
Release-Notes for Debian 14 (forky)

—DRAFT—

Debian Documentation Team

2026-08-30

CONTENTS

1	Introduction —DRAFT—	3
1.1	Reporting bugs on this document	3
1.2	Contributing upgrade reports	3
1.3	Sources for this document	4
2	What’s new in Debian 14 —DRAFT—	5
2.1	Supported architectures	5
2.2	What’s new in the distribution?	5
2.2.1	Desktops and well known packages	5
3	Installation System —DRAFT—	7
3.1	What’s new in the installation system?	7
3.2	Installing Debian Pure Blends	7
3.3	Cloud installations	7
3.4	Container and Virtual Machine images	8
4	Upgrades from Debian 13 (trixie) —DRAFT—	9
4.1	Preparing for the upgrade	9
4.1.1	Back up any data or configuration information	9
4.1.2	Inform users in advance	9
4.1.3	Prepare for downtime on services	10
4.1.4	Prepare for recovery	10
4.1.5	Prepare a safe environment for the upgrade	11
4.2	Start from “pure” Debian	11
4.2.1	Upgrade to Debian 13 (trixie)	12
4.2.2	Upgrade to latest point release	12
4.2.3	Debian Backports	12
4.2.4	Prepare the package database	12
4.2.5	Remove obsolete packages	12
4.2.6	Remove non-Debian packages	12
4.2.7	Clean up leftover configuration files	13
4.2.8	The non-free and non-free-firmware components	13
4.2.9	The proposed-updates section	13
4.2.10	Unofficial sources	13
4.2.11	Disabling APT pinning	13
4.2.12	Check package status	13
4.3	Preparing APT sources files	14
4.3.1	Adding APT Internet sources	14
4.3.2	Adding APT sources for a local mirror	15
4.3.3	Adding APT sources from optical media	16

4.4	Upgrading packages	16
4.4.1	Recording the session	16
4.4.2	Updating the package list	17
4.4.3	Make sure you have sufficient space for the upgrade	17
4.4.4	Stop monitoring systems	18
4.4.5	Minimal system upgrade	18
4.4.6	Upgrading the system	19
4.5	Possible issues during upgrade	19
4.5.1	Full-upgrade fails with “Could not perform immediate configuration”	19
4.5.2	Expected removals	19
4.5.3	Conflicts or Pre-Depends loops	20
4.5.4	File conflicts	20
4.5.5	Configuration changes	20
4.5.6	Change of session to console	20
4.6	Upgrading your kernel and related packages	21
4.6.1	Installing a kernel metapackage	21
4.7	Cleanup after the upgrade	21
4.8	Cleaning up automatically installed packages	22
4.9	Obsolete packages	22
4.9.1	Purging removed packages	22
4.9.2	Transitional dummy packages	23
5	Issues to be aware of for forky —DRAFT—	25
5.1	Things to be aware of while upgrading to forky	25
5.1.1	Things to do before rebooting	25
5.2	Items not limited to the upgrade process	25
5.2.1	Limitations in security support	25
5.3	Obsolescence and deprecation	26
5.3.1	Noteworthy obsolete packages	26
5.3.2	Deprecated components for forky	26
5.4	Known severe bugs	26
6	More information on Debian —DRAFT—	27
6.1	Further reading	27
6.2	Getting help	27
6.2.1	Mailing lists	27
6.2.2	Internet Relay Chat	27
6.3	Reporting bugs	27
6.4	Contributing to Debian	28
7	Managing your trixie system before the upgrade —DRAFT—	29
7.1	Upgrading your trixie system	29
7.2	Checking your APT configuration	29
7.3	Performing the upgrade to latest trixie release	30
7.4	Removing obsolete configuration files	30
8	Contributors to the Release Notes —DRAFT—	31

The Debian Documentation Project <<https://www.debian.org/doc>>.

Last updated on: 2026-08-30

This document is free software; you can redistribute it and/or modify it under the terms of the GNU General Public License, version 2, as published by the Free Software Foundation.

This program is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU General Public License for more details.

You should have received a copy of the GNU General Public License along with this program; if not, the license text can also be found at <https://www.gnu.org/licenses/gpl-2.0.html> and in `/usr/share/common-licenses/GPL-2` on Debian systems.

INTRODUCTION —DRAFT—

This document informs users of the Debian distribution about major changes in version 14 (codenamed forky).

The release notes provide information on how to upgrade safely from release 13 (codenamed trixie) to the current release and inform users of known potential issues they could encounter in that process.

You can get the most recent version of this document from <https://www.debian.org/releases/forky/releasenotes>.

Caution

Note that it is impossible to list every known issue and that therefore a selection has been made based on a combination of the expected prevalence and impact of issues.

Please note that we only support and document upgrading from the previous release of Debian (in this case, the upgrade from trixie). If you need to upgrade from older releases, we suggest you read previous editions of the release notes and upgrade to trixie first.

1.1 Reporting bugs on this document

We have attempted to test all the different upgrade steps described in this document and to anticipate all the possible issues our users might encounter.

Nevertheless, if you think you have found a bug (incorrect information or information that is missing) in this documentation, please file a bug in the [bug tracking system](#) against the **release-notes** package. You might first want to review the [existing bug reports](#) in case the issue you've found has already been reported. Feel free to add additional information to existing bug reports if you can contribute content for this document.

We appreciate, and encourage, reports providing patches to the document's sources. You will find more information describing how to obtain the sources of this document in [Sources for this document](#).

1.2 Contributing upgrade reports

We welcome any information from users related to upgrades from trixie to forky. If you are willing to share information please file a bug in the [bug tracking system](#) against the **upgrade-reports** package with your results. We request that you compress any attachments that are included (using gzip).

Please include the following information when submitting your upgrade report:

- The status of your package database before and after the upgrade: **dpkg**'s status database available at `/var/lib/dpkg/status` and **apt**'s package state information, available at `/var/lib/apt/extended_states`. You should have made a backup before the upgrade as described at [Back up any data or configuration information](#), but you can also find backups of `/var/lib/dpkg/status` in `/var/backups`.

- Your `apt` logs, available at `/var/log/apt/term.log`, or your `aptitude` logs, available at `/var/log/aptitude`.

Note

You should take some time to review and remove any sensitive and/or confidential information from the logs before including them in a bug report as the information will be published in a public database.

1.3 Sources for this document

The source of this document is in reStructuredText format, using the sphinx converter. The HTML version is generated using `sphinx-build -b html`. The PDF version is generated using `sphinx-build -b latex`. Sources for the Release Notes are available in the Git repository of the *Debian Documentation Project*. You can use the [web interface](#) to access its files individually through the web and see their changes. For more information on how to access Git please consult the [Debian Documentation Project VCS information pages](#).

WHAT'S NEW IN DEBIAN 14 —DRAFT—

The [Wiki](#) has more information about this topic.

2.1 Supported architectures

The following are the officially supported architectures for Debian 14:

- 64-bit PC (amd64)
- 64-bit ARM (arm64)
- ARMv7 (EABI hard-float ABI, armhf)
- 64-bit little-endian PowerPC (ppc64el)
- 64-bit little-endian RISC-V (riscv64)
- IBM System z (s390x)

Additionally, on 64-bit PC systems, a partial 32-bit userland (i386) is available.

