Qtile Python tiling WM

Screen one per monitor, holds the bar
Group workspace, shown on one screen
Layout tiling rule of the group
Window one client

CONCEPTS

Super is mod4 in the default config.py (~/.config/qtile/config.py). Every binding on this sheet is a default you can change.

Config is plain Python. Keys call lazy objects (lazy.window.kill()): they are queued and only run on the key press.

Groups are not tied to a screen: a group is pulled to the focused screen when you switch to it.

FOCUS AND MOVE

Super + H / Super + L
Focus left / right

Super + J / Super + K
Focus down / up

Super + Space
Focus the other pane

Super + Shift + H / Super + Shift + L
Move window left / right

FOCUS AND MOVE (cont.)

Super + Shift + J / Super + Shift + K
Move window down / up

Super + Ctrl + H / Super + Ctrl + L
Grow window left / right

Super + Ctrl + J / Super + Ctrl + K
Grow window down / up

Super + N
Reset all window sizes

WINDOWS

Super + Enter
Launch a terminal

Super + W
Kill the focused window

Super + F
Toggle fullscreen

Super + T Toggle floating

Super + R
Open the prompt widget

Super + Button1 › drag
Move a floating window

WINDOWS (cont.)

Super + Button3 › drag
Resize a floating window

GROUPS AND LAYOUTS

Super + 1-9
Switch to group

Super + Shift + 1-9
Send window to group and follow

Super + Tab Next layout

Layouts (names for config)

Layout	Idea
`Columns`	Default: columns of stacked windows
`Max`	One window fills the screen
`MonadTall`	Main pane left, rest stacked right
`MonadWide`	Main pane top, rest below
`Bsp`	Binary tree splits
`TreeTab`	Tabs listed in a side tree

Columns and Max are active by default; the others are commented out in the default layouts list.

SESSION

Super + Ctrl + R
Reload the config

Super + Ctrl + Q
Shut down Qtile

LAZY COMMANDS

`lazy.spawn("<app>")`
Run an application

`lazy.spawncmd()`
Open the prompt on the bar

`lazy.reload_config()`
Reload the config

`lazy.restart()`
Restart (X11 keeps windows)

`lazy.next_layout()` / `lazy.prev_layout()`
Cycle layouts of the group

`lazy.screen.next_group()` / `prev_group()`
Move to adjacent group

`lazy.screen.toggle_group()`
Back to the last group

`lazy.group["<name>"].toscreen()`
Show a group

LAZY COMMANDS (cont.)

`lazy.window.kill()`
Close the focused window

`lazy.window.togroup("<name>")`
Move window to a group

`lazy.window.toggle_floating()`
Float or tile the window

`lazy.window.toggle_fullscreen()`
Fullscreen or restore

CLI

`qtile start`
Start a session (run from .xinitrc)

`qtile start -b wayland`
Start with the Wayland backend

`qtile start -c <file>`
Use another config file

`qtile check`
Check the config for errors

`qtile shell`
Interactive command shell

`qtile cmd-obj`
List root commands