NAME

apt_preferences − Preference control file for APT

DESCRIPTION

The APT preferences file /etc/apt/preferences and the fragment files in the /etc/apt/preferences.d/ folder can be used to control which versions of packages will be selected for installation.

Several versions of a package may be available for installation when the **sources.list**(5) file contains references to more than one distribution (for example, stable and testing). APT assigns a priority to each version that is available. Subject to dependency constraints, **apt−get** selects the version with the highest priority for installation. The APT preferences override the priorities that APT assigns to package versions by default, thus giving the user control over which one is selected for installation.

Several instances of the same version of a package may be available when the **sources.list**(5) file contains references to more than one source. In this case **apt−get** downloads the instance listed earliest in the **sources.list**(5) file. The APT preferences do not affect the choice of instance, only the choice of version.

Preferences are a strong power in the hands of a system administrator but they can become also their biggest nightmare if used without care! APT will not question the preferences, so wrong settings can lead to uninstallable packages or wrong decisions while upgrading packages. Even more problems will arise if multiple distribution releases are mixed without a good understanding of the following paragraphs. Packages included in a specific release aren't tested in (and therefore don't always work as expected in) older or newer releases, or together with other packages from different releases. You have been warned.

Note that the files in the /etc/apt/preferences.d directory are parsed in alphanumeric ascending order and need to obey the following naming convention: The files have either no or "pref" as filename extension and only contain alphanumeric, hyphen (−), underscore (_) and period (.) characters. Otherwise APT will print a notice that it has ignored a file, unless that file matches a pattern in the Dir::Ignore−Files−Silently configuration list − in which case it will be silently ignored.

APT's Default Priority Assignments

If there is no preferences file or if there is no entry in the file that applies to a particular version then the priority assigned to that version is the priority of the distribution to which that version belongs. It is possible to single out a distribution, "the target release", which receives a higher priority than other distributions do by default. The target release can be set on the **apt−get** command line or in the APT configuration file /etc/apt/apt.conf. Note that this has precedence over any general priority you set in the /etc/apt/preferences file described later, but not over specifically pinned packages. For example,

apt−get install −t testing *some−package*

APT::Default−Release "stable";

If the target release has been specified then APT uses the following algorithm to set the priorities of the versions of a package. Assign:

priority 1
: to the versions coming from archives which in their Release files are marked as "NotAutomatic: yes" but *not* as "ButAutomaticUpgrades: yes" like the Debian experimental archive, as well as versions that are not phased on this systems.

priority 100
: to the version that is already installed (if any) and to the versions coming from archives which in their Release files are marked as "NotAutomatic: yes" and "ButAutomaticUpgrades: yes" like the Debian backports archive since squeeze−backports.

priority 500
: to the versions that do not belong to the target release.

priority 990

to the versions that belong to the target release.
The highest of those priorities whose description matches the version is assigned to the version.

If the target release has not been specified then APT simply assigns priority 100 to all installed package versions and priority 500 to all uninstalled package versions, except versions coming from archives which in their Release files are marked as "NotAutomatic: yes" − these versions get the priority 1 or priority 100 if it is additionally marked as "ButAutomaticUpgrades: yes".

APT then applies the following rules, listed in order of precedence, to determine which version of a package to install.

- Never downgrade unless the priority of an available version exceeds 1000. ("Downgrading" is installing a less recent version of a package in place of a more recent version. Note that none of APT's default priorities exceeds 1000; such high priorities can only be set in the preferences file. Note also that downgrading a package can be risky.)
- Install the highest priority version.
- If two or more versions have the same priority, install the most recent one (that is, the one with the higher version number).
- If two or more versions have the same priority and version number but either the packages differ in some of their metadata or the −−reinstall option is given, install the uninstalled one.

In a typical situation, the installed version of a package (priority 100) is not as recent as one of the versions available from the sources listed in the **sources.list**(5) file (priority 500 or 990). Then the package will be upgraded when **apt−get install** *some−package* or **apt−get upgrade** is executed.

More rarely, the installed version of a package is *more* recent than any of the other available versions. The package will not be downgraded when **apt−get install** *some−package* or **apt−get upgrade** is executed.

Sometimes the installed version of a package is more recent than the version belonging to the target release, but not as recent as a version belonging to some other distribution. Such a package will indeed be upgraded when **apt−get install** *some−package* or **apt−get upgrade** is executed, because at least *one* of the available versions has a higher priority than the installed version.