You can read more about port status, and port-specific information for your architecture at the [Debian port web pages](#).

2.2 What's new in the distribution?

2.2.1 Desktops and well known packages

This new release of Debian comes with a lot more software than its predecessor trixie; the distribution includes over 14116 new packages, for a total of over 69830 packages. Most of the software in the distribution has been updated: over 44326 software packages (this is 63% of all packages in trixie). Also, a significant number of packages (over 8844, 12% of the packages in trixie) have for various reasons been removed from the distribution. You will not see any updates for these packages and they will be marked as “obsolete” in package management front-ends; see *[Obsolete packages](#)*.

Debian again ships with several desktop applications and environments. Among others it now includes the desktop environments GNOME 48, KDE Plasma 6.3, LXDE 13, LXQt 2.1.0, and Xfce 4.20.

Productivity applications have also been upgraded, including the office suites:

- LibreOffice is upgraded to version 25;
- GNUMcash is upgraded to 5.10;

Among many others, this release also includes the following software updates:

Package	Version in 13 (trixie)	Version in 14 (forky)
Apache	2.4.62	2.4.65
Bash	5.2.15	5.2.37
BIND DNS Server	9.18	9.20
Cryptsetup	2.6	2.7
curl/libcurl	7.88.1	8.14.1
Emacs	28.2	30.1
Exim (default email server)	4.96	4.98
GCC, the GNU Compiler Collection (default compiler)	12.2	14.2
GIMP	2.10.34	3.0.4
GnuPG	2.2.40	2.4.7
Inkscape	1.2.2	1.4
the GNU C library	2.36	2.41
Linux kernel	6.1 series	6.12 series
LLVM/Clang toolchain	13.0.1 and 14.0 (default) and 15.0.6	19 (default), 17 and 18 available
MariaDB	10.11	11.8
Nginx	1.22	1.26
OpenJDK	17	21
OpenLDAP	2.5.13	2.6.10
OpenSSH	9.2p1	10.0p1
OpenSSL	3.0	3.5
Perl	5.36	5.40
PHP	8.2	8.4
Postfix	3.7	3.10
PostgreSQL	15	17
Python 3	3.11	3.13
Qt 5	5.15.8	5.15.15
Qt 6	6.4.2	6.8.2
Rustc	1.63	1.85
Samba	4.17	4.22
Systemd	252	257
Vim	9.0	9.1

INSTALLATION SYSTEM —DRAFT—

The Debian Installer is the official installation system for Debian. It offers a variety of installation methods. The methods that are available to install your system depend on its architecture.

Images of the installer for forky can be found together with the Installation Guide on the Debian website (<https://www.debian.org/releases/forky/debian-installer/>).

The Installation Guide is also included on the first media of the official Debian DVD (CD/blu-ray) sets, at:

```
/doc/install/manual/language/index.html
```

You may also want to check the errata for debian-installer at <https://www.debian.org/releases/forky/debian-installer#errata> for a list of known issues.

3.1 What's new in the installation system?

There has been a lot of development on the Debian Installer since its previous official release with Debian 13, resulting in improved hardware support and some exciting new features or improvements.

If you are interested in an overview of the changes since trixie, please check the release announcements for the forky beta and RC releases available from the Debian Installer's [news history](#).

3.2 Installing Debian Pure Blends

A selection of Debian Pure Blends, such as Debian Junior, Debian Science, or Debian FreedomBox, can now be accessed directly in the installer - see the [installation-guide](#).

For information about Debian Pure Blends, visit <https://www.debian.org/blends/> or the [wiki](#).

3.3 Cloud installations

The [cloud team](#) publishes Debian forky for several popular cloud computing services including:

- Amazon Web Services
- Microsoft Azure
- OpenStack
- Plain VM

Cloud images provide automation hooks via `cloud-init` and prioritize fast instance startup using specifically optimized kernel packages and grub configurations. Images supporting different architectures are provided where appropriate and the cloud team endeavors to support all features offered by the cloud service.

The cloud team will provide updated images until the end of the LTS period for forky. New images are typically released for each point release and after security fixes for critical packages. The cloud team’s full support policy is available on the [Cloud Image Lifecycle page](#).

More details are available at <https://cloud.debian.org/> and on the [wiki](#).

3.4 Container and Virtual Machine images

Multi-architecture Debian forky container images are available on [Docker Hub](#). In addition to the standard images, a “slim” variant is available that reduces disk usage.

UPGRADES FROM DEBIAN 13 (TRIXIE) —DRAFT—

4.1 Preparing for the upgrade

We suggest that before upgrading you also read the information in *Issues to be aware of for forky —DRAFT—*. That chapter covers potential issues which are not directly related to the upgrade process but could still be important to know about before you begin.

4.1.1 Back up any data or configuration information

Before upgrading your system, it is strongly recommended that you make a full backup, or at least back up any data or configuration information you can't afford to lose. The upgrade tools and process are quite reliable, but a hardware failure in the middle of an upgrade could result in a severely damaged system.

The main things you'll want to back up are the contents of `/etc`, `/var/lib/dpkg`, `/var/lib/apt/extended_states` and the output of:

```
$ dpkg --get-selections '*' # (the quotes are important)
```

If you use `aptitude` to manage packages on your system, you will also want to back up `/var/lib/aptitude/pkgstates`.

The upgrade process itself does not modify anything in the `/home` directory. However, some applications (e.g. parts of the Mozilla suite, and the GNOME and KDE desktop environments) are known to overwrite existing user settings with new defaults when a new version of the application is first started by a user. As a precaution, you may want to make a backup of the hidden files and directories (“dotfiles”) in users’ home directories. This backup may help to restore or recreate the old settings. You may also want to inform users about this.

Any package installation operation must be run with superuser privileges, so either log in as `root` or use `su` or `sudo` to gain the necessary access rights.

The upgrade has a few preconditions; you should check them before actually executing the upgrade.

4.1.2 Inform users in advance

It's wise to inform all users in advance of any upgrades you're planning, although users accessing your system via an `ssh` connection should notice little during the upgrade, and should be able to continue working.

If you wish to take extra precautions, back up or unmount the `/home` partition before upgrading.

You will have to do a kernel upgrade when upgrading to `forky`, so a reboot will be necessary. Typically, this will be done after the upgrade is finished.

4.1.3 Prepare for downtime on services

There might be services that are offered by the system which are associated with packages that will be included in the upgrade. If this is the case, please note that, during the upgrade, these services will be stopped while their associated packages are being replaced and configured. During this time, these services will not be available.

The precise downtime for these services will vary depending on the number of packages being upgraded in the system, and it also includes the time the system administrator spends answering any configuration questions from package upgrades. Notice that if the upgrade process is left unattended and the system requests input during the upgrade there is a high possibility of services being unavailable¹ for a significant period of time.

If the system being upgraded provides critical services for your users or the network², you can reduce the downtime if you do a minimal system upgrade, as described in *Minimal system upgrade*, followed by a kernel upgrade and reboot, and then upgrade the packages associated with your critical services. Upgrade these packages prior to doing the full upgrade described in *Upgrading the system*. This way you can ensure that these critical services are running and available through the full upgrade process, and their downtime is reduced.

