GNU-Linux setup

In this document I will be making a comprehensive guide to my gnu/linux setup, mostly for my personal future reference but also for anyone else who might be interested :)

Change default shell to zsh

While the default shell in most cases - bash - is pretty good, I prefer the zsh mostly because of its support of plugins (auto-suggest and syntax-highlighting).

```
sudo pacman -S zsh
```

After this I like to install `oh-my-zsh` framework:

```
sh -c "$(curl -fsSL
https://raw.githubusercontent.com/ohmyzsh/ohmyzsh/master/tools/install.sh)
" # install oh-my-zsh for some cool themes
```

I like to use the `powerlevel10k` theme for which some fonts are prerequisite:

- MesloLGS NF Regular.ttf
- MesloLGS NF Bold.ttf
- MesloLGS NF Italic.ttf
- MesloLGS NF Bold Italic.ttf
  Now go to the settings of the terminal application and force it to use the MesloLGS font(otherwise the `powerlevel10k` theme has trouble with the symbols and unicode(?) characters).
  Then,

```
git clone --depth=1 https://github.com/romkatv/powerlevel10k.git
${ZSH_CUSTOM:-$HOME/.oh-my-zsh/custom}/themes/powerlevel10k
```

and set `ZSH_THEME="powerlevel10k/powerlevel10k"` in `~/.zshrc`.

Run `exec zsh` and proceed with the rest of the customisation in `powerlevel10k`.

- autosuggesions plugin

  `git clone https://github.com/zsh-users/zsh-autosuggestions.git $ZSH_CUSTOM/plugins/zsh-autosuggestions`
- zsh-syntax-highlighting plugin

`git clone https://github.com/zsh-users/zsh-syntax-highlighting.git $ZSH_CUSTOM/plugins/zsh-syntax-highlighting`

- zsh-fast-syntax-highlighting plugin

  `git clone https://github.com/zdharma-continuum/fast-syntax-highlighting.git ${ZSH_CUSTOM:-$HOME/.oh-my-zsh/custom}/plugins/fast-syntax-highlighting`
- zsh-autocomplete plugin

  `git clone --depth 1 -- https://github.com/marlonrichert/zsh-autocomplete.git $ZSH_CUSTOM/plugins/zsh-autocomplete`

Now add the plugins to your `.zshrc` :
`plugins=(git zsh-autosuggestions zsh-syntax-highlighting fast-syntax-highlighting zsh-autocomplete)`

I like to use `alt + enter` to accept the autosuggestion. To do that, add `bindkey '^[ ' autosuggest-accept` to your `.zshrc`.
Warning: Make sure you disable any `alt + enter` shortcut that might be set by default by your desktop environment (In KDE Plasma, you can find it in Global Shortcut section in settings, in GNOME you can find it in Keyboard section in settings, fiddle around with other desktop environment to find settings to disable the shortcut :)

Installing nix package manager

Although I have only recently learnt about the nix package manager, I was quickly drawn to it because of its underlying philosophy and ease of use.

I tried to install nix package manager on fedora first but encountered some issues. I think SELinux needs to be set to `Permissive` first and then the command needs to be run. However I did install it without any errors on endeavourOS and that is what I am going to document for now:

- Download the installation script:
  `curl --proto '=https' --tlsv1.2 -sSfL https://nixos.org/nix/install -o nix-install.sh`
- (Optional) View it:
  `less ./nix-install.sh`
- Make it executable:
  `chmod +x ./install.sh`
- Execute the script:
  `./install.sh`

Now to have the nix daemon launched at boot time, one needs to enable the `nix-daemon.service` service:

```
sudo systemctl start nix-daemon.service
sudo systemctl enable nix-daemon.service
```

Now add channels and update them using :

```
nix-channel --add https://nixos.org/channels/nixpkgs-unstable
nix-channel --update
```

In place of `nixpkgs-unstable` you can also have stable: `nixos-22.11`, large channels(provide binary builds for the full breadth of Nixpkgs): `nixos-unstable` and small channels(identical to large channels, but contain fewer binaries. This means they update faster, but require more to be built from source): `nixos-unstable-small`.

