NAME

dpkg - package manager for Debian

SYNOPSIS

dpkg [*option...*] *action*

WARNING

This manual is intended for users wishing to understand **dpkg**'s command line options and package states in more detail than that provided by **dpkg --help**.

It should *not* be used by package maintainers wishing to understand how **dpkg** will install their packages. The descriptions of what **dpkg** does when installing and removing packages are particularly inadequate.

DESCRIPTION

dpkg is a tool to install, build, remove and manage Debian packages. The primary and more user-friendly front-end for **dpkg** is **aptitude(1)**. **dpkg** itself is controlled entirely via command line parameters, which consist of exactly one action and zero or more options. The action-parameter tells **dpkg** what to do and options control the behavior of the action in some way.

dpkg can also be used as a front-end to **dpkg-deb(1)** and **dpkg-query(1)**. The list of supported actions can be found later on in the **ACTIONS** section. If any such action is encountered **dpkg** just runs **dpkg-deb** or **dpkg-query** with the parameters given to it, but no specific options are currently passed to them, to use any such option the back-ends need to be called directly.

INFORMATION ABOUT PACKAGES

dpkg maintains some usable information about available packages. The information is divided in three classes: **states**, **selection states** and **flags**. These values are intended to be changed mainly with **dselect**.

Package states

not-installed
: The package is not installed on your system.

config-files
: Only the configuration files of the package exist on the system.

half-installed
: The installation of the package has been started, but not completed for some reason.

unpacked
: The package is unpacked, but not configured.

half-configured
: The package is unpacked and configuration has been started, but not yet completed for some reason.

triggers-awaited
: The package awaits trigger processing by another package.

triggers-pending
: The package has been triggered.

installed
: The package is correctly unpacked and configured.

Package selection states

install The package is selected for installation.

hold A package marked to be on **hold** is not handled by **dpkg**, unless forced to do that with option **--force-hold**.

deinstall
The package is selected for deinstallation (i.e. we want to remove all files, except configuration files).

purge The package is selected to be purged (i.e. we want to remove everything from system directories, even configuration files).

Package flags

reinst-required
A package marked **reinst-required** is broken and requires reinstallation. These packages cannot be removed, unless forced with option **--force-remove-reinstreq**.

ACTIONS

-i, **--install** *package-file...*
Install the package. If **--recursive** or **-R** option is specified, *package-file* must refer to a directory instead.

Installation consists of the following steps:

1. Extract the control files of the new package.

2. If another version of the same package was installed before the new installation, execute *prerm* script of the old package.

3. Run *preinst* script, if provided by the package.

4. Unpack the new files, and at the same time back up the old files, so that if something goes wrong, they can be restored.

5. If another version of the same package was installed before the new installation, execute the *postrm* script of the old package. Note that this script is executed after the *preinst* script of the new package, because new files are written at the same time old files are removed.

6. Configure the package. See **--configure** for detailed information about how this is done.

--unpack *package-file...*
Unpack the package, but don't configure it. If **--recursive** or **-R** option is specified, *package-file* must refer to a directory instead.

--configure *package...*|**-a**|**--pending**
Configure a package which has been unpacked but not yet configured. If **-a** or **--pending** is given instead of *package*, all unpacked but unconfigured packages are configured.

To reconfigure a package which has already been configured, try the dpkg-reconfigure(8) command instead.

Configuring consists of the following steps:

1. Unpack the conffiles, and at the same time back up the old conffiles, so that they can be restored if something goes wrong.

2. Run *postinst* script, if provided by the package.

--triggers-only *package...*|**-a**|**--pending**
Processes only triggers. All pending triggers will be processed. If package names are supplied only those packages' triggers will be processed, exactly once each where necessary. Use of this option may leave packages in the improper **triggers-awaited** and **triggers-pending** states. This can be fixed later by running: **dpkg --configure --pending**.

-r, **--remove** *package...*|**-a**|**--pending**
Remove an installed package. This removes everything except conffiles, which may avoid having to reconfigure the package if it is reinstalled later (conffiles are configuration files

that are listed in the *DEBIAN/conffiles* control file). If **-a** or **--pending** is given instead of a package name, then all packages unpacked, but marked to be removed in file */var/lib/dpkg/status*, are removed.

