United Parcel Service of America, Inc.

Debian's Advanced Package Tool

Packman

APT is a powerful front-end for the Debian GNU/Linux package manager dpkg. This article shows you how to use APT for daily tasks. BY MARTIN LOSCHWITZ

Although the Debian's package management program, *dpkg* is quite powerful, at times it tends to provide its users with too little support – just like *rpm*. For example, the Debian package manager does not allow you to remove dependancies between packages, and you will not find a mechanism for automatically updating previously installed software.

The first attempt to resolve these issues dates back to the Debian Version 0.93 R6 with *dselect* providing an interactive front-end for *dpkg*. *dselect* then later developed into a genuine all round tool that removes package dependencies, updates pre-installed packages (if required) and generally provides a whole bunch of use functions.

But as the functionality and thus the number of packages in Debian increased, *dselect* became increasingly much more complex – with usability issues even for experienced users. Obviously a new alternative was required, a tool that could perform the same tasks as *dselect* but still provide ease of use.

Version 2.1 (alias Slink) of Debian saw the introduction of APT, the Advanced Package Tool. Just like *dselect*, APT is a front-end for *dpkg* and should not be seen as a replacement for the Debian package manager *dpkg*. In contrast to *dselect* you can use the APT syntax to manage *dpkg* from the command line. The tool will provide a home for a collection of useful programs designed to handle multiple tasks: *apt-get* installs and removes the packages, automatically removing any previous dependencies, while *apt-cache* is ideally suited to browsing package lists, and *apt-zip* allows you to keep computers that are not attached to the Internet up to date without too many headaches.

This article first looks into the basic functions of APT, and shows you how to use *apt-get* and *apt-cache* for your daily work. It goes on to introduce three more programs, namely – *auto-apt*, *apt-file*, and *aptitude* – that are not part of the APT package itself, but extremely useful add-ons. Table 1 shows an overview of the most important functions of all the programs belonging to the APT suite.

Package Sources

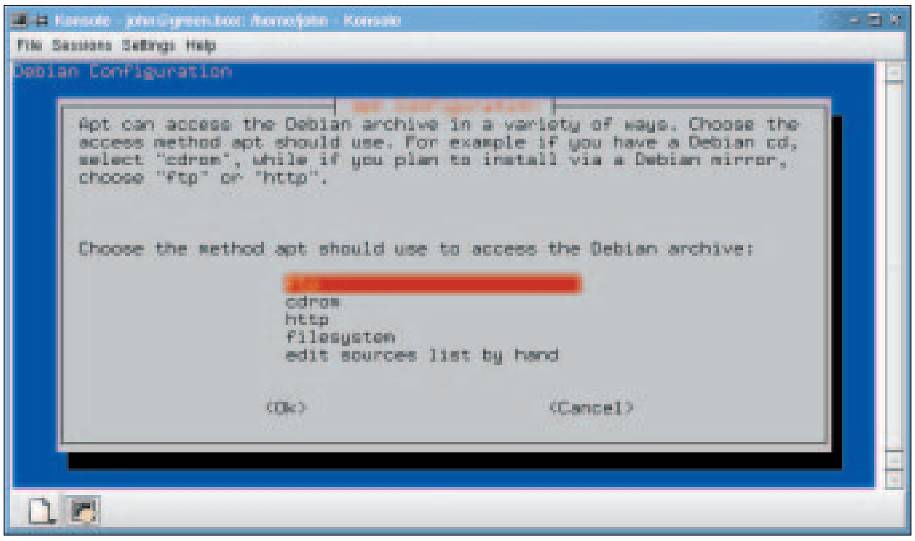
Just like *dselect*, APT refers to an internal database of information on the whole range of packages that are available on the system. With respect to the package database *apt-get* is probably the most important tool in the whole APT group. It ensures that your package lists are kept up to date.

But *apt-get* can do far more than simply maintain the package database – the tool can also offer an intelligent download manager, an all round talent that can download packages from given addresses, ensures that any dependencies are complied with and hands over the packages to the Debian package manager, *dpkg*, where they will be installed.

