# ===================================================================

# This file contains the default startup commands for ranger.

# To change them, it is recommended to create the file

# ~/.config/ranger/rc.conf and add your custom commands there.

#

# If you copy this whole file there, you may want to set the environment

# variable RANGER\_LOAD\_DEFAULT\_RC to FALSE to avoid loading it twice.

#

# The purpose of this file is mainly to define keybindings and settings.

# For running more complex python code, please create a plugin in "plugins/" or

# a command in "commands.py".

#

# Each line is a command that will be run before the user interface

# is initialized. As a result, you can not use commands which rely

# on the UI such as :delete or :mark.

# ===================================================================

# ===================================================================

# == Options

# ===================================================================

# Which viewmode should be used? Possible values are:

# miller: Use miller columns which show multiple levels of the hierarchy

# multipane: Midnight-commander like multipane view showing all tabs next

# to each other

set viewmode miller

#set viewmode multipane

# How many columns are there, and what are their relative widths?

set column\_ratios 1,3,4

# Which files should be hidden? (regular expression)

set hidden\_filter ^\.|\.(?:pyc|pyo|bak|swp)$|^lost\+found$|^\_\_(py)?cache\_\_$

# Show hidden files? You can toggle this by typing 'zh'

set show\_hidden true

# Ask for a confirmation when running the "delete" command?

# Valid values are "always", "never", "multiple" (default)

# With "multiple", ranger will ask only if you delete multiple files at once.

set confirm\_on\_delete multiple

# Use non-default path for file preview script?

# ranger ships with scope.sh, a script that calls external programs (see

# README.md for dependencies) to preview images, archives, etc.

#set preview\_script ~/.config/ranger/scope.sh

# Use the external preview script or display simple plain text or image previews?

set use\_preview\_script true

# Automatically count files in the directory, even before entering them?

set automatically\_count\_files true

# Open all images in this directory when running certain image viewers

# like feh or sxiv? You can still open selected files by marking them.

set open\_all\_images true

# Be aware of version control systems and display information.

set vcs\_aware false

# State of the three backends git, hg, bzr. The possible states are

# disabled, local (only show local info), enabled (show local and remote

# information).

set vcs\_backend\_git enabled

set vcs\_backend\_hg disabled

set vcs\_backend\_bzr disabled

# Use one of the supported image preview protocols

set preview\_images true

# Set the preview image method. Supported methods:

#

# \* w3m (default):

# Preview images in full color with the external command "w3mimgpreview"?

# This requires the console web browser "w3m" and a supported terminal.

# It has been successfully tested with "xterm" and "urxvt" without tmux.

#

# \* iterm2:

# Preview images in full color using iTerm2 image previews

# (http://iterm2.com/images.html). This requires using iTerm2 compiled

# with image preview support.

#

# \* urxvt:

# Preview images in full color using urxvt image backgrounds. This

# requires using urxvt compiled with pixbuf support.

#

# \* urxvt-full:

# The same as urxvt but utilizing not only the preview pane but the

# whole terminal window.

set preview\_images\_method w3m

# Use a unicode "..." character to mark cut-off filenames?

set unicode\_ellipsis false

# Show dotfiles in the bookmark preview box?

set show\_hidden\_bookmarks true

# Which colorscheme to use? These colorschemes are available by default:

# default, jungle, snow, solarized

set colorscheme default

# Preview files on the rightmost column?

# And collapse (shrink) the last column if there is nothing to preview?

set preview\_files true

set preview\_directories true

set collapse\_preview true

# Save the console history on exit?

set save\_console\_history true

# Draw the status bar on top of the browser window (default: bottom)

set status\_bar\_on\_top false

# Draw a progress bar in the status bar which displays the average state of all

# currently running tasks which support progress bars?

set draw\_progress\_bar\_in\_status\_bar true

# Draw borders around columns?

set draw\_borders true

# Display the directory name in tabs?

set dirname\_in\_tabs false

# Enable the mouse support?

set mouse\_enabled true

# Display the file size in the main column or status bar?

set display\_size\_in\_main\_column true

