```
-- Standard awesome library
local gears = require("gears")
local awful = require("awful")
awful.rules = require("awful.rules")
require("awful.autofocus")
-- Widget and layout library
local wibox = require("wibox")
-- Theme handling library
local beautiful = require("beautiful")
-- Notification library
local naughty = require("naughty")
local menubar = require("menubar")
-- Vicious library
vicious = require ("vicious")

-- {{{ Error handling
-- Check if awesome encountered an error during startup and fell back to
-- another config (This code will only ever execute for the fallback config)
if awesome.startup_errors then
    naughty.notify({ preset = naughty.config.presets.critical,
                     title = "Oops, there were errors during startup!",
                     text = awesome.startup_errors })
end

-- Handle runtime errors after startup
do
    local in_error = false
    awesome.connect_signal("debug::error", function (err)
        -- Make sure we don't go into an endless error loop
        if in_error then return end
        in_error = true

        naughty.notify({ preset = naughty.config.presets.critical,
                         title = "Oops, an error happened!",
                         text = err })
        in_error = false
    end)
end
-- }}}

-- {{{ Variable definitions
-- Themes define colours, icons, font and wallpapers.
beautiful.init("/usr/share/awesome/themes/default/theme.lua")

-- This is used later as the default terminal and editor to run.
terminal = "xterm"
editor = os.getenv("EDITOR") or "vim"
editor_cmd = terminal .. " -e " .. editor

-- Default modkey.
-- Usually, Mod4 is the key with a logo between Control and Alt.
-- If you do not like this or do not have such a key,
-- I suggest you to remap Mod4 to another key using xmodmap or other tools.
-- However, you can use another modifier like Mod1, but it may interact with
others.
modkey = "Mod4"

-- Table of layouts to cover with awful.layout.inc, order matters.
```

```
local layouts =
{
    awful.layout.suit.tile,
    awful.layout.suit.tile.left,
    awful.layout.suit.tile.bottom,
    awful.layout.suit.tile.top,
    awful.layout.suit.fair,
    awful.layout.suit.fair.horizontal,
    awful.layout.suit.spiral,
    awful.layout.suit.spiral.dwindle,
    awful.layout.suit.max,
    awful.layout.suit.max.fullscreen,
    awful.layout.suit.magnifier,
    awful.layout.suit.floating
}
-- }}}

-- {{{ Wallpaper
if beautiful.wallpaper then
    for s = 1, screen.count() do
        gears.wallpaper.maximized(beautiful.wallpaper, s, true)
      --gears.wallpaper.maximized("/home/cheon/Pictures/wallpaper10.jpg", s, true)
    end
end
-- }}}

-- {{{ Tags
-- Define a tag table which hold all screen tags.
tags = {}
for s = 1, screen.count() do
    -- Each screen has its own tag table.
    tags[s] = awful.tag({ "</>", "#!", "->" }, s, layouts[1])
end
-- }}}

-- {{{ Menu
-- Create a laucher widget and a main menu
myawesomemenu = {
   { "manual", terminal .. " -e man awesome" },
   { "edit config", editor_cmd .. " " .. awesome.conffile },
   { "restart", awesome.restart },
   { "quit", awesome.quit }
}

developments = {
-- { "Eclipse", "sh -c 'SWT_GTK3=0 eclipse'" }
       { "IntelliJ idea", "sh -c idea.sh" }
}


games = {
   { "Minecraft", "minecraft" },
   { "Nethack", terminal .. " -e nethack" },
   { "Openra", "openra" },
   { "Steam", "sh -c 'STEAM_FRAME_FORCE_CLOSE=1 steam' %U" }
}

graphics = {
```

```
   { "Gimp", "gimp" },
   { "Ristretto", "ristretto "}
}

network = {
   { "Firefox", "firefox" },
   { "Lantern", "lantern" }
}

office = {
   { "Calibre", "calibre" },
   { "Galcalator", "galculator" },
   { "Mousepad", "mousepad" }
}

system = {
      {"Bleachbit", "bleachbit" },
      { "Virtualbox", "virtualbox" }
}

cheon = {
      { "Lock", "sh -c 'sleep 0.5 && xset dpms force off; slock' " },
      { "Restart", "reboot" },
      { "Shutdown", "shutdown now" }
}

mymainmenu = awful.menu({ items = { { "awesome", myawesomemenu,
beautiful.awesome_icon },
                                    { "open terminal", terminal },
                                                    { "Email Reader",
"thunderbird" },
                                                    { "File Manager", "thunar" },
                                                    { "Web Broswer", "firefox" },
                                                    { "Develop", developments },
                                                    { "Games", games },
                                                    { "Graphics", graphics },
                                                    { "Network", network },
                                                    { "Office", office },
                                                    { "System", system },