You will need to perform some simple configuration tasks to maximize the power of *apt-get*. Use the */etc/apt/sources.list* file to type the source path of the so-called index file that *apt-get* will later use to generate the index database. You can alternatively use:

- a normal text editor (*vim*, *emacs*, *nano*),
- the *apt-setup* program and
- the *apt-cdrom* program (if working with CD ROMs).

The *apt-setup* menu is shown in Figure 1. When prompted, you can say *yes* to both of the questions on non-free and contrib software. According to Debian, a software program is non-free if it contravenes the Debian Free Software Guidelines[1]. The contrib is reserved for programs that are free themselves, but depend on software from the non-free category.


Konsole - john@green.box: /home/john - Konsole
File Sessions Settings Help

```
Debian Configuration

                         Apt Configuration
  Apt can access the Debian archive in a variety of ways. Choose the
  access method apt should use. For example if you have a Debian cd,
  select "cdrom", while if you plan to install via a Debian mirror,
  choose "ftp" or "http".


  Choose the method apt should use to access the Debian archive:

                 ftp
                 cdrom
                 http
                 filesystem
                 edit sources list by hand

            <Ok>                      <Cancel>
```

Figure 1: Defining source paths for installation packages. *apt-setup* helps the root user to create the APT configuration file */etc/apt/sources.list*.

If you intend to use CD ROMs as your installation source, you simply have to place the CDs into your CD ROM drive, ensuring that you keep to the correct sequence, type *apt-cdrom add* as *root*, and then let the program take care of all the remaining steps.

Sources List

You may not be able to avoid some of the manual editing of the *sources.list* file. Before you start, you should familiarize yourself with the syntax (see Listing 1 for an example). Each line conforms to the following syntax:

```
Type URI Distribution [Cat1] ↩
[Cat2] ...
```

There are two types available: *deb* (for pre-compiled packages) and *deb-src* (for the package sources).
The Uniform Resource Identifier, URI, contains the home directory of the distribution, and also a reference to the position of the required files, the options being: a local file system (*file*), a CD (*cdrom*) or the Internet via *http* or *ftp*. The Distribution column is intended for the path to the source relative to the base directory you supplied. Normally this will be a directory called *stable* alias *potato* (containing the stable Debian version prior to release), *testing* alias *woody* (the current test candidate for the next stable release – refer to the box "Stable, Testing and Unstable") or *unstable* alias *sid* (the developer version).

Users who prefer a secure system are recommended to use the list entry for *security.debian.org*. This ensures that security patches are automatically applied when you update a package. The system administrator can then define either a single or multiple categories of distribution packages that can be selected on installation (as an example *main* for the packages belonging to the basic distribution or the categories or *contrib* and *non-free* referred to earlier).

After modifying */etc/apt/sources.list* to reflect your requirements, you can use the *apt-get update* command to update the package database. If the file contains *ftp* or *http* URIs, the local package database will be synchronized with the package lists on the official Debian servers. If you are performing a CD ROM only installation, then you will not need this update.

If you need to use a proxy server to update your package database via the Internet, the *http_proxy* and *ftp_proxy* environment variables will make your life easier for you. The download protocol

Listing 1: *sources.list*-Example

```
deb ftp://ftp.uk.debian.org/debian stable main non-free contrib
deb-src ftp://ftp.uk.debian.org/debian stable main non-free contrib
deb ftp://ftp.ticklers.org/debian-non-US stable/non-US main contrib ↩
non-free
deb-src ftp://ftp.ticklers.org/debian-non-US stable/non-US main ↩
contrib non-free
deb cdrom:[Debian GNU/Linux 2.2 r0_Potato_-Official i386 Binary-1 ↩
(20000814)]/ unstable main
deb http://security.debian.org stable/updates main contrib non-free
```

in *sources.list* defines which of these options you will need to set. If your proxy server's address is *192.168.0.20* and the port number is *8080*, then type in the following to set the *http_proxy* variable in *bash* or *zsh*:

```
export http_proxy=➘
"http://192.168.0.20:8080"
```

If you are using an FTP proxy, then you simply assign these values to the *ftp_proxy* variable instead of *http_proxy*.

