

4th Edition



LINUX

IN A NUTSHELL

A Desktop Quick Reference

O'REILLY®

*Ellen Siever,
Stephen Figgins & Aaron Weber*

LINUX

IN A NUTSHELL

Other Linux resources from O'Reilly

Related titles	Linux in a Nutshell	LPI Linux Certification in a Nutshell
	Linux Network Administrator's Guide	Learning Red Hat Linux
	Running Linux	Linux Server Hacks
	Linux Device Drivers	Linux Security Cookbook
	Understanding the Linux Kernel	Managing RAID on Linux
	Building Secure Servers with Linux	Linux Web Server CD Bookshelf
		Building Embedded Linux Systems

Linux Books Resource Center

linux.oreilly.com is a complete catalog of O'Reilly's books on Linux and Unix and related technologies, including sample chapters and code examples.



ONLamp.com is the premier site for the open source web platform: Linux, Apache, MySQL and either Perl, Python, or PHP.

Conferences

O'Reilly & Associates bring diverse innovators together to nurture the ideas that spark revolutionary industries. We specialize in documenting the latest tools and systems, translating the innovator's knowledge into useful skills for those in the trenches. Visit *conferences.oreilly.com* for our upcoming events.



Safari Bookshelf (*safari.oreilly.com*) is the premier online reference library for programmers and IT professionals. Conduct searches across more than 1,000 books. Subscribers can zero in on answers to time-critical questions in a matter of seconds. Read the books on your Bookshelf from cover to cover or simply flip to the page you need. Try it today with a free trial.

LINUX

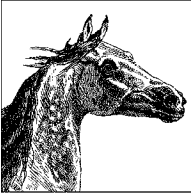
IN A NUTSHELL

Fourth Edition

*Ellen Siever, Stephen Figgins,
and Aaron Weber*

O'REILLY®

Beijing • Cambridge • Farnham • Köln • Paris • Sebastopol • Taipei • Tokyo



5

Red Hat and Debian Package Managers

This chapter describes the two major Linux packaging systems: the Red Hat Package Manager (RPM) and the Debian GNU/Linux Package Manager.

When you install applications on your Linux system, most often you'll find a binary or a source package containing the application you want, instead of (or in addition to) a *.tar.gz* file. A package is a file containing the files necessary to install an application. However, while the package contains the files you need for installation, the application might require the presence of other files or packages that are not included, such as particular libraries (and even specific versions of the libraries), to actually be able to run. Such requirements are known as *dependencies*.

Package management systems offer many benefits. As a user, you may want to query the package database to find out what packages are installed on the system and their versions. As a system administrator, you need tools to install and manage the packages on your system. And if you are a developer, you need to know how to build a package for distribution.

Among other things, package managers do the following:

- Provide tools for installing, updating, removing, and managing the software on your system.
- Allow you to install new or upgraded software directly across a network.
- Tell you what software package a particular file belongs to or what files a package contains.
- Maintain a database of packages on the system and their state, so you can find out what packages or versions are installed on your system.
- Provide dependency checking, so you don't mess up your system with incompatible software.
- Provide PGP, MD5, or other signature verification tools.
- Provide tools for building packages.

Any user can list or query packages. However, installing, upgrading, or removing packages generally requires superuser privileges. This is because the packages normally are installed in systemwide directories that are writable only by root. Sometimes you can specify an alternate directory to install a package into your home directory or into a project directory where you have write permission.

Both RPM and the Debian Package Manager back up old files before installing an updated package. Not only does this let you go back if there is a problem, but also ensures that you don't lose your changes (to configuration files, for example).

The Red Hat Package Manager

The Red Hat Package Manager (RPM) is a freely available packaging system for software distribution and installation. In addition to Red Hat and Red Hat–based distributions, both SuSE and Caldera are among the Linux distributions that use RPM.

Using RPM is straightforward. A single command, **rpm**, has options to perform all package management functions except building packages.* For example, to find out if the Emacs editor is installed on your system, you could say:

```
% rpm -q emacs
emacs-21.2-18
```

The **rpmbuild** command is used to build both binary and source packages.

The rpm Command

RPM packages are built, installed, and queried with the **rpm** command. RPM package names usually end with a *.rpm* extension. **rpm** has a set of modes, each with its own options. The format of the **rpm** command is:

```
rpm [options] [packages]
```

With a few exceptions, as noted in the lists of options that follow, the first option specifies the **rpm** mode (install, query, update, etc.), and any remaining options affect that mode.

Options that refer to packages are sometimes specified as *package-name* and sometimes as *package-file*. The package name is the name of the program or application, such as **gif2png**. The package file is the name of the RPM file, such as *gif2png-2.4.6-1.i386.rpm*.

RPM provides a configuration file for specifying frequently used options. The default global configuration is usually */usr/lib/rpm/rpmrc*, the local system configuration file is */etc/rpmrc*, and users can set up their own *\$HOME/.rpmrc* files. You can use the **--showrc** option to show the values RPM will use for all the options that may be set in an *rpmrc* file:

```
rpm --showrc
```

* In older versions of RPM, the build options were part of the **rpm** command.

The **rpm** command includes FTP and HTTP clients, so you can specify an *ftp://* or *http://* URL to install or query a package across the Internet. You can use an FTP or HTTP URL wherever *package-file* is specified in the commands presented here.

Any user can query the RPM database. Most of the other functions require super-user privileges.

General options

The following options can be used with all modes:

--dbpath *path*

Use *path* as the path to the RPM database instead of the default */var/lib/rpm*.

-?, --help

Print a long usage message (running **rpm** with no options gives a shorter usage message).

--pipe *command*

Pipe the **rpm** output to *command*.

--quiet

Display only error messages.

--rcfile *filelist*

Get configuration from the files in the colon-separated *filelist*. If **--rcfile** is specified, there must be at least one file in the list and the file must exist. *filelist* defaults to */var/lib/rpm/rpmrc:/usr/lib/rpm/redhat/rpmrc:~/rpmrc*.

--root *dir*

Perform all operations within the directory tree rooted at *dir*.

-v Verbose. Print progress messages.

--version

Print the version number of **rpm**.

-vv Print debugging information.

Install, upgrade, and freshen options

Use the **install** command to install or upgrade an RPM package. The **install** syntax is:

```
rpm -i [install-options] package_file ...
rpm --install [install-options] package_file ...
```

To install a new version of a package and remove an existing version at the same time, use the **upgrade** command instead:

```
rpm -U [install-options] package_file ...
rpm --upgrade [install-options] package_file ...
```

If the package doesn't already exist on the system, **-U** acts like **-i** and installs it. To prevent that behavior, you can **freshen** a package instead; in that case, **rpm** upgrades the package only if an earlier version is already installed. The **freshen** syntax is:

```
rpm -F [install-options] package_file ...
rpm --freshen [install-options] package_file ...
```

package-file can be specified as an FTP or HTTP URL to download the file before installing it. See “FTP/HTTP options.”

The installation and upgrade options are:

--aid

If **rpm** suggests additional packages, add them to the list of package files.

