

ARCH LINUX

installation recommendations
guides
cheat sheets

1. General Installation Overview (<10 GB)

CORE-PACKAGES:

base	dosfstools	mc
linux	exfatprogs	7zip
linux-firmware	exfat-utils	grub
base-devel	ntfs-3g	efibootmgr
git	btrfs-progs	arch-install-scripts
	iwd	
	nano	vol: 2.54 GB (4.52 allocated)

>Install Drivers

```
a) proprietary (xf86-video-intel/...)
b) universal (   xf86-video-vesa,
                 xf86-video-ati,
                 xf86-video-amdgpu,
                 xf86-video-fbdev)
```

(dedicated vs universal | [see ArchLinux.org > Install on a Removable Medium](https://www.archlinux.org/docs/install-on-a-removable-medium/)
Tips and Tricks)

```
sof-firmware
alsa-firmware
alsa-lib      →      2.92 GB used (5.52 allocated)
```

:CORE-ACKAGES

```
nasm          →      probably, maybe
polkit        →      has something to do with login and privileges. Probably good to have.
```

```
//setup kernel
//setup grub
//=====
//configure environment:

→      add users, set passwords
(      see: Crystals Linux (Setup 1: Environment)

//
//=====
```

DESKTOP:

```
sd1
sd12_gfx
sd12_image
sd12_mixer
alsa-utils

> default
(+ noto-fonts (from arabic to
whatever) (100 MB))
+ ttf-bitstream-vera #Tall
+ ttf-croscore #variety
+ ttf-roboto #thin
+ ttf-liberation #flat

sway
swaylock
swaybar
wmenu
swaybg
i3status
foot

(brightnessctl)
nasm
fbterm
tmux

xdg-desktop-portal-gtk
libreoffice
AUR: sublime-text
AUR: woodland
→ 4.39 used
(5.52 allocated)
```

:DESKTOP

```
//=====
//configure "flash suite":
```

suggestions:

- > text based greeter: CDM is a good start to familiarize yourself with AUR packages. It starts AFTER login and acts as Selection Menu for Environment selection.
- > autostart into ...: See CDM.
- > text-based login: Or: Auto-Login. Perhaps configure it as
- > logout-condition.

- install Crystals
(see: Crystals Linux (Setup 2: cromp)

```
//
//=====
```

DEVBOX:

???

:DEVBOX

AUR installation (trizen):

- make sure locale is setup properly:

```
→ nano /etc/locale.gen
→ uncomment your locale (UTF-8)
[i.e.: I use de_DE.UTF-8 and
en_US.UTF-8 and then call locale-gen
(as root) //if the locale isn't
installed, some applications spit out
ugly warnings.
```

- install:

```
→ base-devel
→ git
```

- add meta to wheel:

```
→ nano /etc/sudoers
→ uncomment the line '%wheel
ALL=(ALL:ALL) ALL'
→ run 'usermod -aG wheel meta'
- download and install trizen:
→ run 'git clone
https://aur.archlinux.org/trizen.git &&
cd trizen && makepkg -si'
→ enter root ('su')
→ download reported packages
→ !!!exit root!!!
→ run 'makepkg -si' again
:(trizen) installation AUR
```

COMMANDS:

user management:

```
add user: useradd (-m) name #-m to create homedir
remove user: userdel -r username
rename user: usermod -l newname oldname
user detail: id user
list groups: cat /etc/group
add user to group: usermod -aG group,soup username // gpasswd -a user group
remove user frm grp: gpasswd -d user group
add group: groupadd (-g groupid) (-U user,sers) name
remove group: groupdel group
```

see also: [/etc/skel/](#) and [/etc/default/useradd](#)