Of course, *http_proxy* and *ftp_proxy* are not only available in *apt-get* for the *upgrade* function, they can be used for any other task that requires *apt-get* to access the Internet.

As previously mentioned, *apt-get* is an excellent choice for the installation and removing of the software. To install a package the *root* user simply types the *apt-get install packagename* command. *apt-get* will then check to see if the required package is in the package database, and if the dependencies with respect to pre-installed packages can be respected. If required, missing packages will be downloaded from the medium specified in */etc/apt/sources.lists* and passed to the package manager (*dpkg*), for installation. Listing 2 shows us such an example.

If the recently installed package (*libdbmusic0-dev* in our example) does not fulfill all of your expectations, the *root* user can use the command *apt-get* with the *remove* keyword to remove the rogue package and any that depend on it. If you then use *apt-get remove packagename* to deinstall a package, you may find that a few files belonging to the package cannot be deleted.

Files in the */etc* directory are tagged as *config-files*, for example, and cannot be removed by a simple deletion process. This avoids destroying any of the settings you may have made. To make sure that a package has been completely removed from your computer system, you will need to use *apt-get* with the *--purge* option.

The complete syntax, which is used for deinstalling the *libdbmusic0-dev* package is thus *apt-get --purge remove libdb➘music0-dev*.

Listing 2: Installing *libdbmusico-dev* with *apt-get*

```
<0>minerva[1003]:~# apt-get install libdbmusic0-dev
Reading Package Lists... Done
Building Dependency Tree... Done
The following extra packages will be installed:  libdbmusic0
The following NEW packages will be installed:
  libdbmusic0 libdbmusic0-dev
0 packages upgraded, 2 newly installed, 0 to remove and 0 not
upgraded.
Need to get 273kB of archives. After unpacking 1110kB will be used.
Do you want to continue? [Y/n] y
Get:1 ftp://ftp.ticklers.org unstable/main libdbmusic0 0.2.0-2
[260kB]
Get:2 ftp://ftp.ticklers.org unstable/main libdbmusic0-dev 0.2.0-2
[13.1kB]
Fetched 273kB in 4s (59.9kB/s)
Selecting previously deselected package libdbmusic0.
(Reading database ... 68160 files and directories currently
installed.)
Unpacking libdbmusic0 (from .../libdbmusic0_0.2.0-2_i386.deb) ...
Selecting previously deselected package libdbmusic0-dev.
Unpacking libdbmusic0-dev (from .../libdbmusic0-dev_0.2.0-2_i386.deb)
...
Setting up libdbmusic0 (0.2.0-2) ...

Setting up libdbmusic0-dev (0.2.0-2) ...

<0>minerva[1004]:~#
```

Changing Versions Without Headaches

One *apt-get* feature that you may well be interested in is the ability to upgrade the current distribution on your own machine using the *apt-get upgrade* option. A variant of this functionality (*apt-get dist-upgrade*) will be particularly convenient when the next Debian Release Version 3.1 (alias Sarge) becomes available, allowing you to avoid the trouble, time and frustration often involved with installing new versions.

To update your musty Debian Potato installation to Woody, you must first edit */etc/apt/sources.list*. You need to replace each *stable* or *potato* with *woody*, for example, the previous entry

```
deb ftp://ftp.uk.debian.org/➘
debian stable U main non-free ➘
contrib
```

would need to be changed to

```
deb ftp://ftp.uk.debian.org/➘
debian woody U main non-free ➘
contrib
```

for the update. If you prefer to use the current packages and are prepared to take the risk of possible instability, you can even type *sid* to replace *woody*.