set display\_size\_in\_status\_bar true

# Display files tags in all columns or only in main column?

set display\_tags\_in\_all\_columns true

# Set a title for the window?

set update\_title false

# Set the title to "ranger" in the tmux program?

set update\_tmux\_title false

# Shorten the title if it gets long? The number defines how many

# directories are displayed at once, 0 turns off this feature.

set shorten\_title 3

# Show hostname in titlebar?

set hostname\_in\_titlebar true

# Abbreviate $HOME with ~ in the titlebar (first line) of ranger?

set tilde\_in\_titlebar false

# How many directory-changes or console-commands should be kept in history?

set max\_history\_size 20

set max\_console\_history\_size 50

# Try to keep so much space between the top/bottom border when scrolling:

set scroll\_offset 8

# Flush the input after each key hit? (Noticeable when ranger lags)

set flushinput true

# Padding on the right when there's no preview?

# This allows you to click into the space to run the file.

set padding\_right true

# Save bookmarks (used with mX and `X) instantly?

# This helps to synchronize bookmarks between multiple ranger

# instances but leads to \*slight\* performance loss.

# When false, bookmarks are saved when ranger is exited.

set autosave\_bookmarks true

# You can display the "real" cumulative size of directories by using the

# command :get\_cumulative\_size or typing "dc". The size is expensive to

# calculate and will not be updated automatically. You can choose

# to update it automatically though by turning on this option:

set autoupdate\_cumulative\_size false

# Turning this on makes sense for screen readers:

set show\_cursor false

# One of: size, natural, basename, atime, ctime, mtime, type, random

set sort natural

# Additional sorting options

set sort\_reverse false

set sort\_case\_insensitive true

set sort\_directories\_first true

set sort\_unicode false

# Enable this if key combinations with the Alt Key don't work for you.

# (Especially on xterm)

set xterm\_alt\_key false

# Whether to include bookmarks in cd command

set cd\_bookmarks true

# Changes case sensitivity for the cd command tab completion

set cd\_tab\_case sensitive

# Use smart tab completion with less typing? E.g. ":cd /f/b/b<tab>" yields ":cd /foo/bar/baz".

set cd\_tab\_smart false

# Avoid previewing files larger than this size, in bytes. Use a value of 0 to

# disable this feature.

set preview\_max\_size 0

# Add the highlighted file to the path in the titlebar

set show\_selection\_in\_titlebar true

# The delay that ranger idly waits for user input, in milliseconds, with a

# resolution of 100ms. Lower delay reduces lag between directory updates but

# increases CPU load.

set idle\_delay 2000

# When the metadata manager module looks for metadata, should it only look for

# a ".metadata.json" file in the current directory, or do a deep search and

# check all directories above the current one as well?

set metadata\_deep\_search false

# Clear all existing filters when leaving a directory

set clear\_filters\_on\_dir\_change false

# Disable displaying line numbers in main column

set line\_numbers false

# Save tabs on exit

set save\_tabs\_on\_exit false

# Enable scroll wrapping - moving down while on the last item will wrap around to

# the top and vice versa.

set wrap\_scroll false

# Set the global\_inode\_type\_filter to nothing. Possible options: d, f and l for

# directories, files and symlinks respectively.

set global\_inode\_type\_filter

# ===================================================================

# == Local Options

# ===================================================================

# You can set local options that only affect a single directory.

# Examples:

# setlocal path=~/downloads sort mtime

# ===================================================================

# == Command Aliases in the Console

# ===================================================================

alias e edit

alias q quit

alias q! quit!

alias qa quitall

alias qa! quitall!

alias qall quitall

alias qall! quitall!

alias setl setlocal

alias filter scout -prt

alias find scout -aeit

alias mark scout -mr

alias unmark scout -Mr

alias search scout -rs

alias search\_inc scout -rts

alias travel scout -aefklst

# ===================================================================

# == Define keys for the browser

# ===================================================================

# Basic

map Q quit!

map q quit

copymap q ZZ ZQ

map R reload\_cwd

map F set freeze\_files!

map <C-r> reset

map <C-l> redraw\_window

map <C-c> abort

map <esc> change\_mode normal

map ~ set viewmode!

