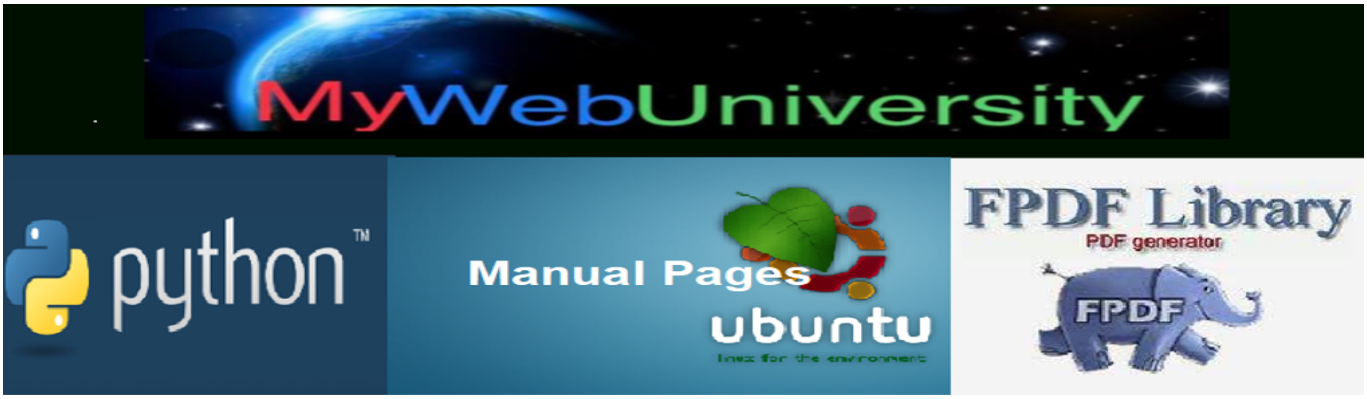

Rocky Enterprise Linux 9.2 Manual Pages on command 'deb-control.5'

$ man deb-control.5

deb-control(5) dpkg suite deb-control(5)

NAME

deb-control - Debian binary packages' master control file format

SYNOPSIS

DEBIAN/control

DESCRIPTION

Each Debian binary package contains a control file in its control member, and its deb822(5) format is a subset of the master debian/control file in Debian source packages, see deb-src-control(5).

This file contains a number of fields. Each field begins with a tag, such as Package or Version (case insensitive), followed by a colon, and the body of the field (case sensitive unless stated otherwise). Fields are delimited only by field tags. In other words, field text may be multiple lines in length, but the installation tools will generally join lines when processing the body of the field (except in the case of the Description field, see below).

FIELDS

Package: package-name (required)

The value of this field determines the package name, and is used to generate file names by most installation tools.

Package-Type: deb|udeb|type

This field defines the type of the package. udeb is for size-constrained packages used by the debian installer. deb is the default value, it is assumed if the field is absent. More types might be added in the future.

Version: version-string (required)

Typically, this is the original package's version number in whatever form the program's author uses. It may also include a Debian revision number (for non-native packages). The exact format and sorting algorithm are described in deb-version(7).

Maintainer: fullname-email (recommended)

Should be in the format ?Joe Bloggs <jbloggs@foo.com>?, and is typically the person who created the package, as opposed to the author of the software that was packaged.

Description: short-description (recommended)

long-description

The format for the package description is a short brief summary on the first line (after the Description field). The following lines should be used as a longer, more detailed description. Each line of the long description must be preceded by a space, and blank lines in the long description must contain a single ?.? following the preceding space.

Section: section

This is a general field that gives the package a category based on the software that it installs. Some common sections are utils, net, mail, text, x11, etc.

Priority: priority

Sets the importance of this package in relation to the system as a whole. Common priorities are required, standard, optional, extra, etc.

The Section and Priority fields usually have a defined set of accepted values based on the specific distribution policy.

Installed-Size: size

The approximate total size of the package's installed files, in KiB units. The algorithm to compute the size is described in deb-substvars(5).