4.1.4 Prepare for recovery

Although Debian tries to ensure that your system stays bootable at all times, there is always a chance that you may experience problems rebooting your system after the upgrade. Known potential issues are documented in this and the next chapters of these Release Notes.

For this reason it makes sense to ensure that you will be able to recover if your system should fail to reboot or, for remotely managed systems, fail to bring up networking.

If you are upgrading remotely via an ssh link it is recommended that you take the necessary precautions to be able to access the server through a remote serial terminal. There is a chance that, after upgrading the kernel and rebooting, you will have to fix the system configuration through a local console. Also, if the system is rebooted accidentally in the middle of an upgrade there is a chance you will need to recover using a local console.

For emergency recovery we generally recommend using the *rescue mode* of the forky Debian Installer. The advantage of using the installer is that you can choose between its many methods to find one that best suits your situation. For more information, please consult the section “Recovering a Broken System” in chapter 8 of the Installation Guide (at <https://www.debian.org/releases/forky/installmanual>) and the [Debian Installer FAQ](#).

If that fails, you will need an alternative way to boot your system so you can access and repair it. One option is to use a special rescue or *live install* image. After booting from that, you should be able to mount your root file system and chroot into it to investigate and fix the problem.

Debug shell during boot using initrd

The **initramfs-tools** package includes a debug shell³ in the initrds it generates. If for example the initrd is unable to mount your root file system, you will be dropped into this debug shell which has basic commands available to help trace the problem and possibly fix it.

Basic things to check are: presence of correct device files in `/dev`; what modules are loaded (`cat /proc/modules`); output of `dmesg` for errors loading drivers. The output of `dmesg` will also show what device files have been assigned to which disks; you should check that against the output of `echo $ROOT` to make sure that the root file system is on the expected device.

If you do manage to fix the problem, typing `exit` will quit the debug shell and continue the boot process at the point it failed. Of course you will also need to fix the underlying problem and regenerate the initrd so the next boot won't fail again.

¹ If the debconf priority is set to a very high level you might prevent configuration prompts, but services that rely on default answers that are not applicable to your system will fail to start.

² For example: DNS or DHCP services, especially when there is no redundancy or failover. In the DHCP case end-users might be disconnected from the network if the lease time is lower than the time it takes for the upgrade process to complete.

³ This feature can be disabled by adding the parameter `panic=0` to your boot parameters.

Debug shell during boot using systemd

If the boot fails under systemd, it is possible to obtain a debug root shell by changing the kernel command line. If the basic boot succeeds, but some services fail to start, it may be useful to add `systemd.unit=rescue.target` to the kernel parameters.

Otherwise, the kernel parameter `systemd.unit=emergency.target` will provide you with a root shell at the earliest possible point. However, this is done before mounting the root file system with read-write permissions. You will have to do that manually with:

```
# mount -o remount,rw /
```

Another approach is to enable the systemd “early debug shell” via the `debug-shell.service`. On the next boot this service opens a root login shell on `tty9` very early in the boot process. It can be enabled with the kernel boot parameter `systemd.debug-shell=1`, or made persistent with `systemctl enable debug-shell` (in which case it should be disabled again when debugging is completed).

More information on debugging a broken boot under systemd can be found in the [Freedesktop.org Diagnosing Boot Problems](https://freedesktop.org/Diagnosing-Boot-Problems) article.

4.1.5 Prepare a safe environment for the upgrade

Important

If you are using some VPN services (such as **tiny**) consider that they might not be available throughout the upgrade process. Please see [Prepare for downtime on services](#).

In order to gain extra safety margin when upgrading remotely, we suggest that you run upgrade processes in a virtual console provided by the `screen` or `tmux` programs, which enables safe reconnection and ensures the upgrade process is not interrupted even if the remote connection process temporarily fails.

In case `tmux` was upgraded to a new major version you may get an error on attach: “open terminal failed: not a terminal”. You can still access the old session with:

```
# /proc/$(pgrep "tmux: server")/exe attach
```

Users of the watchdog daemon provided by the **micro-evtd** package should stop the daemon and disable the watchdog timer before the upgrade, to avoid a spurious reboot in the middle of the upgrade process:

```
# service micro-evtd stop
# /usr/sbin/microap1 -a system_set_watchdog off
```

4.2 Start from “pure” Debian

The upgrade process described in this chapter has been designed for “pure” Debian stable systems. APT controls what is installed on your system. If your APT configuration mentions additional sources besides `trixie`, or if you have installed packages from other releases or from third parties, then to ensure a reliable upgrade process you may wish to begin by removing these complicating factors.

APT is moving to a different format for configuring where it downloads packages from. The files `/etc/apt/sources.list` and `*.list` files in `/etc/apt/sources.list.d/` are replaced by files still in that directory but with names ending in `.sources`, using the new, more readable (deb822 style) format. For details see [sources.list\(5\)](#). Examples of APT configurations in these notes will be given in the new deb822 format.

If your system is using multiple sources files then you will need to ensure they stay consistent.

4.2.1 Upgrade to Debian 13 (trixie)

Only upgrades from Debian 13 (trixie) are supported. Display your Debian version with:

```
$ cat /etc/debian_version
```

Please follow the instructions in the Release Notes for Debian 13 at <https://www.debian.org/releases/trixie/releasenotes> to upgrade to Debian 13 first if needed.

4.2.2 Upgrade to latest point release

This procedure assumes your system has been updated to the latest point release of trixie. If you have not done this or are unsure, follow the instructions in *Upgrading your trixie system*.

4.2.3 Debian Backports

Debian Backports allows users of Debian stable to run more up-to-date versions of packages (with some tradeoffs in testing and security support). The Debian Backports Team maintains a subset of packages from the next Debian release, adjusted and recompiled for usage on the current Debian stable release.

Packages from trixie-backports have version numbers lower than the version in forky, so they should upgrade normally to forky in the same way as “pure” trixie packages during the distribution upgrade. While there are no known potential issues, the upgrade paths from backports are less tested, and correspondingly incur more risk.

Caution

While regular Debian Backports are supported, there is no clean upgrade path from **sloppy** backports (which use APT sources entries referencing trixie-backports-sloppy).

As with *Unofficial sources*, users are advised to remove “trixie-backports” entries from their APT sources files before the upgrade. After it is completed, they may consider adding “forky-backports” (see <https://backports.debian.org/Instructions/>).

For more information, consult the [Backports Wiki](#) page.

4.2.4 Prepare the package database

You should make sure the package database is ready before proceeding with the upgrade. If you are a user of another package manager like **aptitude** or **synaptic**, review any pending actions. A package scheduled for installation or removal might interfere with the upgrade procedure. Note that correcting this is only possible if your APT sources files still point to “trixie” and not to “stable” or “forky”; see *Checking your APT configuration*.

4.2.5 Remove obsolete packages

It is a good idea to *remove obsolete packages* from your system before upgrading. They may introduce complications during the upgrade process, and can present security risks as they are no longer maintained.

