    color = {
      diff-highlight = {
        oldNormal = "red bold";
        oldHighlight = "red bold 52";
        newNormal = "green bold";
        newHighlight = "green bold 22";
      };
      diff = {
        meta = 227;
        frag = "magenta bold";
        commit = "227 bold";
        old = "red bold";
        new = "green bold";
        whitespace = "red reverse";
      };
      status = {
        added = "green";
        changed = "yellow";
        untracked = "cyan";
      };
    };

    commit.template = "${config.xdg.dataHome}/git/commit.template";

    core.pager = "diff-so-fancy | less –tabs=4 -RFX";

    credential = {
      helper = "${pkgs.gitAndTools.pass-git-helper}/bin/pass-git-helper";
      useHttpPath = true;
    };

    diff = {
      sopsdiffer = {
        textconv = "sops -d";
      };
    };

    difftool = {
      pdfdiffer.cmd = "diff-pdf –view \"$LOCAL\" \"$REMOTE\"";
      prompt = false;
      trustExitCode = true;
    };

    pull.ff = "only";

    rerere.enabled = true;

    ui.color = true;
```

```
      url."git@gitlab.sportradar.ag:" = {
        insteadOf = "https://gitlab.sportradar.ag/";
      };
    };

    xdg.configFile."pass-git-helper/git-pass-mapping.ini" = {
      source = ./git-pass-mapping.ini;
    };

    xdg.dataFile."git/commit.template" = {
      source = ./commit.template;
    };

}
```

Aliases

⟨*config/git/aliases.nix* 181⟩≡
```
{ config, lib, ... }:

{

  programs.fish.shellAbbrs = lib.mkIf (config.programs.fish.enable) {
    g = "git";
    gd = "git d";
    gdc = "git dc";
    gm = "git merge";
    gs = "git st";
    gt = "git tree";
  };

  programs.git.aliases = rec {
    ap = "add -p";
    bm = "branch --merged";
    bnm = "branch --no-merged";
    ca = "commit --amend";
    cam = "${ca} -m";
    can = "${ca} --no-edit";
    cans = "${can} -S";
    cas = "${ca} -S";
    casm = "${cas} -m";
    cm = "commit -m";
    cnm = "commit -nm";
    co = "checkout";
    cob = "${co} -b";
    cp = "cherry-pick";
    cpa = "${cp} --abort";
    cpc = "${cp} --continue";
    cs = "crypt status";
    cse = "crypt status -e";
    csm = "commit -S -m";
    csu = "crypt status -u";
    d = "diff";
    dad = "add";
    dc = "diff --cached";
    ds = "diff --stat";
    ffco = "flow feature checkout";
    ffr = "flow feature rebase";
    ffs = "flow feature start";
    frf = "flow release finish";
    frfs = "flow release finish -s";
    frs = "flow release start";
    r = "reset";
    rb = "rebase";
    rba = "rebase --abort";
    rbc = "rebase --continue";
    rbi = "rebase --interactive";
    rbs = "rebase --skip";
    rest = "reset";
    rh = "reset --hard";
```

```
      sa = "stash apply";
      sk = "stash -keep-index";
      sl = "stash list";
      sp = "stash pop";
      spa = "stash -patch";
      ss = "stash save";
      st = "status -s";
      stat = "status";
      tree = "log -all -graph -oneline";
    };

  }
```

GPG

⟨*config/gpg.nix* 182⟩≡
```
  { config, ... }:

  {

    programs.gpg = {
      enable = true;
      settings = {
        default-key = config.accounts.email.accounts.primary.gpg.key;
        keyid-format = "long";
        no-emit-version = true;
      };
    };

    services.gpg-agent = {
      enable = true;
      defaultCacheTtl = 28800;
      enableSshSupport = true;
      maxCacheTtl = 28800;
    };

  }
```

htop

⟨*config/htop.nix* 183⟩≡
```
  { ... }:

  {

    programs.htop.enable = true;

  }
```

jq

⟨*config/jq.nix* 184⟩≡
```
{ ... }:

{

  programs.jq.enable = true;

}
```

⟨*config/kitty.nix* 185⟩≡
```
{ pkgs, ... }:

