

Dotfiles

- [Sway Config \(Archive\)](#)
- [Automatic Cloudflare DNS Update](#)

Sway Config (Archive)

This sway config is here for record purposes and may not be up to date. Visit my [dotfiles Github repository](#) to see what I currently use.

Sway Config

```
# Default config for sway
#
# Copy this to ~/.config/sway/config and edit it to your liking.
#
# Read `man 5 sway` for a complete reference.

### Variables
#
# Logo key. Use Mod1 for Alt.
set $mod Mod4
# Home row direction keys, like vim
set $left h
set $down j
set $up k
set $right l
# Your preferred terminal emulator
#set $term kitty
set $term foot -f "Noto Sans Mono:size=12"
# Your preferred application launcher
# Note: pass the final command to swaymsg so that the resulting window can be opened
# on the original workspace that the command was run on.
#set $menu dmenu_path | dmenu | xargs swaymsg exec --
set $menu bemenu-run --fn "Source Code Pro" 20 | xargs swaymsg exec --

### Output configuration
#
# Default wallpaper (more resolutions are available in /usr/share/backgrounds/sway/)
#output * bg /usr/share/backgrounds/sway/Sway_Wallpaper_Blue_1920x1080.png fill
```

```

#
# Example configuration:
#
# output HDMI-A-1 resolution 1920x1080 position 1920,0
#
# You can get the names of your outputs by running: swaymsg -t get_outputs

### Idle configuration
#
# Example configuration:
#
# exec swayidle -w \
#     timeout 300 'swaylock -f -c 000000' \
#     timeout 600 'swaymsg "output * power off"' resume 'swaymsg "output * power on"' \
#     before-sleep 'swaylock -f -c 000000'
#
# This will lock your screen after 300 seconds of inactivity, then turn off
# your displays after another 300 seconds, and turn your screens back on when
# resumed. It will also lock your screen before your computer goes to sleep.

### Input configuration
#
# Example configuration:
#
# input "2:14:SynPS/2_Synaptics_TouchPad" {
#     dwt enabled
#     tap enabled
#     natural_scroll enabled
#     middle_emulation enabled
# }
#
# You can get the names of your inputs by running: swaymsg -t get_inputs
# Read `man 5 sway-input` for more information about this section.

input type:touchpad {
    ␣tap enabled
    ␣natural_scroll enabled
    ␣accel_profile "flat"
}

```

Key bindings

#

Basics:

#

Start a terminal

bindsym \$mod+Return exec \$term

Kill focused window

bindsym \$mod+q kill

Start your launcher

bindsym \$mod+Shift+d exec \$menu

Drag floating windows by holding down \$mod and left mouse button.

Resize them with right mouse button + \$mod.

Despite the name, also works for non-floating windows.

Change normal to inverse to use left mouse button for resizing and right

mouse button for dragging.

floating_modifier \$mod normal

Reload the configuration file

bindsym \$mod+Shift+c reload

Exit sway (logs you out of your Wayland session)

bindsym \$mod+Shift+e exec swaynag -t warning -m 'You pressed the exit shortcut. Do you really want to exit sway? This will end your Wayland session.' -B 'Yes, exit sway' 'swaymsg exit'

#

Moving around:

#

Move your focus around

bindsym \$mod+\$left focus left

bindsym \$mod+\$down focus down

bindsym \$mod+\$up focus up

bindsym \$mod+\$right focus right

Or use \$mod+[up|down|left|right]