4.2.6 Remove non-Debian packages

Below there are two methods for finding installed packages that did not come from Debian, using either **apt** or **apt-forktracer**. Please note that neither of them are 100% accurate (e.g. the **apt** example will list packages that were once provided by Debian but no longer are, such as old kernel packages).

```
$ apt list '?narrow(?installed, ?not(?origin(Debian)))'  
$ apt-forktracer | sort
```

4.2.7 Clean up leftover configuration files

A previous upgrade may have left unused copies of configuration files; *old versions* of configuration files, versions supplied by the package maintainers, etc. Removing leftover files from previous upgrades can avoid confusion. Find such leftover files with:

```
# find /etc -name '*.dpkg-*' -o -name '*.ucf-*' -o -name '*.merge-error'
```

4.2.8 The non-free and non-free-firmware components

If you have non-free firmware installed it is recommended to add `non-free-firmware` to your APT sources.

4.2.9 The proposed-updates section

If you have listed the `proposed-updates` section in your APT sources files, you should remove it before attempting to upgrade your system. This is a precaution to reduce the likelihood of conflicts.

4.2.10 Unofficial sources

If you have any non-Debian packages on your system, you should be aware that these may be removed during the upgrade because of conflicting dependencies. If these packages were installed by adding an extra package archive in your APT sources files, you should check if that archive also offers packages compiled for forky and change the source item accordingly at the same time as your source items for Debian packages.

Some users may have *unofficial* backported “newer” versions of packages that *are* in Debian installed on their trixie system. Such packages are most likely to cause problems during an upgrade as they may result in file conflicts⁴. *Possible issues during upgrade* has some information on how to deal with file conflicts if they should occur.

4.2.11 Disabling APT pinning

If you have configured APT to install certain packages from a distribution other than stable (e.g. from testing), you may have to change your APT pinning configuration (stored in `/etc/apt/preferences` and `/etc/apt/preferences.d/`) to allow the upgrade of packages to the versions in the new stable release. Further information on APT pinning can be found in `apt_preferences(5)`.

4.2.12 Check package status

Regardless of the method used for upgrading, it is recommended that you check the status of all packages first, and verify that all packages are in an upgradable state. The following command will show any packages which have a status of Half-Installed or Failed-Config, and those with any error status.

```
$ dpkg --audit
```

You could also inspect the state of all packages on your system using `aptitude` or with commands such as

```
$ dpkg -l
```

or

```
# dpkg --get-selections '*' > ~/curr-pkgs.txt
```

Alternatively you can also use `apt`.

⁴ Debian’s package management system normally does not allow a package to remove or replace a file owned by another package unless it has been defined to replace that package.

```
# apt list --installed > ~/curr-pkgs.txt
```

It is desirable to remove any holds before upgrading. If any package that is essential for the upgrade is on hold, the upgrade will fail.

```
$ apt-mark showhold
```

If you changed and recompiled a package locally, and didn't rename it or put an epoch in the version, you must put it on hold to prevent it from being upgraded.

The “hold” package state for apt can be changed using:

```
# apt-mark hold package_name
```

Replace `hold` with `unhold` to unset the “hold” state.

If there is anything you need to fix, it is best to make sure your APT sources files still refer to trixie as explained in *Checking your APT configuration*.

4.3 Preparing APT sources files

Before starting the upgrade you must reconfigure APT to add sources for forky and typically remove sources for trixie.

As mentioned in *Start from “pure” Debian*, we recommend that you use the new deb822-style format, so you would have to replace `/etc/apt/sources.list` and any `*.list` files in `/etc/apt/sources.list.d/` by only one file named `debian.sources` in `/etc/apt/sources.list.d/` (if you haven't done so already). An example is given below of how this file should typically look.

APT will consider all packages that can be found via any configured archive, and install the package with the highest version number, giving priority to the first entry in the files. Thus, if you have multiple mirror locations, list first the ones on local hard disks, then CD-ROMs, and then remote mirrors.

A release can often be referred to both by its codename (e.g. “trixie”, “forky”) and by its status name (i.e. “oldstable”, “stable”, “testing”, “unstable”). Referring to a release by its codename has the advantage that you will never be surprised by a new release and for this reason is the approach taken here. It does of course mean that you will have to watch out for release announcements yourself. If you use the status name instead, you will just see loads of updates for packages available as soon as a release has happened.

Debian provides two announcement mailing lists to help you stay up to date on relevant information related to Debian releases:

- By [subscribing to the Debian announcement mailing list](#), you will receive a notification every time Debian makes a new release. Such as when “forky” changes from e.g. “testing” to “stable”.
- By [subscribing to the Debian security announcement mailing list](#), you will receive a notification every time Debian publishes a security announcement.

4.3.1 Adding APT Internet sources

On new installations the default is for APT to be set up to use the Debian APT CDN service, which should ensure that packages are automatically downloaded from a server near you in network terms. As this is a relatively new service, older installations may have configuration that still points to one of the main Debian Internet servers or one of the mirrors. If you haven't done so yet, it is recommended to switch over to the use of the CDN service in your APT configuration.

To make use of the CDN service, the correct configuration for APT (assuming you are using `main` and `non-free-firmware`) is the following in `/etc/apt/sources.list.d/debian.sources`:

Types: deb
 URIs: <https://deb.debian.org/debian>
 Suites: forky forky-updates
 Components: main non-free-firmware
 Signed-By: /usr/share/keyrings/debian-archive-keyring.gpg

Types: deb
 URIs: <https://security.debian.org/debian-security>
 Suites: forky-security
 Components: main non-free-firmware
 Signed-By: /usr/share/keyrings/debian-archive-keyring.gpg

Note

Starting in trixie the `.gpg` keyring pathnames are now backwards compatibility symlinks to the new `.pgp` canonical pathnames, which will eventually get removed.

Make sure to remove any of the old sources files.

However, if you get better results using a specific mirror that is close to you in network terms instead of the CDN service, then the mirror URI can be substituted in the URIs line as (for instance) “URIs: <https://mirrors.kernel.org/debian>”.

If you want to use packages from the `contrib` or `non-free` components, you may add these names to all the `Components:` lines.

After adding your new sources, disable the previously existing archive entries in the APT sources files by placing a hash sign (`#`) in front of them.

4.3.2 Adding APT sources for a local mirror

Instead of using remote package mirrors, you may wish to modify the APT sources files to use a mirror on a local disk (possibly mounted over NFS).

For example, your package mirror may be under `/var/local/debian/`, and have main directories like this:

```
/var/local/debian/dists/forky/main/...
/var/local/debian/dists/forky/contrib/...
```

To use this with **apt**, add the following to your `/etc/apt/sources.list.d/debian.sources` file:

Types: deb
 URIs: <file:/var/local/debian>
 Suites: forky
 Components: main non-free-firmware
 Signed-By: /usr/share/keyrings/debian-archive-keyring.gpg

Note

Starting in trixie the `.gpg` keyring pathnames are now backwards compatibility symlinks to the new `.pgp` canonical pathnames, which will eventually get removed.

Again, after adding your new sources, disable the previously existing archive entries.

4.3.3 Adding APT sources from optical media

If you want to use *only* DVDs (or CDs or Blu-ray Discs), comment out the existing entries in all the APT sources files by placing a hash sign (#) in front of them.