--allfiles

Install or upgrade all files.

--badreloc

Used with **--relocate** to force relocation even if the package is not relocatable.

--excludedocs

Don't install any documentation files.

--excludepath *path*

Don't install any file whose filename begins with *path*.

--force

Force the installation. Equivalent to using all of **--replacepkgs**, **--replacefiles**, and **--oldpackage**.

-h, --hash

Print 50 hash marks as the package archive is unpacked. Use with **-v** or **--verbose** for a nicer display.

--ignorearch

Install even if the binary package is intended for a different architecture.

--ignoreos

Install binary package even if the operating systems don't match.

--ignoresize

Don't check disk space availability before installing.

--includedocs

Install documentation files. This is needed only if **excludedocs: 1** is specified in an *rpmrc* file.

--justdb

Update the database only; don't change any files.

--nodeps

Don't check whether this package depends on the presence of other packages.

--nodigest

Don't verify package or header digests.

--noorder

Don't reorder packages to satisfy dependencies before installing.

--nopost

Don't execute any post-install script.

--nopostun

Don't execute any post-uninstall script.

--nopre

Don't execute any pre-install script.

- nopreun**
Don't execute any pre-uninstall script.
- noscripts**
Don't execute any pre-install or post-install scripts. Equivalent to specifying all of **--nopre**, **--nopost**, **--nopreun**, and **--nopostun**.
- nosignature**
Don't verify package or header signatures.
- nosuggest**
Don't suggest packages that provide a missing dependency.
- notriggerin**
Don't execute any install trigger scriptlet.
- notriggerun**
Don't execute any uninstall trigger scriptlet.
- notriggerpostun**
Don't execute any post-uninstall trigger scriptlet.
- notriggers**
Don't execute any scripts triggered by package installation.
- oldpackage**
Allow an upgrade to replace a newer package with an older one.
- percent**
Print percent-completion messages as files are unpacked. Useful for running **rpm** from other tools.
- prefix *path***
Set the installation prefix to *path* for relocatable binary packages.
- relocate *oldpath=newpath***
For relocatable binary files, change all file paths from *oldpath* to *newpath*. Can be specified more than once to relocate multiple paths.
- repackage**
Repackage the package files before erasing. Rename the package as specified by the macro **%_repackage_name_fmt** and save it in the directory specified by the macro **%_repackage_dir** (by default */var/tmp*).
- replacefiles**
Install the packages even if they replace files from other installed packages.
- replacepkgs**
Install the packages even if some of them are already installed.
- test**
Go through the installation to see what it would do, but don't actually install the package. This option lets you test for problems before doing the installation.

Query options

The syntax for the **query** command is:

```
rpm -q [package-options] [information-options]  
rpm --query [package-options] [information-options]
```

There are two subsets of query options. *Package selection* options determine what packages to query, and *information selection* options determine what information to provide.

Package selection options

package_name

Query the installed package *package_name*.

-a, --all

Query all installed packages.

-f file, --file file

Find out what package owns *file*.

--fileid md5

Query package with the specified MD5 digest.

-g group, --group group

Find out what packages have group *group*.

--hrid sha1

Query package with the specified SHA1 digest in the package header.

-p package_file, --package package_file

Query the uninstalled package *package_file*, which can be a URL. If *package_file* is not a binary package, it is treated as a text file containing a package manifest, with each line of the manifest containing a path or one or more whitespace-separated glob expressions to be expanded to paths. These paths are then used instead of *package_file* as the query arguments. The manifest can contain comments that begin with a hash mark (#).

--pkgid md5

Query the package with a package identifier that is the given MD5 digest of the combined header and contents.

--querybynumber num

Query the *num*th database entry. Useful for debugging.

-qf, --queryformat num

Specify the format for displaying the query output, using tags to represent different types of data (e.g., NAME, FILENAME, DISTRIBUTION). The format specification is a variation of the standard **printf** formatting, with the type specifier omitted and replaced by the name of the header tag inclosed in brackets ({}). For example:

{NAME}

The tag names are case-insensitive. Use **--querytags** (see “Miscellaneous options”) to view a list of available tags. The tag can be followed by *:type* to get a different output format type. The possible types are:

armor

Wrap a public key in ASCII armor.

base64

Encode binary data as base64.

date

Use **strftime(3)** “%c” format.

day
Use `strftime(3)` “%a %b %d %Y” format.

depflags
Format dependency flags.

fflags
Format file flags.

hex
Use hexadecimal format.

octal
Use octal format.

perms
Format file permissions.

shescape
Escape single quotes for use in a script.

triggertype
Display trigger suffix.

--specfile *specfile*
Query *specfile* as if it were a package. Useful for extracting information from a spec file.

--tid *tid*
List packages with the specified transaction identifier (*tid*). The *tid* is a Unix timestamp. All packages installed or erased in a single transaction have the same *tid*.

--triggeredby *pkg*
List packages that are triggered by the installation of package *pkg*.

--whatrequires *capability*
List packages that require the given capability to function.

--whatprovides *capability*
List packages that provide the given capability.

Information selection options

-c, --configfiles
List configuration files in the package. Implies **-l**.

--changelog
Display the log of change information for the package.

-d, --docfiles
List documentation files in the package. Implies **-l**.

--dump
Dump information for each file in the package. This option must be used with at least one of **-l**, **-c**, or **-d**. The output includes the following information in this order:

path size mtime md5sum mode owner group isconfig isdoc rdev symlink

--filesbypkg
List all files in each package.

- i, --info**
Display package information, including the name, version, and description. Formats the results according to **--queryformat** if specified.
- l, --list**
List all files in the package.
- last**
List packages by install time, with the latest packages listed first.
- provides**
List the capabilities this package provides.
- R, --requires**
List any packages this package depends on.
- s, --state**
List each file in the package and its state. The possible states are **normal**, **not installed**, or **replaced**. Implies **-l**.
- scripts**
List any package-specific shell scripts used during installation and uninstallation of the package.
- triggers, --triggerscript**
Display any trigger scripts in the package.

Uninstall options

The syntax for **erase**, the uninstall command, is:

```
rpm -e package_name ...
rpm --erase package_name ...
```

The uninstall options are:

- allmatches**
Remove all versions of the package. Only one package should be specified; otherwise, an error results.
- nodeps**
Don't check dependencies before uninstalling the package.
- nopostun**
Don't run any post-uninstall scripts.
- noprereun**
Don't run any pre-uninstall scripts.
- noscripts**
Don't execute any pre-uninstall or post-uninstall scripts. Equivalent to **--noprereun --nopostun**.
- notriggerpostun**
Don't execute any post-uninstall scripts triggered by the removal of this package.
- notriggers**
Don't execute any scripts triggered by the removal of this package. Equivalent to **--notriggerun --notriggerpostun**.

--notriggerun

Don't execute any uninstall scripts triggered by the removal of this package.

--repackage

Repackage the files before uninstalling them. Rename the package as specified by the macro `_%repackage_name_fmt` and save it in the directory specified by the macro `_%repackage_dir` (by default `/var/tmp`).

