Debian 快速參考手冊

Osamu Aoki <osamu\#at\#debian.org>
Asho Yeh <asho\#at\#debian.org.tw>
'作者' 23

CVS, 週四 一月 18 11:54:37 UTC 2007

摘要

Debian 快速參考手冊 (http://qref.sourceforge.net/) 的目標在對 Debian 系統進行簡要的介紹，就如同一本**快速參考手冊**。本文件是 Debian 參考手冊 (http://qref.sourceforge.net/) 的節錄。

版權聲明

Contents

Chapter 1

序言

該文件的主旨爲 “quick reference” 但內容已經擴充許多。 儘管如此， **Keep it short and simple** (KISS) 是本文寫作的主導原則。

1.1 文件慣例

“Debian 快速參考手册” 透過簡短的 `bash` 命令提供許多資訊。

參考：

- `bash(1)` 表示 **Unix manual** 頁面資訊。
- `info libc` 表示 **GNU TEXINFO** 頁面資訊。

1.2 Debian distributions 的基本概念

Debian 同時維護三個不同的發行版本： 這些版本是：

- `stable`：— 適用於架設產品化伺服器，該版本只會更新安全性修正的套件。
- `testing`：— 工作站的首選，該版本包含了較新版本的桌面軟體以及測試。
- `unstable`: — Cutting edge。Debian 開發者的選擇。

當`unstable`的軟體在第一個禮拜或更久沒有 release-critical(RC)的臭蟲時，便會自動移動到`testing`。

Debian 發行版本也有代號。 在Sarge發行(2005六月)之前，三個版本的的代號爲Woody(stable), Sarge(testing), Sid(unstable)。當 Sarge 發行後，三個版本代號爲 Sarge, Etch 和 Sid。當 Etch 發行後，`stable`和`unstable`版本將變成Etch和Sid;一個新的`testing`版本(從`stable`版本複製)將會產生並賦予一個新代號。

訂閱debian-devel-announce@lists.debian.org這個 mailing list 來取得 Debian 的重大公告。

如果您想要使用的軟體版本比目前發行版本提供的還新，那麼您可以跟著'將 distribution 升級至 stable, testing 或 unstable' 3這一節來升級系統的版本，或者您可只升級該軟體。如果升級該軟體不是那麼方便或造成更多問題，您可以考慮 '把套件引入 stable 系統' 17 這一節所提的 backport 套件。

Chapter 2

將 distribution 升級至 stable, testing 或 unstable

2.1 從 Potato 到 Woody

以下的步驟是獨立出來介紹的，因爲 Potato 的 APT 並不支援apt_preferences(5)文件中的功能。

編輯/etc/apt/sources.list只留下 Woody 的來源之後，請依照下列步驟升級 APT 以及核心程式到 Woody 的版本：

```
# apt-get update
# apt-get install libc6 perl libdb2 debconf
# apt-get install apt apt-utils dselect dpkg
```

接下來請升級系統到 Woody。

```
# apt-get upgrade
# apt-get dist-upgrade
```

2.2 升級前的準備

您可以透過網路安裝來升級目前的發行版本到另外一個。以下說明完成的方法。

首先取得 stable 的 repositories：

```
# cd /etc/apt
# cp -f sources.list sources.list.old
# :>sources.list
# apt-setup noprobe
```

如果您想升級到 testing，請加入 testing 的來源到該檔。如果您想升級到 unstable，請加入 unstable 的來源到該檔。

```
# cd /etc/apt
# grep -e "^deb " sources.list >srcs
# :>sources.list
# cp -f srcs sources.list
# sed -e "s/stable/testing/" srcs >>sources.list
# sed -e "s/stable/unstable/" srcs >>sources.list
# apt-get update
# apt-get install apt apt-utils
```

請看 '體驗 Debian 套件管理' 6 中說明調效 /etc/apt/sources.list 和 /etc/apt/preferences 的藝術。

2.3 升級

當您按照上述的說明設定好 /etc/apt/sources.list 和 /etc/apt/preferences 之後，您就可以進行升級的動作了。

值得注意的地方，從套件上載到unstable並移植到testing的這段時間內，會造成 Debian testing延遲安裝含有安全性修正的軟體。

參閱 'Debian 套件管理系統' 5 了解套件處理基礎，遇到問題時請參閱 'APT 升級的錯誤排除方法' 11。

2.3.1 使用 dselect

如果系統裝了許多套件的 -dev 等套件，推薦下面使用 dselect 的操作方法來進行套件的精細操作 (fine-grained package control)。

```
# dselect update   # 在升級前要先執行這一步
# dselect select   # 選擇附加的套件
```

當執行 dselect 時，所有你目前的套件都會被選擇，dselect 會基於 Depends，Suggests 和 Recommends 來提示你附加的套件，如果不想添加任何套件，只需輸入 Q 退出 dselect。

```
# dselect install
```

在安裝過程中，必須回答一些有關套件設定的問題，準備好你的筆記本花點時間處理它們。參閱 'dselect' 8。

使用 dselect看看，**他做得還不錯 :)**

Chapter 3

Debian 套件管理系統

進階級套件管理工具 aptitude 是首選的 APT 前端程式。它會紀錄額外安裝的軟體並解決惱人的相依性問題。aptitude也會移除掉不被已安裝軟體需要的套件。它內建了一個套件過濾器，但比較難上手。

synaptic 是目前首選的以 Gtk 爲 toolkit 的視窗化 APT 前端程式。它的套件過濾器就比aptitude來的友善且簡單多了。更多的功能和支援請參閱Debian Package Tags (http://debtags.alioth.debian.org/)。