{

  programs.kitty = {
    enable = true;
    extraConfig = "
      include theme.conf
    ";
    font = {
      name = "Iosevka Term";
      package = (pkgs.nerdfonts.override { fonts = [ "Iosevka" ]; });
    };
    keybindings = {
      "kitty_mod+enter" = "new_window_with_cwd";
      "kitty_mod+k" = "
        combine : clear_terminal scrollback active : send_text normal \x0c
      ";
    };
    settings = {
      editor = "emacsclient -nw -a \"\"\"";
      font_size = 24;
      kitty_mod = "ctrl+shift";
      scrollback_lines = -1;
      shell = ".";
      term = "xterm";
      window_border_width = 0;
    };
  };

  xdg.configFile."kitty/theme.conf".source = builtins.fetchurl {
    url = "https://raw.githubusercontent.com/dexpota/kitty-themes/c4bee86ce8372d3426620e27086d6afa6873a98c/themes/S
    sha256 = "0vwjcyd0asqrbs4437k2b80lqlanl79q03kk4pjnfdf379jklrm4";
  };

}
```

man

⟨*config/man.nix* 186⟩≡

```
{ ... }:

{

  programs.man.enable = true;

}
```

nixpkgs

⟨*config/nixpkgs/default.nix* 187⟩≡

```
{ ... }:

{

  nixpkgs.config = import ./nixpkgs-config.nix;

  xdg.configFile."nixpkgs/config.nix".source = ./nixpkgs-config.nix;

}
```

⟨*config/nixpkgs/nixpkgs-config.nix* 188⟩≡

```
{
  allowBroken = false;
  allowUnfree = true;
  allowUnsupportedSystem = false;
}
```

⟨*config/password-store.nix* 189⟩≡

```
{ config, pkgs, ... }:

{
  home.packages = with pkgs; [
    gitAndTools.pass-git-helper
  ];

  programs.password-store = {
    enable = true;
    package = pkgs.pass.withExtensions (exts: with exts; [
      pass-genphrase
      pass-otp
      pass-tomb
      pass-update
    ]);
    settings = {
      PASSWORD_STORE_DIR = "${config.home.homeDirectory}/.password-store";
    };
  };

}
```

rebar3

⟨*config/rebar3.nix* 190⟩≡

```
{ ... }:

{

  xdg.configFile."rebar3/rebar.config".text = "
    {plugins, [rebar3_hex]}.
  ";

}
```

Starship

Starship[18] is the minimal, blazing-fast, and infinitely customizable prompt for any shell!

[18] https://starship.rs

⟨*config/starship.nix* 191⟩≡

```
{ lib, ... }:
{
  programs.starship = {
    enable = true;
    settings = {
      add_newline = false;
      character = {
        success_symbol = "[λ](bold green)";
        error_symbol = "[λ](bold red)";
      };
      format = builtins.concatStringsSep "" (
        lib.splitString "\n" "
          $username
          $hostname
          $shlvl
          $kubernetes
          $directory
          $git_branch
          $git_commit
          $git_state
          $git_status
          $hg_branch
          $docker_context
          $package
          $cmake
          $dart
          $dotnet
          $elixir
          $elm
          $erlang
          $golang
          $helm
          $java
          $julia
          $nim
          $nodejs
          $ocaml
          $perl
          $php
          $purescript
          $python
          $ruby
          $rust
          $swift
          $terraform
          $zig
          $nix_shell
          $conda
          $memory_usage
```

```
        $aws
        $gcloud
        $openstack
        $env_var
        $crystal
        $cmd_duration
        $custom
        $lua
        $jobs
        $battery
        $time
        $line_break
        $status
        $character
      ”
    );
    git_branch.symbol = ”⎇ ”;
    git_commit.tag_disabled = false;
    git_status = {
      ahead = ”↑”${count}”;
      behind = ”↓”${count}”;
      diverged = ”⇕ ↑”${ahead_count}↓”${behind_count}”;
      staged = ”+$count”;
    };
    kubernetes.disabled = false;
    nix_shell = {
      format = ”via [$symbol$state]($style) ”;
      impure_msg = ”ι”;
      pure_msg = ”ρ”;
      symbol = ”❄”;
    };
    stats.disabled = false;
    time.disabled = false;
    };
  };
}
```

Taskwarrior

Config

⟨*config/taskwarrior/default.nix* 192⟩≡   193 ▷
```
  { config, lib, pkgs, ... }:

