Setting up i3 Window Manager

Table of Contents

This brief guide was initially created as supplementary material to a series of YouTube videos (part1, part2) demonstrating a minimal Debian Linux installation with the i3 window manager.

You can find my i3, i3blocks config files and modules in my github repo.

```
#############################################################################

# ooooo        ooooo ooooo      ooo ooooo     ooo ooooooo  ooooo
# `888'        `888' `888b.     `8' `888'     `8'  `8888    d8'
# 888          888   8 `88b.    8   888       8     Y888..8P
# 888          888   8   `88b.  8   888       8      `8888'
# 888          888   8     `88b.8   888       8     .8PY888.
# 888       o  888   8       `888   `88.    .8'    d8'  `888b
# o888oooooood8 o888o o8o        `8     `YbodP'    o888o  o88888o

# ooooo      ooo   .oooooo.   ooo        ooooo       .o.       oooooooooo.
# `888b.     `8'  d8P'  `Y8b  `88.       .888'      .888.      `888'   `Y8b
# 8 `88b.    8  888      888  888b     d'888      .8"888.      888      888
# 8   `88b.  8  888      888  8 Y88. .P  888     .8' `888.     888      888
# 8     `88b.8  888      888  8  `888'   888    .88ooo8888.    888      888
# 8       `888  `88b    d88'  8    Y     888   .8'     `888.   888     d88'
# o8o        `8   `Y8bood8P'  o8o        o888o o88o     o8888o o888bood8P'

#############################################################################
```

1. Debian minimal installation

Download the network iso installation medium from official Debian webpage and begin the installation process. During the stage of selecting a Desktop Environment uncheck all options except of standard system utilities.

When the installation is finished reboot and enter your credentials (username, password) in the TTY (console like screen).

1.1. Useful packages installation

After login into TTY using your credentials install the next packages to have a bare minimum of functionality after the first reboot.

```
$ sudo apt install i3 i3blocks lightdm x11-xserver-utils -y
$ sudo apt install pulseaudio nm-tray fonts-font-awesome -y
$ sudo apt install curl git feh -y
$ sudo apt install vim chromium-browser xfe xfe-themes mousepad -y
$ sudo reboot
```

2. i3 configuration

After the first reboot you will be prompted to the lightdm login screen (display server) requesting your username and password. Entering your credentials succesfully your will be introduced to a welcome screen of i3 window manager asking you to select Win or Alt button as the mod key of your configuration.

Choose the default Win mod in order to be compatible with the provided configuration here, however, you can change it later and make the respective adjustments in the config file.

When you choose the default mod key an i3 configuration file is created by default in ~/.config/i3 folder with the filename config.

You can edit the `~/.config/i3/config` file with a text editor to add or modify the configuration according to your preferences. Next we define some useful configurations which you could add into your config file.

2.1. Useful configs

Keyboard layout

Edit the /etc/locale.gen file and uncomment the layouts you want to use, for example search for the Greek UTF-8 layout, uncomment it and regenerate the locales

```
$ sudo cat /etc/locale.gen | grep GR
# el_GR ISO-8859-7
# el_GR.UTF-8 UTF-8
# el_GR@euro ISO-8859-7

$ sudo vim /etc/locale.gen
```

```
# el_GR ISO-8859-7
el_GR.UTF-8 UTF-8
# el_GR@euro ISO-8859-7

$ sudo locale-gen
$ sudo reboot
```

and set keyboard layouts (here usa and greek) to be toggled by setting a keybind (here alt+shift combination) in the i3 config file.

```
exec "setxkbmap 'us,gr' -option 'grp:alt_shift_toggle'"
```

Volume control shortcuts

```
# Use pactl to adjust volume in PulseAudio.
set $refresh_i3status killall -SIGUSR1 i3status
bindsym XF86AudioRaiseVolume exec --no-startup-id pactl set-sink-volume @DEFAULT_SINK@
+10% && $refresh_i3status
bindsym XF86AudioLowerVolume exec --no-startup-id pactl set-sink-volume @DEFAULT_SINK@
-10% && $refresh_i3status
bindsym XF86AudioMute exec --no-startup-id pactl set-sink-mute @DEFAULT_SINK@ toggle
&& $refresh_i3status
bindsym XF86AudioMicMute exec --no-startup-id pactl set-source-mute @DEFAULT_SOURCE@
toggle && $refresh_i3status
```

Screen brigtness controls

```
# Screen brightness controls
#bindsym XF86MonBrightnessUp exec xbacklight + 20 # increase screen brightness
#bindsym XF86MonBrightnessDown exec xbacklight - 20 # decrease screen brightness
#bindsym XF86MonBrightnessDown exec xrandr --output LVDS-1 --brightness 0.6
#bindsym XF86MonBrightnessUp exec xrandr --output LVDS-1 --brightness 1
```