bindsym \$mod+Left focus left

bindsym \$mod+Down focus down

bindsym \$mod+Up focus up

```
bindsym $mod+Right focus right
```

```
# Move the focused window with the same, but add Shift
```

```
bindsym $mod+Shift+$left move left
```

```
bindsym $mod+Shift+$down move down
```

```
bindsym $mod+Shift+$up move up
```

```
bindsym $mod+Shift+$right move right
```

```
# Ditto, with arrow keys
```

```
bindsym $mod+Shift+Left move left
```

```
bindsym $mod+Shift+Down move down
```

```
bindsym $mod+Shift+Up move up
```

```
bindsym $mod+Shift+Right move right
```

```
#
```

```
# Workspaces:
```

```
#
```

```
# Switch to workspace
```

```
bindsym $mod+1 workspace number 1
```

```
bindsym $mod+2 workspace number 2
```

```
bindsym $mod+3 workspace number 3
```

```
bindsym $mod+4 workspace number 4
```

```
bindsym $mod+5 workspace number 5
```

```
bindsym $mod+6 workspace number 6
```

```
bindsym $mod+7 workspace number 7
```

```
bindsym $mod+8 workspace number 8
```

```
bindsym $mod+9 workspace number 9
```

```
bindsym $mod+0 workspace number 10
```

```
# Move focused container to workspace
```

```
bindsym $mod+Shift+1 move container to workspace number 1, workspace number 1
```

```
bindsym $mod+Shift+2 move container to workspace number 2, workspace number 2
```

```
bindsym $mod+Shift+3 move container to workspace number 3, workspace number 3
```

```
bindsym $mod+Shift+4 move container to workspace number 4, workspace number 4
```

```
bindsym $mod+Shift+5 move container to workspace number 5, workspace number 5
```

```
bindsym $mod+Shift+6 move container to workspace number 6, workspace number 6
```

```
bindsym $mod+Shift+7 move container to workspace number 7, workspace number 7
```

```
bindsym $mod+Shift+8 move container to workspace number 8, workspace number 8
```

```
bindsym $mod+Shift+9 move container to workspace number 9, workspace number 9
```

```
bindsym $mod+Shift+0 move container to workspace number 10, workspace number 10
```

```
# Note: workspace numbers can have any name you want, not just numbers.
```

```
# We just use 1-10 as the default.
```

```
␣# Set monitor variables
␣set $internal_monitor eDP-1
␣set $second_monitor DP-1

␣# Assign workspaces to seperate monitors

␣output $internal_monitor pos 0 0
    output $second_monitor pos 1080 0

␣workspace 1 output $internal_monitor
␣workspace 2 output $internal_monitor
␣workspace 3 output $internal_monitor
␣workspace 4 output $internal_monitor
␣workspace 5 output $internal_monitor
␣workspace 6 output $second_monitor
␣workspace 7 output $second_monitor
␣workspace 8 output $second_monitor
␣workspace 9 output $second_monitor
␣workspace 10 output $second_monitor

#
# Layout stuff:
#
# You can "split" the current object of your focus with
# $mod+b or $mod+v, for horizontal and vertical splits
# respectively.
bindsym $mod+n splith
bindsym $mod+v splitv

# Switch the current container between different layout styles
bindsym $mod+s layout stacking
bindsym $mod+z layout tabbed
bindsym $mod+e layout toggle split

# Make the current focus fullscreen
bindsym $mod+f fullscreen

# Toggle the current focus between tiling and floating mode
```

```
bindsym $mod+Shift+space floating toggle
```

```
# Swap focus between the tiling area and the floating area
```

```
bindsym $mod+space focus mode_toggle
```

```
# Move focus to the parent container
```

```
bindsym $mod+a focus parent
```

```
#
```

```
# Scratchpad:
```

```
#
```

```
# Sway has a "scratchpad", which is a bag of holding for windows.
```

```
# You can send windows there and get them back later.
```

```
# Move the currently focused window to the scratchpad
```

```
bindsym $mod+Shift+minus move scratchpad
```

```
# Show the next scratchpad window or hide the focused scratchpad window.
```

```
# If there are multiple scratchpad windows, this command cycles through them.
```

```
bindsym $mod+minus scratchpad show
```

```
#
```

```
# Resizing containers:
```

```
#
```

```
mode "resize" {
```

```
    # left will shrink the containers width
```

```
    # right will grow the containers width
```

```
    # up will shrink the containers height
```

```
    # down will grow the containers height
```

```
    bindsym $left resize shrink width 100px
```

```
    bindsym $down resize grow height 100px
```

```
    bindsym $up resize shrink height 100px
```

```
    bindsym $right resize grow width 100px
```

```
# Ditto, with arrow keys
```

```
bindsym Left resize shrink width 100px
```

```
bindsym Down resize grow height 100px
```

```
bindsym Up resize shrink height 100px
```

```
bindsym Right resize grow width 100px
```

```
# Return to default mode
```

```
bindsym Return mode "default"
bindsym Escape mode "default"
}
bindsym $mod+r mode "resize"

#
# Status Bar:
#
# Read `man 5 sway-bar` for more information about this section.
bar {
    position top

    # When the status_command prints a new line to stdout, swaybar updates.
    # The default just shows the current date and time.
    #status_command while date +%Y-%m-%d %l:%M:%S %p'; do sleep 1; done
    status_command while ~/.config/sway/status.sh; do sleep 1; done

    colors {
        statusline #ffffff
        background #323232aa
        inactive_workspace #323232aa #323232aa #000000
    }
}

font "Source Code Pro"
pango_markup enabled
separator_symbol "$"
}

#
# Border Controls
#

default_border pixel
bindsym $mod+Shift+b exec swaymsg border toggle

#
# Startup Applications
#
```



```
exec swaybg -i $(find ~/Pictures/wallpapers -type f | shuf -n1) -m fill &
```

```
#
```

```
# Power Menu
```

```
#
```

```
bindsym $mod+x mode "(k) lock, (l) logout, (r) reboot, (s) shutdown"
```

```
mode "(k) lock, (l) logout, (r) reboot, (s) shutdown" {
```

```
    bindsym k exec swaylock -c 000000, mode "default"
```

```
    bindsym l exec swaymsg exit, mode "default"
```

```
    bindsym r exec reboot, mode "default"
```

```
    bindsym s exec shutdown now, mode "default"
```

```
    bindsym Return mode "default"
```

```
    bindsym Escape mode "default"
```

```
}
```

```
#
```

```
# Special Keybindings
```

```
#
```

```
# Map Caps Lock to Left Control
```

```
input type:keyboard {
```

```
    xkb_options caps:ctrl_modifier
```

```
}
```

```
# Applications
```

```
bindsym $mod+w exec flatpak run io.gitlab.librewolf-community
```

```
# Volume
```

```
bindsym XF86AudioRaiseVolume exec pactl set-sink-volume @DEFAULT_SINK@ +5% && pactl set-sink-mute @DEFAULT_SINK@ 0
```

```
bindsym XF86AudioLowerVolume exec pactl set-sink-volume @DEFAULT_SINK@ -5% && pactl set-sink-mute @DEFAULT_SINK@ 0
```

```
bindsym XF86AudioMute exec pactl set-sink-mute @DEFAULT_SINK@ toggle
```

```
bindsym XF86AudioMicMute exec pactl set-source-mute @DEFAULT_SOURCE@ toggle
```

```
# Media
```

```
bindsym XF86AudioPlay exec playerctl play-pause
```

```
bindsym XF86AudioNext exec playerctl next
```

```
bindsym XF86AudioPrev exec playerctl previous
```

```
bindsym XF86Tools exec playerctl play-pause
```

```
bindsym XF86Favorites exec playerctl next
```

```
bindsym XF86Bluetooth exec playerctl previous
```

```
#Sticky Window
```

```
bindsym $mod+Shift+Return sticky toggle
```

```
# Brightness
```

```
bindsym XF86MonBrightnessUp exec brightnessctl set +10%
```

```
bindsym XF86MonBrightnessDown exec brightnessctl set --min-value=1 10%-
```

```
# Rotation
```

```
exec swaymsg input 1386:20824:Wacom_Pen_and_multitouch_sensor_Finger map_to_output  
$internal_monitor
```

```
bindsym $mod+Mod1+k exec swaymsg output eDP-1 transform normal && swaymsg input  
1386:20824:Wacom_Pen_and_multitouch_sensor_Finger map_to_output $internal_monitor && swaymsg  
input 1386:20824:Wacom_Pen_and_multitouch_sensor_Pen map_to_output $internal_monitor
```

```
bindsym $mod+Mod1+j exec swaymsg output eDP-1 transform 180 && swaymsg input  
1386:20824:Wacom_Pen_and_multitouch_sensor_Finger map_to_output $internal_monitor && swaymsg  
input 1386:20824:Wacom_Pen_and_multitouch_sensor_Pen map_to_output $internal_monitor
```

```
bindsym $mod+Mod1+l exec swaymsg output eDP-1 transform 90 && swaymsg input  
1386:20824:Wacom_Pen_and_multitouch_sensor_Finger map_to_output $internal_monitor && swaymsg  
input 1386:20824:Wacom_Pen_and_multitouch_sensor_Pen map_to_output $internal_monitor
```

```
bindsym $mod+Mod1+h exec swaymsg output eDP-1 transform 270 && swaymsg input  
1386:20824:Wacom_Pen_and_multitouch_sensor_Finger map_to_output $internal_monitor && swaymsg  
input 1386:20824:Wacom_Pen_and_multitouch_sensor_Pen map_to_output $internal_monitor
```

```
include /etc/sway/config.d/*
```

