

APT HOWTO

1.8.2 - Ноябрь 2002

Gustavo Noronha Silva kov@debian.org

Аннотация

Этот документ должен помочь пользователю разобраться с принципами работы с утилитой управления пакетами Debian, APT. Его цель - облегчить жизнь новым пользователям Debian и помочь тем, кто хочет получить более глубокие знания по администрированию системы. Документ был создан для проекта Debian для того, чтобы улучшить поддержку пользователей этого дистрибутива.

Заметка об авторских правах

Copyright © 2001, 2002 Gustavo Noronha Silva

Это руководство лицензировано по положениям GNU FDL (Free Documentation License). Оно было написано в надежде, что будет полезным обществу, но при этом не дается никаких гарантий; вы пользуетесь им на свой риск.

Глава 1 - Введение

В начале был `.tar.gz`. Пользователи должны были сначала скомпилировать программы, которые они хотели использовать на своих системах GNU/Linux. Когда создавался Debian, возникла необходимость в системе управления пакетами, установленными на машине. Этой системе было дано имя `dpkg`. Этот известный пакет первым появился на GNU/Linux, прежде чем Red Hat решил создать собственную систему `rpm`.

Затем перед создателями GNU/Linux быстро возникла новая дилемма. Им понадобился способ для быстрого, практического и эффективного пути установки пакетов, который должен автоматически обслуживать зависимости и сохранять настроечные файлы при обновлении. Тут снова, Debian вышел вперед и предложил миру APT, Advanced Packaging Tool, который затем был портирован Conectiva`ой для использования с `rpm` и был адаптирован некоторыми другими дистрибутивами.

Это руководство не рассматривает `apt-rpm`, как называется порт APT от Conectiva, но "заплаты" к документу по этой теме приветствуются.

Это руководство основано на следующем выпуске Debian, Sarge.

Глава 2 - Основная настройка

2.1 Файл `/etc/apt/sources.list`

Как часть своей работы, APT использует файл, который содержит список 'источников', из которых могут быть скачаны пакеты. Это файл `/etc/apt/sources.list`.

Обычно этот файл имеет следующий формат:

```
deb http://site.http.org/debian distribution раздел1 раздел2 раздел3
deb-src http://site.http.org/debian distribution раздел1 раздел2 раздел3
```

Конечно, вышеприведенные записи являются просто примером и не должны использоваться. Первое слово в каждой строке, либо `deb`, либо `deb-src`, указывает тип архива: либо это двоичные (binary) пакеты (`deb`), которые являются пред-компилированными пакетами, которые готовы к использованию, либо пакеты с исходными текстами (`deb-src`), которые являются первоначальными исходными текстами программ с управляющим файлом Debian (`.dsc`) и файлом `diff.gz`, содержащим изменения, необходимые для `дебианизации' программы.

Обычно в `sources.list` помещается следующее:

```
# See sources.list(5) for more information, especialy
# Remember that you can only use http, ftp or file URIs
# CDROMs are managed through the apt-cdrom tool.
deb http://http.us.debian.org/debian stable main contrib non-free
deb http://non-us.debian.org/debian-non-US stable/non-US main contrib non-free
deb http://security.debian.org stable/updates main contrib non-free
```

```
# Uncomment if you want the apt-get source function to work
#deb-src http://http.us.debian.org/debian stable main contrib non-free
#deb-src http://non-us.debian.org/debian-non-US stable/non-US main contrib non-free
```

Эти строки необходимы для базовой установки Debian. Первая строка `deb` указывает на официальный архив, вторая - на не-США архив и третья - на архив обновлений безопасности Debian.

Две последние строки закомментированы (начинаются с `#`), так что `apt-get` будет их игнорировать. Эти строки `deb-src` указывают на пакеты исходных текстов Debian, если вы часто скачиваете исходные тексты программ для тестирования и перекомпиляции, раскомментируйте их.

Файл `/etc/apt/sources.list` может содержать несколько типов строк. АРТ знает как обращаться с архивами типов `http`, `ftp`, `file` (локальные файлы, напр., каталог, содержащий смонтированную файловую систему ISO9660) и `ssh`, насколько я знаю.

2.2 Как использовать АРТ локально

Иногда, вы можете располагать кучей пакетов `.deb`, которые вам хотелось бы устанавливать с помощью АРТ, чтобы зависимости обрабатывались автоматически.

Чтобы это сделать, создайте каталог и поместите в него `.deb`'ы, которые вам нужны. Например:

```
mkdir /root/debs
```

Вы можете изменить установки определений файла `control` из `debian`-пакета напрямую для вашего репозитория с помощью файла `override`. В этом файле вы можете определить какие-нибудь опции для перекрытия тех, которые приходят с пакетом. Это может выглядеть так:

```
package priority section
```

`package` - это имя пакета, `priority` - имеет значения `low`, `medium` или `high`, а `section` - это название раздела, в котором он находится. Имя файла не имеет значения, позже вы будете указывать его в качестве аргумента для команды `dpkg-scanpackages`. Если вы не хотите составлять файл `override`, тогда просто указывайте `/dev/null` при вызове `dpkg-scanpackages`.

Из каталога `/root` дайте команду:

```
dpkg-scanpackages debs file | gzip > debs/Packages.gz
```

In the above line, *file* is the override file, the command generates a file `Packages.gz` that contains various informations about the packages, which are used by АРТ. To use the packages, finally, add: В вышеприведенной команде, *file* - это файл `override`, команда генерирует файл `Packages.gz`, который содержит различную информацию о пакетах, которые используются АРТ. Чтобы использовать пакеты, наконец, добавьте:

```
deb file:/root debs/
```

После всех этих манипуляций вы можете использовать команды АРТ как обычно. Вы также можете создать и репозиторий для исходных текстов. Для этого используется та же процедура, но помните, что вам нужны файлы `.orig.tar.gz`, `.dsc` и `.diff.gz` в каталоге, и вместо `Packages.gz` надо использовать `Sources.gz`. Также нужно использовать другую программу. Это программа `dpkg-scansources`. Командная строка выглядит примерно так:

```
dpkg-scansources debs | gzip > debs/Sources.gz
```

Обратите внимание на то, что программе `dpkg-scansources` не нужен `override` файл. Строка в `sources.list`:

```
deb-src file:/root debs/
```

2.3 Определение наилучшего зеркала для включения в файл `source.list`: `netselect`, `netselect-apt`

Наиболее часто возникаемый вопрос, в основном у новичков: "какое зеркало Debian включить в `sources.list`?". Для выбора зеркала есть множество способов. Эксперты вероятно пользуются сценариями, которые измеряют продолжительность `ping`'ов до некоторых заркал. Но такая программа для вас уже имеется: **netselect**.