To install a package `pkg`,

```
nix-env -iA nixpkgs.pkg
```

To do away with writing `nix-env -iA` every time installing a package, you can add an alias to your `.zshrc`:

```
alias nix-install="nix-env -iA"
```

To uninstall a package `pkg`:

```
nix-env --uninstall pkg
```

Check list of installed program:

```
nix-env -q
```

One issue that I ran into was that programs installed by nix package manager did not show up in application menu (of GNOME in my case). The problem was that application menus scan for `*.desktop` files under `/usr/share/applications` but nix keeps the `.desktop` file of packages installed by it in `~/.nix-profile/share/applications`. One way to resolve the problem is to create a system link of the latter to the former:

```
sudo ln -s ~/.nix-profile/share/applications/* /usr/share/applications
```

However this command needs to be run every time you install a new package using nix. So you can create an alias of it in your `.zshrc`:

```
alias show-nixapp-in-menu="sudo ln -s ~/.nix-profile/share/applications/*
/usr/share/applications"
```

Warning: In case you want to uninstall a program installed by nix and reinstall it using `pacman` or `yay` or your default package manager, don't forget to delete the `.desktop` file of the program after uninstalling it with nix otherwise there will be conflict when your default package would want to create a `.desktop` file in `/usr/share/applications` and find that it already exists and thus exit the installation process with error.

Sidenote: I installed vscode with nix while using Whitesur-gtk-icon-theme. I found out that the icon of vscode was that of a text editor and not of vscode. To change the icon, go to the correspoding `.desktop` file of vscode(generally in `/usr/share/applications`) and change all instances of `Icon=code` to `Icon=visual-studio-code`. However soon I found out that installing vscode with nix would not open external links in browser which prevented me from using `settings sync`. Unfortunately the only fix I could find was to install vscode using `yay` or `pacman` (I used `yay`).

To enable installation of unfree programs with nix, you need to add:

```
{
  # Enable searching for and installing unfree packages
  allowUnfree = true;
}
```

to `~/.config/nixpkgs/config.nix`. If the file or directory is not present, create them.

Packages I installed with nix:
`brave`, `obsidian`, `vlc`.

Distro and DE specific setups

While writing this, I am using EndeavourOS with GNOME desktop environmnet(yes I know it sounds odd, believe me: I used to hate GNOME for being a memory hog, but after getting a beefed up system and suffering with broken KDE Plasma setups for a long time, I took refuge in GNOME, and you know what ? It's not that bad. At least GNOME 44 is pretty good and provides a stable while reasonably customizable experience).

By default, GNOME was using Wayland but I switched to using Xorg(call me backdated!).

Firstly I install a few customization apps:
`gnome-tweaks`, `extension-manager`, `sushi` (preinstalled with gnome).
These are names of the packages as required by `pacman`.

Next I install a few extensions:

- Blur my shell
- Compiz alike magic lamp effect
- Dash to Dock
- Just Perfection

- Media Controls
- Net speed Simplified
- User Themes

I use `Whitesur-gtk` theme for both shell and icons:

```
git clone https://github.com/vinceliuice/WhiteSur-gtk-theme.git
git clone https://github.com/vinceliuice/WhiteSur-icon-theme.git
```

Install the bold version of icons with nord theme using:

```
./install.sh -b -t nord
```

use the `-a` flag for alternative design of icons.

Install nord version of the theme with Nautilus set to `glassy` mode, Monterey style and maximized window to rounded using:

```
./install.sh --nord -N glassy -m --round
```

