Linux & nvim

1. 提升 Linux 的 shell 使用体验

1.1. Awesome-alternatives-in-rust

- ls → exa
- cat → bat
- rm → rip

1.2. Some scripts written by yourself

- gif

```
#! /bin/bash
gif_file=~/demo.gif
let x y w h

getwin() {
    XWININFO=$(xwininfo)
    read x < <(awk -F: '/Absolute upper-left X/{print $2}' <<< "$XWININFO")
    read y < <(awk -F: '/Absolute upper-left Y/{print $2}' <<< "$XWININFO")
    read w < <(awk -F: '/Width/{print $2}' <<< "$XWININFO")
    read h < <(awk -F: '/Height/{print $2}' <<< "$XWININFO")
}

getregion() {
    xywh=($(xrectsel "%x %y %w %h")) || exit -1
        let x=${xywh[0]} y=${xywh[1]} w=${xywh[2]} h=${xywh[3]}
    }

case $1 in
    1) getwin ;;
    2) getregion ;;
    3) let x=0 y=0 w=1920 h=1080 ;;
    *)
        echo 1: 选择窗口
        echo 2: 选择区域
        echo 3: 全屏
        exit
        ;;
esac

[ "$2" ] \
    && byzanz-record -x $x -y $y -w $w -h $h -v $gif_file --exec="$2 $3 $4 $5 $6 $7 $8 $9 $10" \
    || byzanz-record -x $x -y $y -w $w -h $h -v $gif_file
    sleep 1
```

- ssh

menu

```
# menu
################################################
key=''
_echo_green() { echo -e "\033[32m$1\033[0m"; }
_get_char() { SAVEDSTTY=`stty -g`; stty -echo; stty raw; dd if=/dev/tty bs=1 count=1 2> /dev/null; stty -raw; stty echo; stty $SAVEDSTTY; }
_list() {
    text=''
    for tab in ${menu_tabs[@]}; do
        test ${tab} = ${menu_tabs[$tab_index]} && text=$text' \033[32m'$tab'\033[0m'  || text=$text' '$tab
    done
    echo -e $text

    for item in ${menu_items[@]}; do
        test ${item} = ${menu_items[$item_index]} && _echo_green " > ${item}" || echo "   ${item}"
    done
}
_key() {
    # 计算新的tab_index
    tab_index=$(($tab_index+$1))
    len=${#menu_tabs[*]}
    test $tab_index -lt 0 && tab_index=$((len - 1))
    test $tab_index -gt $((len - 1)) && tab_index=0

    # 计算新的item_index
    item_index=$(($item_index+$2))
    len=${#menu_items[*]}
    test $item_index -lt 0 && item_index=$((len - 1))
    test $item_index -gt $((len - 1)) && item_index=0

    clear

    pre_hook
    _list
```

```
    after_hook
}

###############################################

function pre_hook() { :; }
function after_hook() { :; }
menu_tabs=()
menu_items=()

# 调用menu方法展开菜单
# 上下左右移动tab或item，回车选中 q Q ctrl-c 退出脚本
menu() {
    _key 0 0
    while :; do
        key=`_get_char`
        case "$key" in
            'q'|'Q'|'^C') exit 1 ;;
            '
') break ;;
            '^[')
                secondchar=`_get_char`
                thirdchar=`_get_char`
                case "$thirdchar" in
                    A) _key 0 -1 ;;
                    B) _key 0 1 ;;
                    D) _key -1 0 ;;
                    C) _key 1 0 ;;
                esac ;;
            'k') _key 0 -1 ;;
            'j') _key 0 1 ;;
            'h') _key -1 0 ;;
            'l') _key 1 0 ;;
        esac
```

```
    done
}
```

ssh

```
#!/bin/bash

source ./menu.sh

menu_items=("腾讯云lighthouse" "腾讯云root" "树莓派-lan" "树莓派-wlan" )

pre_hook() {
    echo '请选择item:'
}

cmds[0]='ssh lighthouse@xxx.xxx.xxx.xxx'
cmds[1]='ssh root@xxx.xxx.xxx.xxx'
cmds[2]='ssh pi@10.42.0.174'
cmds[3]='ssh pi@10.xxx.xxx.xxx'

after_hook() {
    echo
    echo '   '${cmds[$item_index]}
}

case $1 in
    ls)
        menu
        echo 连接${menu_items[$item_index]}
        exec ${cmds[$item_index]}
        ;;
    *) ssh $*;;
esac
```

尝试利用 menu 和 gif 这两个脚本完成一个可以用键盘上下选择模式的 gif 录制工具

```
#!/bin/bash

source ./menu.sh

menu_items=( "窗口" "区域" "全屏" )

pre_hook() {
    echo '请选择item:'
}

cmds[0]="./gif.sh 1"
cmds[1]="./gif.sh 2"
cmds[2]="./gif.sh 3"

after_hook() {
    echo
    echo '   '${cmds[$item_index]}
}

case $1 in
    ls)
        menu
        echo 截图:${menu_items[$item_index]}
        exec ${cmds[$item_index]}
        ;;
    *) ./gif.sh $*;;
esac
```