Чтобы установить `netselect`, сделайте как обычно:

```
apt-get install netselect
```

При запуске программы без параметров отображается справка. При запуске с указанными в качестве параметров, разделенных пробелами, именами хостов (зеркал), она выдаст оценку и один из хостов. Эта оценка учитывает ожидаемую продолжительность пинга и количество переходов (хостов, которые должны быть пройдены до того, как пакеты дойдут до цели) и обратна пропорциональна ожидаемой скорости скачивания (т.е., чем она меньше, тем лучше). Программа указывает имя хоста, который имеет наименьшую оценку (полный список оценок можно просмотреть с помощью опции `-vv`). См. пример:

```
bash$ netselect ftp.debian.org http.us.debian.org ftp.at.debian.org download.unesp.br ftp.debian.org.br
365 ftp.debian.org.br
bash$
```

Это означает, что из указанных в качестве параметров зеркал, наилучшим является `ftp.debian.org.br`, с оценкой 365. (Внимание!! Приведенные значения действительны только для моей машины, и для других машин все может быть совсем по-другому).

Сейчас, просто поместите найденное netselect зеркало в файл /etc/apt/sources.list (см. [Файл /etc/apt/sources.list, раздел 2.1](#)) и следуйте советам в [Управление пакетами, Глава 3](#).

Примечание: список зеркал можно всегда найти в файле http://www.debian.org/mirror/mirrors_full.

Начиная с версии 0.3, пакет netselect включает сценарий **netselect-apt**, который делает вышеописанные процедуры автоматически. Просто введите в качестве параметра имя дерева дистрибутива (по умолчанию это stable) и файл sources.list будет настроен на наилучшие зеркала main и non-US и будет сохранен в текущем каталоге. Следующий пример генерирует sources.list стабильного дистрибутива:

```
bash$ ls sources.list
ls: sources.list: File or directory not found
bash$ netselect-apt stable
(...)
bash$ ls -l sources.list
sources.list
bash$
```

Помните: Файл sources.list генерируется в текущем каталоге, и должен быть перемещен в каталог /etc/apt. Далее следуйте советам в [Управление пакетами, Глава 3](#).

2.4 Добавление CD-ROM в файл sources.list

Если вы предпочитает использовать для автоматической установки или обновления пакетов с помощью ваш CD-ROM, то можете поместить его в файл sources.list. Для этого используется программа apt-cdrom:

```
apt-cdrom add
```

Debian CD-ROM должен находиться в приводе. Программа смонтирует CD-ROM и, если диск является Debian CD, то найдет на нем информацию о пакетах. Если у вас CD-ROM необычной конфигурации, то вы можете использовать следующие опции:

```
-h      - справка к программе
-d directory - точка монтирования CD-ROM
-r      - Переименовать распознанный CD-ROM
-m      - Не монтировать
-f      - Ускоренный режим, не проверять пакеты файлов
-a      - Thorough scan mode
```

Например:

```
apt-cdrom -d /home/kov/mycdrom add
```

Также вы можете идентифицировать CD-ROM без добавления его в список:

```
apt-cdrom ident
```

Обратите внимание, что эта программа работает только, если ваш CD-ROM правильно настроен в системном файле настроек /etc/fstab.

Глава 3 - Управление пакетами

3.1 Обновление списка доступных пакетов

Система пакетов использует собственную базу данных для слежения за установленными, не установленными и доступными для установки пакетами. Программа apt-get использует эту базу данных для определения пути установки пакетов, требуемых пользователем, и для определения того, какие дополнительные пакеты нужны, чтобы выбранные пакеты работали корректно.

Чтобы обновить этот список, вы должны использовать команду apt-get update. Эта команда просматривает списки пакетов в архивах, указанных в файле /etc/apt/sources.list; см. более полные сведения об этом файле в [Файл /etc/apt/sources.list, раздел 2.1](#).

Было бы неплохо запускать эту команду регулярно, чтобы ваша система всегда была в курсе об обновлениях доступных пакетов, особенно обновлениях безопасности.

3.2 Установка пакетов

Наконец-то, ожидание окончено! Ваш sources.list готов и списки доступных пакетов обновлены, теперь вам только остается дать команду apt-get, чтобы установить предпочитаемый пакет. Например, вы можете запустить:

```
apt-get install xchat
```

APT будет искать в своей базе данных наиболее свежие версии пакетов и будет скачивать их из соответствующих архивов так, как они указаны в sources.list. В случаях, когда пакет зависит от другого -- как в варианте ниже -- APT проверит зависимости и установит необходимые пакеты. См. пример:

```
[root]@[/] # apt-get install nautilus
Reading Package Lists... Done
Building Dependency Tree... Done
The following extra packages will be installed:
  bonobo libmedusa0 libnautilus0
The following NEW packages will be installed:
  bonobo libmedusa0 libnautilus0 nautilus
0 packages upgraded, 4 newly installed, 0 to remove and 1 not upgraded.
Need to get 8329kB of archives. After unpacking 17.2MB will be used.
Do you want to continue? [Y/n]
```

Пакет nautilus зависит от совместно-используемых библиотек, поэтому АРТ скачает их из архива. Если бы вы указали имена этих библиотек в командной строке apt-get, то АРТ не стал бы спрашивать у вас разрешения продолжать установку; он бы автоматически решил, что вы хотите установить все эти пакеты.

Это означает, что АРТ запрашивает подтверждения только тогда, когда нужно установить пакеты, которые не были перечислены в командной строке.

Вам могут быть полезны следующие опции apt-get:

```
-h справка
-d только загрузить - не устанавливать и не распаковывать архивы
-s ничего не делать реально, имитировать выполнение
-y предполагается ответ Yes на все вопросы, сами вопросы не выводить
-f продолжать, даже если проверка целостности не удачна
-u плюс ко всему показывать список обновленных пакетов
```

В одной строке можно задавать несколько пакетов. Скачанные из сети файлы помещаются в каталог для дальнейшей установки /var/cache/apt/archives.

Также, вы можете указывать пакеты для удаления в той же командной строке. Просто поместите '-' сразу после имени удаляемого пакета, например:

```
[root]@[/] # apt-get install nautilus gnome-panel-
Reading Package Lists... Done
Building Dependency Tree... Done
The following extra packages will be installed:
  bonobo libmedusa0 libnautilus0
The following packages will be REMOVED:
  gnome-applets gnome-panel gnome-panel-data gnome-session
The following NEW packages will be installed:
  bonobo libmedusa0 libnautilus0 nautilus
0 packages upgraded, 4 newly installed, 4 to remove and 1 not upgraded.
Need to get 8329kB of archives. After unpacking 2594kB will be used.
Do you want to continue? [Y/n]
```

Более подробную информацию об удалении пакетов можно найти в разделе [Удаление пакетов, раздел 3.3](#).