爲了減少 Debian 檔案庫 (repository) 的網路負擔並加速您的下載速度，您可以考慮從 Debian 鏡射站台下載。

如果您的區網內需要安裝重複的套件到多台電腦上，請在使用 APT 下載套件時，考慮使用 squid 設定本地端的 HTTP proxy。必要的話，設定http_proxy環境變數或加入 http 設定到/etc/apt/apt.conf。

儘管 apt_preferences(5) 的 pinning 功能十分強大，但造成的影響是難以偵錯和管理。除非熟悉該工具才考慮採用。

chroot 適合於需要同時結合系統的穩定性和使用最新版軟體的情況。

本章是基於 Woody 之後的系統所撰寫的，有些資訊只適用於 Sarge 或更新的系統。

3.1 介紹

如果你沒精力閱讀完所有的開發文件，那麼先看看本章的內容，然後就開始體驗 Debian testing/unstable 的威力吧 :-)

3.1.1 主要的套件管理工具

```
dpkg       - 安裝 Debian 套件的工具
apt-get    - APT 安裝套件的指令
aptitude   - 進階級的文字介面的 APT 前端工具
```

```
synaptic   - 圖形介面的 APT 前端工具
dselect    - 使用選單介面的套件管理工具
tasksel    - 安裝 task
```

這些工具並非是爲了取代對方而產生的，相反的，他們甚至能共用彼此。例如說 dselect 能搭配使用 APT 和 dpkg來安裝套件。

APT 使用 /var/lib/apt/lists/* 來追蹤可用的套件，而 dpkg 則是使用 /var/lib/dpkg/available 。如果直接用 aptitude 或類似工具來安裝套件的話，別忘了要使用 dselect 的 [U]pdate 選項，或在執行 dselect update 來更新 /var/lib/dpkg/available 。

在處理套件相依性的方式上， apt-get 會自動搜尋下載相依的套件，但不會額外安裝該軟體推薦或建議的套件。

相反地，aptitude 可以設定成是否要額外安裝“推薦”或“建議”的套件。

dselect 在套件的選擇方面提供了選單方式的操作，會列出該軟體推薦或建議的套件並個別決定是否要安裝。

3.1.2 方便的工具

```
dpkg-reconfigure  - 重新設定一個已經安裝的套件
                    (如果它是使用 debconf 的話)
dpkg-source       - 管理套件源碼檔案
dpkg-buildpackage - 自動重新編建套件檔案
apt-cache         - check package archive in local cache
```

3.2 體驗 Debian 套件管理

3.2.1 設定 APT

參閱‘升級前的準備’3來設定sources.list。[1]

3.2.2 tasks 安裝

您可以安裝一個許多軟體集合的套件來規畫特定用途的 Debian 系統。而該集合就叫作“Task”。

安裝 tasks 最簡單的方法就是在安裝系統過程中，執行 tasksel。請記得先執行

```
dselect update
```

[1]如果您的系統以testing 或 unstable爲主，您可以移除/etc/apt/sources.list 和 /etc/apt/preferences中的 stable敘述，因爲testing初使是拷貝自stable。

。

建議使用 aptitude 來安裝 tasks，而這也是它的特色之一。它能在您選擇好 tasks 並準備安裝之前再額外刪除掉您不需要的軟體。

3.2.3 aptitude

aptitude 是一套全新的套件安裝系統，類似於 dselect，不同的是針對 APT 重新設計的。它也能當作 apt-get 另一個指令介面且完全相容 apt-get的參數喔。請參閱 aptitude(1) 和 /usr/share/doc/aptitude/README。

從您開始安裝軟體，建議使用aptitude來代替所有的安裝工具，不然會失去了aptitude所建立的套件追蹤清單。這會使您無法移除多餘的套件。

aptitude 的功能鍵如下(大部分爲小寫)：

```
Keystroke   Action
F10         Menu
?           Help for keystroke (complete listing)
u           Update package archive information
           Mark the package to be upgraded or newly installed
-           Mark the package to be removed (keep config)
_           Mark the package to be purged (remove config)
=           Place the package on hold
U           Mark all upgradable packages to be upgraded
g           Download and install selected packages
q           Quit current screen and save changes
x           Quit current screen and discard changes
Enter       View information about a package
C           View a package's changelog
l           Change the limit for the displayed packages
/           Search for the first match
\           Repeat the last search
```

如同apt-get，aptitude在安裝軟體時也會解決掉惱人的相依性問題。 aptitude 也能設定成是否要安裝軟體額外推薦或建議的軟體。 利用主畫面選單上的F10 -> Options -> Dependency handling來改變預設的安裝策略。

aptitude 的其他功能：

- aptitude 能存取所有版本的套件。
- aptitude 的動作會紀錄在/var/log/aptitude。
- aptitude 能輕鬆地追蹤本地端建立的套件並列在 “Obsolete and Locally Created Packages”。
- aptitude 內建強大的收尋引擎並過濾顯示的套件。 mutt的使用者會很容易上手，因爲 expression 的文法是來自於 mutt。請參閱 /usr/share/doc/aptitude/README 的 “SEARCHING, LIMITING, AND EXPRESSIONS”。
- aptitude內建 su 的功能所以一般使用者皆可以執行直到安裝或移除軟體時再取得管理者的權限。