                                                    { "cheon", cheon }
                                  }
                         })

mylauncher = awful.widget.launcher({ image =
"/home/cheon/.config/awesome/archlinux-artwork-1.6/wmlogos/archlinux-wm-
awesome.svg",
                                     menu = mymainmenu })

-- Menubar configuration
menubar.utils.terminal = terminal -- Set the terminal for applications that
require it
-- }}}

-- {{{ Wibox
-- Create a textclock widget
mytextclock = awful.widget.textclock()

-- Create a wibox for each screen and add it
```

```
mywibox = {}
mypromptbox = {}
mylayoutbox = {}
mytaglist = {}
mytaglist.buttons = awful.util.table.join(
                    awful.button({ }, 1, awful.tag.viewonly),
                    awful.button({ modkey }, 1, awful.client.movetotag),
                    awful.button({ }, 3, awful.tag.viewtoggle),
                    awful.button({ modkey }, 3, awful.client.toggletag),
                    awful.button({ }, 4, function(t)
awful.tag.viewnext(awful.tag.getscreen(t)) end),
                    awful.button({ }, 5, function(t)
awful.tag.viewprev(awful.tag.getscreen(t)) end)
                    )
mytasklist = {}
mytasklist.buttons = awful.util.table.join(
                     awful.button({ }, 1, function (c)
                                              if c == client.focus then
                                                  c.minimized = true
                                              else
                                                  -- Without this, the following
                                                  -- :isvisible() makes no sense
                                                  c.minimized = false
                                                  if not c:isvisible() then
                                                      awful.tag.viewonly(c:tags()
[1])
                                                  end
                                                  -- This will also un-minimize
                                                  -- the client, if needed
                                                  client.focus = c
                                                  c:raise()
                                              end
                                          end),
                     awful.button({ }, 3, function ()
                                              if instance then
                                                  instance:hide()
                                                  instance = nil
                                              else
                                                  instance = awful.menu.clients({
                                                      theme = { width = 250 }
                                                  })
                                              end
                                          end),
                     awful.button({ }, 4, function ()
                                              awful.client.focus.byidx(1)
                                              if client.focus then
client.focus:raise() end
                                          end),
                     awful.button({ }, 5, function ()
                                              awful.client.focus.byidx(-1)
                                              if client.focus then
client.focus:raise() end
                                          end))

for s = 1, screen.count() do
    -- Create a promptbox for each screen
    mypromptbox[s] = awful.widget.prompt()
    -- Create an imagebox widget which will contains an icon indicating which
```

```
layout we're using.
    -- We need one layoutbox per screen.
    mylayoutbox[s] = awful.widget.layoutbox(s)
    mylayoutbox[s]:buttons(awful.util.table.join(
                           awful.button({ }, 1, function ()
awful.layout.inc(layouts, 1) end),
                           awful.button({ }, 3, function ()
awful.layout.inc(layouts, -1) end),
                           awful.button({ }, 4, function ()
awful.layout.inc(layouts, 1) end),
                           awful.button({ }, 5, function ()
awful.layout.inc(layouts, -1) end)))
    -- Create a taglist widget
    mytaglist[s] = awful.widget.taglist(s, awful.widget.taglist.filter.all,
mytaglist.buttons)

    -- Create a tasklist widget
    mytasklist[s] = awful.widget.tasklist(s,
awful.widget.tasklist.filter.currenttags, mytasklist.buttons)

    -- Create a memory widget
    memwidget = wibox.widget.textbox()
    vicious.register(memwidget, vicious.widgets.mem, " Memory:$1% ", 13)

    -- Create a CPU widget
    cpuwidget = wibox.widget.textbox()