Если вы обнаружили, что пакет испортился, или просто хотите переустановить файлы пакета более новой доступной версии, то вы можете использовать опцию --reinstall, например:

```
[root]@[/] # apt-get --reinstall install gdm
Reading Package Lists... Done
Building Dependency Tree... Done
0 packages upgraded, 0 newly installed, 1 reinstalled, 0 to remove and 1 not upgraded.
Need to get 0B/182kB of archives. After unpacking 0B will be used.
Do you want to continue? [Y/n]
```

При создании этого руководства АРТ имел версию 0.5.3, которая была текущей версией в Debian `unstable' (sid) на время написания. Если вы установили эту версию, то вам доступны некоторые дополнительные возможности: вы можете использовать команды типа apt-get install пакет/дистрибутив для установки пакетов из указанных дистрибутивов, или apt-get install пакет=версия. Например:

```
apt-get install nautilus/unstable
```

установит nautilus из дистрибутива `unstable' (нестабильный), даже если вы работаете на `stable' (стабильном). Для `distribution' допускаются значения stable, testing и unstable.

Вы можете предпочесть использование ключа -t для выбора целевого дистрибутива, указывающего apt-get предпочесть указанный дистрибутив при обработке зависимостей.

ВАЖНО: `unstable' версия Debian - это самая первая версия, в которой появляются новейшие версии пакетов Debian. Этот дистрибутив подвергается воздействию изменений, сделанных в пакетах, и маленьким, и большим, каждое из которых может воздействовать на множество пакетов или систему в целом. По этой причине, эта версия дистрибутива *не* должна использоваться неопытными пользователями или теми, кто ожидает стабильности.

Дистрибутив `testing' (тестируемый) несколько лучше `unstable' в отношении стабильности, но на производстве лучше все-таки использовать стабильный дистрибутив.

3.3 Удаление пакетов

Если пакет вам больше не нужен, то вы можете удалить его из вашей системы, используя АРТ. Чтобы это сделать просто введите: apt-get remove package. Например:

```
[root]@[/] # apt-get remove gnome-panel
```

```

Reading Package Lists... Done
Building Dependency Tree... Done
The following packages will be REMOVED:
  gnome-applets gnome-panel gnome-panel-data gnome-session
0 packages upgraded, 0 newly installed, 4 to remove and 1 not upgraded.
Need to get 0B of archives. After unpacking 14.6MB will be freed.
Do you want to continue? [Y/n]

```

Из вышеприведенного примера можно видеть, что АРТ также предлагает удалить пакеты, от которых зависит пакет, предлагаемый вами к удалению. Способа удалить пакет при помощи АРТ без удаления пакетов, от которых он зависит, нет.

Вышеприведенная команда `apt-get` удалит пакеты, но оставит их настроечные файлы, если они есть. Для полного удаления пакета запустите:

```

[root]@[/] # apt-get --purge remove gnome-panel
Reading Package Lists... Done
Building Dependency Tree... Done
The following packages will be REMOVED:
  gnome-applets* gnome-panel* gnome-panel-data* gnome-session*
0 packages upgraded, 0 newly installed, 4 to remove and 1 not upgraded.
Need to get 0B of archives. After unpacking 14.6MB will be freed.
Do you want to continue? [Y/n]

```

Обратите внимание на '*' после имен. Это указывает на то, что будут удалены настроечные файлы для каждого из этих пакетов.

Аналогично действию `install`, вы можете инвертировать действие `remove` для определенных пакетов. В случае удаления, если вы добавите знак '+' справа от имени пакета, то пакет будет установлен, а не удален.

```

[root]@[/] # apt-get --purge remove gnome-panel nautilus+
Reading Package Lists... Done
Building Dependency Tree... Done
The following extra packages will be installed:
  bonobo libmedusa0 libnautilus0 nautilus
The following packages will be REMOVED:
  gnome-applets* gnome-panel* gnome-panel-data* gnome-session*
The following NEW packages will be installed:
  bonobo libmedusa0 libnautilus0 nautilus
0 packages upgraded, 4 newly installed, 4 to remove and 1 not upgraded.
Need to get 8329kB of archives. After unpacking 2594kB will be used.
Do you want to continue? [Y/n]

```

Обратите внимание на то, что `apt-get` распечатывает дополнительные пакеты, которые будут установлены (т.е., пакеты, которые будут установлены для правильной работы запрашиваемых к установке пакетов), удаляемые пакеты и затем пакеты, которые будут установлены (также включая дополнительные пакеты).

3.4 Обновление пакетов

Обновления пакетов - это конек системы АРТ. Это может быть сделано одной командой: `apt-get upgrade`. Вы можете использовать эту команду для обновления пакетов в том же самом дистрибутиве, равно как и при обновлении существующего дистрибутива до нового, хотя для последнего действия предназначена команда `apt-get dist-upgrade`; см. подробности в разделе [Обновление нового выпуска, раздел 3.5](#).

Эту команду полезно запускать с опцией `-u`. Эта опция заставляет АРТ показывать полный список пакетов, предназначенных для обновления. Без нее вы будете обновляться вслепую. АРТ последние версии каждого из пакетов и установит их в правильном порядке. Поэтому очень важно запустить перед этим `apt-get update`.

См. раздел [Обновление списка доступных пакетов, раздел 3.1](#). Вот пример:

```

[root]@[/] # apt-get -u upgrade
Reading Package Lists... Done
Building Dependency Tree... Done
The following packages have been kept back
  cpp gcc lilo
The following packages will be upgraded
  adduser ae apt autoconf debhelper dpkg-dev esound esound-common ftp indent
  ipchains isapnptools libaudiofile-dev libaudiofile0 libesd0 libesd0-dev
  libgtk1.2 libgtk1.2-dev liblockfile1 libnewt0 liborbit-dev liborbit0
  libstdc++2.10-glibc2.2 libtiff3g libtiff3g-dev modconf orbit procs psmisc
29 packages upgraded, 0 newly installed, 0 to remove and 3 not upgraded.
Need to get 5055B/5055kB of archives. After unpacking 1161kB will be used.
Do you want to continue? [Y/n]

```

Процесс очень прост. Обратите внимание, что в первых строках `apt-get` говорит, что некоторые пакеты были `kept back`. Это означает, что новые версии некоторых пакетов не будут установлены по некоторым причинам. Возможными причинами могут быть битые зависимости (пакет, от которого он зависит не доступен для скачивания) или новые зависимости (пакет последней версии зависит от новых пакетов).

В первом случае прозрачного решения не существует. Для второго случая, должна помочь команда `apt-get install` для указанного в вопросе пакета, так как это приведет к скачиванию зависимости. Даже более

прозрачное решение сосостоит в использовании dist-upgrade. См. раздел [Обновление нового выпуска, раздел 3.5](#).

3.5 Обновление нового выпуска

Эта возможность АРТ позволяет вам обновлять всю систему Debian за один прием, или через Internet, или с нового CD (купленного или скачанного в виде образа ISO).

Это действие также используется при изменении отношений между установленными пакетами. Команда apt-get upgrade оставит эти пакеты нетронутыми (kept back).

Например, предположим, что вы используете выпуск 0 стабильной версии Debian и вы покупаете CD с выпуском 3. Вы можете использовать АРТ для обновления вашей системы с этого нового CD. Для этого используйте команду apt-cdrom (см. раздел [Добавление CD-ROM в файл sources.list, раздел 2.4](#)) для добавления CD в ваш файл /etc/apt/sources.list и запустите apt-get dist-upgrade.