--test

Don't really uninstall anything; just go through the motions. Use with `-vv` for debugging.

Verify options

The syntax for the `verify` command is:

```
rpm -V|-y|--verify [package-selection-options] [verify-options]
```

Verify mode compares information about the installed files in a package with information about the files that came in the original package, and displays any discrepancies. The information compared includes the size, MD5 sum, permissions, type, owner, and group of each file. Uninstalled files are ignored.

The package selection options include those available for query mode. In addition, the following `verify` options are available:

--nodeps

Ignore package dependencies.

--nodigest

Ignore package or header digests.

--nofiles

Ignore attributes of package files.

--nogroup

Ignore group ownership errors.

--nolinkto

Ignore symbolic link errors.

--nomd5

Ignore MD5 checksum errors.

--nomode

Ignore file mode (permissions) errors.

--nordev

Ignore major and minor device number errors.

--nomtime

Ignore modification time errors.

--noscripts

Ignore any verify script.

--nosignature

Ignore package or header signatures.

--nosize

Ignore file size errors.

--nouser

Ignore user ownership errors.

The output is formatted as an eight-character string, possibly followed by an attribute marker, and then the filename. The possible attribute markers are:

c	Configuration file
d	Documentation file
g	Ghost file (contents not included in package)
l	License file
r	Readme file

Each of the eight characters in the string represents the result of comparing one file attribute to the value of that attribute from the RPM database. A period (.) indicates that the file passed that test. The following characters indicate failure of the corresponding test:

S	MD5 sum
D	Device
G	Group
L	Symlink
M	Mode (includes permissions and file type)
S	File size
T	Mtime
U	User

Database rebuild options

The syntax of the command to rebuild the RPM database is:

```
rpm --rebuilddb [options]
```

You also can build a new database:

```
rpm --initdb [options]
```

The options available with the database rebuild mode are the **--dbpath**, **--root**, and **-v** options described earlier under “General options.”

Signature check options

RPM packages may have a PGP signature built into them. PGP configuration information is read from the *rpmrc* file. There are three types of digital signature options: you can check signatures, add signatures to packages, and import signatures.

The syntax of the signature check mode is:

```
rpm --checksig package_file...
rpm -K package_file...
```

The signature checking options **-K** and **--checksig** check the digests and signatures contained in the specified packages to insure the integrity and origin of the packages. Note that RPM now automatically checks the signature of any package when it is read; this option is still useful, however, for checking all headers and signatures associated with a package.

The following options are available for use with signature check mode:

- nogpg**
Don't check any GPG signatures.
- nomd5**
Don't check any MD5 signatures.
- nopgp**
Don't check any PGP signatures.

The syntax for adding signatures to binary packages is:

```
rpm --addsign binary-pkgfile...  
rpm --resign binary-pkgfile...
```

Both **--addsign** and **--resign** generate and insert new signatures, replacing any that already exist in the specified binary packages.*

The syntax for importing signatures is:

```
rpm --import public-key
```

The **--import** option is used to import an ASCII public key to the RPM database so that digital signatures for packages using that key can be verified. Imported public keys are carried in headers, and keys are kept in a ring, which can be queried and managed like any package file.

Miscellaneous options

Several additional **rpm** options are available:

- querytags**
Print the tags available for use with the **--queryformat** option in query mode.
- setperms *packages***
Set file permissions of the specified packages to those in the database.
- setugids *packages***
Set file owner and group of the specified packages to those in the database.
- showrc**
Show the values **rpm** will use for all options that can be set in an *rpmrc* file.

FTP/HTTP options

The following options are available for use with FTP and HTTP URLs in install, update, and query modes.

* In older versions of RPM, **--addsign** was used to add new signatures without replacing existing ones, but currently both options work the same way and replace any existing signatures.

--ftpport *port*

Use *port* for making an FTP connection on the proxy FTP server instead of the default port. Same as specifying the macro `%_ftpport`.

--ftpproxy *host*

Use *host* as the proxy server for FTP transfers through a firewall that uses a proxy. Same as specifying the macro `%_ftpproxy`.

--httpport *port*

Use *port* for making an HTTP connection on the proxy HTTP server instead of the default port. Same as specifying the macro `%_httpport`.

--httpproxy *host*

Use *host* as the proxy server for HTTP transfers. Same as specifying the macro `%_httpproxy`.

The rpmbuild Command

The **rpmbuild** command is used to build RPM packages. The syntax for **rpmbuild** is:

```
rpmbuild -[b|t]step [build-options] spec-file ...
```

Specify **-b** to build a package directly from a spec file, or **-t** to open a tarred, gzipped file and use its spec file.

Both forms take the following single-character *step* arguments, listed in the order they would be performed:

- p** Perform the prep stage, unpacking source files and applying patches.
- l** Do a list check, expanding macros in the files section of the spec file and verifying that each file exists.
- c** Perform the build stage. Done after the prep stage; generally equivalent to doing a **make**.
- i** Perform the install stage. Done after the prep and build stages; generally equivalent to doing a **make install**.
- b** Build a binary package. Done after prep, build, and install.
- s** Build a source package. Done after prep, build, and install.
- a** Build both binary and source packages. Done after prep, build, and install.

The general **rpm** options described earlier in “General options” can be used with **rpmbuild**.

The following additional options can also be used when building an **rpm** file with **rpmbuild**:

--buildroot *dir*

Override the **BuildRoot** tag with *dir* when building the package.

--clean

Clean up (remove) the build files after the package has been made.

--nobuild

Go through the motions, but don't execute any build stages. Used for testing spec files.

--rmsource

Remove the source files when the build is done. Can be used as a standalone option with **rpm** to clean up files separately from creating the packages.

--rmspec

Remove the spec file when the build is done. Like **--rmsource**, **--rmspec** can be used as a standalone option with **rpmbuild**.

--short-circuit

Can be used with **-bc** and **-bi** to skip previous stages.

--sign

Add a GPG signature to the package for verifying its identity and origin.

--target platform

When building the package, set the macros **%_target**, **%_target_arch**, and **%_target_os** to the value indicated by *platform*.

Two other options can be used standalone with **rpmbuild** to recompile or rebuild a package:

--rebuild source-pkgfile...

Like **--recompile**, but also build a new binary package. Remove the build directory, the source files, and the spec file once the build is complete.

--recompile source-pkgfile...

Install the named source package, and prep, compile, and install the package.

Finally, the **--showrc** option is used to show the current **rpmbuild** configuration:

```
rpmbuild --showrc
```

This option shows the values that will be used for all options that can be set in an *rpmrc* file.