status.sh

```
date=$(date +%Y-%m-%d %l:%M:%S %p')
```

```
wifi_details=$(nmcli -t -f active,ssid,freq,rate,signal dev wifi list --rescan no | grep -e "^yes:" | sed  
"s/yes://g;s:/ /g")
```

```
wired_info=$(ip addr | grep 'state UP' -A2 | tail -n1 | awk '{print $2}' | cut -f1 -d'/')
if [ -z "$wifi_details" ]; then
network_info="Wired $wired_info"
else
network_info="$wifi_details%"
fi

SINK=0
current_brightness=$(brightnessctl | grep Current | cut -d '(' -f2 | cut -d '%' -f1)
current_sink_muted=$(pactl list sinks | grep '^[:space:]]Mute:' | head -n $(( $SINK + 1 )) | tail -n 1 | cut -d
":" -f2 | sed -e 's/^[ \t]*//')

if [ "$current_sink_muted" = "yes" ]; then
current_volume="muted"
else
current_volume=$(pactl list sinks | grep '^[:space:]]Volume:' | head -n $(( $SINK + 1 )) | tail -n 1 | sed -e
's,.* \([0-9][0-9]*\)%.*,\1,')
fi
current_mem=$(free | grep Mem | awk '{print $3/$2 * 100.0}')
current_mem_rounded=`printf "%.2f" $current_mem`
current_cpu=$(100-$(vmstat 1 2|tail -1|awk '{print $15}'))
current_cpu_rounded=`printf "%.2f" $current_cpu`
battery_level=$(cat /sys/class/power_supply/BAT0/capacity)
battery_status=$(cat /sys/class/power_supply/BAT0/status)
if [ "$battery_status" = "Charging" ]; then
battery_prefix="+"
fi

if [ "$battery_level" -le "15" ]; then
battery_color="red"
else
battery_color="springgreen"
fi

echo "<span foreground='limegreen' weight='bold'>$network_info</span> <span foreground='palegreen'
weight='bold'>CPU:$current_cpu%</span> <span foreground='mediumspringgreen'
weight='bold'>MEM:$current_mem_rounded</span> <span foreground='$battery_color'
weight='bold'>BATT:$battery_prefix$battery_level</span> <span foreground='mediumseagreen'
weight='bold'>LUM:$current_brightness</span> <span foreground='yellowgreen'"