Важно иметь в виду, что АРТ всегда высматривает новейшие версии пакетов. Поэтому, если в вашем /etc/apt/sources.list были указаны архивы, которые имеют более новые версии пакетов, чем версия этого CD, то АРТ должен скачивать пакеты с них.

В примере из раздела [Обновление пакетов, раздел 3.4](#), мы видели, что некоторые пакеты были kept back. Сейчас мы решим эту проблему с помощью действия dist-upgrade:

```
[root]@[/] # apt-get -u dist-upgrade
Reading Package Lists... Done
Building Dependency Tree... Done
Calculating Upgrade... Done
The following NEW packages will be installed:
  cpp-2.95 cron exim gcc-2.95 libident libopenldap-runtime libopenldap1
  libpcre2 logrotate mailx
The following packages have been kept back
  lilo
The following packages will be upgraded
  adduser ae apt autoconf cpp debhelper dpkg-dev esound esound-common ftp gcc
  indent ipchains isapnptools libaudiofile-dev libaudiofile0 libesd0
  libesd0-dev libgtk1.2 libgtk1.2-dev liblockfile1 libnewt0 liborbit-dev
  liborbit0 libstdc++2.10-glibc2.2 libtiff3g libtiff3g-dev modconf orbit
  procps psmisc
31 packages upgraded, 10 newly installed, 0 to remove and 1 not upgraded.
Need to get 0B/7098kB of archives. After unpacking 3118kB will be used.
Do you want to continue? [Y/n]
```

Обратите внимание, что сейчас пакеты были обновлены, а также установлены новые пакеты (новые зависимости пакетов). Обратите внимание также, что lilo по-прежнему остается kept back. Вероятно здесь более серьезная причина, чем новая зависимость. Мы можем ее найти командой:

```
[root]@[/] # apt-get -u install lilo
Reading Package Lists... Done
Building Dependency Tree... Done
The following extra packages will be installed:
  cron debconf exim libident libopenldap-runtime libopenldap1 libpcre2
  logrotate mailx
The following packages will be REMOVED:
  debconf-tiny
The following NEW packages will be installed:
  cron debconf exim libident libopenldap-runtime libopenldap1 libpcre2
  logrotate mailx
The following packages will be upgraded
  lilo
1 packages upgraded, 9 newly installed, 1 to remove and 31 not upgraded.
Need to get 225kB/1179kB of archives. After unpacking 2659kB will be used.
Do you want to continue? [Y/n]
```

Как было указано выше, lilo имеет новый конфликт с пакетом debconf-tiny, который означает, что он не может быть установлен (или обновлен) без удаления debconf-tiny.

Чтобы узнать как сохраняется или удаляется пакет, вы можете использовать:

```
# apt-get -o Debug::pkgProblemResolver=yes dist-upgrade
Reading Package Lists... Done
Building Dependency Tree... Done
Calculating Upgrade... Starting
Starting 2
Investigating python1.5
Package python1.5 has broken dep on python1.5-base
  Considering python1.5-base 0 as a solution to python1.5 0
  Holding Back python1.5 rather than change python1.5-base
Investigating python1.5-dev
Package python1.5-dev has broken dep on python1.5
  Considering python1.5 0 as a solution to python1.5-dev 0
  Holding Back python1.5-dev rather than change python1.5
```

```
Try to Re-Instate python1.5-dev
Done
Done
The following packages have been kept back
gs python1.5-dev
0 packages upgraded, 0 newly installed, 0 to remove and 2 not upgraded.
```

Таким образом легко заметить, что пакет python1.5-dev не может быть установлен из-за неудовлетворительной зависимости: python1.5.

3.6 Использование АРТ с dselect

dselect - это программа, которая помогает пользователям выбирать пакеты Debian для инсталляции. Он выглядит в чем-то громоздким и даже раздражает, но со временем вы можете привыкнуть к его консольному, основанному на ncurses интерфейсу.

Одной из особенностей dselect является ее умение работать с "рекомендуемыми" и "предлагаемыми" пакетами при установке выбранных пакетов. Чтобы использовать эту программу, запустите `dselect` от имени root. Выберите в качестве метода доступа 'apt'. В действительности в этом нет необходимости, но если вы не используете CD-ROM и хотите скачивать пакеты из Интернет, то это наилучший способ для dselect.

Чтобы достичь более глубоких знаний по эксплуатации dselect, прочтите документацию по dselect на странице Debian <http://www.debian.org/doc/ddp>.

Когда закончите выбирать в dselect, используйте команду:

```
apt-get -u dselect-upgrade
```

как в нижеприведенном примере:

```
[root]@[/] # apt-get -u dselect-upgrade
Reading Package Lists... Done
Building Dependency Tree... Done
The following packages will be REMOVED:
  lbxproxy
The following NEW packages will be installed:
  bonobo console-tools-libs cpp-3.0 enscript expat fingerd gcc-3.0
  gcc-3.0-base icepref klogd libdigest-md5-perl libfnlib0 libft-perl
  libgc5-dev libgcc300 libhtml-clean-perl libltdl0-dev libsasl-modules
  libstdc++3.0 metamail nethack proftpd-doc psfontmgr python-newt talk tidy
  util-linux-locales vacation xbill xplanet-images
The following packages will be upgraded
  debian-policy
1 packages upgraded, 30 newly installed, 1 to remove and 0 not upgraded.
Need to get 7140kB of archives. After unpacking 16.3MB will be used.
Do you want to continue? [Y/n]
```

Сравните это с результатом работы команды apt-get dist-upgrade на той же системе:

```
[root]@[/] # apt-get -u dist-upgrade
Reading Package Lists... Done
Building Dependency Tree... Done
Calculating Upgrade... Done
The following packages will be upgraded
  debian-policy
1 packages upgraded, 0 newly installed, 0 to remove and 0 not upgraded.
Need to get 421kB of archives. After unpacking 25.6kB will be freed.
Do you want to continue? [Y/n]
```

Обратите внимание на то, что многие из вышеуказанных пакетов были установлены потому, что другие пакеты "предлагали" или "рекомендовали" их. Другие были установлены или удалены (в случае, например, lbxproxy) в результате сделанного нами выбора во время работы в dselect. Dselect может быть мощным инструментом, когда используется вместе с АРТ.

3.7 Как сохранить смешанную систему

Люди часто используют тестируемый дистрибутив, потому что он более сыбилен, чем нестабильный и более новый, чем стабильный. Однако пользователи, которые хотят работать с последними версиями некоторых пакетов, но при этом не доверяют им из-за боязни дестабилизировать всю систему, имеют возможность работать со смешанными тестируемыми/стабильными системами. Или по-другому - более консервативным пользователям возможно нужна смешанная стабильная/тестируемая система.