RPM Examples

Query the RPM database to find Emacs-related packages:

```
% rpm -q -a | grep emacs
```

Query an uninstalled package, printing information about the package and listing the files it contains:

```
% rpm -qpil ~/downloads/bash2-doc-2.03-8.i386.rpm
```

Install a package (assumes superuser privileges):

```
% rpm -i sudo-1.5.3-6.i386.rpm
```

The Debian Package Manager

Debian GNU/Linux provides several package management tools, primarily intended to facilitate the building, installation, and management of binary packages. Debian package names generally end in *.deb*. The Debian package management tools include:

dpkg

The original Debian packaging tool. Used to install or uninstall packages or as a frontend to **dpkg-deb**. Getting and installing packages is usually done with **apt-get**, but **dpkg** is still commonly used to install a package that is already on your system. In fact, **apt-get** calls **dpkg** to do the installation once it's gotten the package.

dpkg-deb

Lower-level packaging tool. Used to create and manage the Debian package archives. Accepts and executes commands from **dpkg** or can be called directly.

dselect

An interactive frontend to **dpkg**.

The Advanced Package Tool (APT)

APT is a modern, user-friendly package management tool that consists of a number of commands. The most frequently used of these commands is **apt-get**, which is used to download and install a Debian package. **apt-get** can be run from the command line or selected as a method from **dselect**. One of the features of **apt-get** is that you can use it to get and install packages across the Internet by specifying an FTP or HTTP URL. You can also use it to upgrade all packages currently installed on your system in a single operation. Note that this results in a large download and will take a long time on a slow Internet connection.

Each of these tools is described in detail in “Debian Package Manager Command Summary.”

Files

Some important files used by the Debian package management tools are:

control

Comes with each package. Documents dependencies; contains the name and version of the package, a description, maintainer, installed size, and so on.

conffiles

Comes with each package. Contains a list of the configuration files associated with the package.

preinst, postinst, prerm, postrm

Scripts that can be included in a package to be run before installation, after installation, before removal, or after removal of the package.

/var/lib/dpkg/available

Contains information about packages available on the system.

/var/lib/dpkg/status

Contains information about the status of packages available on the system.

/etc/apt/sources.list

A list for APT of package sources, used to locate packages. The sources are listed one per line, in order of preference.

/etc/apt/apt.conf

The main APT configuration file.

/etc/apt/apt_preferences

A preferences file that controls various aspects of APT, such as letting a user select the version or release of a package to install.

/etc/dpkg/dpkg.cfg

A configuration file containing default options for **dpkg**.

Package Priorities

Every Debian package has a priority associated with it, indicating how important the package is to the system. The priorities are:

Required

The package is essential to the proper functioning of the system.

Important

The package provides important functionality that enables the system to run well.

Standard

The package is included in a standard system installation.

Optional

The package is one that you might want to install, but you can omit it if you are short on disk space, for example.

Extra

The package either conflicts with other packages that have a higher priority, has specialized requirements, or is one that you would want to install only if you need it.

Package and Selection States

The possible states that a package can be in are:

config-files

Only the configuration files for the package are present on the system.

half-configured

The package is unpacked and configuration was started but not completed.

half-installed

Installation was started but not completed.

installed

The package is unpacked and configured.

not-installed

The package is not installed.

unpacked

The package is unpacked but not configured.

The possible package selection states are:

deinstall

The package has been selected for deinstallation (i.e., for removal of everything but configuration files).

install

The package has been selected for installation.

purge

The package has been selected to be purged (i.e., for removal of everything including the configuration files).

Package Flags

Two possible package flags can be set for a package:

hold

The package should not be handled by **dpkg** unless forced with the **--force-hold** option.

reinst-required

The package is broken and needs to be reinstalled. Such a package cannot be removed unless forced with the **--force-reinstreq** option.

Scripts

In addition to the commands described in the next section, a number of shell and Perl scripts are included with the package manager for use in managing and building packages:

apt-setup

An interactive script for adding download sources to the *sources.list* file. (Perl script)

dpkg-architecture

Determine and set the build and host architecture for package building. (Perl script)

dpkg-checkbuilddeps

Check installed packages against the build dependencies and build conflicts listed in the control file. (Perl script)

dpkg-buildpackage

Help automate package building. (Shell script)

dpkg-distaddfile

Add an entry for a file to *debian/files*. (Perl script)

dpkg-divert

Create and manage the list of diversions, used to override the default location for installing files. (Perl script)

dpkg-genchanges

Generate an upload control file from the information in an unpacked built source tree and the files it has generated. (Perl script)

dpkg-gencontrol

Read information from an unpacked source tree and display a binary package control file on standard output. (Perl script)

dpkg-name

Rename Debian packages to their full package names. (Shell script)

dpkg-parsechangelog

Read and parse the changelog from an unpacked source tree and write the information to standard output in machine-readable form. (Perl script)

dpkg-preconfigure

Let packages ask questions prior to installation. (Perl script)

dpkg-reconfigure

Reconfigure a package that is already installed. (Perl script)

dpkg-scanpackages

Create a *Packages* file from a tree of binary packages. The *Packages* file is used by **dselect** to provide a list of packages available for installation. (Perl script)

dpkg-shlibdeps

Calculate shared library dependencies for named executables. (Perl script)

dpkg-source

Pack and unpack Debian source archives. (Perl script)

dpkg-statoverride

Manage the list of stat overrides, which let **dpkg** override file ownership and mode when a package is installed. (Perl script)

Debian Package Manager Command Summary

For the **apt-** commands, options can be specified on the command line or set in the configuration file. Boolean options set in the configuration file can be overridden on the command line in a number of different ways, such as **--no-opt** and **-opt=no**, where *opt* is the single-character or full name of the option.

apt-cache

apt-cache [*options*] *command*

Perform low-level operations on the APT binary cache, including the ability to perform searches and produce output reports from package metadata. Useful for finding out information about packages.

Commands

add *files*

Add the specified package index files to the source cache.

depends *pkgs*

For each specified package, show a list of dependencies and packages that can fulfill the dependency.

dotty *pkgs*

Graph the relationships between the specified packages. The default is to trace out all dependent packages; turn this behavior off by setting the **APT::Cache::GivenOnly** configuration option.

dump

List every package in the cache. Used for debugging.

dumpavail

Print a list of available packages to standard output, suitable for use with **dpkg**.

gencaches

Build source and package caches from the sources in the file *sources.list* and from */var/lib/dpkg/status*. Equivalent to running **apt-get check**.

pkgnames [*prefix*]

Print a list of packages in the system. If *prefix* is specified, print only packages whose names begin with that prefix. Most useful with the **--generate** option.

policy [*pkgs*]

Print detailed information about the priority selection of each specified package. With no arguments, print the priorities of each source. Useful for debugging issues related to the *preferences* file.

search *regex*

Search package names and descriptions of all available package files for the specified regular expression and print the name and short description of each matching package. With **--full**, the output is identical to that from the **show** command. With **--names-only**, only the package name is searched.

show *pkgs*

Display the package records for each specified package. Similar to running **dpkg --print-avail**.

showpkg *pkgs*

Display information about the specified packages. For each package, the output includes the available versions, packages that depend on this package, and packages that this package depends on.

stats

Display statistics about the cache.

unmet

Display the unmet dependencies in the package cache.

Options

-a, --all-versions

Print full records for all available versions. For use with the **show** commands. The configuration option is **APT::Cache::AllVersions**.

--all-names

Cause **pkgnames** to print all names, including virtual packages and missing dependencies. The configuration option is **APT::Cache::AllNames**.

