System update preparations

- How to free space on the root filesystem (before an update)

How to free space on the root filesystem (before an update)

This is a copy! The original has been posted by nephros in the SFOS forum and may be more up-to-date.

Having enough space on the root filesystem before an update is a common problem.

This guide intends to give some hints how to free up space.

First things first:

Do not ignore the recommended free space requirements in the Release Notes, and the Update tool!

OTA updates frequently fail if these recommendations are ignored.

Also: Free space on the `/home` partition is not relevant here. Freeing up caches, or removing other data like Pictures will **not** help.
Neither will uninstalling Android Apps, as these are also installed in Home.

That being said, `/home` should also have some free space (at least 1GB) before an update is attempted. Do not update while your `/home` partition is full.

WARNING WARNING WARNING

#0 You presumably have a brain. Use it. (And do use your own brain, not some hallucinating large language model's ("AI") imitation of one.)
Double check instructions, seek help if you're not sure about something.

If you break something using the content below, you get to keep all the pieces.

#1 If possible, all the removal/freeing up space should be done **before** selecting "Download Update" in the UI.

#2 Always make sure you are dealing with files in the root (`/`) partition only. There are many other partitions mounted (`/apex`, `/vendor` and so on). IGNORE THEM. Do not delete anything from there. You have a high chance of bricking your device if you do, and it will also do no good for the issue we are trying to solve.
Just ignore them.

#3 Some sources will list the `rpm -e` or `rpm --erase` command.
Never use `rpm` to install or uninstall packages. (It is okay to use `rpm -q` to query information.)

#4 Naturally, you have backups. And have *tested* these backups.

The tools

While possible, finding things to remove using the UI is difficult.

We will be using command-line tools in this guide. This also means you will have to enable Developer Mode, at least during these activities.

If you're not familiar with the Linux/UNIX command line, or are uncomfortable using it, that's ok. You can still try if you're brave, but please try to understand what you're going to do, and do not blindly copy-and-paste commands from here. And you're almost always safe until the point where you become root, i.e. invoke `devel-su`.

Tip: Using command line tools in the Terminal app is a bit tedious. We recommend logging in via SSH from a proper computer. (There is also a ssh guide for Windows, but we're talking about proper computers here.)

Install the tools required like so:

```
# become root
devel-su

# while pkcon works for uninstalling, zypper can detect unnecessary dependencies and remove them too. So we use zypper:
pkcon install zypper
# refresh the package list
zypper ref

# ncdu is a graphical tool for showing disk usage.
```

```
zypper in ncdu
```

1. Finding space eaters

Checking free space

You can check the free space on the root filesystem like this:

```
df -h /
```

Example output:

```
nemo@PGXperiiia10:~ $ df -h /
Filesystem              Size      Used Available Use% Mounted on
/dev/sailfish/root       4.7G      4.4G    333.7M  93% /
```

Note the `Available` column. We want to maximize this.
You should regularly check this to see whether you are done. Of course, the more free space, the better.

Checking whether something is installed:

Either tool can be used:

```
pkcon search --filter installed SEARCHTERM
zypper search --installed-only SEARCHTERM
```

Example output:

```
nemo@PGXperiiia10:~ $ zypper search --installed-only whisper
Loading repository data...
Reading installed packages...

S  | Name                          | Summary                                       | Type
---+-------------------------------+-----------------------------------------------+--------
i+ | harbour-whisperfish           | Private messaging using Signal for SailfishOS | package
i+ | harbour-whisperfish-shareplugin | Share plugin for Whisperfish                | package
```

```
nemo@PGXperiiia10:~ $ pkcon search --filter installed whisper
Searching by
details
Starting

Refreshing software
list
Querying

Finished

Installed      harbour-whisperfish-0.6.0.beta.40-1.aarch64 (installed)       Private messaging using Signal for
SailfishOS
Installed      harbour-whisperfish-shareplugin-0.6.0.beta.20-0.sf4.aarch64 (installed) Share plugin for Whisperfish
```

Finding out what is using up space

Something like `rpm -qa --queryformat '%{SIZE} %{NAME} \n' | sort -n -r | head -20` will list the 20 largest installed RPMs.

