[previous] [Contents] [1] [2] [3] [4] [5] [6] [7] [8] [9] [10] [11] [12] [A] [B] [C] [D] [E] [F] [G] [next]

Debian Policy Manual
Chapter 11 - Customized programs

11.1 Architecture specification strings

If a program needs to specify an *architecture specification string* in some place, it should select one of the strings provided by `dpkg-architecture -L`. The strings are in the format *`os-arch`*, though the OS part is sometimes elided, as when the OS is Linux.

Note that we don't want to use *`arch`*`-debian-linux` to apply to the rule *`architecture-vendor-os`* since this would make our programs incompatible with other Linux distributions. We also don't use something like *`arch`*`-unknown-linux`, since the `unknown` does not look very good.

11.1.1 Architecture wildcards

A package may specify an architecture wildcard. Architecture wildcards are in the format `any` (which matches every architecture), *`os`*-any, or any-*`cpu`*. [100]

11.2 Daemons

The configuration files `/etc/services`, `/etc/protocols`, and `/etc/rpc` are managed by the `netbase` package and must not be modified by other packages.

If a package requires a new entry in one of these files, the maintainer should get in contact with the `netbase` maintainer, who will add the entries and release a new version of the `netbase` package.

The configuration file `/etc/inetd.conf` must not be modified by the package's scripts except via the `update-inetd` script or the `DebianNet.pm` Perl module. See their documentation for details on how to add entries.

If a package wants to install an example entry into `/etc/inetd.conf`, the entry must be preceded with exactly one hash character (#). Such lines are treated as "commented out by user" by the `update-inetd` script and are not changed or activated during package updates.

11.3 Using pseudo-ttys and modifying wtmp, utmp and lastlog

Some programs need to create pseudo-ttys. This should be done using Unix98 ptys if the C library supports it. The resulting program must not be installed setuid root, unless that is required for other functionality.

The files `/var/run/utmp`, `/var/log/wtmp` and `/var/log/lastlog` must be installed writable by group `utmp`. Programs which need to modify those files must be installed setgid `utmp`.

11.4 Editors and pagers

Some programs have the ability to launch an editor or pager program to edit or display a text document. Since there are lots of different editors and pagers available in the Debian distribution, the system administrator and each user should have the possibility to choose their preferred editor and pager.

In addition, every program should choose a good default editor/pager if none is selected by the user or system administrator.

Thus, every program that launches an editor or pager must use the EDITOR or PAGER environment variable to determine the editor or pager the user wishes to use. If these variables are not set, the programs `/usr/bin/editor` and `/usr/bin/pager` should be used, respectively.

These two files are managed through the `dpkg` "alternatives" mechanism. Every package providing an editor or pager must call the `update-alternatives` script to register as an alternative for `/usr/bin/editor` or `/usr/bin/pager` as appropriate. The alternative should have a slave alternative for `/usr/share/man/man1/editor.1.gz` or `/usr/share/man/man1/pager.1.gz` pointing to the corresponding manual page.

If it is very hard to adapt a program to make use of the EDITOR or PAGER variables, that program may be configured to use `/usr/bin/sensible-editor` and `/usr/bin/sensible-pager` as the editor or pager program respectively. These are two scripts provided in the `sensible-utils` package that check the EDITOR and PAGER variables and launch the appropriate program, and fall back to `/usr/bin/editor` and `/usr/bin/pager` if the variable is not set.

A program may also use the VISUAL environment variable to determine the user's choice of editor. If it exists, it should take precedence over EDITOR. This is in fact what `/usr/bin/sensible-editor` does.

It is not required for a package to depend on `editor` and `pager`, nor is it required for a package to provide such virtual packages.[101]

11.5 Web servers and applications

This section describes the locations and URLs that should be used by all web servers and web applications in the Debian system.

1. Cgi-bin executable files are installed in the directory

   `/usr/lib/cgi-bin`

   or a subdirectory of that directory, and the script

   `/usr/lib/cgi-bin/.../`*`cgi-bin-name`*

   should be referred to as

   `http://localhost/cgi-bin/.../`*`cgi-bin-name`*

2. (Deleted)

3. Access to images

It is recommended that images for a package be stored in `/usr/share/images/`*package* and may be referred to through an alias `/images/` as

```
http://localhost/images/<package>/<filename>
```

4. Web Document Root

   Web Applications should try to avoid storing files in the Web Document Root. Instead they should use the /usr/share/doc/*package* directory for documents and register the Web Application via the `doc-base` package. If access to the web document root is unavoidable then use

   ```
   /var/www/html
   ```

   as the Document Root. This might be just a symbolic link to the location where the system administrator has put the real document root.