3.2.4 dselect

從 Stable 發行到現在爲止(包含Potato)，dselect是主要的套件維護工具。當 Sarge 發行後，您可以考慮使用 aptitude 來取代。

當你啓動程式時， dselect 會自動選取所有 “Required” 、 “Important” 和 “Standard” 分類的套件。

雖然dselect 的使用介面有點怪，但大部分的人都已經習慣了。有四主要的指令 (都是大寫的指令！)：

按鍵	動作
Q	離開。確認目前所選取的並離開程式。 (override dependencies)
R	回復 (Revert)！ I did not mean it.
D	不管你 (Damn it)！我不管 dselect 怎麼想的。照我的做就是了！
U	都照建議 (sUggested) 的來做

使用 D 和 Q 可以選擇有衝突的選項。要小心地使用這些指令。

在 /etc/dpkg/dselect.cfg 中加上一行 “expert” 選項以減少干擾。

對於速度慢的機器，可以在其它速度快的機器上執行 dselect 先選好套件，然後用 apt-get install 來安裝。

3.2.5 使用 APT 來維持 Debian 發行版本

請編輯/etc/apt/preferences並加入以下的說明來維持系統爲 testing 版本：

```
Package: *
Pin: release a=testing
Pin-Priority: 800

Package: *
Pin: release a=stable
Pin-Priority: 600
```

要注意的是追蹤testing版本會有延誤安裝安全性修正軟體的副作用。這樣的軟體是因爲上傳到unstable並移植到testing的這段期間所造成的延誤。

更多且複雜的範例請參考 apt_preferences(5)，允許您做更多的事情，例如安裝 unstable 的套件還能把系統維持在testing。

關於限制替定軟體在特定的版本的範例可以在 examples subdirectory (http://www.debian.org/doc/manuals/debian-reference/examples/) 找到 preferences.testing 和 preferences.unstable。

如果您混用不同的發行版本，例如 testing 和 stable 或 unstable 和 stable，您終究會安裝到testing或unstable版本的核心軟體，例如libc6，而這樣的行爲無法保證系統無臭蟲存在。您必須特別小心。

另外一個例子，preferences.stable會強制降級所有的軟體到stable。

Debian 不支援將某個 **套件**降級到先前的發行版本。但新版的套件出問題時，重裝舊版的套件是被允許的。您可以在本地端的 /var/cache/apt/archives/ 或遠地端的 http://snapshot.debian.net/ 找到先前發行的版本。請參考 '使用 dpkg 來救援' 11。

Debian 也不支援將某個 **發行版本** 降級到先前的版本且這樣做往往會造成很多問題。如果您願意冒險的話，也是值得嘗試看看。

3.2.6 aptitude，apt-get和apt-cache 命令

當像之前的例子一樣跟隨著 testing，您可以下列的指令來管理系統。

- aptitude update (或 apt-get update
  以上動作會更新檔案庫中最新的套件列表
- aptitude upgrade (或 apt-get upgrade 或 aptitude dist-upgrade 或 apt-get dist-upgrade)
  這樣會跟隨testing版本 — 他們會安裝目前testing版本上可提供升級的套件以及其相依的軟體。[2]
- apt-get dselect-upgrade
  這會跟隨testing版本 — 上述指令升級的方法是依照dselect的處理方式來決定的。
- aptitude install *package*/unstable
  由 unstable distribution 安裝 *package* ，並由 testing distribution 安裝相依的套件。
- aptitude install -t unstable *package*
  將 unstable 的 Pin-Priority 設爲 990，會由 unstable distribution 安裝 *package*，也從 unstable distribution 安裝相依的套件。
- apt-cache policy *foo bar ...*
  檢查 *foo bar ...* 套件的狀態。
- aptitude show *foo bar ...* | less (或 apt-cache show *foo bar ...* | less)
  檢查 *foo bar ...* 套件的資料。
- aptitude install *foo=2.2.4-1*
  安裝 *foo* 套件的特定版本 *2.2.4-1* 。

[2] upgrade與dist-upgrade不同的地方是對於那些升級套件的相依性問題的處理方式。請參閱apt-get(8)來了解更多細節。 aptitude upgrade 和 aptitude dist-upgrade 是aptitude 的組合指令。按下 e可以全螢幕顯示。

- `aptitude install` *foo bar-*
  安裝 *foo* 套件，並移除 *bar* 套件
- `aptitude remove` *bar*
  移除 *bar* 套件但不刪除設定檔。
- `aptitude purge` *bar*
  移除 *bar* 並刪除其設定檔。

在上面的例子中，在apt-get 中使用 -u 選項會列出所有要升級的套件列表，並在動作前請示使用者。aptitude預設也是如此。下面的作法也會使 apt-get完成上述的動作：

```
$ cat >> /etc/apt/apt.conf << .
// 總是列出要升級的套件 (-u)
APT::Get::Show-Upgraded "true";
.
```

搭配 --no-act 來模擬這些安裝，移除...套件等動作。

3.3 Debian 生存指令

掌握了這些知識，就能讓你享受無窮盡的"升級"了 :-)