Startup applications

You can autolaunch your prefered applications after login, here we use an example of the flatpak Nextcloud client.

```
# Nextcloud client
exec --no-startup-id flatpak run com.nextcloud.desktopclient.nextcloud
```

Launchers

You can also create launchers of your favourite apps using keybinds. Here we will set a keybind for rofi which is an app launcher.

```
bindsym $mod+d exec "rofi -show drun"
```

Workspaces names and apps assignment

We can define our custom names for i3 workspaces as in the following example.

```
set $ws1 "1:Web"
set $ws2 "2:Media"
set $ws3 "3:Office"
set $ws4 "4:Dev"
set $ws5 "5:Misc"
```

Another useful configuration is to assign applications to open in specific workspaces. In order to do that we need first to identify the WM_CLASS of a window/app.

First, launch the desired application and then open a terminal. Inside the terminal execute the command `xprop | grep WM_CLASS` and you will see a cross-cursor which you should place over the application window and left click.

After the left mouse click an output will appear in the terminal with the WM_class of the application, for example,

```
WM_CLASS(STRING) = "xfe", "Xfe"
```

is the output for the Sublime Text editor. We can assign this window class to open on a specific workspace as in the following i3 config command

```
for_window [class="Xfe"] move to workspace $ws3
```

which will assign Sublime Text editor to launch each time on workspace with variable name $ws4.

3. i3blocks (i3 status bar)

We will use the i3block status bar which offers a natively feel and support of the i3 window manager. Each entry/widget/block that appears on the i3bar is defined in a separate module file which is loaded from the i3blocks configuration file.

In order to use i3blocks status bar we need to comment out the default i3status bar and set the path of our i3blocks config file.

First of all, create an empty i3blocks folder inside user's hidden home config folder and inside that create an empty modules folder.

```
$ mkdir ~/.config/i3blocks
$ mkdir ~/.config/i3blocks/modules
```

Next we create an empty i3blocks config file inside i3blocks config folder we just created.

```
$ cd ~/.config/i3blocks
$ touch config
```

During the installation of i3blocks (section 1) a default configuration file is created in `/etc/i3blocks.conf` and a default folder in `/usr/share/i3blocks` contains some modules. You could replace the empty config file with the default config and also copy the contents of the aforementioned folder inside our modules folder.

However, since we try to create a clean i3blocks status bar and add modules one by one, we will copy only the module named disk as a simple example into our modules folder and edit our empty config file accordingly.

```
$ sudo cp ~/usr/share/i3blocks/disk ~/.config/i3blocks/modules/
$ sudo chown linuxnomad:linuxnomad ~/.config/i3blocks/modules/disk
$ chmod +x ~/.config/i3blocks/modules/disk
```

Note: With the last two commands we gave ownership of the module to user and made it executable which is mandatory otherwise it will not appear on the status bar! Make sure to replace `linuxnomad:linuxnomad` with your own username.

Next we open our empty i3blocks config file and specify our modules folder path.

```
vim ~/.config/i3blocks/config

#--- global preferences ---#
command=~/.config/i3blocks/modules/$BLOCK_NAME

separator=true
separator_block_width=10
markup=pango

#--- modules section ---#
[disk]
label=disk_usage:
instance=/
interval=30
```

As you can see we load the disk module by its name into square brackets, we set a test name as label [1]

Instance and interval are properties identified by i3bar protocol and i3blocks respectively. Note that this disk module setup and its properties (label, instance, interval) were found inside the default i3blocks config (`/etc/i3blocks.conf`) and you can understand more about them by reading the contents of the disk module (`~/.config/i3blocks/modules/disk`) and from the links above.

We also suggest reading the contents of the default i3blocks config since there are many modules setup therein with various properties/keywoards/options that will help you understand more about them and how you could create your own modules.

You can find an i3blocks configuration file and modules in my github repo.

Another important block/widget that can be added on the i3block bar is application launchers, i.e. clickable launchers of applications. You can set an application launcher following the example below

```
[web]
full_text=WEB
color=#029CD9
command=chromium >/dev/null
separator=false
```

where we define the block name [web], its full_text appearance on the status bar, the command to launch the chromium application with its output piped to /dev/null device and a false value in separator to hide the separator symbol on the bar.

4. Network Manager

In case you use a wifi connection it is possible that the wifi device is not connected and identified by the Network Manager applet nm-tray. However, you might still have access to the internet through the loopback (lo) device if you connected to your wifi network during the Debian installation.