Removing of a package consists of the following steps:

1. Run *prerm* script

2. Remove the installed files

3. Run *postrm* script

-P, **--purge** *package...*|**-a**|**--pending**
Purge an installed or already removed package. This removes everything, including conffiles. If**-a** or **--pending** is giv en instead of a package name, then all packages unpacked or removed, but marked to be purged in file */var/lib/dpkg/status*, are purged.

Note: some configuration files might be unknown to **dpkg** because they are created and handled separately through the configuration scripts. In that case, **dpkg** won't remove them by itself, but the package's *postrm* script (which is called by **dpkg**), has to take care of their removal during purge. Of course, this only applies to files in system directories, not configuration files written to individual users' home directories.

Purging of a package consists of the following steps:

1. Remove the package, if not already removed. See **--remove** for detailed information about how this is done.

2. Run *postrm* script.

-V, **--verify** [*package-name...*]
Verifies the integrity of *package-name* or all packages if omitted, by comparing information from the files installed by a package with the files metadata information stored in the **dpkg** database. The origin of the files metadata information in the database is the binary packages themselves. That metadata gets collected at package unpack time during the installation process.

Currently the only functional check performed is an md5sum verification against the stored value in the files database. It will only get checked if the database contains the file md5sum. To check for any missing metadata in the database, the **--audit** command can be used.

The output format is selectable with the **--verify-format** option, which by default uses the **rpm** format, but that might change in the future, and as such, programs parsing this command output should be explicit about the format they expect.

--update-avail, **--merge-avail** [*Packages-file*]
Update **dpkg**'s and **dselect**'s idea of which packages are available. With action **--merge-avail**, old information is combined with information from *Packages-file*. With action **--update-avail**, old information is replaced with the information in the *Packages-file*. The *Packages-file* distributed with Debian is simply named *Packages*. If the *Packages-file* argument is missing or named **-** then it will be read from standard input (since dpkg 1.17.7). **dpkg** keeps its record of available packages in */var/lib/dpkg/available*.

A simpler one-shot command to retrieve and update the *available* file is **dselect update**. Note that this file is mostly useless if you don't use **dselect** but an APT-based frontend: APT has its own system to keep track of available packages.

-A, **--record-avail** *package-file...*
Update **dpkg** and **dselect**'s idea of which packages are available with information from the package *package-file*. If **--recursive** or **-R** option is specified, *package-file* must refer to a directory instead.

--forget-old-unavail
Now **obsolete** and a no-op as **dpkg** will automatically forget uninstalled unavailable packages.

--clear-avail
Erase the existing information about what packages are available.

-C, **--audit** [*package-name...*]
Performs database sanity and consistency checks for *package-name* or all packages if omitted. For example, searches for packages that have been installed only partially on your system or that have missing, wrong or obsolete control data or files. **dpkg** will suggest what to do with them to get them fixed.

--get-selections [*package-name-pattern...*]
Get list of package selections, and write it to stdout. Without a pattern, non-installed packages (i.e. those which have been previously purged) will not be shown.

--set-selections
Set package selections using file read from stdin. This file should be in the format '*package state*', where state is one of **install**, **hold**, **deinstall** or **purge**. Blank lines and comment lines beginning with '#' are also permitted.

The *available* file needs to be up-to-date for this command to be useful, otherwise unknown packages will be ignored with a warning. See the **--update-avail** and **--merge-avail** commands for more information.

--clear-selections
Set the requested state of every non-essential package to deinstall. This is intended to be used immediately before **--set-selections**, to deinstall any packages not in list given to **--set-selections**.

--yet-to-unpack
Searches for packages selected for installation, but which for some reason still haven't been installed.

--add-architecture *architecture*
Add *architecture* to the list of architectures for which packages can be installed without using **--force-architecture**. The architecture **dpkg** is built for (i.e. the output of **--print-architecture**) is always part of that list.

--remove-architecture *architecture*
Remove *architecture* from the list of architectures for which packages can be installed without using **--force-architecture**. If the architecture is currently in use in the database then the operation will be refused, except if **--force-architecture** is specified. The architecture **dpkg** is built for (i.e. the output of **--print-architecture**) can never be removed from that list.

--print-architecture
Print architecture of packages **dpkg** installs (for example, i386).