3.3.1 檢查 Debian 中的 bugs 並尋求幫助

如你使用某個套件出現問題，在尋求幫助或發送錯誤報告之前請確認查看過下列網站（ lynx 、 links 和 w3m 都很好用）：

```
$ lynx http://bugs.debian.org/
$ lynx http://bugs.debian.org/package-name  # 如果你知道套件名稱
$ lynx http://bugs.debian.org/bugnumber     # 如果你知道錯誤序號
```

在 Google (www.google.com) 中使用關鍵字 "site:debian.org" 搜索。

如有疑問，可閱讀說明文件。設定 CDPATH 如下：

```
export CDPATH=.:/usr/local:/usr/share/doc
```

並輸入

```
$ cd packagename
$ pager README.Debian # 如果存在的話
$ mc
```

3.3.2 APT 升級的錯誤排除方法

在升級 unstable或testing 時可能會遇到在‘升級’4 所描述的套件相依性問題。在大多數情況下，是因爲將要升級的套件所相依性套件不存在。這個問題可用下面的方法來解決：

```
# aptitude dist-upgrade
```

如果這也沒辦法的話，就重複使用下列之一的方法至到問題自動解決：

```
# aptitude -f upgrade          # 即使遇到錯誤也繼續升級
... 或
# aptitude -f dist-upgrade     # 即使遇到錯誤也繼續 dist-upgrade
```

有些升級用的 script 的確有問題，所以會持續出現狀況。通常要解決這個狀況，你最好能檢查一下這些討厭套件中的 /var/lib/dpkg/info/*packagename*.{post-,pre-}{install,removal} script，並執行：

```
# dpkg --configure -a     # 設定所有部分安裝的套件
```

如果 script 抱怨它找不到設定檔的話，在 /etc/ 中找找對應的設定檔。如果你找到的是個有 .dpkg-new 的副檔名（或是類似的東西），就把它的後綴去除掉（用 mv ）。

在安裝 unstable或testing 系統時也可能遇到相依性問題。可用這個方法巧妙的解決：

```
# aptitude -f install package # override broken dependencies
```

要修正這些問題，另一個可用的方法是使用 equivs 套件。請參閱 /usr/share/doc/equivs/README.Debian 。

3.3.3 使用 dpkg 來救援

如果你在使用 APT 遇到死胡同，那麼可以從 Debian 鏡射站台下載套件並使用dpkg來安裝。如果您還沒連上網路，可以鎖住 /var/cache/apt/archives/ 的快取檔案。

```
# dpkg -i fetchmail_6.2.5-4_i386.deb
```

如果您嘗試安裝套件卻因爲相依性問題失敗的話，請搭配--ignore-depends--force-depends或其他參數來執行dpkg。dpkg(8)有更詳盡的介紹。

3.3.4 回復套件的選取狀態的資料

不論是什麼原因，如果 /var/lib/dpkg/status 亂掉了的話，則 Debian 系統會失去套件的選取狀態的資料，這是很糟糕的事。到 /var/lib/dpkg/status-old 或 /var/backups/dpkg.status.* 找找看舊的 /var/lib/dpkg/status 檔。

因爲 /var/backups/ 這個目錄中有很多重要的系統資料，所以把它放到分開的分割區會是個不錯的主意。

如果連舊的 /var/lib/dpkg/status 檔也找不到了，你仍可以從 /usr/share/doc/ 目錄來回復這些資料。

```
# ls /usr/share/doc | \
  grep -v [A-Z] | \
  grep -v '^texmf$' | \
  grep -v '^debian$' | \
  awk '{print $1 " install"}' | \
  dpkg --set-selections
# dselect --expert # 重新安裝系統，如果需要的話去除一些選取
```

3.3.5 在 /var 崩潰後救援系統

因爲 /var 目錄下包含了常被更動的資料，如 mail ，所以比較容易會有損壞。把它放到獨立的分割區可以減少風險。如果災難發生了，你必需重建 /var 以回復 Debian 系統。

從相同或較舊版本的最簡化的 Debian 系統中取得 /var 目錄內容的架構，例如 var.tar.gz (http://people.debian.org/~osamu/pub/)，將它放入受損系統的根目錄，然後

```
# cd /
# mv var var-old       # 如果還留下有用的資料的話
# tar xvzf var.tar.gz  # 使用 Woody 架構的檔案
# aptitude             # 或是用 dselect
```

上述步驟應可使系統恢復工作。使用'回復套件的選取狀態的資料' 12 中描述的技術來加快套件選取資料的恢復。([FIXME] ：這個流程需要更多的實驗來驗證。)

3.3.6 把套件安裝到一個無法開機的系統

用 Debian 救援磁片/CD 或是一個多重開機的 Linux 系統的其它分割區來開機進入 Linux 。在 /target 掛上這個無法開機的系統，並使用 dpkg 的 chroot 安裝模式。

```
# dpkg --root /target -i packagefile.deb
```

然後設定並解決問題。

By the way, if a broken `lilo` is all that prevents booting, you can boot using a standard Debian rescue disk. At boot prompt, assuming the root partition of your Linux installation is in `/dev/hda12` and you want runlevel 3, enter:

```
boot: rescue root=/dev/hda12 3
```

Then you are booted into an almost fully functional system with the kernel on floppy disk. (There may be minor glitches due to lack of kernel features or modules.)