`ncdu -rx /` can be used to find space-eaters. It is a graphical (TUI) app which lets you navigate the file system. Always use the `-x` switch, it ensures that you only look at the correct partition, and not other mounted file systems. `-r` means read-only, making sure you won't unintentionally delete any files.

Here's a "screenshot" of `ncdu`:

```
ncdu 1.22 ~ Use the arrow keys to navigate, press ? for
help                                                                                    [read-only]
--- /usr
-------------------------------------------------------------------------------------------------------------------------------------------------------------
------------
. 939.9 MiB [##########################]
/share
  845.4 MiB [#######################   ] /lib64
  456.8 MiB [############              ] /bin
  322.7 MiB [########                  ] /libexec
  167.6 MiB [####                      ] /lib
   40.4 MiB [#                         ] /sbin
```

```
  34.0 MiB [                ] /include
 140.0 KiB [                ] /local
  12.0 KiB [                ] /src
e   4.0 KiB [                ] /games
@   0.0   B [                ]  tmp
```

Tip: `ncdu` can drop you into a shell in the selected directory. This is the procedure I use to uninstall space eaters:

- launch `ncdu -rx /`
- navigate to a large directory (usually this will be something in `/usr/share` or `/usr/lib`.
- press `b` (short for bash) to drop you into a shell
- in the shell session, use `rpm -qf FILENAME` to find which package owns the file(s)
- optionally issue `devel-su pkcon remove -u packagename` to get rid of the space-eater.
- press `Ctrl-D` or issue the `exit` command to get back into `ncdu`.
- repeat.

Of course you can also use `Ctrl-Z` to send ncdu to the background, and call it back with `fg`.

Note: You primarily should be looking for stuff under `/usr/share`. Maybe `/usr/lib*`, but that's dangerous. No matter what you do, things like `/system` `/vendor` and the like are absolutely off-limits (see the note above).

Keeping backups

If you want you can download a back-up of any RPM package you remove. If you do, do store it under `/home` :wink: Image not found or type unknown

Example:

```
nemo@PGXperiiia10:~ $ mkdir ~/Downloads/RPM-Packages/
nemo@PGXperiiia10:~ $ pkcon download ~/Downloads/RPM-Packages/ PACKAGENAME
```

2. Removing Applications

While you can remove apps by long-pressing on their icons, or from Jolla Store, uninstalling this way may leave dependencies installed. So again, we use zypper.

Again, removing Android Apps is not necessary or meaningful.

With the exception of AppSupport, uninstalling an app does not remove your user data, and you can simply re-install them after the update.

As a rule of thumb, packages starting with `harbour-` are generally safe to be removed and can be re-installed later. Be careful with anything with `sailfishos` or `jolla` in the name. Do not touch things with `patterns`, `hybris`, `droid`, or `lipstick` in the name. Oh, and `glibc` stays.

Invoking zypper

WARNING WARNING WARNING
Zypper will warn you if you try to uninstall critical packages.
If zypper says:

```
This request will break your system!
```

it means it.
Never ignore these warnings. Never choose `ignore`. Never use `zypper --no-confirm`.

Do use `--dry-run` if you're not sure!

The general procedure is invoking zypper like this, replacing `"[PACKAGENAME]"` by the RPM package name (not necessarily the application name) to uninstall.

```
# become root
devel-su
# check what will be done
zypper remove -u --details --dry-run "[PACKAGENAME]"
# do it
zypper remove -u "[PACKAGENAME]"
```

Note the `-u` parameter, this will cause zypper to also uninstall dependencies which are not required any more.

Here's a list of Apps which are quite large, and if you can, you should remove them first.

Jolla apps

- Andoid™ AppSupport
  AppSupport is one of the largest packages around and uninstalling it will quickly free up a lot of space. However, uninstalling this may have unintended side effects.
- Mozilla Location Services data

Most other Jolla apps are quite small and you can probably just leave them alone.