Чтобы этого добиться, поместите следующую строку в /etc/apt/apt.conf:

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

Затем, при установке пакетов из нестабильного дистрибутива, просто используйте ключ -t:

```
# apt-get -t unstable install имя_пакета
```

Не забывайте, что для того, чтобы использовать пакеты этой версии Debian, нужно внести дополнения в файл `/etc/apt/sources.list`. Применительно к нашему примеру, нам нужно добавить строки источников для дистрибутива `unstable` после аналогичных строк для дистрибутива `testing`.

3.8 Как обновлять пакеты из указанных версий Debian

`apt-show-versions` обеспечивает безопасный путь обновления системы для пользователей смешанных дистрибутивов, без получения лишних частей нестабильного дистрибутива, чем это требуется в действительности. К примеру, можно обновить только нестабильные пакеты командой:

```
# apt-get install `apt-show-versions -u -b | grep unstable`
```

3.9 Как сохранять указанные версии установленных пакетов (сложный способ)

Могло случиться так, что вы изменяете что-то в пакете и не имеете времени или не хотите портить эти изменения в новую версию программы. Или, к примеру, вы можете просто собираться обновить ваш дистрибутив Debian до версии 3.0, на при этом хотите по-прежнему использовать некоторые пакеты из Debian 2.2. Вы можете "приколоть" (pin) установленные версии пакетов так, чтобы они не обновлялись. Это делается просто. Вам всего лишь надо отредактировать файл `/etc/apt/preferences`.

Его формат прост:

```
Package: <package>
Pin: <pin definition>
Pin-Priority: <pin's priority>
```

Например, чтобы предотвратить модификацию пакета `sylpheed` версии 0.4.99, которую я поправил на предмет "reply-to-list", я ввел следующее:

```
Package: sylpheed
Pin: version 0.4.99*
```

Обратите внимание, что я использовал * (звездочка). Это "маска"; она означает, что я хочу "приколоть" (pin) все версии, начиная с 0.4.99. Это нужно потому, что в Debian-версиях пакетов в номере версии присутствует "номер редакции Debian", а я не хочу отменять установку этих редакций. Так, например, версии 0.4.99-1 и 0.4.99-10 будут установлены, как только станут доступны. Обратите внимание, что если вы внесли свои изменения в пакет, то такой способ вам не подойдет.

Поле `Pin-Priority` необязательно; если оно не указано, то по умолчанию оно равно 989.

Давайте рассмотрим работу приоритетов. Приоритет менее 0 указывает, что пакет никогда не должен устанавливаться. Приоритет от 0 до 100 означает пакеты, которые не установлены и которые не имеют доступных версий. Они не входят в процесс выбора версий. Приоритет 100 назначается установленному пакету - для замены установленной версии пакета другой версией, заменяющий пакет должен иметь приоритет выше 100.

Приоритеты выше 100 указывают, что пакет должен быть установлен. Обычно, установленная версия пакета изменяется только при обновлении до новой версии. Любые приоритеты от 100 до 1000 (включительно) определяют это типичное поведение. Пакет с таким приоритетом не будет даунгрейдиться до доступной версии с меньшим номером версии. Для примера, если у меня установлен `sylpheed 0.5.3` и определен `pin` на `sylpheed 0.4.99` с приоритетом 999, то пакет 0.4.99 *не* будет установлен из-за значения `pin`. Чтобы можно было установить пакет меньшей версии, значение `pin` должно иметь приоритет выше 1000.

`pin` может быть определен для `version`, `release` или `origin` пакета.

При указании `pin` для версий, как мы видели, для указания нескольких версий за один раз поддерживаются литеральные номера версий, равно как и маски.

Опция `release` зависит от файла `Release` из репозитория APT или с CD. Эта опция может не использоваться вовсе, если вы используете репозитории пакетов, которые не обеспечивают этот файл. Вы можете увидеть содержимое файлов `Release`, которые вы используете в каталоге `/var/lib/apt/lists/`. Параметры для опции `release`: `a` (archive -- архив), `c` (components -- компоненты), `v` (version -- версия), `o` (origin -- источник) и `l` (label -- метка).

Пример:

```
Package: *
Pin: release v=2.2*,a=stable,c=main,o=Debian,l=Debian
Pin-Priority: 1001
```

В этом примере, мы выбираем версию 2.2* Debian (которая может быть 2.2g2, 2.2g3 -- это "точечные выпуски", которые обычно включают исправления безопасности и другие важные обновления), стабильный репозиторий, раздел `main` (как противовес `contrib` или `non-free`) и источник и метку Debian. Источник (`o=`) определяет, кто создал данный файл `Release`, метка (`l=`) определяет имя дистрибутива: Debian для самого Debian и Progeny для дистрибутива Progeny, например. Пример файла `Release`:

```
$ cat /var/lib/apt/lists/ftp.debian.org.br_debian_dists_potato_main_binary-i386_Release
Archive: stable
```

Version: 2.2r3
Component: main
Origin: Debian
Label: Debian
Architecture: i386

Глава 4 - Весьма полезные помощники

4.1 Как установить локально скомпилированные пакеты: `equivs`

Иногда, люди хотят использовать определенную версию программы, доступную только в исходных текстах, без пакета Debian. Но этому может помешать система пакетов. Предположим, вы хотите скомпилировать новую версию вашего почтового сервера. Все отлично, но многие пакеты в Debian зависят от MTA. Так как вы устанавливаете нечто, собранное вами вручную, то система пакетов об этом не будет знать.

Тут на сцену выходит весь в белом пакет `equivs`. Чтобы его использовать, установите пакет с тем же именем. Он создает пустой пакет, который содержит полный набор зависимостей, уведомляя таким образом систему пакетов, что все зависимости удовлетворены.

Прежде, чем мы начнем, хорошо бы сообщить вам, что есть более безопасные способы компиляции программ, для которых уже есть пакеты Debian с различными опциями, и что в этом случае лучше не использовать `equivs` для замены зависимостей, если вы не уверены в том, что все делаете правильно. См. подробности в разделе [Работа с пакетами исходных текстов, Глава 6](#).

Давайте вернемся к примеру с MTA, вы только что установили свежесобранный postfix и переходите к установке mutt. Сразу же вы обнаружите, что mutt нужен установленный MTA. Но ведь он у вас уже есть.

Зайдите в какой-нибудь каталог (`/tmp`, например) и запустите:

```
# equivs-control name
```

Замените *name* на имя управляющего файла, который вы создаете. Будет создан следующий файл:

```
Section: misc
Priority: optional
Standards-Version: 3.0.1

Package: <enter package name; defaults to equivs-dummy>
Version: <enter version here; defaults to 1.0>
Maintainer: <your name and email address; defaults to username>
Pre-Depends: <packages>
Depends: <packages>
Recommends: <packages>
Suggests: <package>
Provides: <(virtual)package>
Architecture: all
Copyright: <copyright file; defaults to GPL2>
Changelog: <changelog file; defaults to a generic changelog>
Readme: <README.Debian file; defaults to a generic one>
Extra-Files: <additional files for the doc directory, commaseperated>
Description: <short description; defaults to some wise words>
    long description and info
.
    second paragraph
```

Нам просто нужно поправить его, как нам нужно. Формат полей и их описаний понятен визуально, так что давайте сделаем требуемое:

```
Section: misc
Priority: optional
Standards-Version: 3.0.1
```

```
Package: mta-local
Provides: mail-transport-agent
```

Да, это все. mutt зависит от mail-transport-agent, это виртуальный пакет, обеспечиваемый всеми MTA, я мог бы использовать имя пакета mail-transport-agent, но я придерживаюсь принятой схемы именования, используя Provides.