Phased Updates

APT understands a field called Phased−Update−Percentage which can be used to control the rollout of a new version. It is an integer between 0 and 100.

A system's eligibility to a phased update is determined by seeding random number generator with the package source name, the version number, and /etc/machine−id, and then calculating an integer in the range [0, 100]. If this integer is larger than the Phased−Update−Percentage, the version is pinned to 1, and thus held back. Otherwise, normal policy rules apply.

In case you have multiple systems that you want to receive the same set of updates, you can set APT::Machine−ID to a UUID such that they all phase the same, or set APT::Get::Never−Include−Phased−Updates or APT::Get::Always−Include−Phased−Updates to true such that APT will never/always consider phased updates.

The Effect of APT Preferences

The APT preferences file allows the system administrator to control the assignment of priorities. The file consists of one or more multi−line records separated by blank lines. Records can have one of two forms, a specific form and a general form.

- The specific form assigns a priority (a "Pin−Priority") to one or more specified packages with a specified version or version range. For example, the following record assigns a high priority to all versions of the perl package whose version number begins with "5.20". Multiple packages can be separated by spaces.

  Package: perl
  Pin: version 5.20*
  Pin−Priority: 1001

- The general form assigns a priority to all of the package versions in a given distribution (that is, to all the versions of packages that are listed in a certain Release file) or to all of the package versions coming from a particular Internet site, as identified by the site's fully qualified domain name.

  This general−form entry in the APT preferences file applies only to groups of packages. For example, the following record assigns a high priority to all package versions available from the local site.

  Package: *
  Pin: origin ""
  Pin−Priority: 999

  A note of caution: the keyword used here is "origin" which can be used to match a hostname. The following record will assign a high priority to all versions available from the server identified by the hostname "ftp.de.debian.org"

  Package: *
  Pin: origin "ftp.de.debian.org"
  Pin−Priority: 999

  This should *not* be confused with the Origin of a distribution as specified in a Release file. What follows the "Origin:" tag in a Release file is not an Internet address but an author or vendor name, such as "Debian" or "Ximian".

  The following record assigns a low priority to all package versions belonging to any distribution whose Archive name is "unstable".

  Package: *
  Pin: release a=unstable
  Pin−Priority: 50

  The following record assigns a high priority to all package versions belonging to any distribution whose Codename is "bookworm".

  Package: *
  Pin: release n=bookworm
  Pin−Priority: 900

  The following record assigns a high priority to all package versions belonging to any release whose Archive name is "stable" and whose release Version number is "11".

  Package: *
  Pin: release a=stable, v=11
  Pin−Priority: 500

The effect of the comma operator is similar to an "and" in logic: All conditions must be satisfied for the pin to match. There is one exception: For any type of condition (such as two "a" conditions), only the last such condition is checked.

Matching packages in the Package field

The Package field specifies the package that a pinning priority is applied to. The field can either contain a binary package name, a source package name (prefixed with "src:"), a **glob**(7) expression or a regular expression (surrounded by slashes). Multiple package names, **glob**(7) expressions and regular expressions can be listed separated by whitespace in which case the record will match any of the matched packages.

By default, only packages of the native architecture are matched. To match binary packages of any

architecture, add the :any suffix to the package name. You can also limit matching to a specific architecture by appending the architecture name to the package name, separated by a colon character.

For example, the following example uses a glob expression and a regular expression to assign the priority 500 to all packages from experimental where the name starts with gnome (as a **glob**(7)−like expression) or contains the word kde (as a POSIX extended regular expression surrounded by slashes).

```
Package: gnome* /kde/
Pin: release a=experimental
Pin−Priority: 500
```

The rule for those expressions is that they can occur anywhere where a string can occur. Thus, the following pin assigns the priority 990 to all packages from a release starting with hirsute.

```
Package: *
Pin: release n=hirsute*
Pin−Priority: 990
```

If a regular expression occurs in a Package field, the behavior is the same as if this regular expression were replaced with a list of all package names it matches. It is undecided whether this will change in the future; thus you should always list wild−card pins first, so later specific pins override it. The pattern "*" in a Package field is not considered a **glob**(7) expression in itself.

To pin all binaries produced by the apt source package of this APT's version to 990, you can do:

```
Package: src:apt
Pin: version 2.4.8
Pin−Priority: 990
```

Source package pinning can be combined with regular expressions and glob patterns, and can also take a binary architecture.