The next step is to synchonize the local package database with the package lists on the official servers using *apt-get update*. Of course, all of these steps will work just as well if you are using CD ROMs. In this case you can simply use the *apt-cdrom* command to insert the package list contents on the CD ROMs into the local package database and then launch *apt-get update*.

After updating the package database, type the *apt-get dist-upgrade* command. *apt-get* will then compare the versions of the pre-installed packages with the new versions in the package database. If the program discovers that the new package database lists a more recent version than the one actually installed on your machine, it will automatically download and install the package.

You may notice that the command *apt-get dist-upgrade* has downloaded some packages, but *dpkg* cannot install them correctly, since some packages that they depend on are missing. In this case, repeatedly launch *apt-get dist-upgrade* until the installation has been completed for all the packages on your list.

But make sure you have enough free space in the */var* partition on your hard disk: *apt-get* will download packages to */var/cache/apt/archives*, but not delete them after completing its task. To free up the disk space used by packages you no longer need, the *root* user can type the *apt-get clean* command.

For users that prefer to keep a stable system instead of moving to Debian Unstable, but still require a few packages from the unstable version (because the Woody version is too old, for example), the post Debian Woody *apt-get* version offers a special function: Just enter the following line in */etc/apt/apt.conf*

```
APT::Default-Release "testing";
```

and add suitable *deb* and *deb-src* entries for *unstable* in */etc/apt/sources.list*, and you will be able to install Sid packages in Woody by typing *apt-get -t unstable install packagename*.

You can use the inverse functionality: The entry

```
APT::Default-Release "unstable";
```

in */etc/apt/apt.conf* and the right entries in */etc/apt/sources.list* will allow you to install Woody packages in Sid by typing *apt-get -t testing install packagename*.

APT-Cache

apt-cache is the second most important program after *apt-get* in the APT program group. It provides the user with direct access to the package database and is thus extremely useful.

You can type *apt-cache search identd* to browse the package database for any packages containing the *identd* keyword as part of their name or the package description, for example. If the results contain a package that seems interesting you can use *apt-cache show packagename* to display the package details. If you need information on a package's dependencies, or require a list of packages that depend on the current package, you can type *apt-cache showpkg packagename*.

Where are you?

For practical purposes you will normally want to use *apt-cache* and the *search* keyword to browse the package list. This is hardly surprising as Debian GNU/ Linux Woody comprises almost 9,300 packages, and that makes it impossible for you to remember every individual package.

But searching with *apt-cache* has its limitations. If a program's *./configure* script complains about not being able to find *glut.h*, *apt-cache search glut.h* will not provide you with any results and you will then be no closer to home than you were previously.

Two additional programs are available in this scenario: *auto-apt* and *apt-file*. Neither program belongs to the official APT package and you will need to use *apt-get install auto-apt* or *apt-get install apt-file* to install them separately.

Just like APT, both *auto-apt* and *apt-file* use internal databases that are generated after installing the package. To do so, both programs use individual *Contents-** files that depend on the architecture of your machine and are available in the *debian/dists/Distribution*directory on any Debian server.

To generate the internal package database, you will need to launch both *apt-file* and *auto-apt* with the *update* argument. You are also recommended to update the internal database of both of the programs at regular intervals to achieve the best possible results for search operations.

The search operation is launched in a similar way to *apt-cache*, i.e. it uses the *search* keyword. To find the *glut.h* file

TABLE 1: COMMAND LINE REFERENCE

Command	Function
apt-get update	Updates the local package database
apt-get upgrade	Updates all the packages on a system if the current release includes an update
apt-get dist-upgrade	Similar to *apt-get upgrade* but will install or remove packages to satisfy dependancies
apt-get install *Packagename*	Installs a package
apt-get remove *Packagename*	Deinstalls a package
apt-get --purge remove *Packagename*	Deletes a package completely
apt-cache search *Packagename*	Searches the package database for a package
apt-cache show *Packagename*	Displays information for a selected package
apt-cache showpkg *Packagename*	Displays details on the dependencies for a selected package
auto-apt search *file*	Searches for a file in the package database
auto-file search *file*	Searches for a file in the package database and returns more detailed information

Stable, Testing and Unstable

Debian GNU/Linux comes in three different flavors: "stable", "testing" and "unstable", aka "potato", "woody" and "sid", which are the names of the current distributions.