Сейчас вам нужно только построить пакет:

```
# equivs-build name
dh_testdir
touch build-stamp
dh_testdir
dh_testroot
dh_clean -k
# Add here commands to install the package into debian/tmp.
touch install-stamp
dh_testdir
```

```
dh_testroot
dh_installdocs
dh_installchangelogs
dh_compress
dh_fixperms
dh_installdeb
dh_gencontrol
dh_md5sums
dh_builddeb
dpkg-deb: building package `name' in `./name_1.0_all.deb'.
```

The package has been created.

Attention, the package has been created in the current directory,

И установите полученный .deb файл.

Очевидно, существует несколько применений для equivs. Одно из них, например, - создание пакета my-favorites, который зависит от программ, которые вы обычно устанавливаете. Просто включите свое воображение, но будьте осторожны.

Важно заметить, что примеры управляющих файлов есть в каталоге /usr/share/doc/equivs/examples. Посмотрите их.

4.2 Удаление неиспользуемых файлов локали: localepurge

Многие пользователи Debian используют только одну локаль. Бразильские пользователи Debian, например, обычно используют локаль pt_BR и не интересуются локалью es.

localepurge - это очень полезный инструмент для таких пользователей. Вы можете освободить много места, если оставите только те локали, которые вы действительно используете. Просто дайте команду apt-get install localepurge.

Пакет очень легко настраивается, debconf обеспечивает его пошаговую настройку. Однако будьте осторожны при ответе на первый вопрос, неправильный ответ может повлечь удаление всех файлов локалей, даже тех, которые вы используете. После этого восстановить их можно будет только переустановкой всех пакетов, их предоставляющих.

4.3 Как узнать, какие пакеты можно обновить

apt-show-versions - это программа, которая показывает, какие пакеты в системе можно обновить и еще некоторую полезную информацию. Опция -u отображает список обновляемых пакетов:

```
$ apt-show-versions -u
libeel0/unstable upgradeable from 1.0.2-5 to 1.0.2-7
libeel-data/unstable upgradeable from 1.0.2-5 to 1.0.2-7
```

Глава 5 - Получение информации о пакетах.

Для системы АРТ существует несколько оболочек, которые значительно упрощают просмотр пакетов, которые доступны для установки или уже установлены, а также для поиска разделов, приоритетов, описания пакетов и т.д.

Но... наша цель состоит в изучении самого АРТ. Так как же найти имя пакета, который вам надо установить?

Для такой задачи мы располагаем несколькими ресурсами. Начнем с apt-cache. Эта программа используется системой АРТ для управления ее базами данных. Мы только вкратце оглядим ее более практические приложения.

5.1 Нахождение имен пакетов

Например, допустим, что вы хотите вспомнить старые добрые денечки Atari 2600. Вы хотите использовать АРТ для установки эмулятора Atari и скачивания некоторых игр. Можете сделать так:

```
[root]@[/] # apt-cache search atari
atari-fdisk-cross - Partition editor for Atari (running on non-Atari)
circuslinux - The clowns are trying to pop balloons to score points!
madbomber - A Kaboom! clone
tcs - Character set translator.
atari800 - Atari emulator for svgalib/X/curses
stella - Atari 2600 Emulator for X windows
xmess-x - X binaries for Multi-Emulator Super System
```

Мы нашли несколько пакетов, имеющих отношение к тому, что нам требуется, вместе с краткими описаниями. Чтобы посмотреть более полные описания, можно дать команду:

```
[root]@[/] # apt-cache show stella
Package: stella
Priority: extra
Section: non-free/otherosfs
Installed-Size: 830
Maintainer: Tom Lear <tom@trap.mtview.ca.us>
Architecture: i386
Version: 1.1-2
Depends: libc6 (>= 2.1), libstdc++2.10, xlib6g (>= 3.3.5-1)
Filename: dists/potato/non-free/binary-i386/otherosfs/stella_1.1-2.deb
Size: 483430
MD5sum: 11b3e86a41a60fa1c4b334dd96c1d4b5
Description: Atari 2600 Emulator for X windows
Stella - это portable emulator of the old Atari 2600 video-game console
written in C++. You can play most Atari 2600 games with it. The latest
news, code and binaries for Stella can be found at:
http://www4.ncsu.edu/~bwmott/2600
```

В этом выводе мы получили множество сведений о пакете, который вы хотите (или не хотите) установить, вместе с полным описанием пакета. Если пакет уже установлен в системе или есть более новая версия, то вы увидите информацию об обеих версиях. Например:

```
[root]@[/] # apt-cache show lilo
Package: lilo
Priority: important
Section: base
Installed-Size: 271
Maintainer: Russell Coker <russell@coker.com.au>
Architecture: i386
Version: 1:21.7-3
Depends: libc6 (>= 2.2.1-2), debconf (>=0.2.26), logrotate
Suggests: lilo-doc
Conflicts: manpages (<<1.29-3)
Filename: pool/main/l/lilo/lilo_21.7-3_i386.deb
Size: 143052
MD5sum: 63fe29b5317fe34ed8ec3ae955f8270e
Description: LInux LOader - The Classic OS loader can load Linux and others
This Package contains lilo (the installer) and boot-record-images to
install Linux, OS/2, DOS and generic Boot Sectors of other OSes.
.
You can use Lilo to manage your Master Boot Record (with a simple text screen)
or call Lilo from other Boot-Loaders to jump-start the Linux kernel.
```

```
Package: lilo
Status: install ok installed
Priority: important
Section: base
Installed-Size: 190
Maintainer: Vincent Renardias <vincent@debian.org>
Version: 1:21.4.3-2
Depends: libc6 (>= 2.1.2)
Recommends: mbr
Suggests: lilo-doc
Description: LInux LOader - The Classic OS loader can load Linux and others
This Package contains lilo (the installer) and boot-record-images to
install Linux, OS/2, DOS and generic Boot Sectors of other OSes.
.
You can use Lilo to manage your Master Boot Record (with a simple text screen)
or call Lilo from other Boot-Loaders to jump-start the Linux kernel.
```

Обратите внимание, что первый в списке - доступный пакет, а второй - уже установленный. Для получения более общей информации о пакете, вы можете использовать:

```
[root]@[/] # apt-cache showpkg penguin-command
Package: penguin-command
Versions:
1.4.5-1(/var/lib/apt/lists/download.sourceforge.net_debian_dists_unstable_main_binary-i386_Packages)/var/lib/dpkg/status

Reverse Depends:
Dependencies:
1.4.5-1 - libc6 (2 2.2.1-2) libpng2 (0 (null)) libsdl-mixer1.1 (2 1.1.0) libsdl1.1 (0 (null)) zlib1g (2 1:1.1.3)
Provides:
1.4.5-1 -
Reverse Provides:
```

А чтобы посмотреть только его зависимости:

```
[root]@[/] # apt-cache depends penguin-command
penguin-command
Depends: libc6
Depends: libpng2
```

```
Depends: libsdl-mixer1.1
Depends: libsdl1.1
Depends: zlib1g
```

В общем, мы имеем полный арсенал для поиска имен нужных нам пакетов.

