Hyprland Dynamic tiling Wayland compositor

CONCEPTS

Mod is Super in the stock config (local mainMod = "SUPER"). Every key below is a default bind from the example hyprland.lua and is yours to change.

Since 0.55 the config is Lua (hyprland.lua); the old hyprlang syntax is deprecated. A bind is hl.bind("SUPER + Q", hl.dsp.exec_cmd("kitty")), keys joined with ' + '.

Monitor an output
- **Workspace** numbered or named
  - **Window** tiled, floating or grouped

Dispatchers are actions (hl.dsp.*) that you pass to hl.bind(), or to hl.dispatch() inside a Lua function. Lua binds run on the compositor loop, so never block in them.

LAUNCH AND CLOSE

Mod + Q Open terminal

Mod + R Open app launcher

Mod + E Open file manager

Mod + C
Close focused window

Mod + M Quit Hyprland

FOCUS AND MOVE

Mod + Arrows
Focus left / right / up / down

`hl.dsp.window.move({ direction = "left" })`
Move window in a direction

`hl.dsp.window.swap({ direction = "left" })`
Swap with the neighbour

`hl.dsp.window.cycle_next()`
Focus next window

`hl.dsp.focus({ last = true })`
Focus the previous window

`hl.dsp.focus({ monitor = "right" })`
Focus another monitor

FOCUS AND MOVE (cont.)

`hl.dsp.window.move({ monitor = "right" })`
Send window to another monitor

LAYOUT AND STATE

Mod + V Toggle floating

Mod + J Toggle split direction

`hl.dsp.window.fullscreen()`
Toggle fullscreen

`hl.dsp.window.resize({ x = 10, y = 0, relative = true })`
Resize by pixels

`hl.dsp.group.toggle()`
Make / dissolve a tabbed group

WORKSPACES

Mod + 1-9
Switch to workspace N

Mod + Shift + 1-9
Move window to workspace N

Mod + Scroll
Next / previous open workspace

Mod + S
Toggle scratchpad workspace

Mod + Shift + S
Send window to scratchpad

`focus({ workspace = "name:<name>" })`
Go to a named workspace

`focus({ workspace = "previous" })`
Back to previous workspace

`focus({ workspace = "e+1" })`
Next open workspace

MOUSE

Mod + LMB Drag a window

Mod + RMB Resize a window

BIND SYNTAX

`hl.bind(<keys>, <dispatcher>, { <flags> })`
Define a keybind

`hl.bind("SUPER + code:28", ...)`
Bind by keycode

BIND SYNTAX (cont.)

`hl.unbind("SUPER + O")`
Remove a bind

`hl.dsp.exec_cmd(<cmd>)`
Run a shell command

`{ repeating = true }`
Repeat while held

`{ locked = true }`
Work on the lock screen

`{ release = true }`
Fire on key release

SUBMAPS

`hl.dsp.submap("<name>")`
Enter a submap

`hl.dsp.submap("reset")`
Leave to the global binds

`hl.define_submap("<name>", function() ... end)`
Define the binds in a submap

MEDIA AND GLOBAL

`XF86AudioRaiseVolume`
Volume key bind with wpctl

HYPRCTL

`hyprctl reload`
Reload the config

`hyprctl dispatch '<lua dispatcher>'`
Run a dispatcher now

`hyprctl clients`
List windows with properties

`hyprctl monitors`
List monitors

`hyprctl activewindow`
Show the focused window

`hyprctl workspaces`
List workspaces

`hyprctl binds`
List all registered binds

`hyprctl devices`
List keyboards, mice and switches

`hyprctl configerrors`
Show config parse errors

`hyprctl eval '<lua>'`
Run Lua in the compositor

`hyprctl -j <command>`
JSON output