--print-foreign-architectures
Print a newline-separated list of the extra architectures **dpkg** is configured to allow packages to be installed for.

--compare-versions *ver1 op ver2*
Compare version numbers, where *op* is a binary operator. **dpkg** returns success (zero result) if the specified condition is satisfied, and failure (nonzero result) otherwise. There are two groups of operators, which differ in how they treat an empty *ver1* or *ver2*. These treat an empty version as earlier than any version: **lt le eq ne ge gt**. These treat an empty version as later than any version: **lt-nl le-nl ge-nl gt-nl**. These are provided only for compatibility with control file syntax: < << <= = >= >> >.

-?, **--help**
Display a brief help message.

--force-help
Give help about the **--force-***thing* options.

-Dh, **--debug=help**
Give help about debugging options.

--version
Display **dpkg** version information.

dpkg-deb actions
See **dpkg-deb(1)** for more information about the following actions.

-b, **--build** *directory* [*archive*|*directory*]
Build a deb package.
-c, **--contents** *archive*
List contents of a deb package.
-e, **--control** *filename* [*directory*]
Extract control-information from a package.
-x, **--extract** *archive directory*
Extract the files contained by package.
-X, **--vextract** *archive directory*
Extract and display the filenames contained by a package.
-f, **--field** *archive* [*control-field...*]
Display control field(s) of a package.
--fsys-tarfile *archive*
Display the filesystem tar-file contained by a Debian package.
-I, **--info** *archive* [*control-file...*]
Show information about a package.

dpkg-query actions
See **dpkg-query(1)** for more information about the following actions.

-l, **--list** *package-name-pattern...*
List packages matching given pattern.
-s, **--status** *package-name...*
Report status of specified package.
-L, **--listfiles** *package-name...*
List files installed to your system from *package-name*.
-S, **--search** *filename-search-pattern...*
Search for a filename from installed packages.
-p, **--print-avail** *package-name...*
Display details about *package-name*, as found in */var/lib/dpkg/available*. Users of APT-based frontends should use **apt-cache show** *package-name* instead.

OPTIONS

All options can be specified both on the command line and in the **dpkg** configuration file */etc/dpkg/dpkg.cfg* or fragment files (with names matching this shell pattern [0-9a-zA-Z_-]*) on the configuration directory */etc/dpkg/dpkg.cfg.d/*. Each line in the configuration file is either an option (exactly the same as the command line option but without leading hyphens) or a comment (if it starts with a #).

--abort-after=*number*
Change after how many errors **dpkg** will abort. The default is 50.

-B, **--auto-deconfigure**
When a package is removed, there is a possibility that another installed package depended on the removed package. Specifying this option will cause automatic deconfiguration of the package which depended on the removed package.

-D*octal*, **--debug=***octal*
Switch debugging on. *octal* is formed by bitwise-orring desired values together from the list below (note that these values may change in future releases). **-Dh** or **--debug=help** display these debugging values.

Number Description 1 Generally helpful progress information 2 Invocation and status of maintainer scripts 10 Output for each file processed 100 Lots of output for each file processed 20 Output for each configuration file 200 Lots of output for each configuration file 40 Dependencies and conflicts 400 Lots of dependencies/conflicts output 10000 Trigger activation and processing 20000 Lots of output regarding triggers 40000 Silly amounts of output regarding triggers 1000 Lots of drivel about e.g. the dpkg/info dir 2000 Insane amounts of drivel

--force-*things*, **--no-force-***things*, **--refuse-***things*

Force or refuse (**no-force** and **refuse** mean the same thing) to do some things. *things* is a comma separated list of things specified below. **--force-help** displays a message describing them. Things marked with (*) are forced by default.

Warning: These options are mostly intended to be used by experts only. Using them without fully understanding their effects may break your whole system.

all: Turns on (or off) all force options.

downgrade(*): Install a package, even if newer version of it is already installed.