Protected: yes|no

This field is usually only needed when the answer is yes. It denotes a package that is required for proper booting of the system. dpkg(1) or any other installation tool will not allow a Protected package to be removed (at least not without using one of the force options).

Supported since dpkg 1.20.1.

Essential: yes|no

This field is usually only needed when the answer is yes. It denotes a package that

is required for proper operation of the system. dpkg(1) or any other installation tool will not allow an Essential package to be removed (at least not without using one of the force options).

Build-Essential: yes|no

This field is usually only needed when the answer is yes, and is commonly injected by the archive software. It denotes a package that is required when building other packages.

Architecture: arch|all (required)

The architecture specifies which type of hardware this package was compiled for. Common architectures are amd64, armel, i386, powerpc, etc. Note that the all value is meant for packages that are architecture independent. Some examples of this are shell and Perl scripts, and documentation.

Origin: name

The name of the distribution this package is originating from.

Bugs: url

The url of the bug tracking system for this package. The current used format is bts-type://bts-address, like debbugs://bugs.debian.org.

Homepage: url

The upstream project home page url.

Tag: tag-list

List of tags describing the qualities of the package. The description and list of supported tags can be found in the debtags package.

Multi-Arch: no|same|foreign|allowed

This field is used to indicate how this package should behave on a multi-arch installations.

no This value is the default when the field is omitted, in which case adding the field with an explicit no value is generally not needed.

same

This package is co-installable with itself, but it must not be used to satisfy the dependency of any package of a different architecture from itself.

foreign

This package is not co-installable with itself, but should be allowed to satisfy a non-arch-qualified dependency of a package of a different arch from itself (if a

dependency has an explicit arch-qualifier then the value foreign is ignored).

allowed

This allows reverse-dependencies to indicate in their Depends field that they accept this package from a foreign architecture by qualifying the package name with :any, but has no effect otherwise.

Source: source-name [(source-version)]

The name of the source package that this binary package came from, if it is different than the name of the package itself. If the source version differs from the binary version, then the source-name will be followed by a source-version in parenthesis. This can happen for example on a binary-only non-maintainer upload, or when setting a different binary version via ?dpkg-gencontrol -v?.

Subarchitecture: value

Kernel-Version: value

Installer-Menu-Item: value

These fields are used by the debian-installer and are usually not needed. See /usr/share/doc/debian-installer/devel/modules.txt from the debian-installer package for more details about them.

Depends: package-list

List of packages that are required for this package to provide a non-trivial amount of functionality. The package maintenance software will not allow a package to be installed if the packages listed in its Depends field aren't installed (at least not without using the force options). In an installation, the postinst scripts of packages listed in Depends fields are run before those of the packages which depend on them. On the opposite, in a removal, the prerm script of a package is run before those of the packages listed in its Depends field.

Pre-Depends: package-list

List of packages that must be installed and configured before this one can be installed. This is usually used in the case where this package requires another package for running its preinst script.

Recommends: package-list

Lists packages that would be found together with this one in all but unusual installations. The package maintenance software will warn the user if they install a package without those listed in its Recommends field.

Suggests: package-list

Lists packages that are related to this one and can perhaps enhance its usefulness, but without which installing this package is perfectly reasonable.

The syntax of Depends, Pre-Depends, Recommends and Suggests fields is a list of groups of alternative packages. Each group is a list of packages separated by vertical bar (or ?pipe?) symbols, ?|?. The groups are separated by commas. Commas are to be read as ?AND?, and pipes as ?OR?, with pipes binding more tightly. Each package name is optionally followed by an architecture qualifier appended after a colon ?:?, optionally followed by a version number specification in parentheses.

An architecture qualifier name can be a real Debian architecture name (since dpkg 1.16.5) or any (since dpkg 1.16.2). If omitted, the default is the current binary package architecture. A real Debian architecture name will match exactly that architecture for that package name, any will match any architecture for that package name if the package has been marked as Multi-Arch: allowed.

A version number may start with a ?>>?, in which case any later version will match, and may specify or omit the Debian packaging revision (separated by a hyphen). Accepted version relationships are ?>>? for greater than, ?<<? for less than, ?>=? for greater than or equal to, ?<=? for less than or equal to, and ?=? for equal to.