Getting started:

when uncertain what option to use, type command --help

```
show directory: ls (path) (ls -l, ls -lA)
dir, DIR (when initialized)
```

```
change directory: cd .. =up,
/ =root,
./ =current,
~/ =home
```

```
NOTE: \ is the 'escape' character
denoting the next character to be
literal (i.e.> ~/this\ there.txt
(to have spaces in filenames))
```

```
copy file cp (options) file folder/(name)
move/rename file mv
remove file rm (-r (recursively->folders))
edit file nano (^=ctrl, M=alt, ctrl+o=save)
browser mc (access menu with F9)
```

pacman:

```
install: pacman -S
search (install): pacman -Ss
remove: pacman -R
remove(+): pacman -Rns
clean orphans: pacman -Qdtq | pacman -Rns -
clear cache: pacman -Scc (or: trizen -Sc)
list all: pacman -Q
package status: pacman -Qi <package>
search (installed) pacman -Qs <pattern>
list explicit: pacman -Qet
```

btrfs:

```
btrfs filesystem usage path
btrfs subvolume create path
btrfs subvolume delete path
btrfs subvolume list -t path
```

other:

```
active services: systemctl --type=service
disk space: df (option -i for inodes)
fix ntfs: ntfsfix -d /dev/sdXY
```



```
#my laptop doesn't wake up from
#hibernation (also on Windows)
Section "Extensions"
    Options "DPMS" "false"
EndSection

Section "ServerFlags"
    Option "BlankTime" "0"
# Option "StandbyTime" "10"
# Option "SuspendTime" "20"
# Option "OffTime" "30"
EndSection
```

fonts:

useful recommendations I found on a website:

```
ttf-hanazono
ttf-liberation
adobe-source-
code-pro-fonts
han-sans-otc-fonts
han-serif-otc-fonts
sans-fonts
serif-fonts
```

pacman -Ss ttf and such (liberation) can give you an index.

ALTERNATIVES: DISCLAIMER :STUFF

This is a condensed setup. Alternatives ... thinking of ... at this point, [nixos](#) or [gentoo](#). Ambition last night had to use a 128 sGB sd card that is in first place extended storage to double as a fallback/flash environment. Finally I wasn't able to decide and started prepping a Flash Drive. All the relevant isos and packages from gentoo – for my architecture – nixos, arch and arch32 ... but I suppose I'm good for now.

The sd installation should be simple and universal. So – I'm not sure what to do there. However – it might also be worth installing gentoo. The idea is to for once not get in the way of having an actual use for my computer – but eventually I might get a fine-tuned gentoo system that I should then be able to port onto my main drive. On the other hand, nixos might be a great way to do the other thing. In essence I should be able to specify a bunch of settings – and so it should be able to sustain a simple but adaptive environment.

Primarily both are to be dormant – but never active while a "CLX-ROOT" is plugged in (safety kink). Instead to CLX-ROOT it is a passive device. The System (Hardware+) finally is in this case subject to the "Flash System", as it has to deal with any kind of vulnerabilities plugged into it at any rate. Hence it can serve as tertiary workstation and direct interface with the flash vault; Which can in turn be wired into the ROOT. This way it is well justifiable 'best practice'.

From here on out, if your ideas of a desktop diverge from the herein produced call, you would probably either look towards `xdg-desktop-portal-gnome` or `kde`. After some experience here and there, I found that `nemo` can get the work done just as much as `dolphin` – and its own tree view may even be better. Because it doesn't boast as much complexity, the relevant shortcuts are usually there and easily accessible. Problems with qt based programs probably emerge in how the different packages resolve themselves relative to their design intent and condition.

For now I learned that `xdg-desktop-portal` – it has imprinted itself into my consciousness – is probably the most pivotal aspect of it all; So, if you install 'kde' or 'gnome' – they in turn run on their own portals – invoke them as a dependency and subsequently run off of what the different packages are set up to do. Point being that this way we're coming at it sideways. Properly done, first the desktop portals have to be properly dealt with – and then on top of that you would install something else. KDE and GNOME are at that point merely options to extend ontop of the given "core".