For example, let's pin all binaries for all architectures produced by any source package containing apt in its name to 990:

```
Package: src:*apt*:any
Pin: version *
Pin−Priority: 990
```

The :any suffix makes sure to select binary packages from any architecture. Without that suffix, apt implicitly assumes the :native suffix which would only select packages from the native architecture.

How APT Interprets Priorities

Priorities (P) assigned in the APT preferences file must be positive or negative integers. They are interpreted as follows (roughly speaking):

P >= 1000
: causes a version to be installed even if this constitutes a downgrade of the package

990 <= P < 1000
: causes a version to be installed even if it does not come from the target release, unless the installed version is more recent

500 <= P < 990
: causes a version to be installed unless there is a version available belonging to the target release or the installed version is more recent

100 <= P < 500
: causes a version to be installed unless there is a version available belonging to some other distribution or the installed version is more recent

0 < P < 100

causes a version to be installed only if there is no installed version of the package

P < 0
: prevents the version from being installed

P = 0
: has undefined behaviour, do not use it.

The first specific−form record matching an available package version determines the priority of the package version. Failing that, the priority of the package is defined as the maximum of all priorities defined by generic−form records matching the version. Records defined using patterns in the Pin field other than "*" are treated like specific−form records.

For example, suppose the APT preferences file contains the three records presented earlier:

```
Package: perl
Pin: version 5.20*
Pin−Priority: 1001

Package: *
Pin: origin ""
Pin−Priority: 999

Package: *
Pin: release unstable
Pin−Priority: 50
```

Then:

- The most recent available version of the perl package will be installed, so long as that version's version number begins with "5.20". If *any* 5.20* version of perl is available and the installed version is 5.24*, then perl will be downgraded.
- A version of any package other than perl that is available from the local system has priority over other versions, even versions belonging to the target release.
- A version of a package whose origin is not the local system but some other site listed in **sources.list**(5) and which belongs to an unstable distribution is only installed if it is selected for installation and no version of the package is already installed.

Determination of Package Version and Distribution Properties

The locations listed in the **sources.list**(5) file should provide Packages and Release files to describe the packages available at that location.

The Packages file is normally found in the directory .../dists/*dist−name*/*component*/*arch*: for example, .../dists/stable/main/binary−i386/Packages. It consists of a series of multi−line records, one for each package available in that directory. Only two lines in each record are relevant for setting APT priorities:

the Package: line
: gives the package name

the Version: line
: gives the version number for the named package

The Release file is normally found in the directory .../dists/*dist−name*: for example, .../dists/stable/Release, or .../dists/bullseye/Release. It consists of a single multi−line record which applies to *all* of the packages in the directory tree below its parent. Unlike the Packages file, nearly all of the lines in a Release file are relevant for setting APT priorities:

the Archive: or Suite: line
: names the archive to which all the packages in the directory tree belong. For example, the line "Archive: stable" or "Suite: stable" specifies that all of the packages in the directory tree below the

parent of the Release file are in a stable archive. Specifying this value in the APT preferences file would require the line:

Pin: release a=stable

the Codename: line
names the codename to which all the packages in the directory tree belong. For example, the line "Codename: bookworm" specifies that all of the packages in the directory tree below the parent of the Release file belong to a version named bookworm. Specifying this value in the APT preferences file would require the line:

Pin: release n=bookworm

the Version: line
names the release version. For example, the packages in the tree might belong to Debian release version 11. Note that there is normally no version number for the testing and unstable distributions because they have not been released yet. Specifying this in the APT preferences file would require one of the following lines.

Pin: release v=11
Pin: release a=stable, v=11
Pin: release 11

the Component: line
names the licensing component associated with the packages in the directory tree of the Release file. For example, the line "Component: main" specifies that all the packages in the directory tree are from the main component, which entails that they are licensed under terms listed in the Debian Free Software Guidelines. Specifying this component in the APT preferences file would require the line:

Pin: release c=main

the Origin: line
names the originator of the packages in the directory tree of the Release file. Most commonly, this is Debian. Specifying this origin in the APT preferences file would require the line:

Pin: release o=Debian

the Label: line
names the label of the packages in the directory tree of the Release file. Most commonly, this is Debian. Specifying this label in the APT preferences file would require the line:

Pin: release l=Debian

All of the Packages and Release files retrieved from locations listed in the **sources.list**(5) file are stored in the directory /var/lib/apt/lists, or in the file named by the variable Dir::State::Lists in the apt.conf file. For example, the file debian.lcs.mit.edu_debian_dists_unstable_contrib_binary−i386_Release contains the Release file retrieved from the site debian.lcs.mit.edu for binary−i386 architecture files from the contrib component of the unstable distribution.