Breaks: package-list

Lists packages that this one breaks, for example by exposing bugs when the named packages rely on this one. The package maintenance software will not allow broken packages to be configured; generally the resolution is to upgrade the packages named in a Breaks field.

Conflicts: package-list

Lists packages that conflict with this one, for example by containing files with the same names. The package maintenance software will not allow conflicting packages to be installed at the same time. Two conflicting packages should each include a Conflicts line mentioning the other.

Replaces: package-list

List of packages files from which this one replaces. This is used for allowing this package to overwrite the files of another package and is usually used with the Conflicts field to force removal of the other package, if this one also has the same files as the conflicted package.

The syntax of Breaks, Conflicts and Replaces is a list of package names, separated by commas (and optional whitespace). In the Breaks and Conflicts fields, the comma should be read as ?OR?. An optional architecture qualifier can also be appended to the package name with the same syntax as above, but the default is any instead of the binary package architecture. An optional version can also be given with the same syntax as above for the Breaks, Conflicts and Replaces fields.

Enhances: package-list

This is a list of packages that this one enhances. It is similar to Suggests but in the opposite direction.

Provides: package-list

This is a list of virtual packages that this one provides. Usually this is used in the case of several packages all providing the same service. For example, sendmail and exim can serve as a mail server, so they provide a common package (?mail-transport-agent?) on which other packages can depend. This will allow sendmail or exim to serve as a valid option to satisfy the dependency. This prevents the packages that depend on a mail server from having to know the package names for all of them, and using ?|? to separate the list.

The syntax of Provides is a list of package names, separated by commas (and optional whitespace). An optional architecture qualifier can also be appended to the package name with the same syntax as above. If omitted, the default is the current binary package architecture. An optional exact (equal to) version can also be given with the same syntax as above (honored since dpkg 1.17.11).

Built-Using: package-list

This field lists extra source packages that were used during the build of this binary package. This is an indication to the archive maintenance software that these extra source packages must be kept whilst this binary package is maintained. This field must be a list of source package names with strict ?=? version relationships. Note that the archive maintenance software is likely to refuse to accept an upload which declares a Built-Using relationship which cannot be satisfied within the archive.

Built-For-Profiles: profile-list (obsolete)

This field used to specify a whitespace separated list of build profiles that this binary packages was built with (since dpkg 1.17.2 until 1.18.18). The information previously found in this field can now be found in the .buildinfo file, which

supersedes it.

Auto-Built-Package: reason-list

This field specifies a whitespace separated list of reasons why this package was auto-generated. Binary packages marked with this field will not appear in the debian/control master source control file. The only currently used reason is debug-symbols.

Build-Ids: elf-build-id-list

This field specifies a whitespace separated list of ELF build-ids. These are unique identifiers for semantically identical ELF objects, for each of these within the package.

The format or the way to compute each build-id is not defined by design.

EXAMPLE

```
Package: grep
Essential: yes
Priority: required
Section: base
Maintainer: Wichert Akkerman <wakkerma@debian.org>
Architecture: sparc
Version: 2.4-1
Pre-Depends: libc6 (>= 2.0.105)
Provides: rgrep
Conflicts: rgrep
Description: GNU grep, egrep and fgrep.
 The GNU family of grep utilities may be the "fastest grep in the west".
 GNU grep is based on a fast lazy-state deterministic matcher (about
 twice as fast as stock Unix egrep) hybridized with a Boyer-Moore-Gosper
 search for a fixed string that eliminates impossible text from being
 considered by the full regexp matcher without necessarily having to
 look at every character. The result is typically many times faster
 than Unix grep or egrep. (Regular expressions containing backreferencing
 will run more slowly, however).
```

BUGS

The Build-Ids field uses a rather generic name out of its original context within an ELF

object, which serves a very specific purpose and executable format.

SEE ALSO

deb822(5), deb-src-control(5), deb(5), deb-version(7), debtags(1), dpkg(1), dpkg-deb(1).