    vicious.register(cpuwidget, vicious.widgets.cpu, " CPU:$1% ")

    -- Create a volume widget
    volwidget = wibox.widget.textbox()
    vicious.register(volwidget, vicious.widgets.volume,
      function(widget, args)
          local label = { ["♫"] = " ♬", ["♩"] = "♩" }
          return  label[args[2]].. args[1] .. "%"
            end, 1, "Master")

    -- Create a battery widget
    batwidget = wibox.widget.textbox()
    vicious.register(batwidget, vicious.widgets.bat, " $2%($3)", 61, "BAT0")

    -- Create a MPD widget
    mpdwidget = wibox.widget.textbox()
    vicious.register(mpdwidget, vicious.widgets.mpd,
    function (mpdwidget, args)
        if args["{state}"] == "Stop" then
            return " - "
        else
            return args["{Artist}"]..' - '.. args["{Title}"]
        end
    end, 10)

    -- Create the wibox
    mywibox[s] = awful.wibox({ position = "top", screen = s })

    -- Widgets that are aligned to the left
    local left_layout = wibox.layout.fixed.horizontal()
    left_layout:add(mylauncher)
    left_layout:add(mytaglist[s])
```

```
    left_layout:add(mypromptbox[s])

    -- Widgets that are aligned to the right
    local right_layout = wibox.layout.fixed.horizontal()
    if s == 1 then right_layout:add(wibox.widget.systray()) end
    right_layout:add(memwidget)
    right_layout:add(cpuwidget)
    right_layout:add(mpdwidget)
    right_layout:add(volwidget)
    right_layout:add(batwidget)
    right_layout:add(mytextclock)
    right_layout:add(mylayoutbox[s])

    -- Now bring it all together (with the tasklist in the middle)
    local layout = wibox.layout.align.horizontal()
    layout:set_left(left_layout)
    layout:set_middle(mytasklist[s])
    layout:set_right(right_layout)

    mywibox[s]:set_widget(layout)
end
-- }}}

-- {{{ Mouse bindings
root.buttons(awful.util.table.join(
    awful.button({ }, 3, function () mymainmenu:toggle() end),
    awful.button({ }, 4, awful.tag.viewnext),
    awful.button({ }, 5, awful.tag.viewprev)
))
-- }}}

-- {{{ Key bindings
globalkeys = awful.util.table.join(
    awful.key({ modkey,           }, "Left",   awful.tag.viewprev       ),
    awful.key({ modkey,           }, "Right",  awful.tag.viewnext       ),
    awful.key({ modkey,           }, "Escape", awful.tag.history.restore),

    awful.key({ modkey,           }, "j",
        function ()
            awful.client.focus.byidx( 1)
            if client.focus then client.focus:raise() end
        end),
    awful.key({ modkey,           }, "k",
        function ()
            awful.client.focus.byidx(-1)
            if client.focus then client.focus:raise() end
        end),
    awful.key({ modkey,           }, "w", function () mymainmenu:show() end),

    -- Layout manipulation
    awful.key({ modkey, "Shift"   }, "j", function () awful.client.swap.byidx(  1)
end),
    awful.key({ modkey, "Shift"   }, "k", function () awful.client.swap.byidx( -1)
end),
    awful.key({ modkey, "Control" }, "j", function () awful.screen.focus_relative(
1) end),
    awful.key({ modkey, "Control" }, "k", function ()
awful.screen.focus_relative(-1) end),
```

```
    awful.key({ modkey,           }, "u", awful.client.urgent.jumpto),
    awful.key({ modkey,           }, "Tab",
        function ()
            awful.client.focus.history.previous()
            if client.focus then
                client.focus:raise()
            end
        end),

    -- Standard program
    awful.key({ modkey,           }, "Return", function ()
awful.util.spawn(terminal) end),
    awful.key({ modkey, "Control" }, "r", awesome.restart),
    awful.key({ modkey, "Shift"   }, "q", awesome.quit),

    awful.key({ modkey,           }, "l",     function ()
awful.tag.incmwfact( 0.05)    end),
    awful.key({ modkey,           }, "h",     function () awful.tag.incmwfact(-
0.05)    end),
    awful.key({ modkey, "Shift"   }, "h",     function () awful.tag.incnmaster( 1)
end),
    awful.key({ modkey, "Shift"   }, "l",     function () awful.tag.incnmaster(-1)
end),
    awful.key({ modkey, "Control" }, "h",     function () awful.tag.incncol( 1)
end),
    awful.key({ modkey, "Control" }, "l",     function () awful.tag.incncol(-1)
end),
    awful.key({ modkey, "Control" }, "space", function ()
awful.layout.inc(layouts,  1) end),
    awful.key({ modkey, "Shift"   }, "space", function ()
awful.layout.inc(layouts, -1) end),

    awful.key({ modkey, "Control" }, "n", awful.client.restore),
    -- Customer Setting
    awful.key({ modkey, "Control" }, "b", function() awful.util.spawn("firefox")
end),
    awful.key({ modkey, "Control" }, "e", function()
awful.util.spawn("thunderbird") end),
    awful.key({ modkey, "Control" }, "c", function()
awful.util.spawn("galculator") end),