5.2 Применение dpkg для поиска имен пакетов

Один из способов найти имя пакета - знать имя какого-либо важного файла, находящегося в этом пакете. Например, чтобы найти пакет, которому принадлежит некий файл ".h", нужный вам для компиляции, может запустить:

```
[root]@[/] # dpkg -S stdio.h
libc6-dev: /usr/include/stdio.h
libc6-dev: /usr/include/bits/stdio.h
perl: /usr/lib/perl/5.6.0/CORE/nostdio.h
```

или:

```
[root]@[/] # dpkg -S /usr/include/stdio.h
libc6-dev: /usr/include/stdio.h
```

Чтобы найти имя пакета, установленного в вашей системе, что полезно, например, если вы планируете почистить место на диске, дайте команду:

```
[root]@[/] # dpkg -l | grep mozilla
ii mozilla-browser 0.9.6-7 Mozilla Web Browser
```

Проблема этой команды в том, что она может "испортить" имя пакета. В примере выше, полное имя пакета mozilla-browser. Чтобы это исправить, вы можете изменить значение переменной окружения COLUMNS:

```
[kov]@[couve] $ COLUMNS=132 dpkg -l | grep mozilla
ii mozilla-browser 0.9.6-7 Mozilla Web Browser - core and browser
```

или использовать описание или часть его следующим образом:

```
[root]@[/] # apt-cache search "Mozilla Web Browser"
mozilla-browser - Mozilla Web Browser
```

5.3 Установка пакетов "по запросу"

Вы собираете пакет, и вдруг, бах! Ошибка из-за того, что нет какого-то там нужного файла .h. От такого поворота событий вас может спасти программа auto-apt. Она запрашивает какие пакеты установить, если они нужны, останавливает соответствующий процесс и продолжает его после установки пакета.

Для этого, в основном, запустите:

```
auto-apt run command
```

Где `command` - это команда, при выполнении которой может обнаружиться отсутствие некоего файла.

Например:

```
auto-apt run ./configure
```

Она запросит установку необходимых пакетов и вызовет apt-get автоматически. Если вы работаете в X, то графический интерфейс будет по умолчанию заменен на текстовый.

Auto-apt сохраняет базы данных в обновленном состоянии, чтобы быть более эффективной. Это достигается вызовом команд auto-apt update, auto-apt updatedb и auto-apt update-local.

5.4 Как определить, какому пакету принадлежит файл

Если вы хотите установить пакет и не можете найти его название с помощью apt-cache, но знаете имя программы этого пакета или имя любого другого файла из этого пакета, то для поиска имени пакета вы можете воспользоваться программой apt-file. Используется она примерно так:

```
$ apt-file search имя-файла
```

Работает она примерно также как и dpkg -S, но будет показывать также и неустановленные пакеты, которые содержат указанный файл. Команда может применяться также для поиска заголовочных файлов, которые могут потребоваться при компиляции программ, хотя auto-apt в данном случае более хорош, см. [Установка пакетов "по запросу", раздел 5.3](#).

Также вы можете посмотреть список файлов в пакете командой:

```
$ apt-file list имя-пакета
```

apt-file хранит базу данных, в которой отслеживает какие файлы содержат пакеты также, как это делает auto-apt и ему нужно ее обновлять. Это делается так:

```
# apt-file update
```

По умолчанию, apt-file использует базу данных auto-apt, см. [Установка пакетов "по запросу", раздел 5.3](#).

5.5 Как получить информацию об изменениях в пакете.

Каждый пакет устанавливает в свой каталог документации (/usr/share/doc/package) файл, называемый changelog.Debian.gz, который содержит список изменений, сделанных в пакете в последней версии. Вы можете прочесть эти файлы, например, утилитой zless, но в общем-то не слишком удобно после полного обновления системы искать changelog и для каждого обновленного пакета. Можно автоматизировать эту задачу с помощью инструмента apt-listchanges. Для начала надо установить пакет apt-listchanges. В течение инсталляции, Debconf настроит его. Отвечайте на вопросы так, как вам будет удобнее. Опция "Should apt-listchanges be automatically run by apt?" очень полезна, так как показывает список изменений, сделанных в каждом установленном при обновлении пакете, и позволяет вам проанализировать ситуацию перед тем, как продолжить. Опция "Should apt-listchanges prompt for confirmation after displaying changes?" запрашивает у вас разрешения продолжать после чтения списка изменений. Если вы скажете, что не хотите продолжать, то apt-listchanges вернет ошибку и apt прервет установку. После установки apt-listchanges, как только пакеты скачаются (или будут получены с CD или смонтированного диска) с помощью apt, можно будет смотреть списки изменений, сделанных в ранее установленных пакетах.

Глава 6 - Работа с пакетами исходных текстов

6.1 Скачивание пакетов исходных текстов

В мире свободного ПО общепринято изучать исходный код или даже делать исправления ошибок. Чтобы это делать, вы должны скачать исходный текст программ. Система АРТ предлагает легкий путь для получения исходных текстов пакетов для множества программ, составляющих дистрибутив, включая все файлы, необходимые для создания .deb-файлов программ.

Другой способ использования исходных текстов в Debian - адаптация более новых версий программ из, например, нестабильного дистрибутива для использования со стабильным дистрибутивом. Сборка пакета в среде стабильного дистрибутива обеспечит генерацию .deb с зависимостями, соответствующими пакетам, доступным в дистрибутиве.

Для этого запись deb-src в вашем файле /etc/apt/sources.list должна указывать на нестабильный дистрибутив. Она должна быть разрешена (раскомментирована). См. раздел [Файл /etc/apt/sources.list, раздел 2.1](#).

Для скачивания пакетов исходных текстов рекомендуется использовать команду:

```
$ apt-get source packagename
```

Команда должна скачать три файла: .orig.tar.gz, .dsc и .diff.gz. В случае, если пакеты создавались исключительно для Debian, последний из них не скачивается и первый файл не имеет вставки "orig" в своем имени.

Файл .dsc пользуется командой dpkg-source для распаковки пакета исходных текстов в каталог *packagename-version*. Внутри каждого скачанного пакета исходных текстов есть каталог debian/, который содержит файлы, необходимые для создания пакета .deb.

Для автоматического построения пакета из скачиваемых исходных текстов просто укажите ключ -b в командной строке, примерно так:

```
$ apt-get -b source имяпакета
```

Если вы рашаете не создавать .deb при скачивании, то вы можете создать его позже командой:

```
$ dpkg-buildpackage -rfakeroot -uc -b
```

из каталога, который был создан для пакета после скачивания.