map i display\_file

map ? help

map W display\_log

map w taskview\_open

map S shell $SHELL

map : console

map ; console

map ! console shell%space

map @ console -p6 shell %%s

map # console shell -p%space

map s console shell%space

map r chain draw\_possible\_programs; console open\_with%%space

map f console find%space

map cd console cd%space

# Change the line mode

map Mf linemode filename

map Mi linemode fileinfo

map Mm linemode mtime

map Mp linemode permissions

map Ms linemode sizemtime

map Mt linemode metatitle

# Tagging / Marking

map t tag\_toggle

map ut tag\_remove

map "<any> tag\_toggle tag=%any

map <Space> mark\_files toggle=True

map v mark\_files all=True toggle=True

map uv mark\_files all=True val=False

map V toggle\_visual\_mode

map uV toggle\_visual\_mode reverse=True

# For the nostalgics: Midnight Commander bindings

map <F1> help

map <F2> console rename%space

map <F3> display\_file

map <F4> edit

map <F5> copy

map <F6> cut

map <F7> console mkdir%space

map <F8> console delete

map <F10> exit

# In case you work on a keyboard with dvorak layout

map <UP> move up=1

map <DOWN> move down=1

map <LEFT> move left=1

map <RIGHT> move right=1

map <HOME> move to=0

map <END> move to=-1

map <PAGEDOWN> move down=1 pages=True

map <PAGEUP> move up=1 pages=True

map <CR> move right=1

#map <DELETE> console delete

map <INSERT> console touch%space

# VIM-like

copymap <UP> k

copymap <DOWN> j

copymap <LEFT> h

copymap <RIGHT> l

copymap <HOME> gg

copymap <END> G

copymap <PAGEDOWN> <C-F>

copymap <PAGEUP> <C-B>

map J move down=0.5 pages=True

map K move up=0.5 pages=True

copymap J <C-D>

copymap K <C-U>

# Jumping around

map H history\_go -1

map L history\_go 1

map ] move\_parent 1

map [ move\_parent -1

map } traverse

map ) jump\_non

map gh cd ~

map ge cd /etc

map gu cd /usr

map gd cd /dev

map gl cd -r .

map gL cd -r %f

map go cd /opt

map gv cd /var

map gm cd /media

map gM cd /mnt

map gs cd /srv

map gt cd /tmp

map gr cd /

map gR eval fm.cd(ranger.RANGERDIR)

map g/ cd /

map g? cd /usr/share/doc/ranger

# External Programs

map E edit

map du shell -p du --max-depth=1 -h --apparent-size

map dU shell -p du --max-depth=1 -h --apparent-size | sort -rh

map yp yank path

map yd yank dir

map yn yank name

# Filesystem Operations

map = chmod

map cw console rename%space

map a rename\_append

map A eval fm.open\_console('rename ' + fm.thisfile.relative\_path.replace("%", "%%"))

map I eval fm.open\_console('rename ' + fm.thisfile.relative\_path.replace("%", "%%"), position=7)

map pp paste

map po paste overwrite=True

map pP paste append=True

map pO paste overwrite=True append=True

map pl paste\_symlink relative=False

map pL paste\_symlink relative=True

map phl paste\_hardlink

map pht paste\_hardlinked\_subtree

map dD console delete

map dd cut

map ud uncut

map da cut mode=add

map dr cut mode=remove

map dt cut mode=toggle

map yy copy

map uy uncut

map ya copy mode=add

map yr copy mode=remove

map yt copy mode=toggle

# Temporary workarounds

map dgg eval fm.cut(dirarg=dict(to=0), narg=quantifier)

map dG eval fm.cut(dirarg=dict(to=-1), narg=quantifier)

map dj eval fm.cut(dirarg=dict(down=1), narg=quantifier)

map dk eval fm.cut(dirarg=dict(up=1), narg=quantifier)

map ygg eval fm.copy(dirarg=dict(to=0), narg=quantifier)

map yG eval fm.copy(dirarg=dict(to=-1), narg=quantifier)

map yj eval fm.copy(dirarg=dict(down=1), narg=quantifier)

map yk eval fm.copy(dirarg=dict(up=1), narg=quantifier)