-c file, --config-file=file

Specify a configuration file to be read after the default configuration file.

- f, --full**
Print full package records when searching. The configuration option is **APT::Cache::ShowFull**.
- g, --generate**
Automatically regenerate the package cache rather than using the current cache. The default is to regenerate; turn it off with **--no-generate**. The configuration option is **APT::Cache::Generate**.
- h, --help**
Print usage information and exit.
- i, --important**
Print only important dependencies. For use with **unmet**. The configuration option is **APT::Cache::Important**.
- names-only**
Search only on package names, not long descriptions. The configuration option is **APT::Cache::NamesOnly**.
- o, --option**
Set a configuration option. Syntax is **-o group::tool=option**.
- p file, --pkg-cache=file**
Use the specified file for the package cache, the primary cache used by all operations. The configuration option is **Dir::Cache::pkgcache**.
- q, --quiet**
Operate quietly, producing output for logging but no progress indicators. Use **-qq** for even quieter operation. The configuration option is **quiet**.
- recurse**
Run **depends** recursively, so all mentioned packages are printed once. The configuration option is **APT::Cache::RecurseDepends**.
- s file, --src-cache=file**
Specify the source cache file used by **gencaches**. The configuration option is **Dir::Cache::srcpkgcache**.
- v, --version**
Print version information and exit.

apt-cdrom**apt-cdrom** [*options*] *command*

Add a new CD-ROM to APT's list of available sources. The database of CD-ROM IDs that APT maintains is */var/lib/apt/cdroms.list*.

Commands**add**

Add a CD-ROM to the source list.

ident

Print the identity of the current CD-ROM and the stored filename. Used for debugging.

Options

- a, --thorough**
Do a thorough package scan. May be needed with some old Debian CD-ROMs.
- c file, --config-file=file**
Specify a configuration file to be read after the default configuration file.
- d mount-point, --cdrom=mount-point**
Specify the CD-ROM mount point, which must be listed in */etc/fstab*. The configuration option is **Acquire::cdrom::mount**.
- f, --fast**
Do a fast copy, assuming the files are valid and don't all need checking. Specify this only if disk has been run before without error. The configuration option is **APT::CDROM::Fast**.
- h, --help**
Print help message and exit.
- m, --no-mount**
Don't mount or unmount the mount point. The configuration option is **APT::CDROM::NoMount**.
- n, --just-print, --recon, --no-act**
Check everything, but don't actually make any changes. The configuration option is **APT::CDROM::NoAct**.
- o, --option**
Set a configuration option. Syntax is **-o group::tool=option**.
- r, --rename**
Prompt for a new label and rename the disk to the new value. The configuration option is **APT::CDROM::Rename**.
- v, --version**
Print the version information and exit.

apt-config

apt-config [*options*] **shell** *args*
apt-config [*options*] **dump**

An internal program for querying configuration information.

Commands

dump

Display the contents of the configuration space.

shell

Access the configuration information from a shell script. The arguments are in pairs, specifying the name of a shell variable and a configuration value to query. The value may be post-fixed with */x*, where *x* is one of the following letters:

- b** Return true or false.
- d** Return directories.
- f** Return filenames.
- i** Return an integer.

Options

- c file, --config-file=file**
Specify a configuration file to be read after the default configuration file.
- h, --help**
Print help message and exit.
- o, --option**
Set a configuration option. Syntax is **-o group::tool=option**.
- v, --version**
Print the version information and exit.

apt-extract-templates

apt-extracttemplates [*options*] *files*

Extract configuration scripts and templates from the specified Debian package files. For each specified file, a line of output is generated with the following information:

package version template-file config-script

and the template and configuration files are written to the directory specified with **-t** or **--temp-dir** or by the configuration option **APT::ExtractTemplates::TempDir**. The filenames are in the form *template.xxxx* and *config.xxxx*.

Options

- c file, --config-file=file**
Specify a configuration file to be read after the default configuration file.
- h, --help**
Print help message and exit.
- o, --option**
Set a configuration option. Syntax is **-o group::tool=option**.
- t dir, --tempdir=dir**
Write the extracted template files and configuration scripts to the specified directory. The configuration option is **APT::ExtractTemplates::TempDir**.
- v, --version**
Print the version information and exit.

apt-ftparchive

apt-ftparchive [*options*] *command*

Generate *Package* and other index files used to access a distribution source. The files should be generated on the source's origin site.

Commands

clean *config-file*

Clean the databases used by the specified configuration file by removing obsolete records.

contents *path*

Search the specified directory recursively. For each *.deb* file found, read the file list, sort the files by package, and write

the results to standard output. Use with `--db` to specify a binary caching database.

generate *config-file sections*

Build indexes according to the specified configuration file.

packages *paths [override [pathprefix]]*

Generate a package file from the specified directory tree. The optional override file contains information describing how the package fits into the distribution, and the optional path prefix is a string prepended to the filename fields. Equivalent to **dpkg-scanpackages**.

sources *paths [override [pathprefix]]*

Generate a source index file from the specified directory tree. The optional override file contains information used to set priorities in the index file and to modify maintainer information. The optional path prefix is a string prepended to the directory field in the generated source index. Use `--source-override` to specify a different source override file. Equivalent to **dpkg-scansources**.

Options

-c file, --config-file=file

Specify a configuration file to be read after the default configuration file.

--contents

Perform contents generation. If set, and package indexes are being generated with a cache database, the file listing is extracted and stored in the database. Used with **generate**, allows the creation of any contents files. The default is on. The configuration option is **APT::FTPArchive::Contents**.

-d, --db

Use a binary caching database. This option has no effect on **generate**. The configuration option is **APT::FTPArchive::DB**.

--delink

Enable delinking of files when used with the **External-Links** setting. The default is on; turn off with `--no-delink`. The configuration option is **APT::FTPArchive::DeLinkAct**.

-h, --help

Print help message and exit.

--md5

Generate MD5 sums for the index files. The default is on. The configuration option is **APT::FTPArchive::MD5**.

-o, --option

Set a configuration option. Syntax is `-o group::tool=option`.

-q, --quiet

Run quietly, producing logging information but no progress indicators. Use `-qq` for quieter operation. The configuration option is **quiet**.

--read-only

Make the caching databases read-only. The configuration option is **APT::FTPArchive::ReadOnlyDB**.

-s file, --source-override=file

Specify a source override file. For use with the **sources** command. The configuration option is **APT::FTPArchive::SourceOverride**.

-v, --version

Print the version information and exit.

apt-get

apt-get [*options*] *command* [*package...*]

A command-line tool for handling packages. Will eventually be a backend to APT.