Make sure there is a line in `/etc/fstab` that enables mounting your CD-ROM drive at the `/media/cdrom` mount point. For example, if `/dev/sr0` is your CD-ROM drive, `/etc/fstab` should contain a line like:

```
/dev/sr0 /media/cdrom auto noauto,ro 0 0
```

Note that there must be *no spaces* between the words `noauto,ro` in the fourth field.

To verify it works, insert a CD and try running

```
# mount /media/cdrom      # this will mount the CD to the mount point
# ls -alF /media/cdrom    # this should show the CD's root directory
# umount /media/cdrom     # this will unmount the CD
```

Next, run:

```
# apt-cdrom add
```

for each Debian Binary CD-ROM you have, to add the data about each CD to APT's database.

4.4 Upgrading packages

The recommended way to upgrade from previous Debian releases is to use the package management tool `apt`.

Note

`apt` is meant for interactive use, and should not be used in scripts. In scripts one should use `apt-get`, which has a stable output better suitable for parsing.

Don't forget to mount all needed partitions (notably the root and `/usr` partitions) read-write, with a command like:

```
# mount -o remount,rw /mountpoint
```

Next you should double-check that the APT sources entries (in files under `/etc/apt/sources.list.d/`) refer either to “forky” or to “stable”. There should not be any sources entries pointing to trixie.

Note

Sources lines for a CD-ROM might sometimes refer to “unstable”; although this may be confusing, you should *not* change it.

4.4.1 Recording the session

`apt` will log the changed package states in `/var/log/apt/history.log` and the terminal output in `/var/log/apt/term.log`. `dpkg` will, in addition, log all package state changes in `/var/log/dpkg.log`. If you use `aptitude`, it will also log state changes in `/var/log/aptitude`.

If a problem occurs, you will have a log of what happened, and if needed, can provide exact information in a bug report.

The `term.log` will also allow you to review information that has scrolled off-screen. If you are at the system's console, just switch to VT2 (using `Alt+F2`) to review it.

4.4.2 Updating the package list

First the list of available packages for the new release needs to be fetched. This is done by executing:

```
# apt update
```

4.4.3 Make sure you have sufficient space for the upgrade

You have to make sure before upgrading your system that you will have sufficient hard disk space when you start the full system upgrade described in *Upgrading the system*. First, any package needed for installation that is fetched from the network is stored in `/var/cache/apt/archives` (and the `partial/` subdirectory, during download), so you must make sure you have enough space on the file system partition that holds `/var/` to temporarily download the packages that will be installed in your system. After the download, you will probably need more space in other file system partitions in order to both install upgraded packages (which might contain bigger binaries or more data) and new packages that will be pulled in for the upgrade. If your system does not have sufficient space you might end up with an incomplete upgrade that is difficult to recover from.

`apt` can show you detailed information about the disk space needed for the installation. Before executing the upgrade, you can see this estimate by running:

```
# apt -o APT::Get::Trivial-Only=true full-upgrade
[ ... ]
XXX upgraded, XXX newly installed, XXX to remove and XXX not upgraded.
Need to get xx.xMB of archives.
After this operation, AAAMB of additional disk space will be used.
```

Note

Running this command at the beginning of the upgrade process may give an error, for the reasons described in the next sections. In that case you will need to wait until you've done the minimal system upgrade as in *Minimal system upgrade* before running this command to estimate the disk space.

If you do not have enough space for the upgrade, `apt` will warn you with a message like this:

```
E: You don't have enough free space in /var/cache/apt/archives/.
```

In this situation, make sure you free up space beforehand. You can:

- Remove packages that have been previously downloaded for installation (at `/var/cache/apt/archives`). Cleaning up the package cache by running `apt clean` will remove all previously downloaded package files.
- Remove forgotten packages. If you have used `aptitude` or `apt` to manually install packages in `trixie` it will have kept track of those packages you manually installed, and will be able to mark as redundant those packages pulled in by dependencies alone which are no longer needed due to a package being removed. They will not mark for removal packages that you manually installed. To remove automatically installed packages that are no longer used, run:

```
# apt autoremove
```

You can also use `debfoister` to find redundant packages. Do not blindly remove the packages this tool presents, especially if you are using aggressive non-default options that are prone to false positives. It is highly recommended that you manually review the packages suggested for removal (i.e. their contents, sizes, and descriptions) before you remove them.

- Remove packages that take up too much space and are not currently needed (you can always reinstall them after the upgrade). If you have `popularity-contest` installed, you can use `popcon-largest-unused` to list the packages

you do not use that occupy the most space. You can find the packages that just take up the most disk space with `dpigs` (available in the **debian-goodies** package) or with `wajig` (running `wajig size`). They can also be found with **aptitude**. Start `aptitude` in full-terminal mode, select `Views > New Flat Package List`, press `l` and enter `~i`, then press `S` and enter `~installsize`. This will give you a handy list to work with.

- Remove translations and localization files from the system if they are not needed. You can install the **localepurge** package and configure it so that only a few selected locales are kept in the system. This will reduce the disk space consumed at `/usr/share/locale`.
- Temporarily move to another system, or permanently remove, system logs residing under `/var/log/`.
- Use a temporary `/var/cache/apt/archives`: You can use a temporary cache directory from another filesystem (USB storage device, temporary hard disk, filesystem already in use, ...).

Note

Do not use an NFS mount as the network connection could be interrupted during the upgrade.

For example, if you have a USB drive mounted on `/media/usbkey`:

1. remove the packages that have been previously downloaded for installation:

```
# apt clean
```

2. copy the directory `/var/cache/apt/archives` to the USB drive:

```
# cp -ax /var/cache/apt/archives /media/usbkey/
```

3. mount the temporary cache directory on the current one:

```
# mount --bind /media/usbkey/archives /var/cache/apt/archives
```

4. after the upgrade, restore the original `/var/cache/apt/archives` directory:

```
# umount /var/cache/apt/archives
```

5. remove the remaining `/media/usbkey/archives`.

You can create the temporary cache directory on whatever filesystem is mounted on your system.

- Do a minimal upgrade of the system (see [Minimal system upgrade](#)) or partial upgrades of the system followed by a full upgrade. This will make it possible to upgrade the system partially, and allow you to clean the package cache before the full upgrade.

Note that in order to safely remove packages, it is advisable to switch your APT sources files back to trixie as described in [Checking your APT configuration](#).

4.4.4 Stop monitoring systems

As `apt` may need to temporarily stop services running on your computer, it's probably a good idea to stop monitoring services that can restart other terminated services during the upgrade. In Debian, **monit** is an example of such a service.

4.4.5 Minimal system upgrade

In some cases, doing the full upgrade (as described below) directly might remove large numbers of packages that you will want to keep. We therefore recommend a two-part upgrade process: first a minimal upgrade to overcome these conflicts, then a full upgrade as described in [Upgrading the system](#).

To do this, first run:

```
# apt upgrade --without-new-pkgs
```

This has the effect of upgrading those packages which can be upgraded without requiring any other packages to be removed or installed.

The minimal system upgrade can also be useful when the system is tight on space and a full upgrade cannot be run due to space constraints.

If the **apt-listchanges** package is installed, it will (in its default configuration) show important information about upgraded packages in a pager after downloading the packages. Press **q** after reading to exit the pager and continue the upgrade.