.zshrc

I am attaching a copy of my `.zshrc` for my convenience:

```
# Enable Powerlevel10k instant prompt. Should stay close to the top of
~/.zshrc.
# Initialization code that may require console input (password prompts,
[y/n]
# confirmations, etc.) must go above this block; everything else may go
below.
if [[ -r "${XDG_CACHE_HOME:-$HOME/.cache}/p10k-instant-prompt-${(%):-
%n}.zsh" ]]; then
  source "${XDG_CACHE_HOME:-$HOME/.cache}/p10k-instant-prompt-${(%):-
%n}.zsh"
fi

# If you come from bash you might have to change your $PATH.
# export PATH=$HOME/bin:/usr/local/bin:$PATH

# Path to your oh-my-zsh installation.
export ZSH="$HOME/.oh-my-zsh"

# Set name of the theme to load --- if set to "random", it will
# load a random theme each time oh-my-zsh is loaded, in which case,
# to know which specific one was loaded, run: echo $RANDOM_THEME
# See https://github.com/ohmyzsh/ohmyzsh/wiki/Themes
ZSH_THEME="powerlevel10k/powerlevel10k"
```

```
# Set list of themes to pick from when loading at random
# Setting this variable when ZSH_THEME=random will cause zsh to load
# a theme from this variable instead of looking in $ZSH/themes/
# If set to an empty array, this variable will have no effect.
# ZSH_THEME_RANDOM_CANDIDATES=( "robbyrussell" "agnoster" )

# Uncomment the following line to use case-sensitive completion.
# CASE_SENSITIVE="true"

# Uncomment the following line to use hyphen-insensitive completion.
# Case-sensitive completion must be off. _ and - will be interchangeable.
# HYPHEN_INSENSITIVE="true"

# Uncomment one of the following lines to change the auto-update behavior
# zstyle ':omz:update' mode disabled  # disable automatic updates
# zstyle ':omz:update' mode auto      # update automatically without
asking
# zstyle ':omz:update' mode reminder  # just remind me to update when it's
time

# Uncomment the following line to change how often to auto-update (in
days).
# zstyle ':omz:update' frequency 13

# Uncomment the following line if pasting URLs and other text is messed
up.
# DISABLE_MAGIC_FUNCTIONS="true"

# Uncomment the following line to disable colors in ls.
# DISABLE_LS_COLORS="true"

# Uncomment the following line to disable auto-setting terminal title.
# DISABLE_AUTO_TITLE="true"

# Uncomment the following line to enable command auto-correction.
# ENABLE_CORRECTION="true"

# Uncomment the following line to display red dots whilst waiting for
completion.
# You can also set it to another string to have that shown instead of the
default red dots.
# e.g. COMPLETION_WAITING_DOTS="%F{yellow}waiting...%f"
# Caution: this setting can cause issues with multiline prompts in zsh <
5.7.1 (see #5765)
# COMPLETION_WAITING_DOTS="true"

# Uncomment the following line if you want to disable marking untracked
files
```

```
# under VCS as dirty. This makes repository status check for large
repositories
# much, much faster.
# DISABLE_UNTRACKED_FILES_DIRTY="true"

# Uncomment the following line if you want to change the command execution
time
# stamp shown in the history command output.
# You can set one of the optional three formats:
# "mm/dd/yyyy"|"dd.mm.yyyy"|"yyyy-mm-dd"
# or set a custom format using the strftime function format
specifications,
# see 'man strftime' for details.
# HIST_STAMPS="mm/dd/yyyy"

# Would you like to use another custom folder than $ZSH/custom?
# ZSH_CUSTOM=/path/to/new-custom-folder

# Which plugins would you like to load?
# Standard plugins can be found in $ZSH/plugins/
# Custom plugins may be added to $ZSH_CUSTOM/plugins/
# Example format: plugins=(rails git textmate ruby lighthouse)
# Add wisely, as too many plugins slow down shell startup.
plugins=(git zsh-autosuggestions zsh-syntax-highlighting)

source $ZSH/oh-my-zsh.sh
bindkey '^[ ' autosuggest-accept
# User configuration

# export MANPATH="/usr/local/man:$MANPATH"

# You may need to manually set your language environment
# export LANG=en_US.UTF-8

# Preferred editor for local and remote sessions
# if [[ -n $SSH_CONNECTION ]]; then
#   export EDITOR='vim'
# else
#   export EDITOR='mvim'
# fi

# Compilation flags
# export ARCHFLAGS="-arch x86_64"

# Set personal aliases, overriding those provided by oh-my-zsh libs,
# plugins, and themes. Aliases can be placed here, though oh-my-zsh
# users are encouraged to define aliases within the ZSH_CUSTOM folder.
# For a full list of active aliases, run `alias`.
#
```

```
# Example aliases
# alias zshconfig="mate ~/.zshrc"
# alias ohmyzsh="mate ~/.oh-my-zsh"

# To customize prompt, run `p10k configure` or edit ~/.p10k.zsh.
[[ ! -f ~/.p10k.zsh ]] || source ~/.p10k.zsh
alias nix-install="nix-env -iA"
alias show-nixapp-in-menu="sudo ln -s ~/.nix-profile/share/applications/*
/usr/share/applications"
```