Commands

autoclean

Like **clean**, but remove only package files that can no longer be downloaded.

build-dep

Install or remove packages to satisfy the build dependencies for a source package.

clean

Clear the local repository of retrieved package files.

check

Update the package cache and check for broken packages.

dist-upgrade

Like **upgrade**, but also handle dependencies intelligently. See the **-f** option for more information.

dselect-upgrade

Used with **dselect**. Track the changes made by **dselect** to the **Status** field of available packages and take actions necessary to realize that status.

install packages

Install one or more packages. Specify the package name, not the full filename. Other required packages are also retrieved and installed. With a hyphen appended to the package name, the package is removed if it is already installed.

remove packages

Remove one or more packages. Specify the package name, not the full filename. With a plus sign appended to the name, the package is installed.

source packages

Find source packages and download them into the current directory. If specified with **--compile**, the source packages are compiled into binary packages. With **--download-only**, the source packages are not unpacked.

update

Resynchronize the package overview files from their sources. Must be done before an **upgrade** or **dist-upgrade**.

upgrade

Install the latest versions of all packages currently installed. Run **update** first.

Options

-b, --compile, --build

Compile source packages after download. The configuration option is **APT::Get::Compile**.

-c file, --config-file=file

Specify a configuration file to read after the default.

-d, --download-only

Retrieve package files, but don't unpack or install them. The configuration option is **APT::Get::Download-Only**.

--diff-only

Download only the *diff* file from a source archive. The configuration option is **APT::Get::Diff-Only**.

-f, --fix-broken

Try to fix a system with broken dependencies. Can be used alone or with a command. Run with the **install** command if you have problems installing packages. You can run the sequence:

```
apt-get -f install
apt-get dist-upgrade
```

several times to clean up interlocking dependency problems. The configuration option is **APT::Get::Fix-Broken**.

--force-yes

Force yes. Causes APT to continue without prompting if it is doing something that could damage your system. Use with great caution and only if absolutely necessary. The configuration option is **APT::Get::force-yes**.

-h, --help

Display a help message and exit.

--ignore-hold

Ignore a hold placed on a package. Use with **dist-upgrade** to override many undesired holds. The configuration option is **APT::Get::Ignore-Hold**.

--list-cleanup

Erase obsolete files from */var/lib/apt/lists*. The default is on; use **--no-list-cleanup** to turn it off, which you would normally do only if you frequently modify your list of sources. The configuration option is **APT::Get::List-Cleanup**.

-m, --ignore-missing, --fix-missing

Ignore missing or corrupted packages or packages that cannot be retrieved. Can cause problems when used with **-f**. The configuration option is **APT::Get::Fix-Missing**.

--no-download

Disable package downloading; use with **--ignore-missing** to force APT to use only the packages that have already been downloaded. The configuration option is **APT::Get::Download**.

--no-remove

Do not remove any packages; instead, abort without prompting. The configuration option is **APT::Get::Remove**.

--no-upgrade

Do not upgrade packages. Use with **install** to prevent upgrade of packages that are already installed. The configuration option is **APT::Get::Upgrade**.

-o, --option

Set a configuration option. Syntax is **-o group::tool=option**.

--only-source

Do not map the names specified with the **source** command through the binary table. The configuration option is **APT::Get::Only-Source**.

--print-uris

Print URIs of files instead of fetching them. Print path, destination filename, size, and expected MD5 hash. The configuration option is **APT::Get::Print-URIs**.

--purge

Tell **dpkg** to do a purge instead of a remove for items that would be removed. Purging removes packages completely, including any configuration files. The configuration option is **APT::Get::Purge**.

-q, --quiet

Quiet mode. Omit progress indicators and produce only logging output. Use **-qq** to make even quieter. The configuration option is quiet.

--reinstall

Reinstall packages that are already installed, upgrading them to the latest version. The configuration option is **APT::Get::ReInstall**.

-s, --simulate, --just-print, --dry-run, --recon, --no-act

Go through the motions, but don't actually make any changes to the system. The configuration option is **APT::Get::Simulate**.

-t rel, --target-release=rel, --default-release=rel

Retrieve packages only from the specified release. The value of *rel* can be a release number or a value such as "unstable". The configuration option is **APT::Default-Release**.

--tar-only

Download only the TAR file from a source archive. The configuration option is **APT::Get::Tar-Only**.

- trivial-only**
Perform only operations that are considered trivial. The configuration option is **APT::Get::Trivial-Only**.
- u, --show-upgraded**
Print a list of all packages to be upgraded. The configuration option is **APT::Get::Show-Upgraded**.
- v, --version**
Display the version and exit.
- y, --yes, --assume-yes**
Automatically reply “yes” to prompts and run noninteractively. Abort if there is an error. The configuration option is **APT::Get::Assume-Yes**.

apt-sortpkgs

apt-sortpkgs [*options*] *indexfiles*

Sort the records in a source or package index file by package name and write the results to standard output. **apt-sortpkgs** also sorts the internal fields of each record.

Options

- c file, --config-file=file**
Specify a configuration file to read after the default.
- h, --help**
Display a help message and exit.
- o, --option**
Set a configuration option. Syntax is **-o group::tool=option**.
- s, --source**
Order by source index field. The configuration option is **APT::SortPkgs::Source**.
- v, --version**
Display the version and exit.

dpkg

dpkg [*options*] *action*

A tool for installing, managing, and building packages. Serves as a frontend to **dpkg-deb**.

dpkg actions

These actions are carried out by **dpkg** itself:

- A pkgfile, --record-avail pkgfile**
Update the record of available files kept in */var/lib/dpkg/available* with information from *pkgfile*. This information is used by **dpkg** and **dselect** to determine what packages are available. With **-R** or **--recursive**, *pkgfile* must be a directory.
- C, --audit**
Search for partially installed packages and suggest how to get them working.

- clear-avail**
Remove existing information about what packages are available.
- command-fd *n***
Accept commands passed on the file descriptor given by *n*. Note that any additional options set through this file descriptor or on the command line are not reset, but remain for other commands issued during the same session.
- compare-versions *ver1 op ver2***
Perform a binary comparison of two version numbers. The operators **lt le eq ne ge gt** treat a missing version as earlier. The operators **lt-nl le-nl ge-nl gt-nl** treat a missing version as later (where **nl** is “not later”). A third set of operators (**< <= = >= >**) is provided for compatibility with control-file syntax. **dpkg** returns zero for success (i.e., the condition is satisfied) and nonzero otherwise.
- configure [*packages*|-a|--pending]**
Reconfigure one or more unpacked *packages*. If **-a** or **--pending** is given instead of *packages*, configure all packages that are unpacked but not configured.
- Dh, --debug=help**
Print debugging help message and exit.
- force-help**
Print help message about the **--force-list** options and exit. See the **--force-list** option description for the possible values of *list*.
- forget-old-unavail**
Forget about uninstalled, unavailable packages.
- get-selections [*pattern*]**
Get list of package selections and write to standard output. With *pattern* specified, write selections that match the pattern.
- help**
Print help message and exit.
- i *pkgfile*, --install *pkgfile***
Install the package specified as *pkgfile*. With **-R** or **--recursive**, *pkgfile* must be a directory.
- l, --list [*pkg-name-pattern*]**
List all packages whose names match the specified pattern. With no pattern, list all packages in */var/lib/dpkg/available*. The pattern can include standard shell wildcard characters and may have to be quoted to prevent the shell from doing filename expansion.
- L *packages*, --listfiles *packages***
List installed files that came from the specified package or packages.
- license, --licence**
Print **dpkg** license information and exit.