To check about this occasion you need to run the command `nmcli d` and if you get the following output

```
DEVICE   TYPE      STATE                  CONNECTION
wlo1     wifi      unmanaged              --
lo       loopback  connected (externally)  lo
enp0s25  ethernet  unavailable            --
```

it means that you need to make the wifi device manageable. Network Manager does not manage any interface defined in /etc/network/interfaces by default and therefore we have to comment out the wifi device. Using sudo and your text editor of choice you can edit the aforementioned file and result to something like this

```
# The loopback network interface
auto lo
iface lo inet loopback

# The primary network interface
#allow-hotplug wlo1
#iface wlo1 inet dhcp
#       wpa-ssid <SSID>
#       wpa-psk  <pass>
```

where we have commented out the wlo1 (wifi) interface.

After this modification we can reboot into our system and check that nm-tray is able to identify wireless networks.

In case that the `nmcli d` command does not return a wol1 (wifi) device it probably hints that a driver for the wifi device is not installed, although since Debian 12 the netinstaller includes by default even proprietary device drivers.

We can try to identify the available network devices by executing the command

```
lspci -nn | grep -i network
```

and write down our wifi device in order to search for the appropriate driver on the internet. For more information it is advisable to read the Debian manpages.

5. Wallpaper/Background

During the installation of Debian based OS we installed the package `feh` which can be used in a oneline command to set our display background

```
exec_always feh --bg-fill ~/Pictures/debian-black-wallpaper.jpg
```

where we load out prefered image, here from folder ~/Pictures.

6. Touchpad settings

By default the toucpad and input devices settings are stored in `/usr/share/X11/xorg.conf.d/40-libinput.conf`. You can open this file and copy the contents of the touchpad section, for example

```
Section "InputClass"
        Identifier "libinput touchpad catchall"
        MatchIsTouchpad "on"
        MatchDevicePath "/dev/input/event*"
        Driver "libinput"
EndSection
```

into a new config file (`/etc/X11/xorg.conf.d/40-libinput.conf') that will perceed the default settings

```
Section "InputClass"
        Identifier "libinput touchpad catchall"
        MatchIsTouchpad "on"
        MatchDevicePath "/dev/input/event*"
        Driver "libinput"
        Option "NaturalScrolling" "on"
        Option "ScrollMethod" "edge"
EndSection
```

where we have modified our settings by adding the options of "natural scrolling" and "edge scrolling". For the full list of available options you should visit libinput manpage and more about the libinput config files the LibinputTouchpad Debian wiki page.

7. Mount external devices

First we need to plugin our external device (usb, hard disk, etc.) and then run the command

```
$ lsblk -o NAME,UUID,PATH,LABEL,VENDOR,SIZE,FSTYPE

NAME   UUID                                 PATH      LABEL    VENDOR     SIZE FSTYPE
sda                                         /dev/sda           ATA      232.9G
├─sda1 504704de-fe12-4e6f-9a96-d5c829ce6428 /dev/sda1                   231.9G ext4
├─sda2                                      /dev/sda2                       1K
└─sda5 56cfc037-af5b-40db-8d39-4bf2aa0b4a2c /dev/sda5                     976M swap
sdb                                         /dev/sdb           WD         1.8T
└─sdb1 9ED6C007D6BFDE29                     /dev/sdb1 Elements            1.8T ntfs
sr0                                         /dev/sr0           hp        1024M
```

and we should try to identify our device from the output. As we can see the WesternDigital external hard disk labeled Elements with size 1.8T is identified at path /dev/sdb but note that we will always use the numbered partition, i.e. /dev/sdb1 in our case.

In order to give ownership of this drive and its content to our user we need to find the uid, gid and umask values of our user. We can get access to these values by runnin the following commands

```
$ id
$ umask
```

and now we can mount our device on a directory from which we will get access to the contents of the drive

```
sudo mkdir /media/elements-drive
sudo mount /dev/sdb1 /media/elements-drive -o uid=1000,gid=1000,umask=0022
```

where we use the /media directory as the path to host our external drive directory since it is a common practice and we give access to our user with the specified options.

If we want to auto-mount the drive after each boot of the OS we can add the UUID and FSTYPE of the device into the fstab file

```
# <device> <dir> <type> <options> <dump> <fsck>
UUID=9ED6C007D6BFDE29 /media/elements-drive ntfs defaults,noatime,nofail 0 0
```

where the <device> and <type> correspond to the UUID and FSTYPE respectively while <dir> is the mount path. These are the values that we need to modify for any new device we might want to add on the fstab file while the <options>, <dump>, <fsck> can be kept with the same values.

Finally, we can unmount the device by executing the command

```
sudo umount /media/elements-drive
```

and you can read more about mounting filesystems and the available options at the pages mount, fstab.

[1] Instead for the plain text label we can be display - in i3blocks status bar - a beautiful aesthetically Awesome Font Icon since we installed the appropriate fonts package (section "Useful packages installation"). You can find a collection of awesome font icons in https://fontawesome.com/icons which you can copy and paste to the label of your module.