# Searching

map / console search%space

map n search\_next

map N search\_next forward=False

map ct search\_next order=tag

map cs search\_next order=size

map ci search\_next order=mimetype

map cc search\_next order=ctime

map cm search\_next order=mtime

map ca search\_next order=atime

# Tabs

map <C-n> tab\_new

map <C-w> tab\_close

map <TAB> tab\_move 1

map <S-TAB> tab\_move -1

map <A-Right> tab\_move 1

map <A-Left> tab\_move -1

map gt tab\_move 1

map gT tab\_move -1

map gn tab\_new

map gc tab\_close

map uq tab\_restore

map <a-1> tab\_open 1

map <a-2> tab\_open 2

map <a-3> tab\_open 3

map <a-4> tab\_open 4

map <a-5> tab\_open 5

map <a-6> tab\_open 6

map <a-7> tab\_open 7

map <a-8> tab\_open 8

map <a-9> tab\_open 9

#IMAGES

map bg shell wal -i %f

# Sorting

map or set sort\_reverse!

map oz set sort=random

map os chain set sort=size; set sort\_reverse=False

map ob chain set sort=basename; set sort\_reverse=False

map on chain set sort=natural; set sort\_reverse=False

map om chain set sort=mtime; set sort\_reverse=False

map oc chain set sort=ctime; set sort\_reverse=False

map oa chain set sort=atime; set sort\_reverse=False

map ot chain set sort=type; set sort\_reverse=False

map oe chain set sort=extension; set sort\_reverse=False

map oS chain set sort=size; set sort\_reverse=True

map oB chain set sort=basename; set sort\_reverse=True

map oN chain set sort=natural; set sort\_reverse=True

map oM chain set sort=mtime; set sort\_reverse=True

map oC chain set sort=ctime; set sort\_reverse=True

map oA chain set sort=atime; set sort\_reverse=True

map oT chain set sort=type; set sort\_reverse=True

map oE chain set sort=extension; set sort\_reverse=True

map dc get\_cumulative\_size

# Settings

map zc set collapse\_preview!

map zd set sort\_directories\_first!

map zh set show\_hidden!

map <C-h> set show\_hidden!

map zI set flushinput!

map zi set preview\_images!

map zm set mouse\_enabled!

map zp set preview\_files!

map zP set preview\_directories!

map zs set sort\_case\_insensitive!

map zu set autoupdate\_cumulative\_size!

map zv set use\_preview\_script!

map zf console filter%space

copymap zf zz

# Bookmarks

map `<any> enter\_bookmark %any

map '<any> enter\_bookmark %any

map m<any> set\_bookmark %any

map um<any> unset\_bookmark %any

map m<bg> draw\_bookmarks

copymap m<bg> um<bg> `<bg> '<bg>

# Generate all the chmod bindings with some python help:

eval for arg in "rwxXst": cmd("map +u{0} shell -f chmod u+{0} %s".format(arg))

eval for arg in "rwxXst": cmd("map +g{0} shell -f chmod g+{0} %s".format(arg))

eval for arg in "rwxXst": cmd("map +o{0} shell -f chmod o+{0} %s".format(arg))

eval for arg in "rwxXst": cmd("map +a{0} shell -f chmod a+{0} %s".format(arg))

eval for arg in "rwxXst": cmd("map +{0} shell -f chmod u+{0} %s".format(arg))

eval for arg in "rwxXst": cmd("map -u{0} shell -f chmod u-{0} %s".format(arg))

eval for arg in "rwxXst": cmd("map -g{0} shell -f chmod g-{0} %s".format(arg))

eval for arg in "rwxXst": cmd("map -o{0} shell -f chmod o-{0} %s".format(arg))

eval for arg in "rwxXst": cmd("map -a{0} shell -f chmod a-{0} %s".format(arg))

eval for arg in "rwxXst": cmd("map -{0} shell -f chmod u-{0} %s".format(arg))