Warning: At present **dpkg** *does not do any dependency checking on downgrades and therefore will not warn you if the downgrade breaks the dependency of some other package. This can have serious side effects, downgrading essential system components can even make your whole system unusable. Use with care.*

configure-any: Configure also any unpacked but unconfigured packages on which the current package depends.

hold: Process packages even when marked hold.

remove-reinstreq: Remove a package, even if it's broken and marked to require reinstallation. This may, for example, cause parts of the package to remain on the system, which will then be forgotten by **dpkg**.

remove-essential: Remove, even if the package is considered essential. Essential packages contain mostly very basic Unix commands. Removing them might cause the whole system to stop working, so use with caution.

depends: Turn all dependency problems into warnings.

depends-version: Don't care about versions when checking dependencies.

breaks: Install, even if this would break another package.

conflicts: Install, even if it conflicts with another package. This is dangerous, for it will usually cause overwriting of some files.

confmiss: If a conffile is missing and the version in the package did change, always install the missing conffile without prompting. This is dangerous, since it means not preserving a change (removing) made to the file.

confnew: If a conffile has been modified and the version in the package did change, always install the new version without prompting, unless the **--force-confdef** is also specified, in which case the default action is preferred.

confold: If a conffile has been modified and the version in the package did change, always keep the old version without prompting, unless the **--force-confdef** is also specified, in which case the default action is preferred.

confdef: If a conffile has been modified and the version in the package did change, always choose the default action without prompting. If there is no default action it will stop to ask the user unless **--force-confnew** or **--force-confold** is also been given, in which case it will use that to decide the final action.

confask: If a conffile has been modified always offer to replace it with the version in the package, even if the version in the package did not change. If any of **--force-confmiss**, **--force-confnew**, **--force-confold**, or **--force-confdef** is also given, it will be used to decide the final action.

overwrite: Overwrite one package's file with another's file.

overwrite-dir Overwrite one package's directory with another's file.

overwrite-diverted: Overwrite a diverted file with an undiverted version.

unsafe-io: Do not perform safe I/O operations when unpacking. Currently this implies not performing file system syncs before file renames, which is known to cause substantial performance degradation on some file systems, unfortunately the ones that require the safe I/O on the first place due to their unreliable behaviour causing zero-length files on abrupt system crashes.

Note: For ext4, the main offender, consider using instead the mount option **nodelalloc**, which will fix both the performance degradation and the data safety issues, the latter by making the file system not produce zero-length files on abrupt system crashes with any software not doing syncs before atomic renames.

Warning: Using this option might improve performance at the cost of losing data, use with care.

architecture: Process even packages with wrong or no architecture.

bad-version: Process even packages with wrong versions.

bad-path: **PATH** is missing important programs, so problems are likely.

not-root: Try to (de)install things even when not root.

bad-verify: Install a package even if it fails authenticity check.

--ignore-depends=*package*,...
Ignore dependency-checking for specified packages (actually, checking is performed, but only warnings about conflicts are given, nothing else).

--no-act, **--dry-run**, **--simulate**
Do everything which is supposed to be done, but don't write any changes. This is used to see what would happen with the specified action, without actually modifying anything.

Be sure to give **--no-act** before the action-parameter, or you might end up with undesirable results. (e.g. **dpkg --purge foo --no-act** will first purge package foo and then try to purge package --no-act, even though you probably expected it to actually do nothing)

-R, **--recursive**
Recursively handle all regular files matching pattern ***.deb** found at specified directories and all of its subdirectories. This can be used with **-i**, **-A**, **--install**, **--unpack** and

--avail actions.

-G Don't install a package if a newer version of the same package is already installed. This is an alias of **--refuse-downgrade**.

--admindir=*dir*
Change default administrative directory, which contains many files that give information about status of installed or uninstalled packages, etc. (Defaults to */var/lib/dpkg*)

--instdir=*dir*
Change default installation directory which refers to the directory where packages are to be installed. **instdir** is also the directory passed to **chroot(2)** before running package's installation scripts, which means that the scripts see **instdir** as a root directory. (Defaults to */*)

--root=*dir*
Changing **root** changes **instdir** to *dir* and **admindir** to *dir*/**var/lib/dpkg**.

-O, **--selected-only**
Only process the packages that are selected for installation. The actual marking is done with **dselect** or by **dpkg**, when it handles packages. For example, when a package is removed, it will be marked selected for deinstallation.

-E, **--skip-same-version**
Don't install the package if the same version of the package is already installed.