Optional Lines in an APT Preferences Record

Each record in the APT preferences file can optionally begin with one or more lines beginning with the word Explanation:. This provides a place for comments.

EXAMPLES

Tracking Stable

The following APT preferences file will cause APT to assign a priority higher than the default (500) to all package versions belonging to a stable distribution and a prohibitively low priority to package versions belonging to other Debian distributions.

Explanation: Uninstall or do not install any Debian−originated
Explanation: package versions other than those in the stable distro
Package: *
Pin: release a=stable
Pin−Priority: 900

Package: *
Pin: release o=Debian
Pin−Priority: −10

With a suitable **sources.list**(5) file and the above preferences file, any of the following commands will cause APT to upgrade to the latest stable version(s).

apt−get install *package−name*
apt−get upgrade
apt−get dist−upgrade

The following command will cause APT to upgrade the specified package to the latest version from the testing distribution; the package will not be upgraded again unless this command is given again.

apt−get install *package*/testing

Tracking Testing or Unstable

The following APT preferences file will cause APT to assign a high priority to package versions from the testing distribution, a lower priority to package versions from the unstable distribution, and a prohibitively low priority to package versions from other Debian distributions.

Package: *
Pin: release a=testing
Pin−Priority: 900

Package: *
Pin: release a=unstable
Pin−Priority: 800

Package: *
Pin: release o=Debian
Pin−Priority: −10

With a suitable **sources.list**(5) file and the above preferences file, any of the following commands will cause APT to upgrade to the latest testing version(s).

apt−get install *package−name*
apt−get upgrade
apt−get dist−upgrade

The following command will cause APT to upgrade the specified package to the latest version from the unstable distribution. Thereafter, **apt−get upgrade** will upgrade the package to the most recent testing version if that is more recent than the installed version, otherwise, to the most recent unstable version if that is more recent than the installed version.

apt−get install *package*/unstable

Tracking the evolution of a codename release

The following APT preferences file will cause APT to assign a priority higher than the default (500) to all package versions belonging to a specified codename of a distribution and a prohibitively low priority to

package versions belonging to other Debian distributions, codenames and archives. Note that with this APT preference APT will follow the migration of a release from the archive testing to stable and later oldstable. If you want to follow for example the progress in testing notwithstanding the codename changes you should use the example configurations above.

Explanation: Uninstall or do not install any Debian−originated package versions
Explanation: other than those in the distribution codenamed with bookworm or sid
Package: *
Pin: release n=bookworm
Pin−Priority: 900

Explanation: Debian unstable is always codenamed with sid
Package: *
Pin: release n=sid
Pin−Priority: 800

Package: *
Pin: release o=Debian
Pin−Priority: −10

With a suitable **sources.list**(5) file and the above preferences file, any of the following commands will cause APT to upgrade to the latest version(s) in the release codenamed with bookworm.

apt−get install *package−name*
apt−get upgrade
apt−get dist−upgrade

The following command will cause APT to upgrade the specified package to the latest version from the sid distribution. Thereafter, **apt−get upgrade** will upgrade the package to the most recent bookworm version if that is more recent than the installed version, otherwise, to the most recent sid version if that is more recent than the installed version.

apt−get install *package*/sid

FILES

/etc/apt/preferences
: Version preferences file. This is where you would specify "pinning", i.e. a preference to get certain packages from a separate source or from a different version of a distribution. Configuration Item: Dir::Etc::Preferences.

/etc/apt/preferences.d/
: File fragments for the version preferences. Configuration Item: Dir::Etc::PreferencesParts.

SEE ALSO

apt-get(8) **apt-cache**(8) **apt.conf**(5) **sources.list**(5)

BUGS

APT bug page[1]. If you wish to report a bug in APT, please see /usr/share/doc/debian/bug−reporting.txt or the **reportbug**(1) command.

AUTHOR

APT team

NOTES

1. APT bug page
   http://bugs.debian.org/src:apt