  {
```

Install Timewarrior[19].   [19] https://timewarrior.net

⟨*config/taskwarrior/default.nix* 192⟩+≡   ◁ 192 194 ▷
```
    home.packages = with pkgs; [
      timewarrior
    ];
```

SINCE HOME-MANAGER GENERATES A READ-ONLY `.taskrc` FILE, and using contexts[20] requires Taskwarrior to be able to modify the it, use the following hacky workaround.

Set `$TASKRC` to `~/.taskrc-dirty`, which must contain at least `include ~/.taskrc`.

⟨*config/taskwarrior/default.nix* 192⟩+≡   ◁193 195▷

```
home.sessionVariables = {
  TASKRC = "~/.taskrc-dirty";
};
```

DEFINE short Taskwarrior shell aliases.

⟨*config/taskwarrior/default.nix* 192⟩+≡   ◁194 196▷

```
programs.fish.shellAliases = lib.mkIf (config.programs.fish.enable) rec {
  p = "task ls limit:page";
  pp = tbd;
  t = "task limit:page";
  ta = "task add";
  tbd = "task burndown.daily";
  te = "env VISUAL=$EDITOR task edit";
  tl = "task list";
  tm = "task mod";
};
```

CONFIGURE TASKWARRIOR, using home-manager[21].

⟨*config/taskwarrior/default.nix* 192⟩+≡   ◁195 197▷

```
programs.taskwarrior = {
  enable = true;
  colorTheme = "solarized-dark-256";
  config = {
    context.other = "jiraurl.none or -work";
    context.work = "jiraurl.any or +work";
    uda = {
      jiracreatedts = {
        label = "Created At";
        type = "date";
      };
      jiradescription = {
        label = "Jira Description";
        type = "string";
      };
      jiraestimate = {
        label = "Estimate";
        type = "numeric";
      };
      jirafixversion = {
        label = "Fix Version";
        type = "string";
      };
      jiraid = {
        label = "Jira Issue ID";
        type = "string";
      };
      jiraissuetype = {
        label = "Issue Type";
        type = "string";
      };
      jirastatus = {
        label = "Jira Status";
        type = "string";
      };
      jirasummary = {
        label = "Jira Summary";
        type = "string";
      };
      jiraurl = {
        label = "Jira URL";
        type = "string";
      };
    };
  };
};
```

LINK the on-exit Git hook. [22]

⟨*config/taskwarrior/default.nix* 192⟩+≡   ◁196 198▷

```
xdg.dataFile."task/hooks/on-exit-git.sh" = {
  executable = true;
  source = ./on-exit-git.sh;
};
```

⟨*config/taskwarrior/default.nix* 192⟩+≡   ◁197

```
xdg.dataFile."task/hooks/on-modify.timewarrior" = {
  executable = true;
  source = let inherit (pkgs.timewarrior) version; in
    with pkgs; stdenv.mkDerivation {
      pname = "taskwarrior-on-modify.timewarrior";
      inherit version;
      nativeBuildInputs = [ makeWrapper ];
      src = fetchurl {
        url = "https://raw.githubusercontent.com/GothenburgBitFactory/timewarrior/v${version}/ext/on-modify.timew
        sha512 = "sha512-CaaELr0JaDJ/LNvwUGDQUQPoKEOapGkJMycD9ujv0DWgfTQzoYkw/JCDPChreT4xCEJ303sAx4NpUqb+260orw==
      };
      dontUnpack = true;
      installPhase = "
        cp $src $out
      ";
      postInstall = "
        wrapProgram $out −prefix PATH : ${lib.makeBinPath [ python3 ]}
      ";
    };
};

}
```

on-exit Git hook

FIRST, use a fairly portable Bash shebang, and be safe.[23]

⟨*config/taskwarrior/on-exit-git.sh* 199⟩≡   200▷

```
#! /usr/bin/env bash

set -eufo pipefail
```

Run in debug mode[24], if the environment variable DEBUG is nonempty.

⟨*config/taskwarrior/on-exit-git.sh* 199⟩+≡   ◁199 201▷

```
if [ -n "${DEBUG:-}" ]; then
    set -x
fi
```

PARSE the command line arguments.

Don't include the `api` version, `rc` file, or Taskwarrior `version`.