    awful.key({ modkey, "Control" }, "p", function () awful.util.spawn("scrot -e
'mv $f /home/cheon/Pictures/ 2>/dev/null'") end),
    awful.key({ }, "XF86AudioRaiseVolume", function () awful.util.spawn("amixer
set Master 5%+") end),
    awful.key({ }, "XF86AudioLowerVolume", function () awful.util.spawn("amixer
set Master 5%-") end),
    awful.key({ }, "XF86AudioMute", function () awful.util.spawn("amixer set
Master toggle") end),
    awful.key({ }, "XF86MonBrightnessUp", function () awful.util.spawn("xbacklight
-inc 5") end),
    awful.key({ }, "XF86MonBrightnessDown", function ()
awful.util.spawn("xbacklight -dec 5") end),
      awful.key({ }, "XF86AudioNext",function () awful.util.spawn( "mpc next" )
end),
      awful.key({ }, "XF86AudioPrev",function () awful.util.spawn( "mpc prev" )
end),
      awful.key({ }, "XF86AudioPlay",function () awful.util.spawn( "mpc play" )
end),
```

```
      awful.key({ }, "XF86AudioStop",function () awful.util.spawn( "mpc pause" )
end),
    awful.key({ modkey,           }, "b", function ()
mywibox[mouse.screen].visible = not mywibox[mouse.screen].visible end),

    -- Prompt
    awful.key({ modkey },            "r",     function ()
mypromptbox[mouse.screen]:run() end),

    awful.key({ modkey }, "x",
              function ()
                  awful.prompt.run({ prompt = "Run Lua code: " },
                  mypromptbox[mouse.screen].widget,
                  awful.util.eval, nil,
                  awful.util.getdir("cache") .. "/history_eval")
              end),
    -- Menubar
    awful.key({ modkey }, "p", function() menubar.show() end)
)

clientkeys = awful.util.table.join(
    awful.key({ modkey,           }, "f",      function (c) c.fullscreen = not
c.fullscreen  end),
    awful.key({ modkey, "Shift"   }, "c",      function (c) c:kill()
end),
    awful.key({ modkey, "Control" }, "space",  awful.client.floating.toggle
),
    awful.key({ modkey, "Control" }, "Return", function (c)
c:swap(awful.client.getmaster()) end),
    awful.key({ modkey,           }, "o",      awful.client.movetoscreen
),
    awful.key({ modkey,           }, "t",      function (c) c.ontop = not c.ontop
end),
    awful.key({ modkey,           }, "n",
        function (c)
            -- The client currently has the input focus, so it cannot be
            -- minimized, since minimized clients can't have the focus.
            c.minimized = true
        end),
    awful.key({ modkey,           }, "m",
        function (c)
            c.maximized_horizontal = not c.maximized_horizontal
            c.maximized_vertical   = not c.maximized_vertical
        end)
)

-- Bind all key numbers to tags.
-- Be careful: we use keycodes to make it works on any keyboard layout.
-- This should map on the top row of your keyboard, usually 1 to 9.
for i = 1, 9 do
    globalkeys = awful.util.table.join(globalkeys,
        -- View tag only.
        awful.key({ modkey }, "#" .. i + 9,
                  function ()
                        local screen = mouse.screen
                        local tag = awful.tag.gettags(screen)[i]
                        if tag then
                           awful.tag.viewonly(tag)
```

```
                      end
                end),
        -- Toggle tag.
        awful.key({ modkey, "Control" }, "#" .. i + 9,
                function ()
                    local screen = mouse.screen
                    local tag = awful.tag.gettags(screen)[i]
                    if tag then
                       awful.tag.viewtoggle(tag)
                    end
                end),
        -- Move client to tag.
        awful.key({ modkey, "Shift" }, "#" .. i + 9,
                function ()
                    if client.focus then
                        local tag = awful.tag.gettags(client.focus.screen)[i]
                        if tag then
                            awful.client.movetotag(tag)
                        end
                   end
                end),
        -- Toggle tag.
        awful.key({ modkey, "Control", "Shift" }, "#" .. i + 9,
                function ()
                    if client.focus then
                        local tag = awful.tag.gettags(client.focus.screen)[i]
                        if tag then
                            awful.client.toggletag(tag)
                        end
                    end
                end))
end

clientbuttons = awful.util.table.join(
    awful.button({ }, 1, function (c) client.focus = c; c:raise() end),
    awful.button({ modkey }, 1, awful.mouse.client.move),
    awful.button({ modkey }, 3, awful.mouse.client.resize))

-- Set keys
root.keys(globalkeys)
-- }}}

-- {{{ Rules
-- Rules to apply to new clients (through the "manage" signal).
awful.rules.rules = {
    -- All clients will match this rule.