--pre-invoke=*command*
--post-invoke=*command*
Set an invoke hook *command* to be run via "sh -c" before or after the **dpkg** run for the *unpack*, *configure*, *install*, *triggers-only*, *remove*, *purge*, *add-architecture* and *remove-architecture* **dpkg** actions. This option can be specified multiple times. The order the options are specified is preserved, with the ones from the configuration files taking precedence. The environment variable **DPKG_HOOK_ACTION** is set for the hooks to the current **dpkg** action. Note: front-ends might call **dpkg** several times per invocation, which might run the hooks more times than expected.

--path-exclude=*glob-pattern*
--path-include=*glob-pattern*
Set *glob-pattern* as a path filter, either by excluding or re-including previously excluded paths matching the specified patterns during install.

Warning: take into account that depending on the excluded paths you might completely break your system, use with caution.

The glob patterns use the same wildcards used in the shell, were '*' matches any sequence of characters, including the empty string and also '/'. For example, *'/usr/*/READ*'* matches *'/usr/share/doc/package/README'*. As usual, '?' matches any single character (again, including '/'). And '[' starts a character class, which can contain a list of characters, ranges and complementations. See **glob(7)** for detailed information about globbing. Note: the current implementation might re-include more directories and symlinks than needed, to be on the safe side and avoid possible unpack failures, future work might fix this.

This can be used to remove all paths except some particular ones; a typical case is:

--path-exclude=/usr/share/doc/*
--path-include=/usr/share/doc/*/copyright

to remove all documentation files except the copyright files.

These two options can be specified multiple times, and interleaved with each other. Both are processed in the given order, with the last rule that matches a file name making the

decision.

--verify-format *format-name*
Sets the output format for the **--verify** command.

The only currently supported output format is **rpm**, which consists of a line for every path that failed any check. The lines start with 9 characters to report each specific check result, a '**?**' implies the check could not be done (lack of support, file permissions, etc), '**.**' implies the check passed, and an alphanumeric character implies a specific check failed; the md5sum verification is denoted with a '**5**' on the third character. The line is followed by a space and an attribute character (currently '**c**' for conffiles), another space and the pathname.

--status-fd *n*
Send machine-readable package status and progress information to file descriptor *n*. This option can be specified multiple times. The information is generally one record per line, in one of the following forms:

status: *package***:** *status*
Package status changed; *status* is as in the status file.

status: *package* **: error :** *extended-error-message*
An error occurred. Any possible newlines in *extended-error-message* will be converted to spaces before output.

status: *file* **: conffile-prompt : '***real-old***' '***real-new***'** *useredited distedited*
User is being asked a conffile question.

processing: *stage***:** *package*
Sent just before a processing stage starts. *stage* is one of **upgrade**, **install** (both sent before unpacking), **configure**, **trigproc**, **disappear**, **remove**, **purge**.

--status-logger=*command*
Send machine-readable package status and progress information to the shell *command*'s standard input. This option can be specified multiple times. The output format used is the same as in **--status-fd.**

--log=*filename*
Log status change updates and actions to *filename*, instead of the default */var/log/dpkg.log*. If this option is given multiple times, the last filename is used. Log messages are of the form 'YYYY-MM-DD HH:MM:SS status *state pkg installed-version*' for status change updates; 'YYYY-MM-DD HH:MM:SS *action pkg installed-version available-version*' for actions where *action* is one of **install**, **upgrade**, **remove**, **purge**; and 'YYYY-MM-DD HH:MM:SS conffile *filename decision*' for conffile changes where *decision* is either **install** or **keep**.

--no-debsig
Do not try to verify package signatures.

--no-triggers
Do not run any triggers in this run (activations will still be recorded). If used with **--configure** *package* or **--triggers-only** *package* then the named package postinst will still be run even if only a triggers run is needed. Use of this option may leave packages in the improper **triggers-awaited** and **triggers-pending** states. This can be fixed later by running: **dpkg --configure --pending**.

--triggers
Cancels a previous **--no-triggers**.

ENVIRONMENT

HOME
If set, **dpkg** will use it as the directory from which to read the user specific configuration file.

TMPDIR
If set, **dpkg** will use it as the directory in which to create temporary files and directories.

PAGER
The program **dpkg** will execute when displaying the conffiles.

SHELL
The program **dpkg** will execute when starting a new shell.

COLUMNS
Sets the number of columns **dpkg** should use when displaying formatted text. Currently only used by -l.