```
i="${1#api:}"
="${4#rc:}"
rsion="${6#version:}"
```

Parse and store the values of the arguments, `args`, `command`, and `data` directory.

⟨*config/taskwarrior/on-exit-git.sh* 199⟩+≡   ◁200 202▷
```
  args="${2#args:}"
  command="${3#command:}"
  data="${5#data:}"
```

THROUGHOUT THIS SCRIPT, run `git` as if it were started in the Taskwarrior `data` directory, i.e. `git -C "$data"`.

If there are no changes, `exit` successfully, printing an informative message if running in debug mode. `git diff` `exit`s with status code **0** if there are no differences between the working tree and the index.

⟨*config/taskwarrior/on-exit-git.sh* 199⟩+≡   ◁201 203▷
```
  if git -C "$data" diff –quiet; then
      if [ -n "${DEBUG:-}" ]; then
          echo 'No changes to commit'
      fi
      exit 0
```

If present, stage the changes or die trying.

⟨*config/taskwarrior/on-exit-git.sh* 199⟩+≡   ◁202 204▷
```
  elif ! git -C "$data" add -A; then
      echo 'Failed to add files to the index'
      exit 100
```

Try to commit the changes, or else.

Quietly store the current contents of the index in a new commit, along with a log message describing the changes.

⟨*config/taskwarrior/on-exit-git.sh* 199⟩+≡   ◁203 205▷
```
  elif ! git -C "$data" commit -qm "$command: ${args#task $command}"; then
      echo 'Failed to record changes to the repository'
      exit 101
```

If running in debug mode, print a brief summary of the commit.

⟨*config/taskwarrior/on-exit-git.sh* 199⟩+≡   ◁204
```
  elif [ -n "${DEBUG:-}" ]; then
      git -C "$data" log –oneline -1
  fi
```

xmonad

⟨*config/xmonad/default.nix* 206⟩≡

```
{ pkgs, ... }:

{

  home.file.".xmonad/matrix.png".source = ./matrix.png;

  home.packages = with pkgs; [
    flameshot
    # font-awesome_4
    i3lock
    haskellPackages.xmobar
  ];

  programs.rofi = {
    enable = true;
    font = "Iosevka Term 18";
    pass = {
      enable = true;
      stores = [ "~/.password-store" ];
    };
    theme = "purple";
  };

  services.network-manager-applet.enable = true;

  services.pasystray.enable = true;

  services.picom.enable = true;

  services.stalonetray = {
    enable = false;
    config = {
      background = "#001e26";
      decorations = "none";
      dockapp_mode = "none";
      geometry = "5x1-860-0";
      grow_gravity = "NE";
      icon_gravity = "NE";
      icon_size = 43;
      kludges = "force_icons_size";
      no_shrink = false;
      skip_taskbar = true;
      sticky = true;
      transparent = false;
      window_layer = "bottom";
      window_strut = "none";
      window_type = "dock";
    };
  };

  xdg.configFile."xmobar/xmobarrc".source = ../xmobar/xmobarrc;
```

```
xsession = {
  enable = true;
  pointerCursor = {
    package = pkgs.vanilla-dmz;
    name = "Vanilla-DMZ-AA";
    size = 36;
  };
  scriptPath = ".hm-xsession";
  windowManager.xmonad = {
    enable = true;
    enableContribAndExtras = true;
    config = ./xmonad.hs;
  };
};

}
```

Glossary

module "Each NixOS module is a file that handles one logical aspect of the configuration, such as a specific kind of hardware, a service, or network settings."[25] 5

[25] https://nixos.org/nixos/manual/index.html#sec-writing-modules

Acronyms

Bibliography

abbr - manage fish abbreviations. URL https://fishshell.com/docs/current/cmds/abbr.html.

Andrew Gallant. rg - man page. URL https://www.mankier.com/1/rg.

Rafael J. Wysocki. CPU Performance Scaling – The Linux Kernel documentation. URL https://www.kernel.org/doc/html/v4.19/admin-guide/pm/cpufreq.html#cpu-performance-scaling-in-linux.

To-Do