3.3.7 如果 dpkg 指令壞了怎麼辦

如果 dpkg 有問題，就不能安裝任何 .deb 檔了。下列的流程可幫助你來修復這個情況。（在第一行中，你可以把 "links" 換成你喜歡的瀏覽器指令。）

```
$ links http://http.us.debian.org/debian/pool/main/d/dpkg/
  ... 下載正確的 dpkg_version_arch.deb
$ su
password: *****
# ar x dpkg_version_arch.deb
# mv data.tar.gz /data.tar.gz
# cd /
# tar xzfv data.tar.gz
```

如果是 i386 的話，也可以用 http://packages.debian.org/dpkg 。

3.4 Debian 神技之指令

經過這些指令的**啓示**，你將可由無休止的升級地獄中解放出來，到達 Debian **涅盤**。 :-)

3.4.1 檔案中的資訊

在已安裝的套件中尋找特定檔案樣板所屬的套件：

```
$ dpkg {-S|--search} pattern
```

或者搜尋 Debian 檔案庫中類似的：

```
$ wget http://ftp.us.debian.org/debian/dists/sarge/Contents-i386.gz
$ zgrep -e pattern Contents-i386.gz
```

或是使用特殊的套件命令：

```
# aptitude install dlocate
$ dlocate filename          # dpkg -L 和 dpkg -S 的快速版
...
# aptitude install auto-apt # on-demand package installation tool
# auto-apt update           # 建立 auto-apt 的 db 檔
$ auto-apt search pattern
                # 尋找符合某個 pattern 的套件，不論是否安裝
```

3.4.2 套件的資訊

從套件 archive 中尋找並顯示資訊。編輯 /etc/apt/sources.list 以確定 APT 指向適合的 archive(s)。如果想了解 testing/unstable 中的相對應套件與當前系統安裝的套件有何差別，使用 apt-cache policy—會好得多。

```
# apt-get   check            # 更新暫存區並檢查損壞的套件
$ apt-cache search  pattern # 搜尋套件的文字敘述
$ apt-cache policy  package # 套件的 priority/dists 資訊
$ apt-cache show -a package # show description of package in all dists
$ apt-cache showsrc package # show description of matching source package
$ apt-cache showpkg package # 套件的除錯資訊
# dpkg  --audit|-C          # search for partially installed packages
$ dpkg {-s|--status} package ... # 已安裝套件的敘述
$ dpkg -l package ...       # 已安裝套件的狀態 (一行一個)
$ dpkg -L package ...       # 列出套件安裝的檔案
```

apt-cache showsrc 在 Woody release 時沒有文件，但是會動 :)

你也可以在下列的地方找到套件資訊 (我用 mc 來瀏覽) ：

```
/var/lib/apt/lists/*
/var/lib/dpkg/available
```

比較下面的檔案可以正確的了解最近幾個安裝的過程對系統造成了那些改變。

```
/var/lib/dpkg/status
/var/backups/dpkg.status*
```

3.4.3 Unattended installation with APT

For an unattended installation，請在 /etc/apt/apt.conf 加入一行：

```
Dpkg::Options {"--force-confold";}
```

這等同於執行 aptitude -y install *packagename* 或 apt-get -q -y install *packagename*。因爲這個作法是對所有的提示都回答 "yes" 的，所以可能會造成問題，請小心使用。參閱 apt.conf(5) 和 dpkg(1) 。

你可以在稍後用 '重新設定已安裝的套件' 15 的方法重新設定特定的套件。

3.4.4 重新設定已安裝的套件

使用下面的指令來重新設定任何已安裝的套件。

```
# dpkg-reconfigure --priority=medium package [...]
# dpkg-reconfigure --all   # 重新設定所有的套件
# dpkg-reconfigure locales # 產生額外的 locales
# dpkg-reconfigure --p=low xserver-xfree86 # 重新設定 X server
```

如果你要永久設定 debconf 爲對話窗模式，請對 debconf 進行重設定。

有些程式是用做特殊設定的 scripts 。[3]

```
apt-setup      - 建立 /etc/apt/sources.list
install-mbr    - 安裝 Master Boot Record manager
tzconfig       - 設定本地時區
gpmconfig      - 設定 gpm mouse daemon
eximconfig     - 設定 Exim (MTA)
texconfig      - 設定 teTeX
apacheconfig   - 設定 Apache (httpd)
cvsconfig      - 設定 CVS
sndconfig      - 設定音效系統
...
update-alternatives - 設定預設的指令， e.g., vim as vi
update-rc.d         - System-V init script management
update-menus        - Debian 選單系統
...
```

3.4.5 移除和清除套件

移除套件但保留它的設定檔：

```
# aptitude remove package ...
# dpkg   --remove package ...
```

[3] Some *config scripts are disappearing in the newer Sarge releases and the package configuration functionality are moved to the debconf system.

