```
import XMonad import XMonad.Config.Desktop import XMonad.Hooks.DynamicLog (shorten, dynamicLogWithPP, xmobarPP, xmobarColor,wrap ,PP(..)) import qualified XMonad.StackSet as W import qualified Data.Map as M import Data.Maybe (fromJust)

import XMonad.Hooks.WindowSwallowing import XMonad import XMonad.Actions.Volume (lowerVolume, raiseVolume, toggleMute) import XMonad.Util.Dzen import Data.Map (fromList) import Data.Monoid (mappend) import XMonad.Util.EZConfig (additionalKeysP) import XMonad.Util.Run (spawnPipe) import System.IO (hPutStrLn) import XMonad.Hooks.ManageDocks (docks, avoidStruts, manageDocks , ToggleStruts) import XMonad.Util.Hacks (windowedFullscreenFixEventHook, javaHack, trayerAboveXmobarEventHook, trayAbovePanelEventHook, trayerPaddingXmobarEventHook, trayPaddingXmobarEventHook, trayPaddingEventHook)


alert = dzenConfig centered . show . round centered =
      onCurr (center 150 66)
   >=> font "-*-helvetica-*-r-*-*-64-*-*-*-*-*-*-*"
   >=> addArgs ["-fg", "#80c0ff"]
   >=> addArgs ["-bg", "#000040"]

myKeyBindings :: [(String, X () )] myKeyBindings =    [
      ("<XF86AudioLowerVolume>", lowerVolume 4 >>= alert),
      ("<XF86AudioRaiseVolume>", raiseVolume 4 >>= alert),
      ("<XF86AudioMute>",  (const ()) <$> toggleMute),
      ("<XF86MonBrightnessUp>", spawn "light -A 2"),
      ("<XF86MonBrightnessDown>", spawn "light -U 2"),
      ("M-f", spawn "$BROWSER"),
      ("M-e", spawn "st -e $EDITOR"),
      ("M-r", spawn "st -e ranger")
    -- , ((modMask x, xK_b     ), sendMessage ToggleStruts)
   ] myTerminal :: String myTerminal = "st"

windowCount :: X (Maybe String) windowCount = gets $ Just . show . length . W.integrate' . W.stack . W.workspace . W.current . windowset

clickable :: String -> String clickable ws = "<action=xdotool key super+1>"++ws++"</action>"
   -- where i = fromJust $ M.lookup ws myWorkspaceIndices



main :: IO () main = do
  xmobar_process <- spawnPipe "xmobar /home/atticusk/.xmobarrc"
  xmonad $ docks $ (def { terminal = myTerminal           , modMask = mod4Mask
  , handleEventHook    = windowedFullscreenFixEventHook <> swallowEventHook (className =? "Alacritty" <||> className =? "st-256color" <||> className =? "XTerm") (return True) <> trayerPaddingXmobarEventHook
     , layoutHook=avoidStruts $ layoutHook def
    , manageHook=manageHook def <+> manageDocks          , borderWidth = 3
    , logHook = dynamicLogWithPP $ xmobarPP
      { ppOutput = hPutStrLn xmobar_process
      , ppCurrent = xmobarColor "#c792ea" "" . wrap "<box type=Bottom width=2 mb=2 color=#c792ea>" "</box>"         -- Current workspace
      , ppVisible = xmobarColor "#c792ea" "" . clickable              -- Visible but not current workspace
      , ppHidden = xmobarColor "#82AAFF" "" . wrap "<box type=Top width=2 mt=2 color=#82AAFF>" "</box>" . clickable -- Hidden workspaces
```

```
    , ppHiddenNoWindows = xmobarColor "#82AAFF" ""  . clickable     -- Hidden workspaces (no windows)
    , ppTitle = xmobarColor "#b3afc2" "" . shorten 60               -- Title of active window
    , ppSep =  "<fc=#666666> <fn=1>|</fn> </fc>"                    -- Separator character
    , ppUrgent = xmobarColor "#C45500" "" . wrap "!" "!"            -- Urgent workspace
    , ppExtras  = [windowCount]                                     -- # of windows current workspace
    , ppOrder  = :l:t:ex) -> [ws,l]++ex++[t]                        -- order of things in xmobar
  }          } `additionalKeysP` myKeyBindings)
```