# ===================================================================

# == Define keys for the console

# ===================================================================

# Note: Unmapped keys are passed directly to the console.

# Basic

cmap <tab> eval fm.ui.console.tab()

cmap <s-tab> eval fm.ui.console.tab(-1)

cmap <ESC> eval fm.ui.console.close()

cmap <CR> eval fm.ui.console.execute()

cmap <C-l> redraw\_window

copycmap <ESC> <C-c>

copycmap <CR> <C-j>

# Move around

cmap <up> eval fm.ui.console.history\_move(-1)

cmap <down> eval fm.ui.console.history\_move(1)

cmap <left> eval fm.ui.console.move(left=1)

cmap <right> eval fm.ui.console.move(right=1)

cmap <home> eval fm.ui.console.move(right=0, absolute=True)

cmap <end> eval fm.ui.console.move(right=-1, absolute=True)

cmap <a-left> eval fm.ui.console.move\_word(left=1)

cmap <a-right> eval fm.ui.console.move\_word(right=1)

# Line Editing

cmap <backspace> eval fm.ui.console.delete(-1)

cmap <delete> eval fm.ui.console.delete(0)

cmap <C-w> eval fm.ui.console.delete\_word()

cmap <A-d> eval fm.ui.console.delete\_word(backward=False)

cmap <C-k> eval fm.ui.console.delete\_rest(1)

cmap <C-u> eval fm.ui.console.delete\_rest(-1)

cmap <C-y> eval fm.ui.console.paste()

# And of course the emacs way

copycmap <up> <C-p>

copycmap <down> <C-n>

copycmap <left> <C-b>

copycmap <right> <C-f>

copycmap <home> <C-a>

copycmap <end> <C-e>

copycmap <delete> <C-d>

copycmap <backspace> <C-h>

# Note: There are multiple ways to express backspaces. <backspace> (code 263)

# and <backspace2> (code 127). To be sure, use both.

copycmap <backspace> <backspace2>

# This special expression allows typing in numerals:

cmap <allow\_quantifiers> false

# ===================================================================

# == Pager Keybindings

# ===================================================================

# Movement

pmap <down> pager\_move down=1

pmap <up> pager\_move up=1

pmap <left> pager\_move left=4

pmap <right> pager\_move right=4

pmap <home> pager\_move to=0

pmap <end> pager\_move to=-1

pmap <pagedown> pager\_move down=1.0 pages=True

pmap <pageup> pager\_move up=1.0 pages=True

pmap <C-d> pager\_move down=0.5 pages=True

pmap <C-u> pager\_move up=0.5 pages=True

copypmap <UP> k <C-p>

copypmap <DOWN> j <C-n> <CR>

copypmap <LEFT> h

copypmap <RIGHT> l

copypmap <HOME> g

copypmap <END> G

copypmap <C-d> d

copypmap <C-u> u

copypmap <PAGEDOWN> n f <C-F> <Space>

copypmap <PAGEUP> p b <C-B>

# Basic

pmap <C-l> redraw\_window

pmap <ESC> pager\_close

copypmap <ESC> q Q i <F3>

pmap E edit\_file

# ===================================================================

# == Taskview Keybindings

# ===================================================================

# Movement

tmap <up> taskview\_move up=1

tmap <down> taskview\_move down=1

tmap <home> taskview\_move to=0

tmap <end> taskview\_move to=-1

tmap <pagedown> taskview\_move down=1.0 pages=True

tmap <pageup> taskview\_move up=1.0 pages=True

tmap <C-d> taskview\_move down=0.5 pages=True

tmap <C-u> taskview\_move up=0.5 pages=True

copytmap <UP> k <C-p>

copytmap <DOWN> j <C-n> <CR>

copytmap <HOME> g

copytmap <END> G

copytmap <C-u> u

copytmap <PAGEDOWN> n f <C-F> <Space>

copytmap <PAGEUP> p b <C-B>

# Changing priority and deleting tasks

tmap J eval -q fm.ui.taskview.task\_move(-1)

tmap K eval -q fm.ui.taskview.task\_move(0)

tmap dd eval -q fm.ui.taskview.task\_remove()

tmap <pagedown> eval -q fm.ui.taskview.task\_move(-1)

tmap <pageup> eval -q fm.ui.taskview.task\_move(0)

tmap <delete> eval -q fm.ui.taskview.task\_remove()

# Basic

tmap <C-l> redraw\_window

tmap <ESC> taskview\_close

copytmap <ESC> q Q w <C-c>