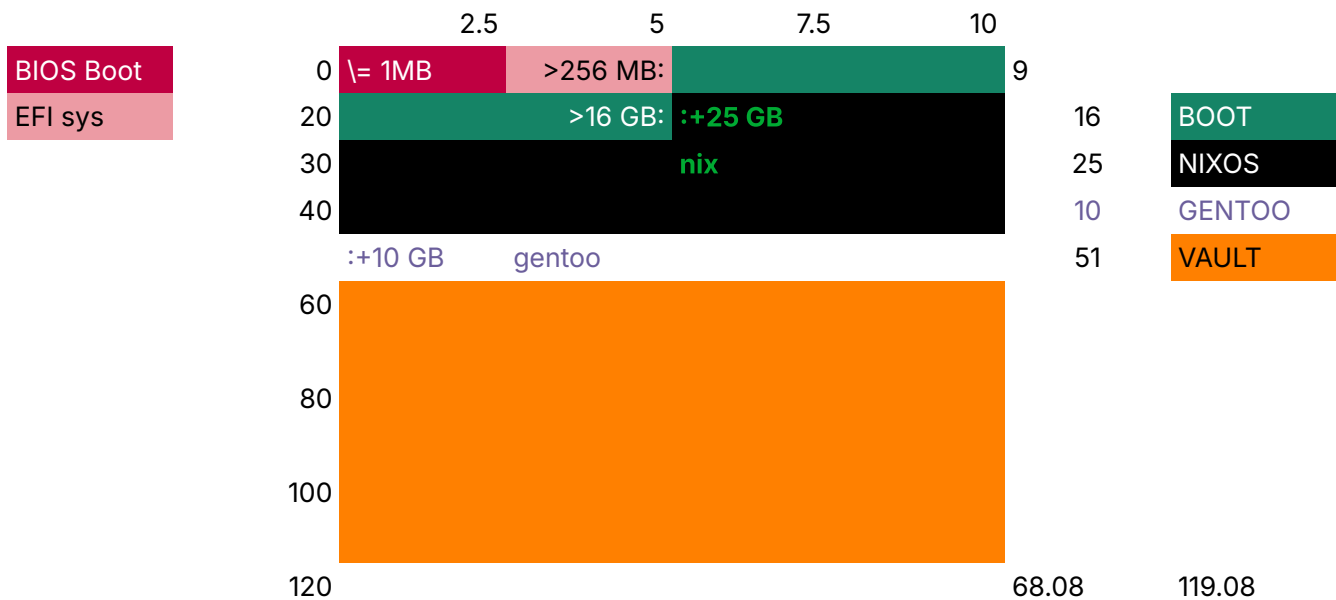
SUGGESTION:

Following these recent recommendations I was able to consolidate my thoughts somewhat. Ultimately I'm left with less than I had yesterday still bargained with, but am still above the 64 GB threshold. The volume is setup to be capable of booting BIOS and UEFI systems with focus on an iso volume and thereon contained images. Instinctively I would want to have a fallback installation – but technically we have two systems to operate with, so that shouldn't be the priority here either.

The calculation for the volume sizes is as follows: BIOS Boot size is required. EFI: Well, my experience has me believe that 256 MB is plenty. The "Data" partition (Boot "in the idea" entails all three) is calculated so:

Part 3:	>33554432	linux
arch linux	1.4326171875	
arch linux 32	0.77734375	
nixos minimal	1.4873046875	
nixos graphical	3.427734375	
gentoo minimal	0.76164436340332	
gentoo tarballs	1.78813934326172	
	(=7.5*250000)/(1024*1024)	
	9.67478370666504	GB
dwarf fortress	1.001953125	
mint	2.845703125	
v	13.522439956665	GB
16	GB	33554432

25 GB for Nixos just instinctively impressed itself on me, and going with 10 for Gentoo is simply because 10 Gigs have so far worked fine for me. Respectively we get to 51 GB total – and some part of me really doesn't like that. But so we're left with:



So, this is how much it gives us. It should absolutely be possible to spawn this setup onto a card via script. Then it comes down to filling it up with isos and managing grub.

=====
dangit. There's still something wrong with the prompt. When they're not properly implemented, the length of the prompt is not calculated properly. ...

=====

CUSTOMIZATION:example:

bootsplash:

```
/etc/default/grub:  
(find line): GRUB_BACKGROUND and edit file  
update /boot/grub/grub.cfg  
    > delete/rename old config  
    grub-mkconfig -o /boot/grub/grub.cfg
```

:example:**CUSTOMIZATION**

=====