移除套件並清除所有的設定檔：

```
# aptitude  purge  package ...
# dpkg    --purge  package ...
```

3.4.6 維持舊的套件

舉例來說，下面的指令可以在 dselect 和 aptitude install *package* 時維持 libc6 和 libc6-dev 的版本：

```
# echo -e "libc6 hold\nlibc6-dev hold" | dpkg --set-selections
```

但這個方法擋不住 aptitude install *package* 。如果要在 aptitude upgrade *package* 或 aptitude dist-upgrade 時自動降級來維持版本的話，在 /etc/apt /preferences 中加入：

```
Package: libc6
Pin: release a=stable
Pin-Priority: 2000
```

在這裡的 "Package:" 不能用類似 "libc6*" 的用法。如果你要所有有關 glibc 的二進位套件都維持同步的話，你需要明確地把它們都列出來。

下面的指令可以列出被維持住的套件：

```
dpkg --get-selections "*"|grep -e "hold$"
```

3.4.7 stable/testing/unstable 混合系統

apt-show-versions 能以 distribution 來列出可用的套件版本。

```
$ apt-show-versions | fgrep /testing | wc
... 在 testing 中的套件數目
$ apt-show-versions -u
... 列出可升級的套件
$ aptitude install `apt-show-versions -u -b | fgrep /unstable`
... 將所有 unstalbe 的套件升級至各自最新的版本
```

3.4.8 刪除暫存的套件檔

用 APT 安裝的套件會在 /var/cache/apt/archives/ 留下暫存的套件檔。這些檔案是可以刪除的。

```
# aptitude autoclean # 只刪除無用的套件檔
# aptitude clean     # 刪除所有的暫存套件檔
```

3.4.9 記錄/複製系統設定

要把套件選取狀態複製到本地端：

```
# dpkg --get-selections "*" >myselections   # 或使用 \*
# debconf-get-selections > debconfsel.txt
```

"*" 使 *myselections* 也包含標記爲 "purge" 的套件。

你可將這個檔案傳到另一台電腦，並用下列的方法把它安裝起來：

```
# dselect update
# debconf-set-selections < debconfsel.txt
# dpkg --set-selections <myselections
# apt-get -u dselect-upgrade    # 或 dselect install
```

3.4.10 把套件引入 stable 系統

將 stable 系統進行部分升級，並在些環境下以重編建的方式來使用套件，這個方法是可以期待的。這個方法可以避免因相依性而對大量套件升級。首先，將下列來源加入 /etc/apt/sources.list ：

```
deb-src http://http.us.debian.org/debian testing \
 main contrib non-free
deb-src http://http.us.debian.org/debian unstable \
 main contrib non-free
```

因爲螢幕輸出的限制，上面 deb-src 中每一項都分成了2行，實際上在 sources.list 中它們都應該是單行。

然後取得源碼套件並編建出本地端的套件：

```
$ apt-get update  # 更新源碼套件的搜尋列表
$ apt-get source package
$ dpkg-source -x package.dsc
$ cd package-version
  ... 檢查必需的套件 ( 在 .dsc 檔中的 Build-depends) 並也一起安裝。
      你也需要 "fakeroot" 套件。

$ dpkg-buildpackage -rfakeroot

  ... 或是 (no sig)
$ dpkg-buildpackage -rfakeroot -us -uc # 如果需要的話，再使用 "debsign"
```

```
   ... 然後就安裝吧
$ su -c "dpkg -i packagefile.deb"
```

通常，你會需要安裝一些以 "-dev" 結尾的套件以滿足相依性的要求。 debsign 在 devscripts 套件中。 auto-apt 可以輕鬆解決相依性的問題。請使用 fakeroot 以避免不必要的 root 帳號使用。

在 Woody 中，這些相依性可被簡化。例如編譯 pine 源碼套件：

```
# apt-get build-dep pine
# apt-get source -b pine
```

3.4.11 本地端的套件 archive

爲了要建立相容於 APT 和 dselect 系統的本地端套件，需要建立 Packages 檔，而且套件檔需要被放在特別的檔案目錄樹中。

可以用下列的方法來建立類似官方 Debian archive 的本地端 deb repository ：

```
# aptitude install dpkg-dev
# cd /usr/local
# install -d pool # 套件實體是放在這裡
# install -d dists/unstable/main/binary-i386
# ls -1 pool | sed 's/_.*$/ priority section/' | uniq > override
# editor override # 調整 priority 和 section
# dpkg-scanpackages pool override /usr/local/ \
   > dists/unstable/main/binary-i386/Packages
# cat > dists/unstable/main/Release << EOF
Archive: unstable
Version: 3.0
Component: main
Origin: Local
Label: Local
Architecture: i386
EOF
# echo "deb file:/usr/local unstable main" \
   >> /etc/apt/sources.list
```

不然，你也可以用一個快速但骯髒的方法來建立本地端的 deb repository ：

```
# aptitude install dpkg-dev
# mkdir /usr/local/debian
# mv /some/where/package.deb /usr/local/debian
```

```
# dpkg-scanpackages /usr/local/debian /dev/null | \
  gzip - > /usr/local/debian/Packages.gz
#  echo "deb file:/usr/local/debian ./" >> /etc/apt/sources.list
```

把這些目錄開放給 HTTP 或 FTP 存取，並在 /etc/apt/sources.list 中設定相對應的位址，就可以遠端存取這些 archives 了。

3.4.12 轉換或安裝外來的二進位套件

alien 可將 Red Hat rpm ， Stampede slp ， Slackware tgz 和 Solaris pkg 的二進位套件檔案格式轉成 Debian 的 deb 套件。如果你不要使用你已安裝在系統中的套件，而想裝來自其他 Linux distribution 的套件的話，你可以使用 alien 來將其轉成你喜愛的套件格式並安裝。alien 也支援 LSB 套件。