DPKG_SHELL_REASON
Defined by **dpkg** on the shell spawned on the conffile prompt to examine the situation. Current valid value: **conffile-prompt**.

DPKG_CONFFILE_OLD
Defined by **dpkg** on the shell spawned on the conffile prompt to examine the situation. Contains the path to the old conffile.

DPKG_CONFFILE_NEW
Defined by **dpkg** on the shell spawned on the conffile prompt to examine the situation. Contains the path to the new conffile.

DPKG_RUNNING_VERSION
Defined by **dpkg** on the maintainer script environment to the version of the currently running **dpkg** instance.

DPKG_MAINTSCRIPT_PACKAGE
Defined by **dpkg** on the maintainer script environment to the (non-arch-qualified) package name being handled.

DPKG_MAINTSCRIPT_PACKAGE_REFCOUNT
Defined by **dpkg** on the maintainer script environment to the package reference count, i.e. the number of package instances with a state greater than **not-installed**. Since dpkg 1.17.2.

DPKG_MAINTSCRIPT_ARCH
Defined by **dpkg** on the maintainer script environment to the architecture the package got built for.

DPKG_MAINTSCRIPT_NAME
Defined by **dpkg** on the maintainer script environment to the name of the script running (preinst, postinst, prerm, postrm).

FILES

*/etc/dpkg/dpkg.cfg.d/[0-9a-zA-Z_-]**
Configuration fragment files.

/etc/dpkg/dpkg.cfg
Configuration file with default options.

/var/log/dpkg.log
Default log file (see */etc/dpkg/dpkg.cfg*(5) and option **--log**).

The other files listed below are in their default directories, see option **--admindir** to see how to change locations of these files.

/var/lib/dpkg/available
: List of available packages.

/var/lib/dpkg/status
: Statuses of available packages. This file contains information about whether a package is marked for removing or not, whether it is installed or not, etc. See section **INFORMATION ABOUT PACKAGES** for more info.

  The status file is backed up daily in */var/backups*. It can be useful if it's lost or corrupted due to filesystems troubles.

The following files are components of a binary package. See **deb(5)** for more information about them:

control
conffiles
preinst
postinst
prerm
postrm
triggers

BUGS

--no-act usually gives less information than might be helpful.

EXAMPLES

To list installed packages related to the editor **vi(1)** (note that **dpkg-query** does not load the *available* file anymore by default, and the **dpkg-query --load-avail** option should be used instead for that):

```
dpkg -l *vi*
```

To see the entries in */var/lib/dpkg/available* of two packages:

```
dpkg --print-avail elvis vim | less
```

To search the listing of packages yourself:

```
less /var/lib/dpkg/available
```

To remove an installed elvis package:

```
dpkg -r elvis
```

To install a package, you first need to find it in an archive or CDROM. The *available* file shows that the vim package is in section editors:

```
cd /media/cdrom/pool/main/v/vim
dpkg -i vim_4.5-3.deb
```

To make a local copy of the package selection states:

```
dpkg --get-selections >myselections
```

You might transfer this file to another computer, and after having updated the *available* file there with your package manager frontend of choice (see https://wiki.debian.org/Teams/Dpkg/FAQ for more details), for example:

```
apt-cache dumpavail | dpkg --merge-avail
```

or with dpkg 1.17.6 and earlier:

```
avail=‘mktemp‘
apt-cache dumpavail >$avail
dpkg --merge-avail $avail
rm $avail
```

you can install it with:

```
dpkg --clear-selections
dpkg --set-selections <myselections
```

Note that this will not actually install or remove anything, but just set the selection state on the

requested packages. You will need some other application to actually download and install the requested packages. For example, run **apt-get dselect-upgrade**.

Ordinarily, you will find that **dselect(1)** provides a more convenient way to modify the package selection states.

ADDITIONAL FUNCTIONALITY

Additional functionality can be gained by installing any of the following packages: **apt**, **aptitude** and **debsums**.

SEE ALSO

aptitude(1), **apt(1)**, **dselect(1)**, **dpkg-deb(1)**, **dpkg-query(1)**, **deb(5)**, **deb-control(5)**, **dpkg.cfg**(5), and **dpkg-reconfigure(8)**.

AUTHORS

See */usr/share/doc/dpkg/THANKS* for the list of people who have contributed to **dpkg**.