"stable" refers to the current stable release of the Debian Linux distribution. Currently, this is Debian GNU/Linux 2.2, codename "Potato". In Debian GNU/Linux's case stable means that no new packages will be incorporated into the distribution with the possible exception of security updates. Debian GNU/Linux 2.2 is already fairly old, and thus only recommended for system administrators who need an absolutely reliable system. However, this will mean doing without some current software such as XFree 4.* or an SSH daemon with built in SSHv2 support.

"Unstable" is part of the Debian GNU/Linux project that will appeal to experienced users and in fact anyone who would like to live "on the bleeding edge". The codename for the current "unstable" version is "Sid". Any new packages will always be placed in "unstable" first and then make the transition to "testing" distribution 10 days later, unless a critical error ("Release Critical Bug") becomes apparent. Debian "unstable" offers both advantages and disadvantages: On the one hand you will always have access to brand new packages, on the other hand the packages in Sid have not been tested, and this can mean a program behaving badly or not running at all. You may not want to run Sid on a production system for this reason.

"testing" is that part of the Debian GNU/Linux project that is due to go "stable" in the near future. Its current codename is "Woody", and when "Woody" is released as the stable version its version number will be 3.0. It is also quite possible that this release might occur on or around the publication date of the current issue. As previously mentioned, "testing" mainly comprises packages from "unstable" that have proved themselves well behaved for a while. Most "Woody" programs work quite well by now so you should not experience any issues if you are running them on your workstation or multimedia machine.

with *auto-apt*, you will thus need the *auto-apt search glut.h* command; *apt-file search glut.h* performs the same task in *apt-file*. The output formats of these two programs are completely different: *apt-file* simply returns the names of any packages found to contain a certain file:

```
<0>minerva[1004]:~#
apt-file search glut.h
libfltk1-dev
glutg3-dev
cint
```

In contrast *auto-apt* returns the complete path to the files and shows the package category, although this does not always provide for easy readability (Listing 4).

```
<0>minerva[1001]:~#
auto-apt search glut.h
usr/include/GL/fglut.h
devel/glutg3-dev
usr/include/GL/glut.h
devel/glutg3-dev
```

User Friendly Dselect

Many users like the way *dselect* looks but are at odds with its complexity. Others prefer the simplicity of *apt-get* and *apt-cache*, but would like the functionality of a program like *dselect* that allows them to interactively select and install packages.

aptitude is an approach to resolving this issue and provides a *dselect* style front-end for *apt-get* and *apt-cache* (see Figure 2). But in contrast to the former, the tool is easy to use and supports the arrow and function keys.

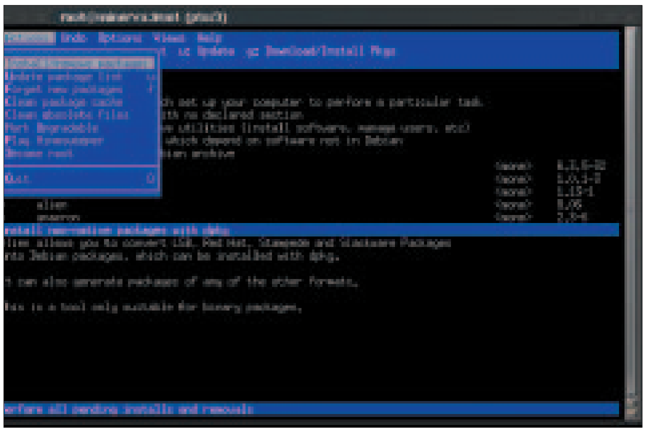
If you run *aptitude* in an X11 terminal emulation, you can simply use your mouse to click on the buttons. Additionally, the intuitive handling and the menu layout enhance *aptitude*'s user-friendliness.