4.4.6 Upgrading the system

Once you have taken the previous steps, you are now ready to continue with the main part of the upgrade. Execute:

```
# apt full-upgrade
```

This will perform a complete upgrade of the system, installing the newest available versions of all packages, and resolving all possible dependency changes between packages in different releases. If necessary, it will install some new packages (usually new library versions, or renamed packages), and remove any conflicting obsoleted packages.

When upgrading from a set of CDs/DVDs/BDs, you will probably be asked to insert specific discs at several points during the upgrade. You might have to insert the same disc multiple times; this is due to inter-related packages that have been spread out over the discs.

New versions of currently installed packages that cannot be upgraded without changing the install status of another package will be left at their current version (displayed as “held back”). This can be resolved by either using **aptitude** to choose these packages for installation or by trying **apt install package**.

4.5 Possible issues during upgrade

The following sections describe known issues that might appear during an upgrade to forky.

4.5.1 Full-upgrade fails with “Could not perform immediate configuration”

In some cases the **apt full-upgrade** step can fail after downloading packages with:

```
E: Could not perform immediate configuration on 'package'. Please see man 5 apt.conf,
under APT::Immediate-Configure for details.
```

If that happens, running **apt full-upgrade -o APT::Immediate-Configure=0** instead should allow the upgrade to proceed.

Another possible workaround for this problem is to temporarily add both trixie and forky sources to your APT sources files and run **apt update**.

4.5.2 Expected removals

The upgrade process to forky might ask for the removal of packages on the system. The precise list of packages will vary depending on the set of packages that you have installed. These release notes give general advice on these removals, but if in doubt, it is recommended that you examine the package removals proposed by each method before proceeding. For more information about packages obsoleted in forky, see *Obsolete packages*.

4.5.3 Conflicts or Pre-Depends loops

Sometimes it's necessary to enable the `APT::Force-LoopBreak` option in APT to be able to temporarily remove an essential package due to a Conflicts/Pre-Depends loop. `apt` will alert you of this and abort the upgrade. You can work around this by specifying the option `-o APT::Force-LoopBreak=1` on the `apt` command line.

It is possible that a system's dependency structure can be so corrupt as to require manual intervention. Usually this means using `apt` or

```
# dpkg --remove package_name
```

to eliminate some of the offending packages, or

```
# apt -f install
# dpkg --configure --pending
```

In extreme cases you might have to force re-installation with a command like

```
# dpkg --install /path/to/package_name.deb
```

4.5.4 File conflicts

File conflicts should not occur if you upgrade from a “pure” trixie system, but can occur if you have unofficial backports installed. A file conflict will result in an error like:

```
Unpacking <package-foo> (from <package-foo-file>) ...
dpkg: error processing <package-foo> (--install):
trying to overwrite `<some-file-name>',
which is also in package <package-bar>
dpkg-deb: subprocess paste killed by signal (Broken pipe)
Errors were encountered while processing:
<package-foo>
```

You can try to solve a file conflict by forcibly removing the package mentioned on the *last* line of the error message:

```
# dpkg -r --force-depends package_name
```

After fixing things up, you should be able to resume the upgrade by repeating the previously described `apt` commands.

4.5.5 Configuration changes

During the upgrade, you will be asked questions regarding the configuration or re-configuration of several packages. When you are asked if any file in the `/etc/init.d` directory, or the `/etc/manpath.config` file should be replaced by the package maintainer's version, it's usually necessary to answer “yes” to ensure system consistency. You can always revert to the old versions, since they will be saved with a `.dpkg-old` extension.

If you're not sure what to do, write down the name of the package or file and sort things out at a later time. You can search in the `/var/log/apt/term.log` file to review the information that was on the screen during the upgrade.

4.5.6 Change of session to console

If you are running the upgrade using the system's local console you might find that at some points during the upgrade the console is shifted over to a different view and you lose visibility of the upgrade process. For example, this may happen in systems with a graphical interface when the display manager is restarted.

To recover the console where the upgrade was running you will have to use `Ctrl+Alt+F1` (if in the graphical startup screen) or `Alt+F1` (if in the local text-mode console) to switch back to the virtual terminal 1. Replace `F1` with the

function key with the same number as the virtual terminal the upgrade was running in. You can also use `Alt+Left Arrow` or `Alt+Right Arrow` to switch between the different text-mode terminals.

4.6 Upgrading your kernel and related packages

This section explains how to upgrade your kernel and identifies potential issues related to this upgrade. You can either install one of the **linux-image-*** packages provided by Debian, or compile a customized kernel from source.

Note that a lot of information in this section is based on the assumption that you will be using one of the modular Debian kernels, together with **initramfs-tools** and **udev**. If you choose to use a custom kernel that does not require an `initrd` or if you use a different `initrd` generator, some of the information may not be relevant for you.

4.6.1 Installing a kernel metapackage

When you full-upgrade from trixie to forky, it is strongly recommended that you install a `linux-image-*` metapackage, if you have not done so before. These metapackages will automatically pull in a newer version of the kernel during upgrades. You can verify whether you have one installed by running:

```
$ dpkg -l 'linux-image*' | grep ^ii | grep -i meta
```

If you do not see any output, then you will either need to install a new `linux-image` package by hand or install a `linux-image` metapackage. To see a list of available `linux-image` metapackages, run:

```
$ apt-cache search linux-image- | grep -i meta | grep -v transition
```

If you are unsure about which package to select, run `uname -r` and look for a package with a similar name. For example, if you see “4.9.0-8-amd64”, it is recommended that you install **linux-image-amd64**. You may also use `apt` to see a long description of each package in order to help choose the best one available. For example:

```
$ apt show linux-image-amd64
```

You should then use `apt install` to install it. Once this new kernel is installed you should reboot at the next available opportunity to get the benefits provided by the new kernel version. However, please have a look at *Things to do before rebooting* before performing the first reboot after the upgrade.

For the more adventurous there is an easy way to compile your own custom kernel on Debian. Install the kernel sources, provided in the **linux-source** package. You can make use of the `deb-pkg` target available in the sources’ makefile for building a binary package. More information can be found in the *Debian Linux Kernel Handbook*, which can also be found as the **debian-kernel-handbook** package.

If possible, it is to your advantage to upgrade the kernel package separately from the main `full-upgrade` to reduce the chances of a temporarily non-bootable system. Note that this should only be done after the minimal upgrade process described in *Minimal system upgrade*.

4.7 Cleanup after the upgrade

Two steps are recommended to clean the upgraded distribution.

- Remove newly redundant or obsolete packages as described in *Make sure you have sufficient space for the upgrade* and *Obsolete packages*. You should review which configuration files they use and consider purging the packages to remove their configuration files. See also *Purging removed packages*.
- Upgrade your APT sources. APT is deprecating the old format used for specifying what repositories to use - see *Preparing APT sources files* and `sources.list(5)`. If you haven’t already switched all your configuration files, you can use the new `apt` feature `apt modernize-sources`.

- Switch your APT sources to use the canonical Debian archive keyring pathnames in *Signed-By*, by replacing the .gpg extension to .pgp, as the backwards compatible .gpg symlinks will eventually disappear.