- p, --print-avail** *package*
Print the details about *package* from */var/lib/dpkg/available*.
- print-architecture**
Print the target architecture.
- print-gnu-build-architecture**
Print the GNU version of the target architecture.
- print-installation-architecture**
Print the host architecture for installation.
- r, --remove** [*packages*|-**a**|-**pending**]
- purge** [*packages*|-**a**|-**pending**]
Remove or purge one or more installed *packages*. Removal gets rid of everything except the configuration files listed in *debian/conffiles*; purging also removes the configuration files. If **-a** or **--pending** is given instead of *packages*, **dpkg** removes or purges all packages that are unpacked and marked (in */var/lib/dpkg/status*) for removing or purging.
- s** *packages*, **--status** *packages*
Report the status of one or more *packages* by displaying the entry in the status database */var/lib/dpkg/status*.
- S** *filename-pattern*, **--search** *filename-pattern*
Search installed packages for a filename. The pattern can include standard shell wildcard characters and may have to be quoted to prevent the shell from doing filename expansion.
- set-selections**
Set package selections based on input file read from standard input.
- unpack** *pkgfile*
Unpack the package, but don't configure it. With **-R** or **--recursive**, *pkgfile* must be a directory.
- update-avail** *pkgs-file*
- merge-avail** *pkgs-file*
Update the record of available files kept in */var/lib/dpkg/available*. This information is used by **dpkg** and **dselect** to determine what packages are available. Update replaces the information with the contents of the *pkgs-file*, distributed as *Packages*. Merge combines the information from *Packages* with the existing information.
- version**
Print **dpkg** version information and exit.
- yet-to-unpack**
Search for uninstalled packages that have been selected for installation.

dpkg-deb actions

The following actions can be specified for **dpkg** and are passed to **dpkg-deb** for execution. Also see **dpkg-deb**.

- b** *dir* [*archive*], **--build** *dir* [*archive*]
Build a package.
- c** *archive*, **--contents** *archive*
List the contents of a package.
- e** *archive* [*dir*], **--control** *archive* [*dir*]
Extract control information from a package.
- f** *archive* [*control-fields*], **--field** *archive* [*control-fields*]
Display the control field or fields of a package.
- I** *archive* [*control-files*], **--info** *archive* [*control-files*]
Show information about a package.
- fsys-tarfile** *archive*
Display the filesystem TAR file contained in a package.
- x** *archive dir*, **--extract** *archive dir*
Extract the files from a package.
- X** *archive dir*, **--vextract** *archive dir*
Extract and display the filenames from a package.

Options

dpkg options can be specified on the command line or set in the configuration file. Each line in the configuration file contains a single option, specified without the leading dash (-).

- abort-after=num**
Abort processing after *num* errors. Default is 50.
- B, --auto-deconfigure**
When a package is removed, automatically deconfigure any other package that depended on it.
- Doctal, --debug=octal**
Turn on debugging, with the *octal* value specifying the desired level of debugging information. Use **-Dh** or **--debug=help** to display the possible values. You can OR the values to get the desired output.
- E, --skip-same-version**
Don't install the package if this version is already installed.
- force-list, --no-force-list, --refuse-list**
Force or refuse to force an operation. *list* is specified as a comma-separated list of options. With **--force**, a warning is printed, but processing continues. **--refuse** and **--no-force** cause processing to stop with an error. The force/refuse options are:
 - all**
Turn all force options on or off.
 - architecture**
Process even if intended for a different architecture.
 - auto-select**
Select or deselect packages to install or remove them. Forced by default.

bad-path

Some programs are missing from the path.

confdef

Always choose the default action for modified configuration files. If there is no default and **confnew** or **confold** is also specified, use that to decide; otherwise, ask the user.

configure-any

Configure any unconfigured package that the package depends on.

conflicts

Permit installation of conflicting packages. Can result in problems from files being overwritten.

confmiss

Always install a missing configuration file. Be careful using this option, since it means overriding the removal of the file.

confnew

Always install the new version of a modified configuration file unless **confdef** is also specified. In that case, use the default action if there is one.

confold

Keep the old version of a modified configuration file unless **confdef** is also specified. In that case, use the default action if there is one.

depends

Turn dependency problems into warnings.

depends-version

Warn of version problems when checking dependencies, but otherwise ignore.

downgrade

Install even if a newer version is already installed. Forced by default.

hold

Process packages even if they are marked to be held.

not-root

Try to install or remove even when not logged on as root.

overwrite

Overwrite a file from one package with the same file from another package. Forced by default.

overwrite-dir

Overwrite one package's directory with a file from another package.

overwrite-diverted

Overwrite a diverted file with an undiverted version.

remove-essential

Remove a package even if it is essential. Note that this can cause your system to stop working.

remove-reinstreq

Remove packages that are broken and are marked to require reinstallation.

-G Don't install a package if a newer version is already installed. Same as **--refuse-downgrade**.

--ignore-depends=pkglist

Dependency problems result only in a warning for the packages in *pkglist*.

--new

New binary package format. This is a **dpkg-deb** option.

--no-act

Go through the motions, but don't actually write any changes. Used for testing. Be sure to specify before the action; otherwise, changes might be written.

--nocheck

Ignore the contents of the control file when building a package. This is a **dpkg-deb** option.

-O, --selected-only

Process only packages that are marked as selected for installation.

--old

Old binary package format. This is a **dpkg-deb** option.

-R, --recursive

Recursively handle *.deb* files found in the directories and their subdirectories specified with **-A**, **--install**, **--unpack**, and **--avail**.

--root=dir, --admindir=dir, --instdir=dir

Change default directories. **admindir** contains administrative files with status and other information about packages; it defaults to */var/lib/dpkg*. **instdir** is the directory in which packages are installed; it defaults to *.*. Changing the **root** directory to *dir* automatically changes **instdir** to *dir* and **admindir** to */dir/var/lib/dpkg*.

--status-fd n

Send the package status information to the specified file descriptor. Can be given more than once.

dpkg-deb

dpkg-deb *action* [*options*]

Backend command for building and managing Debian package archives. Also see **dpkg**; you'll often want to use **dpkg** to pass commands through to **dpkg-deb**, rather than call **dpkg-deb** directly.

Actions

-b dir [*archive*], **--build dir** [*archive*]

Create an *archive* from the filesystem tree starting with directory *dir*. The directory must have a *DEBIAN* subdirectory containing the control file and any other control information.

If *archive* is specified and is a filename, the package is written to that file; if no *archive* is specified, the package is written to *dir.deb*. If the archive already exists, it is replaced. If *archive* is the name of a directory, **dpkg-deb** looks in the control file for the information it needs to generate the package name. (Note that for this reason, you cannot use **--nocheck** with a directory name.)

-c archive, --contents archive

List the filesystem-tree portion of *archive*.

-e archive [dir], --control archive [dir]

Extract control information from *archive* into the directory *dir*, which is created if it doesn't exist.