```

weight='bold'>VOL:\$current_volume <span foreground='lightseagreen'
weight='bold'>\$date "

Automatic Cloudflare DNS Update

Create a script that queries the current IP address, then updates it using Cloudflare's API

```
# Acquire the permanent, publically addressable ipv6 address of the eth0 adapter
ip -6 addr show dev eth0 mngtmpaddr | grep -oE "([0-9a-fA-F]{1,4}:){7}[0-9a-fA-F]{1,4}"
```

```
# Acquire the ipv4 address
curl https://api.ipify.org
```

```
import http.client requests json
```

```
ipv4_data = json.loads(requests.get('https://api.ipify.org?format=json').text)
```

```
ipv4_addr = ipv4_data["ip"]
```

```
# Acquire the permanent, publically addressable ipv6 address of the eth0 adapter
```

```
ipv6_addr = subprocess.run('ip -6 addr show dev eth0 mngtmpaddr | grep -oE "([0-9a-fA-F]{1,4}:){7}[0-9a-fA-F]{1,4}"')
```

```
# Acquire zones
```

```
conn = http.client.HTTPSConnection("api.cloudflare.com")
```

```
headers = {
```

```
    'Content-Type': "application/json",
```

```
    'Authorization': "Bearer <API_TOKEN>"
```

```
}
```

```
conn.request("GET", "/client/v4/zones", headers=headers)
```

```
res = conn.getresponse()
```

```
data = res.read()
```

```
# Modify all DNS records
```

```
...
```