    { rule = { },
      properties = { border_width = beautiful.border_width,
                     border_color = beautiful.border_normal,
                     focus = awful.client.focus.filter,
                              size_hints_honor = false,
                     raise = true,
                     keys = clientkeys,
                     buttons = clientbuttons } },
--    { rule = { class = "MPlayer" },
--      properties = { floating = true } },
    { rule = { class = "pinentry" },
      properties = { floating = true } },
```

```
    { rule = { class = "gimp" },
      properties = { floating = true } },
    { rule = { class = "VirtualBox" },
      properties = { floating = true } },
}
-- }}}

-- {{{ Signals
-- Signal function to execute when a new client appears.
client.connect_signal("manage", function (c, startup)
    -- Enable sloppy focus
    c:connect_signal("mouse::enter", function(c)
        if awful.layout.get(c.screen) ~= awful.layout.suit.magnifier
            and awful.client.focus.filter(c) then
            client.focus = c
        end
    end)

    if not startup then
        -- Set the windows at the slave,
        -- i.e. put it at the end of others instead of setting it master.
        -- awful.client.setslave(c)

        -- Put windows in a smart way, only if they does not set an initial
position.
        if not c.size_hints.user_position and not c.size_hints.program_position
then
            awful.placement.no_overlap(c)
            awful.placement.no_offscreen(c)
        end
    end

    local titlebars_enabled = false
    if titlebars_enabled and (c.type == "normal" or c.type == "dialog") then
        -- buttons for the titlebar
        local buttons = awful.util.table.join(
                awful.button({ }, 1, function()
                    client.focus = c
                    c:raise()
                    awful.mouse.client.move(c)
                end),
                awful.button({ }, 3, function()
                    client.focus = c
                    c:raise()
                    awful.mouse.client.resize(c)
                end)
                )

        -- Widgets that are aligned to the left
        local left_layout = wibox.layout.fixed.horizontal()
        left_layout:add(awful.titlebar.widget.iconwidget(c))
        left_layout:buttons(buttons)

        -- Widgets that are aligned to the right
        local right_layout = wibox.layout.fixed.horizontal()
        right_layout:add(awful.titlebar.widget.floatingbutton(c))
        right_layout:add(awful.titlebar.widget.maximizedbutton(c))
        right_layout:add(awful.titlebar.widget.stickybutton(c))
```

```
        right_layout:add(awful.titlebar.widget.ontopbutton(c))
        right_layout:add(awful.titlebar.widget.closebutton(c))

        -- The title goes in the middle
        local middle_layout = wibox.layout.flex.horizontal()
        local title = awful.titlebar.widget.titlewidget(c)
        title:set_align("center")
        middle_layout:add(title)
        middle_layout:buttons(buttons)

        -- Now bring it all together
        local layout = wibox.layout.align.horizontal()
        layout:set_left(left_layout)
        layout:set_right(right_layout)
        layout:set_middle(middle_layout)

        awful.titlebar(c):set_widget(layout)
    end
end)

client.connect_signal("focus", function(c) c.border_color = beautiful.border_focus
end)
client.connect_signal("unfocus", function(c) c.border_color =
beautiful.border_normal end)
-- }}}

--autorun
function run_once(cmd)
      findme=cmd
      firstspace=cmd:find(" ")
      if firstspace then
            findme=cmd:sub(0, firstspace-1)
      end
      awful.util.spawn_with_shell("pgrep -u $USER -x " .. findme .." > /dev/null
|| (" .. cmd ..")")
end
--run_once("ibus-daemon")
--run_once("nm-applet")
--run_once("xfce4-power-manager")
run_once("xcompmgr")
run_once("devilspie -a")
--[[
do
  local cmdss =
    {"xcompmgr",
     "devilspie -a",
    }
  for _,i in pairs(cmdss) do
    awful.util.spawn(i)
  end
end
--]]
       middle_layout:add(title)
```