-f archive [control-fields], --field archive [control-fields]

Extract information about one or more fields in the control file for *archive*. If no fields are provided, print the entire control file.

-h, --help

Print help information and exit.

-I archive [control-files], --info archive [control-files]

Provide information about binary package *archive*. If no control files are provided, print a summary of the package contents; otherwise, print the control files in the order they were specified. An error message is printed to standard error for any missing components.

--fsys-tarfile archive

Extract the filesystem tree from *archive*, and send it to standard output in **tar** format. Can be used with **tar** to extract individual files from an archive.

--license, --licence

Print the license information and exit.

--version

Print the version number and exit.

-x archive dir, --extract archive dir

-X archive dir, --vextract archive dir

Extract the filesystem tree from *archive* into the specified directory, creating *dir* if it doesn't already exist. **-x (--extract)** works silently, while **-X (--vextract)** lists the files as it extracts them. Do not use this action to install packages; use **dpkg** instead.

Options

-D, --debug

Turn on debugging.

--new

Build a new-style archive format (this is the default).

- nocheck**
Don't check the control file before building an archive. This lets you build a broken archive.
- old**
Build an old-style archive format.

dpkg-query

dpkg-query [*option*] *command*

Display information about packages listed in the **dpkg** database.

Commands

- help**
Print help information and exit.
- l** [*patterns*], **--list** [*patterns*]
List packages whose names match any of the specified patterns. With no pattern specified, list all packages in */var/lib/dpkg/available*. The pattern may need to be in quotes to avoid expansion by the shell.
- L** *packages*, **--list** *packages*
List files installed on your system from each of the specified packages. This command does not list files created by package-specific installation scripts.
- license**, **--licence**
Print the license information and exit.
- p** *package*, **--print-avail** *package*
Display details for the specified package, as found in */var/lib/dpkg/available*.
- S** *patterns*
Search the installed packages for filenames matching one of the specified patterns. At least one pattern must be specified.
- W** [*patterns*], **--show** [*patterns*]
Like **-l**, but the output can be customized with the **--show-format** option.
- version**
Print version information and exit.

Options

- admindir**=*dir*
Use *dir* as the location of the **dpkg** database. The default is */var/lib/dpkg*.
- showformat**=*format*
Specify the output format for **-W/--show**. The format can include the standard escape sequences **\n** (newline), **\r** (carriage return), or **** (backslash). Specify package fields with the syntax **\${var[;width]}**. Fields are right-aligned by default, or left-aligned if *width* is negative.

dpkg-split

dpkg-split [*action*] [*options*]

Split a binary package into smaller pieces and reassemble the pieces, either manually or in automatic mode. The automatic mode maintains a queue of parts for reassembling. Useful for transferring to and from floppy disks.

Actions

-a -o *output part*, **--auto -o** *output part*

Add *part* to the queue for automatic reassembly, and if all the parts are available, reassemble the package as *output*. Requires the use of the **-o** (or **--output**) option, as shown.

-d [*packages*], **--discard** [*packages*]

Discard parts from the automatic-assembly queue. If any *packages* are specified, discard only parts from those packages. Otherwise, empty the queue.

-I *parts*, **--info** *parts*

Print information about the part file or files specified.

-j *parts*, **--join** *parts*

Join the parts of a package file together from the *parts* specified. The default output file is *package-version.deb*.

-l, **--listq**

List the contents of the queue of parts waiting for reassembly, giving the package name, the parts that are on the queue, and the number of bytes.

-s *full-package* [*prefix*], **--split** *full-package* [*prefix*]

Split the package *full-package* into parts, named *prefixNofM.deb*. The prefix defaults to the *full-package* name without the *.deb* extension.

-h, **--help**

Print help message and exit.

--license, **--licence**

Print license information and exit.

--version

Print version information and exit.

Options

--depotdir *dir*

Specify an alternate directory *dir* for the queue of parts waiting for reassembly. Default is */var/lib/dpkg*.

--msdos

Force **--split** output filenames to be MS-DOS-compatible.

-Q, **--npquiet**

Do not print an error message for a part that doesn't belong to a binary package when doing automatic queuing or reassembly.

-O *output*, **--output** *output*

Use *output* as the filename for a reassembled package.

-S num, --partsize num

When splitting, specify the maximum part size (*num*) in kilobytes. Default is 450 KB.

dselect

dselect [*options*] [*action*]

A screen-oriented user frontend to **dpkg**. The primary user interface for installing and managing packages. See **dpkg** and **dpkg-deb** for information on building packages.

Actions

If **dselect** is run with no action specified on the command line, it displays the following menu:

- ```
* 0. [A]ccess Choose the access method to use.
 1. [U]pdate Update list of available packages, if
 possible.
 2. [S]elect Request which packages you want on your
 system.
 3. [I]nstall Install and upgrade wanted packages.
 4. [C]onfig Configure any packages that are
 unconfigured.
 5. [R]emove Remove unwanted software.
 6. [Q]uit Quit dselect.
```

The asterisk (on the first line) shows the currently selected option. Any of the menu items can be specified directly on the command line as an action (**access**, **update**, **select**, **install**, **config**, **remove**, **quit**) to go directly to the desired activity. For example:

```
% dselect access
```

If you enter **quit** on the command line, **dselect** exits immediately without doing anything. An additional command-line action is **menu**, which displays the menu and is equivalent to running **dselect** with no action.

### Options

Options can be specified both on the command line and in the **dselect** configuration file, */etc/dpkg/dselect.cfg*.

**--admindir dir**

Change the directory that holds internal data files to *dir*. Default is */var/lib/dpkg*.

**--color colorspec, --colour colorspec**

Set colors for different parts of the screen, as specified by *colorspec* as follows:

```
screenpart:[fgcolor],[bgcolor][:attr[+attr+...]]
```

This option can be specified multiple times, to override the default colors for different screen parts. Rather than having to specify the colors on the command line each time you run **dselect**, you might prefer to set them in the configuration file. The possible screen parts (going from the top of the screen to the bottom) are:

**title**  
The screen title.

**listhead**  
The header line above the package list.

**list**  
The scrolling list of packages and some help text.

**listsel**  
The selected item in the list.

**pkgstate**  
The text showing the current state of each package.

**pkgstatesel**  
The text showing the current state of the selected package.

**infohead**  
The header line showing the state of the selected package.

**infodesc**  
The short description of the package.

**info**  
The text that displays information such as the package description.

**infofoot**  
The last line of the screen when selecting packages.

**query**  
Query lines.

**helpscreen**  
The color of help screens.  
Either the foreground color, the background color, or both can be specified for each screen part. The colors are given as the standard **curses** colors. After the color specification, you can specify a list of attributes separated by plus signs (+). The possible attributes are **normal**, **standout**, **underline**, **reverse**, **blink**, **bright**, **dim**, and **bold**. Not all attributes work on all terminals.

**--expert**  
Run in expert mode; don't print help messages.

**-D [file], --debug [file]**  
Turn on debugging. Send output to *file* if specified.

**--help**  
Print help message and exit.

**--license, licence**  
Print license information and exit.

**--version**  
Print version information and exit.

---

---