4.8 Cleaning up automatically installed packages

Some packages may have been only installed on your system as dependencies of other packages. With the new release these dependencies could have changed and apt will propose to remove those automatically installed packages. For this run:

```
# apt autoremove
```

4.9 Obsolete packages

Introducing lots of new packages, forky also retires and omits quite a few old packages that were in trixie. It provides no upgrade path for these obsolete packages. While nothing prevents you from continuing to use an obsolete package where desired, the Debian project will usually discontinue security support for it a year after forky’s release⁵, and will not normally provide other support in the meantime. Replacing them with available alternatives, if any, is recommended.

There are many reasons why packages might have been removed from the distribution: they are no longer maintained upstream; there is no longer a Debian Developer interested in maintaining the packages; the functionality they provide has been superseded by different software (or a new version); or they are no longer considered suitable for forky due to bugs in them. In the latter case, packages might still be present in the “unstable” distribution.

“Obsolete and Locally Created Packages” can be listed and purged from the commandline with:

```
$ apt list '?obsolete'
# apt purge '?obsolete'
```

The [Debian Bug Tracking System](#) often provides additional information on why the package was removed. You should review both the archived bug reports for the package itself and the archived bug reports for the [ftp.debian.org pseudo-package](http://ftp.debian.org/pseudo-package).

For a list of obsolete packages for forky, please refer to [Noteworthy obsolete packages](#).

4.9.1 Purging removed packages

It is generally advisable to purge removed packages. This is especially true if these have been removed in an earlier release upgrade (e.g. from the upgrade to trixie) or they were provided by third-party vendors. In particular, old init.d scripts have been known to cause issues.

Caution

Purging a package will generally also purge its log files, so you might want to back them up first.

The following command displays a list of all removed packages that may have configuration files left on the system (if any):

```
$ apt list '?config-files'
```

The packages can be removed by using `apt purge`. Assuming you want to purge all of them in one go, you can use the following command:

⁵ Or for as long as there is not another release in that time frame. Typically only two stable releases are supported at any given time.

```
# apt purge '?config-files'
```

4.9.2 Transitional dummy packages

Some packages from trixie may have been replaced in forky by transitional dummy packages, which are empty placeholders designed to simplify upgrades. If for instance an application that was formerly a single package has been split into several, a transitional package may be provided with the same name as the old package and with appropriate dependencies to cause the new ones to be installed. After this has happened the redundant dummy package can be safely removed.

The package descriptions for transitional dummy packages usually indicate their purpose. However, they are not uniform; in particular, some “dummy” packages are designed to be kept installed, in order to pull in a full software suite, or track the current latest version of some program.

ISSUES TO BE AWARE OF FOR FORKY —DRAFT—

Sometimes, changes introduced in a new release have side-effects we cannot reasonably avoid, or they expose bugs somewhere else. This section documents issues we are aware of. Please also read the errata, the relevant packages' documentation, bug reports, and other information mentioned in *Further reading*.

5.1 Things to be aware of while upgrading to forky

This section covers items related to the upgrade from trixie to forky.

5.1.1 Things to do before rebooting

When `apt full-upgrade` has finished, the “formal” upgrade is complete. For the upgrade to forky, there are no special actions needed before performing a reboot.

5.2 Items not limited to the upgrade process

5.2.1 Limitations in security support

There are some packages where Debian cannot promise to provide minimal backports for security issues. These are covered in the following subsections.

Note

The package **debian-security-support** helps to track the security support status of installed packages.

Security status of web browsers and their rendering engines

Debian 14 includes several browser engines which are affected by a steady stream of security vulnerabilities. The high rate of vulnerabilities and partial lack of upstream support in the form of long term branches make it very difficult to support these browsers and engines with backported security fixes. Additionally, library interdependencies make it extremely difficult to update to newer upstream releases. Applications using the **webkit2gtk** source package (e.g. **epiphany**) are covered by security support, but applications using qtwebengine (source packages **qtwebengine-opensource-src** and **qt6-webengine**) are not.

For general web browser use we recommend Firefox or Chromium. They will be kept up-to-date by rebuilding the current ESR releases for stable. The same strategy will be applied for Thunderbird.

Once a release becomes `oldstable`, officially supported browsers may not continue to receive updates for the standard period of coverage. For example, Chromium will only receive 6 months of security support in `oldstable` rather than the typical 12 months.

Go- and Rust-based packages

The Debian infrastructure currently has problems with rebuilding packages of types that systematically use static linking. With the growth of the Go and Rust ecosystems it means that these packages will be covered by limited security support until the infrastructure is improved to deal with them maintainably.

In most cases if updates are warranted for Go or Rust development libraries, they will only be released via regular point releases.

5.3 Obsolescence and deprecation

5.3.1 Noteworthy obsolete packages

The following is a list of known and noteworthy obsolete packages (see *Obsolete packages* for a description).

The list of obsolete packages includes:

5.3.2 Deprecated components for forky

With the next release of Debian 15 (codenamed duke) some features will be deprecated. Users will need to migrate to other alternatives to prevent trouble when updating to Debian 15.

This includes the following features:

5.4 Known severe bugs

Although Debian releases when it's ready, that unfortunately doesn't mean there are no known bugs. As part of the release process all the bugs of severity serious or higher are actively tracked by the Release Team, so an [overview of those bugs](#) that were tagged to be ignored in the last part of releasing forky can be found in the [Debian Bug Tracking System](#). The following bugs were affecting forky at the time of the release and worth mentioning in this document:

Bug number	Package (source or binary)	Description
1032240	akonadi-backend-mysql	akonadi server not robust against mysql upgrades

MORE INFORMATION ON DEBIAN —DRAFT—

6.1 Further reading

Beyond these release notes and the installation guide (at <https://www.debian.org/releases/forky/installmanual>) further documentation on Debian is available from the Debian Documentation Project (DDP), whose goal is to create high-quality documentation for Debian users and developers, such as the Debian Reference, Debian New Maintainers Guide, the Debian FAQ, and many more. For full details of the existing resources see the [Debian Documentation website](#) and the [Debian Wiki](#).

Documentation for individual packages is installed into `/usr/share/doc/package`. This may include copyright information, Debian specific details, and any upstream documentation.

6.2 Getting help

There are many sources of help, advice, and support for Debian users, though these should only be considered after researching the issue in available documentation. This section provides a short introduction to these sources which may be helpful for new Debian users.

6.2.1 Mailing lists

The mailing lists of most interest to Debian users are the `debian-user` list (English) and other `debian-user-language` lists (for other languages). For information on these lists and details of how to subscribe see <https://lists.debian.org/>. Please check the archives for answers to your question prior to posting and also adhere to standard list etiquette.

6.2.2 Internet Relay Chat

Debian has an IRC channel dedicated to support and aid for Debian users, located on the OFTC IRC network. To access the channel, point your favorite IRC client at `irc.debian.org` and join `#debian`.

Please follow the channel guidelines, respecting other users fully. The guidelines are available at the [Debian Wiki](#).

For more information on OFTC please visit the [website](#).