The program is not part of the Debian GNU/Linux standard offering, meaning that you will need to run *apt-get install aptitude* to install it. After the initial program launch, you might want to press [F10] and then press [U] to ensure that *apt-get* automatically updates its own package database.

After completing these steps you can use the cursor keys, or the mouse on X11 systems, to toggle between the five menu items *Installed Packages*, *Not Installed Packages*, *Obsolete and Locally Created Packages*, *Virtual Packages* and *Tasks*.

Obsolete?

Installed Packages will show you the packages currently installed on your machine. *Not Installed Packages* shows all the packages available in the database but not installed on your system. If you then select *Obsolete and Locally Created Packages*, *aptitude* lists packages that are installed on your machine but do not appear in the *apt* database. These may include packages that were formerly part of a Debian installation but have since been removed, because either the author or the package maintainer decided to ditch the package.



```
Actions  Undo  Options  Views  Help
                     t  u: Update  g: Download/Install Pkgs
Update package list
Forget new packages
Clean package cache     ch set up your computer to perform a particular task.
Clean obsolete files    ith no declared section
Mark Upgradable         e utilities (install software, manage users, etc)
Play Minesweeper        which depend on software not in Debian
Become root             ian archive
Quit
     alien
nstall non-native packages with dpkg
lien allows you to convert LSB, Red Hat, Stampede and Slackware Packages
nto Debian packages, which can be installed with dpkg.

t can also generate packages of any of the other formats.

his is a tool only suitable for binary packages.

erform all pending installs and removals
```

Figure 2: Aptitude – the better alternative to Dselect

Virtual Packages are a speciality of Debian : You can use them to group the various programs that fulfill the same tasks under a generic heading. i.e. the mail reader mutt needs a local mail server in order to transfer mail. The maintainer would normally have to decide which mail server software *mutt* should depend on, and this would annoy any users who prefer exim to sendmail or vice versa. So, instead of making exim mandatory, the maintainer can use a trick and specify a dependency between *mutt* and the virtual *mail-transport-agent* package. The database knows that a mail server must be installed to fulfill this condition.

The *Tasks* packages are of little interest nowadays. They were used to group Debian GNU/Linux 2.2 programs that performed certain tasks. i.e. there was a package called *task-x-window-system* that contained XFree86. This type of package more or less died out when Debian GNU/Linux Woody was released.

If you intend to use *aptitude* to install a package, you may first want to find the package name in the list of *Not Installed Packages*. Place the selection bar on the package name to highlight the package, then press the [+] key. The package name should now be selected (shaded).

You can use this to select the multiple packages for installation. Then press the [G] key (that is, get) to display on overview of the packages you will be installing in the next step. Press the key again to download the selected packages. If any dependencies are discovered where the corresponding packages have not been pre-installed, *aptitude* will find the packages and pass them to *apt-get*.

aptitude is also well suited to deleting packages. Locate the entry for those packages you want to delete in the list of *Installed Packages*, then press the [-] key once and the [G] key twice. If any of the installed packages depend on the ones you delete, *aptitude* will automatically remove them.

But *aptitude* offers more than just installing and removing packages. Just press the [H] key for a complete overview and howtos for the available commands. The documentation is available under */usr/share/doc/aptitude*.

Future Trends

Although the APT programs perform well at present, the developers responsible for *dpkg* and APT have big plans for the future. Interaction between the two is due for enhancement via a tailor-made communication layer. A protocol layer of this type would also mean performance gains.

Let us not forget that work is in progress to enhance its user friendliness. It remains to be seen, if that will actually mean a GUI. The prospect is exciting. ■

INFO

[1] Debian Free Software Guidelines: *http://www.debian.org/social_contract*

[2] APT Howto: *http://www.debian.org/doc/manuals/apt-howto/index.en.html*