Есть разница между методом source команды apt-get и ее другими методами. Метод source может быть использован обычными пользователями, без необходимости иметь права root. Файлы скачиваются в каталог, из которого вызывается команда apt-get source package.

6.2 Пакеты, необходимые для компиляции пакетов исходных текстов

Обычно, для того, чтобы пакет можно было собрать, нужны некоторые заголовочные файлы и совместно-используемые библиотеки. Все пакеты исходных текстов имеют в своих управляющих файлах поле 'Build-Depends:', которое указывает дополнительные пакеты, которые необходимы для построения пакета из исходных текстов.

АРТ имеет простой метод скачивания этих пакетов. Просто запустите apt-get build-dep package, где `package` - это имя пакета, который вы собираетесь построить. Например:

```
[root]@[/] # apt-get build-dep gmc
Reading Package Lists... Done
Building Dependency Tree... Done
The following NEW packages will be installed:
```

```
comerr-dev e2fslibs-dev gdk-implib-dev implib-progs libgnome-dev libgnorba-dev
libgpmg1-dev
0 packages upgraded, 7 newly installed, 0 to remove and 1 not upgraded.
Need to get 1069kB of archives. After unpacking 3514kB will be used.
Do you want to continue? [Y/n]
```

Будут установлены пакеты, необходимые для правильного построения пакета gmc. Важно отметить, что команда не ищет сам пакет исходных текстов. По этой причине вам потребится скачать его отдельной командой `apt-get source`.

Глава 7 - Как исправлять ошибки

7.1 Общие ошибки

Ошибки случаются зачастую из-за невнимательности пользователей. Далее будут рассмотрены несколько частных ошибок и методв их устранения.

Если вы получили пимерно такое сообщение при выполнении команды `apt-get install package...`

```
Reading Package Lists... Done
Building Dependency Tree... Done
W: Couldn't stat source package list 'http://people.debian.org unstable/ Packages'
(/var/state/apt/lists/people.debian.org_%7ekov_debian_unstable_Packages) - stat (2 No such file or directory)
W: You may want to run apt-get update to correct these missing files
E: Couldn't find package penguineyes
```

от вы забыли запустить `apt-get update` после последних изменений в файле `/etc/apt/sources.list`.

Ошибка выглядит вот так:

```
E: Could not open lock file /var/lib/dpkg/lock - open (13 Permission denied)
E: Unable to lock the administration directory (/var/lib/dpkg/), are you root?
```

если вы пытаетесь воспользоваться любым другим методом `apt-get`, кроме `source`, не имея прав `root`.

Похожие ошибки могут возникать при запске двух копий программы `apt-get` в одно и то же время, или даже если вы пытаетесь запустить `apt-get` одновременно с `dpkg`. Одновременно с другими можно запускать только метод `source`.

Если установка прервалась на середине процесса, и вы больше не можете ни установить, ни удалить пакеты, то попробуйте эти две команды:

```
# apt-get -f install
# dpkg --configure -a
```

И затем попробуйте снова. Может потребоваться запустить вторую из вышеуказанных команд более одного раза. Это важный урок для тех любителей приключений, которые используют `'unstable'`.

7.2 Где мне искать помощь?

Если вы в тупике, посмотрите вполне доходчивую документацию для системы пакетов Debian. Вам могут помочь `--help`'ы и страницы руководств, также как и документация в каталоге `/usr/share/doc`, например, в подкаталоге `/usr/share/doc/apt`.

Если документация вас не просвещает, то попробуйте поискать ответ в списках почтовой рассылки Debian. Вы можете найти более подробную информацию об указанных списках рассылки на веб-сайте Debian:

<http://www.debian.org>.

Помните, что эти списки и ресурсы должны использоваться только пользователями Debian; пользователи других систем найдут лучшую поддержку в ресурсах их собственных дистрибутивов.

Глава 8 - Какие дистрибутивы поддерживают АРТ?

Вот названия некоторых дистрибутивов, которые используют АРТ:

Debian GNU/Linux (<http://www.debian.org>) - это дистрибутив, в котором был разработан АРТ

Conectiva (<http://www.conectiva.com.br>) - это дистрибутив, который первым портировал АРТ для использования с `rpm`

Mandrake (<http://www.mandrake.com>)

PLD (<http://pld.org.pl>)

Vine (<http://www.vinelinux.org>)

APT4RPM (<http://apt4rpm.sf.net>)

Alt Linux (<http://www.altlinux.ru/>)

Red Hat (<http://www.redhat.com/>)

Sun Solaris (<http://www.sun.com/>)
SuSE (<http://www.suse.de/>)
Yellow Dog Linux (<http://www.yellowdoglinux.com/>)

Глава 9 - Благодарности

Большое спасибо моим большим друзьям в проекте Debian-BR и в самом Debian, которые постоянно помогают мне и всегда придают сил для труда на благо человечества, также как и помогают мне в стремлении сохранить мир. :)

Также я хочу выразить свою благодарность CIPSGA за посильную помощь нашему проекту и всем свободным проектам, которые несут великие идеи.

И особые благодарности:

Yooseong Yang <yooseong@debian.org> - за перевод руководства на Korean.

Michael Bramer <grisu@debian.org> - за предложение включить раздел о сохранении указанных версий.

Bryan Stillwell <bryan@bokeoa.com> - за различные заплатки и исправления.

Pawel Tecza <pawel.tecza@poczta.fm> - за различные исправления и польский перевод.

Hugo Mora <h.mora@melix.com.mx> - за испанский перевод.

Luca Monducci <luca.mo@tiscali.it> - за итальянский перевод.

Tomohiro KUBOTA <kubota@debian.org> - за японский перевод.

Pablo Lorenzoni <spectra@debian.org> - за написание раздела о netselect.

Steve Langasek <vorlon@netexpress.net> - за перевод руководства на английский.

Arnaldo Carvalho de Melo <acme@conectiva.com.br> - за вклад в список дополнительных дистрибутивов, которые сейчас работают с APT: Mandrake, PLD и Vine.

Erik Rossen <rossen@freesurf.ch> - за совет с переменной COLUMNS для команды dpkg -l.

Ross Boylan <RossBoylan@stanfordalumni.org> - за совет использовать -o Debug::pkgProblemResolver=yes.

Matt Kraai <kraai@debian.org> - за различные заплатки, советы и исправления.

Aaron M. Ucko <ucko@debian.org> - за вычитку и исправления.

Jon eslund <d98-jas@nada.kth.se> - за написание раздела об apt-file.

Глава 10 - Новые версии этого руководства

Это руководство было создано проектом [Debian-BR](#), с целью ежедневной помощи проекту Debian.

Новые версии этого документа будут доступны на странице проекта <http://www.debian.org/doc/ddp>.

Комментарии и критику можно отправлять прямо мне на электронную почту kov@debian.org.