Other Sailfish apps from various sources

- harbour-dsnote

- harbour-machines-vs-machines-sfos
- harbour-multimodal
- harbour-jupii
- harbour-fernschreiber
- harbour-whisperfish
- harbour-maelstrom
- harbour-owncloud
- harbour-saildiscord
- harbour-sailpipe
- stellarium
- harbour-sshazam
- anything Qt6-related from Chum (like Angelfish)

3. Removing other packages

Qt packages from Chum

These are quite a lot of packages, and removing them one by one can be tedious. So we again invoke zypper's dependency magic and call it on the base package:

```
zypper remove -u qt6-qtbase
```

The Chum packages for Qt5.15 install into `/opt`. Depending on the device, `/opt` may or may not be located on the root partition. If it is, removing these makes sense, otherwise, probably not:

```
zypper remove -u opt-qt5-qtbase
```

Afterwards, find any stray packages like so:

```
find /opt/qt5/ -type f -exec rpm -qf {} + | sort -u
find /usr/lib64/qt6/ -type f -exec rpm -qf {} + | sort -u
```

Python or Perl

Apps which depend on Python or Perl can pull in quite large dependencies.

```
rpm -q --whatrequires /usr/bin/python3
rpm -q --whatrequires /usr/bin/perl
```

will list such packages (and the dependencies)

Other large packages (mainly developer tools)

- rclone
- vim (yes, really!) (30MB)
- git (13MB, plus it pulls in perl with 30MB)
- osc, spectacle (because Python)
- any -devel packages
- gcc, gdb, valgrind, cmake, autotools, …

4. Deleting other files

DANGER DANGER DANGER

This part is intended for desperate cases, and should **ONLY** be attempted by users who know what they are doing here.

You have been warned!

I know what I'm doing

The following locations contain system files which can not easily be removed using a package, but may be deleted individually.
Deleting this stuff should be safe, as the files will be re-installed during the update.

Do **NOT** simply delete the whole directory! Rather, delete individual files, e.g. translation files for languages you do not use. And be careful with your wildcards, do use `rm -i`!

- `/usr/share/locale` (43MB),
- `/usr/share/translations` (42MB)
- `/usr/share/ambience` (50MB)
- `/usr/share/fonts/wqy-zenhei` (12MB)
- `/etc/skel/Pictures` (14MB)
- `/etc/skel/Videos` (12MB)

Stuff in `/usr/share/doc` and `/usr/share/man` is *probably* also safe to delete.
Be careful about `/var/`.

You will be tempted by `/usr/lib64/xulrunner-qt5-*/libxul.so`. Leave it. It is not worth the pain.

Others:

- `/var/lib/plocate`
- Privoxy: If using AB2P packages `/usr/share/harbour-privoxy/conf/extra` can be quite large. Delete them, and reinstall the AB2P package after the update.

If something goes wrong

Error recovery is a bit out-of-scope of this guide, and it will depend very much on the actual problem. But we can offer this:

Save your session or command line history

Depending on the terminal you use, you may be able to copy the output of your terminal session somewhere.
Doing this will be very useful when asking for help.

If you're using `screen`, you can scroll back in the buffer by issuing `Ctrl-A Esc`, and use the cursor keys to scroll up. Pressing space will start copy mode. Pressing space again will copy the selected text into the (screen) clipboard. issuing `Ctrl-A ]` will paste these clipboard contents.
So after copying you could do:

```
cat > session.log
Ctrl-A ]
Ctrl-D
```

To save the history of commands you issued, you can

```
history > ~/session_history.txt
```

Package install/uninstall history

The package manager keeps a log of events at `/var/log/zypp/history`.
While that file is a little hard to read, it can be used to find out which packages were installed or uninstalled, whether that was successful, and when that happened.

Shameles Plug: this author's app "Install History" can be used to make the file contents a bit more readable. And you get bar charts!

Restoring files

So you have deleted or otherwise or changed some file and would like to get it back.

You can find out which package owns a file using

```
rpm -qf filename or path
```

You can re-install a package like so:

```
zypper install --force packagename
pkcon install --allow-reinstall packagename
```

If you want to find out the original content of a file (e.g. a config), you can download the RPM, unpack it, and look at the files:

```
# download the package
mkdir ~/Downloads/rpms
pkcon download ~/Downloads/rpms packagename
# find out the exact file name
ls ~/Downloads/rpms
# unpack the contents into a temporary directory
mkdir ~/unpack
cd ~/unpack
rpm2cpio ~/Downloads/rpms/package-1.2.3.aarch64.rpm | cpio -id
# ~/unpack will now contain all the original files.
```