2. nvim config with lua

相关文件

2.1. lua 基础

不同于原版教程，以下资源适用于国内用户：

- 在 Y 分钟内学习 X 关于 Lua 的页面
- Lua 菜鸟教程
- Lua 用户维基
- Lua 的官方参考手册

Lua 是一种非常干净和简单的语言。它很容易学习，特别是如果你有其他编程语言基础的例如 TypeScript / JavaScript 等，会更加容易上手 Lua。注意：Neovim 嵌入的 Lua 版本是 LuaJIT 2.1.0，它与 Lua 5.1 保持兼容（带有几个 5.2 扩展）

2.2. lua api

查阅 nvim to lua guide 或者在 nvim 中按 `:help lua`

2.3. 配置文件结构

```
~/.config/nvim
├── ftdetect
├── ftplugin
├── init.lua
├── lua
│   ├── colorscheme.lua
│   ├── function.lua
│   ├── keymappings.lua
│   ├── nv-cmp
│   │   └── init.lua
│   ├── nv-floaterm
│   │   └── init.lua
│   ├── nv-lsp
│   │   └── init.lua
│   ├── nv-nvimtree
│   │   └── init.lua
│   ├── nv-treesitter
│   │   └── init.lua
│   ├── nv-undotree
│   │   └── init.lua
│   ├── plugins.lua
│   ├── settings.lua
│   └── ...
└── tmp
    ├── backup
    │   └── ...
    └── undo
        └── ...
```

2.4. 加载配置文件到 nvim

```lua
-- ~/.config/nvim/init.lua
require('settings')
require('plugins')
require('function')
require('keymappings')
require('colorscheme')

require('nv-lsp')
require('nv-treesitter')
require('nv-cmp')
require('nv-undotree')
require('nv-floaterm')
require('nv-nvimtree')
```

2.5. Plugins

- 包管理 nvim-packer

```
yay -S nvim-packer-git
```

- 安装插件

```lua
local execute = vim.api.nvim_command
local fn = vim.fn
local install_path = fn.stdpath("data") .. "/site/pack/packer/test/start/packer.nvim"
local packer = require("packer")

if fn.empty(fn.glob(install_path)) > 0 then
    fn.system({"git", "clone", "https://github.com/wbthomason/packer.nvim", install_path})
    execute "packadd packer.nvim"
end
return require("packer").startup(
    {
        function()
            -- Packer can manage itself
            use "wbthomason/packer.nvim"

            -- File tree
            use "cseickel/nvim-tree.lua"

            -- Config my lsp
            use "neovim/nvim-lspconfig"
            use "williamboman/nvim-lsp-installer"
            use "onsails/lspkind-nvim"

            -- Complete of nvim
            use "hrsh7th/nvim-cmp"
            use "hrsh7th/cmp-buffer"
            use "hrsh7th/cmp-nvim-lsp"
            use "hrsh7th/cmp-path"
            use "ray-x/cmp-treesitter"
            use "quangnguyen30192/cmp-nvim-ultisnips"

            -- Snippets
            use "SirVer/ultisnips"
```

```
            use {"Allen191819/vim-snippets", rtp = "."}

            -- treesitter
            use {"nvim-treesitter/nvim-treesitter", run = ":TSUpdate"}
            use "nvim-treesitter/playground"

            -- Undotree
            use "mbbill/undotree"

            -- Floaterm
            use "voldikss/vim-floaterm"

            -- Colorscheme
            use {
                  'nvim-lualine/lualine.nvim',
                  requires = {'kyazdani42/nvim-web-devicons', opt = true}
}           use 'marko-cerovac/material.nvim'
        end,
        config = {
            -- Move to lua dir so impatient.nvim can cache it
            compile_path = vim.fn.stdpath("config") .. "/lua/packer_compiled.lua"
        }
    }
)
```

2.6. 几个插件的细节

- nvim-lspconfig

> 1. 手动安装 lsp
> 2. 自动安装 nvim-lsp-installer

- treesitter
- undotree

```
-- ~/.config/nvim/lua/settings.lua
....
vim.o.backupdir = "/home/allen/.config/nvim/tmp/backup"
vim.o.directory = "/home/allen/.config/nvim/tmp/backup"

if vim.fn.has("persistent_undo") then
    vim.o.undofile = true
    vim.o.undodir = "/home/allen/.config/nvim/tmp/undo"
end
....
```

2.7. 更多的插件 😃

https://github.com/Allen191819/awesome-neovim

2.8. 我的 nvim 配置

https://github.com/Allen191819/nvim-lua

2.9. 我的 Archlinux 配置

https://github.com/Allen191819/dotfiles