6.3 Reporting bugs

We strive to make Debian a high-quality operating system; however that does not mean that the packages we provide are totally free of bugs. Consistent with Debian's "open development" philosophy and as a service to our users, we provide all the information on reported bugs at our own Bug Tracking System (BTS). The BTS can be browsed at <https://bugs.debian.org/>.

If you find a bug in the distribution or in packaged software that is part of it, please report it so that it can be properly fixed for future releases. Reporting bugs requires a valid e-mail address. We ask for this so that we can trace bugs and developers can get in contact with submitters should additional information be needed.

You can submit a bug report using the program `reportbug` or manually using e-mail. You can find out more about the Bug Tracking System and how to use it by reading the reference documentation (available at `/usr/share/doc/debian` if you have **doc-debian** installed) or online at the [Bug Tracking System](#).

6.4 Contributing to Debian

You do not need to be an expert to contribute to Debian. By assisting users with problems on the various user support [lists](#) you are contributing to the community. Identifying (and also solving) problems related to the development of the distribution by participating on the development [lists](#) is also extremely helpful. To maintain Debian's high-quality distribution, [submit bugs](#) and help developers track them down and fix them. The tool `how-can-i-help` helps you to find suitable reported bugs to work on. If you have a way with words then you may want to contribute more actively by helping to write [documentation](#) or [translating](#) existing documentation into your own language.

If you can dedicate more time, you could manage a piece of the Free Software collection within Debian. Especially helpful is if people adopt or maintain items that people have requested for inclusion within Debian. The [Work Needing and Prospective Packages database](#) details this information. If you have an interest in specific groups then you may find enjoyment in contributing to some of Debian's [subprojects](#) which include ports to particular architectures and [Debian Pure Blends](#) for specific user groups, among many others.

In any case, if you are working in the free software community in any way, as a user, programmer, writer, or translator you are already helping the free software effort. Contributing is rewarding and fun, and as well as allowing you to meet new people it gives you that warm fuzzy feeling inside.

MANAGING YOUR TRIxie SYSTEM BEFORE THE UPGRADE —DRAFT—

This appendix contains information on how to make sure you can install or upgrade trixie packages before you upgrade to forky.

7.1 Upgrading your trixie system

Basically this is no different from any other upgrade of trixie you’ve been doing. The only difference is that you first need to make sure your package list still contains references to trixie as explained in *Checking your APT source-list files*.

If you upgrade your system using a Debian mirror, it will automatically be upgraded to the latest trixie point release.

7.2 Checking your APT configuration

If any of the lines in your APT sources files (see [sources.list\(5\)](#)) contain references to “stable”, this is effectively pointing to forky already. This might not be what you want if you are not yet ready for the upgrade. If you have already run `apt update`, you can still get back without problems by following the procedure below.

If you have also already installed packages from forky, there probably is not much point in installing packages from trixie anymore. In that case you will have to decide for yourself whether you want to continue or not. It is possible to downgrade packages, but that is not covered here.

As root, open the relevant APT sources file(s) (such as `/etc/apt/sources.list` or any file under `/etc/apt/sources.list.d/`) with your favorite editor, and check all lines beginning with

- `deb http:`
- `deb https:`
- `deb tor+http:`
- `deb tor+https:`
- `URIs: http:`
- `URIs: https:`
- `URIs: tor+http:`
- `URIs: tor+https:`

for a reference to “stable”. If you find any, change “stable” to “trixie”.

If you have any lines starting with `deb file:` or `URIs: file:`, you will have to check for yourself if the location they refer to contains a trixie or forky archive.

Important

Do not change any lines that begin with `deb cdrom:` or `URIs: cdrom:`. Doing so would invalidate the line and you would have to run `apt-cdrom` again. Do not be alarmed if a `cdrom:` source line refers to “unstable”. Although confusing, this is normal.

If you’ve made any changes, save the file and execute

```
# apt update
```

to refresh the package list.

7.3 Performing the upgrade to latest trixie release

To upgrade all packages to the state of the latest point release for trixie, do

```
# apt full-upgrade
```

7.4 Removing obsolete configuration files

Before upgrading your system to forky, it is recommended to remove old configuration files (such as `*.dpkg-{new,old}` files under `/etc`) from the system.

CONTRIBUTORS TO THE RELEASE NOTES —DRAFT—

Many people helped with the release notes, including, but not limited to

- ADAM D. BARRAT (various fixes in 2013),
- ADAM DI CARLO (previous releases),
- ANDREAS BARTH ABA (previous releases: 2005 - 2007),
- ANDREI POPESCU (various contributions),
- ANNE BEZEMER (previous release),
- BOB HILLIARD (previous release),
- CHARLES PLESSY (description of GM965 issue),
- CHRISTIAN PERRIER BUBULLE (Lenny installation),
- CHRISTOPH BERG (PostgreSQL-specific issues),
- DANIEL BAUMANN (Debian Live),
- DAVID PRÉVOT TAFFIT (Wheezy release),
- EDDY PETRIȘOR (various contributions),
- EMMANUEL KASPER (backports),
- ESKO ARAJÄRVI (rework X11 upgrade),
- FRANS POP FJP (previous release Etch),
- GIOVANNI RAPAGNANI (innumerable contributions),
- GORDON FARQUHARSON (ARM port issues),
- HIDEKI YAMANE HENRICH (contributed and contributing since 2006),
- HOLGER WANSING HOLGERW (contributed and contributing since 2009),
- JAVIER FERNÁNDEZ-SANGUINO PEÑA JFS (Etch release, Squeeze release),
- JENS SEIDEL (German translation, innumerable contributions),
- JONAS MEURER (syslog issues),
- JONATHAN NIEDER (Squeeze release, Wheezy release),
- JOOST VAN BAAL-ILIĆ JOOSTVB (Wheezy release, Jessie release),
- JOSIP RODIN (previous releases),
- JULIEN CRISTAU JCRISTAU (Squeeze release, Wheezy release),

- JUSTIN B RYE (English fixes),
- LAMONT JONES (description of NFS issues),
- LUK CLAES (editors motivation manager),
- MARTIN MICHLMAYR (ARM port issues),
- MICHAEL BIEBL (syslog issues),
- MORITZ MÜHLENHOFF (various contributions),
- NIELS THYKIER NTHYKIER (Jessie release),
- NOAH MEYERHANS (innumerable contributions),
- NORITADA KOBAYASHI (Japanese translation (coordination), innumerable contributions),
- OSAMU AOKI (various contributions),
- PAUL GEVERS ELBRUS (buster release),
- PETER GREEN (kernel version note),
- ROB BRADFORD (Etch release),
- SAMUEL THIBAUT (description of d-i Braille support),
- SIMON BIENLEIN (description of d-i Braille support),
- SIMON PAILLARD SPAILLAR-GUEST (innumerable contributions),
- STEFAN FRITSCH (description of Apache issues),
- STEVE LANGASEK (Etch release),
- STEVE MCINTYRE (Debian CDs),
- TOBIAS SCHERER (description of "proposed-update"),
- VICTORY VICTORY-GUEST (markup fixes, contributed and contributing since 2006),
- VINCENT MCINTYRE (description of "proposed-update"),
- W. MARTIN BORGERT (editing Lenny release, switch to DocBook XML).

This document has been translated into many languages. Many thanks to all the translators!