3.4.13 Automatically install command

auto-apt is an on-demand package installation tool.

```
$ sudo auto-apt update
 ... update database
$ auto-apt -x -y run
Entering auto-apt mode: /bin/bash
Exit the command to leave auto-apt mode.
$ less /usr/share/doc/med-bio/copyright # access non-existing file
 ...  Install the package which provide this file.
 ... Also install dependencies
```

3.4.14 驗證已安裝的套件檔

debsums 以 MD5 偵錯碼的方式來驗證已安裝的套件檔。有些套件並沒有可用的 MD5 偵錯碼。有個可能的暫時性的修正方法提供給系統管理員：

```
# cat >>/etc/apt/apt.conf.d/90debsums
DPkg::Post-Install-Pkgs {"xargs /usr/bin/debsums -sg";};
^D
```

per Joerg Wendland <joergland@debian.org> (untested).

3.5 Debian 其它特別之處

3.5.1 dpkg-divert 指令

檔案移轉是強迫 dpkg 把某個檔案安裝在一個**被轉移過的**位置，而非預設的位置。當 Debian 套件的 scripts 發生衝突時，可用**移轉**來將檔案移開。系統管理者也可以用移轉來多載套件的設定檔，或是用在當安裝套件新版本時，包括有些沒被標記爲 conffiles 而需要被 dpkg 所保留的檔案。。

```
# dpkg-divert [--add]  filename # 加入 "diversion"
# dpkg-divert --remove filename # 刪除 "diversion"
```

平時不要請使用 dpkg-divert ，除非是必要的情況下。

3.5.2 equivs 套件

如果你從源碼來編譯程式，最好是能將它做成本地的 Debian 套件 (*.deb)。最後的手段是使用 equivs 。

```
Package: equivs
Priority: extra
Section: admin
Description: Circumventing Debian package dependencies
 This is a dummy package which can be used to create Debian
 packages, which only contain dependency information.
```

3.5.3 Alternative 指令

如果想用 vi 來執行 vim ，請用 update-alternatives ：

```
# update-alternatives --display vi
...
# update-alternatives --config vi
  Selection    Command
-----------------------------------------------
      1        /usr/bin/elvis-tiny
      2        /usr/bin/vim
*+    3        /usr/bin/nvi

Enter to keep the default[*], or type selection number: 2
```

在 Debian alternatives 系統中的項目都以符號連結的方式放在 /etc/alternatives/ 。

想設定你喜歡的 X 視窗環境的話，請用 update-alternatives 來修改 /usr/bin/x-session-manager 和 /usr/bin/x-window-manager 。

/bin/sh 則就直接符號連結至 /bin/bash 或 /bin/dash 。如果要相容於舊有且 bash 化的 script 的話，用 /bin/bash 會比較好點，但用 /bin/dash 可強迫訓練你與 POSIX 相容。升級至 2.4 Linux kernel 時傾向於將其設定至 /bin/dash 。

3.5.4 Runlevel

大部分的Debian套件在安裝後是被設定在 runlevel 2 到 5 時會自動執行。 因此，Debian 系統的 runlevel 2,3,4,5是沒有差別的，而 Debian 是保留給系統管理者來設定。 自定 runlevels。 這樣的 runlevels 系統是與其他有名的 GNU/Linux 發行版本完全不同。 您或許想取消 runlevel 2 上的xdm 或 gdm 在開機後自動執行 X display 管理器。您也可以試著修改 runlevel 3 的設定。

3.5.5 停止 daemon 服務程式

Debian distribution 非常注重系統安全，許多 daemon 伺服程式都定位在最高安全等級，因而在預設的安裝狀態下，系統只啓動最少的可用的服務程式。

如果你不確定你執行了什麼服務程式（像 Exim ， DHCP …）的話，執行 ps aux 或檢查一下 /etc/init.d/* 和 /etc/inetd.conf 的內容。同時也確認一下 /etc/hosts.deny 。而 pidof 指令也是很有用的（請參閱 pidof(8)）。

在最近的 Debian 版本中，預設 X11 是不允許（遠端） TCP/IP 連接的。經由 SSH 來 X forwarding 也是禁止的。

Appendix A

附錄

A.1 作者

Debian 快速參考手冊 是由 青木 修 (Osamu Aoki) <osamu\#at\#debian.org> 所發起，剛開始時是做爲個人的安裝備忘錄，而後叫做 “Quick Reference . . . ”。許多內容來自於 “debian-user” 郵件列表的彙整庫。也參考了 “Debian Installation Manual” 和 “Debian Release Notes” 。

而後由 Debian Documentation Project (http://www.debian.org/doc/ddp) (DDP) 的積極參與者、也是目前 “The Debian FAQ” 的主要維護者 Josip Rodin 的建議，這份文件更名爲 “Debian 參考手冊” ，並和 “The Debian FAQ” 中的幾個類似參考的章節合併。而 “Debian 快速參考手冊” 則是以節錄的方式形成。

本文件是由下列 QREF 團隊成員所編輯，翻譯和擴充：

- 原本 “Quick Reference. . . ” 的英文原版
  - 青木 修 (Osamu Aoki) <osamu\#at\#debian.org> (所有內容的領導者)
- 英文版校對和後續的貢獻
  - Esko Araj酺vi <edu\#at\#iki.fi> (etch的更新)
  - Thomas Hood <jdthood\#at\#yahoo.co.uk> (網路相關的內容)
  - Brian Nelson <nelson\#at\#bignachos.com> (特別是 X 相關的內容)
  - David Sewell <dsewell\#at\#virginia.edu> (已退休)
  - Jan Michael C Alonzo <jmalonzo\#at\#spaceants.net>
  - Daniel Webb <webb\#at\#robust.colorado.edu>
  - 所有翻譯人員的回饋
- 法文版翻譯
  - Guillaume Erbs <gerbs\#at\#free.fr> (法文版領導者)
  - Renald Casagraude <rcasagraude\#at\#interfaces.fr>
  - Jean-Pierre Delange <adeimantos\#at\#free.fr>
  - Daniel Desages <daniel\#at\#desages.com>
- 義大利文版翻譯
  - Davide Di Lazzaro <mc0315\#at\#mclink.it> (義大利文版領導者)
- 葡萄牙文版翻譯
  - Paulo Rogerio Ormenese <pormenese\#at\#uol.com.br> (葡萄文版領導者)

  - Andre Luis Lopes <andrelop\#at\#ig.com.br>
  - Marcio Roberto Teixeira <marciotex\#at\#pop.com.br>
  - Rildo Taveira de Oliveira <to_rei\#at\#yahoo.com>
  - Raphael Bittencourt Simoes Costa <raphael-bsc\#at\#bol.com.br>
  - Gustavo Noronha Silva <kov\#at\#debian.org> (coordinator)
- 西班牙文版翻譯
  - Walter Echarri <wecharri\#at\#infovia.com.ar> (西班牙文版領導者)
  - Jose Carreiro <ffx\#at\#urbanet.ch>
- 德文版翻譯
  - Jens Seidel <tux-master\#at\#web.de> (德文版領導者)
  - Willi Dyck <wdyck\#at\#gmx.net>
  - Stefan Schroeder <stefan\#at\#fkp.uni-hannover.de>
  - Agon S. Buchholz <asb\#at\#kefk.net>
- 波蘭文版翻譯 — 下列是 PDDP (http://debian.linux.org.pl) 的成員：
  - Marcin Andruszkiewicz
  - Mariusz Centka <mariusz.centka\#at\#debian.linux.org.pl>
  - Bartosz Feński <fenio\#at\#debian.linux.org.pl> (波蘭文版領導者)
  - Radosław Grzanka <radekg\#at\#debian.linux.org.pl>
  - Bartosz 'Xebord' Janowski
  - Jacek Lachowicz
  - Rafał Michaluk
  - Leonard Milcin, Jr.
  - Tomasz Z. Napierała <zen\#at\#debian.linux.org.pl>
  - Oskar Ostafin <cx\#at\#debian.linux.org.pl>
  - Tomasz Piękoś
  - Jacek Politowski
  - Mateusz Prichacz <mateusz\#at\#debian.linux.org.pl>
  - Marcin Rogowski
  - Paweł Różański
  - Mariusz Strzelecki
  - Krzysztof Ścierski
  - Przemysław Adam Śmiejek <tristan\#at\#debian.linux.org.pl>
  - Krzysztof Szynter
  - Mateusz Tryka <uszek\#at\#debian.linux.org.pl>
  - Cezary Uchto
  - Krzysztof Witkowski <tjup\#at\#debian.linux.org.pl>
  - Bartosz Zapałowski <zapal\#at\#debian.linux.org.pl>
- 簡體中文翻譯
  - Hao "Lyoo" LIU<iamlyoo\#at\#163.net>
  - Ming Hua <minghua\#at\#rice.edu>
  - Xiao Sheng Wen <atzlinux\#at\#163.com> (leader: zh-cn)
  - Haifeng Chen <optical.dlz\#at\#gmail.com>
  - Xie Yanbo <xieyanbo\#at\#gmail.com>
  - easthero <easthero\#at\#gmail.com>
- 繁體中文翻譯
  - 葉信佑 (asho) <asho\#at\#debian.org.tw> (繁體中文版領導者)

  - 唐偉清 (wctang) <wctang\#at\#csie.nctu.edu.tw>
- 日文翻譯
  - Shinichi Tsunoda <tsuno\#at\#ngy.1st.ne.jp> (日文版領導者)
  - Osamu Aoki <osamu\#at\#debian.org>
- 芬蘭翻譯
  - Esko Araj酺vi <edu\#at\#iki.fi> (芬蘭版領導者)

A.2 保證

因爲我不是專家，我也不敢自詡對 Debian 或 Linux 瞭若指掌。我所使用的安全性考量可能只適合於家庭使用。

本文件不能取代任何權威指南。

不提供任何保證。所有商標的所有權均屬於個別商標的所有人。

A.3 回饋

歡迎對本文件提出建議和補充。請以 email 寄給 debian-reference 套件或個別翻譯套件的 Debian BTS system (http://bugs.debian.org/)。使用 reportbug 可以更容易的提供一個精確的錯誤報告。你也可以用英文寄 email 至 <osamu\#at\#debian.org> 給 Osamu Aoki (http://people.debian.org/~osamu/) 或是用個別翻譯的語言寄給其翻譯者。