5. Providing httpd and/or httpd-cgi

   All web servers should provide the virtual package `httpd`. If a web server has CGI support it should provide `httpd-cgi` additionally.

   All web applications which do not contain CGI scripts should depend on `httpd`, all those web applications which `do` contain CGI scripts, should depend on `httpd-cgi`.

11.6 Mail transport, delivery and user agents

Debian packages which process electronic mail, whether mail user agents (MUAs) or mail transport agents (MTAs), must ensure that they are compatible with the configuration decisions below. Failure to do this may result in lost mail, broken `From:` lines, and other serious brain damage!

The mail spool is `/var/mail` and the interface to send a mail message is `/usr/sbin/sendmail` (as per the FHS). On older systems, the mail spool may be physically located in `/var/spool/mail`, but all access to the mail spool should be via the `/var/mail` symlink. The mail spool is part of the base system and not part of the MTA package.

All Debian MUAs, MTAs, MDAs and other mailbox accessing programs (such as IMAP daemons) must lock the mailbox in an NFS-safe way. This means that `fcntl()` locking must be combined with dot locking. To avoid deadlocks, a program should use `fcntl()` first and dot locking after this, or alternatively implement the two locking methods in a non blocking way[102]. Using the functions `maillock` and `mailunlock` provided by the `liblockfile*` [103] packages is the recommended way to realize this.

Mailboxes are generally either mode 600 and owned by *user* or mode 660 and owned by *`user`*`:mail`[104]. The local system administrator may choose a different permission scheme; packages should not make assumptions about the permission and ownership of mailboxes unless required (such as when creating a new mailbox). A MUA may remove a mailbox (unless it has nonstandard permissions) in which case the MTA or another MUA must recreate it if needed.

The mail spool is 2775 `root:mail`, and MUAs should be setgid mail to do the locking mentioned above (and must obviously avoid accessing other users' mailboxes using this privilege).

`/etc/aliases` is the source file for the system mail aliases (e.g., postmaster, usenet, etc.), it is the one which the sysadmin and `postinst` scripts may edit. After `/etc/aliases` is edited the program or human editing it must call `newaliases`. All MTA packages must come with a `newaliases` program, even if it does nothing, but older MTA packages did not do this so programs should not fail if `newaliases` cannot be found. Note that because of this, all MTA packages must have `Provides`, `Conflicts` and `Replaces: mail-transport-agent` control fields.

The convention of writing `forward to address` in the mailbox itself is not supported. Use a `.forward` file instead.

The `rmail` program used by UUCP for incoming mail should be `/usr/sbin/rmail`. Likewise, `rsmtp`, for receiving batch-SMTP-over-UUCP, should be `/usr/sbin/rsmtp` if it is supported.

If your package needs to know what hostname to use on (for example) outgoing news and mail messages which are generated locally, you should use the file `/etc/mailname`. It will contain the portion after the username and `@` (at) sign for email addresses of users on the machine (followed by a newline).

Such a package should check for the existence of this file when it is being configured. If it exists, it should be used without comment, although an MTA's configuration script may wish to prompt the user even if it finds that this file exists. If the file does not exist, the package should prompt the user for the value (preferably using `debconf`) and store it in `/etc/mailname` as well as using it in the package's configuration. The prompt should make it clear that the name will not just be used by that package. For example, in this situation the `inn` package could say something like:

```
Please enter the "mail name" of your system.  This is the
hostname portion of the address to be shown on outgoing
news and mail messages.  The default is
syshostname, your system's host name.  Mail
name ["syshostname"]:
```

where *syshostname* is the output of `hostname --fqdn`.

11.7 News system configuration

All the configuration files related to the NNTP (news) servers and clients should be located under `/etc/news`.

There are some configuration issues that apply to a number of news clients and server packages on the machine. These are:

`/etc/news/organization`

A string which should appear as the organization header for all messages posted by NNTP clients on the machine

`/etc/news/server`

Contains the FQDN of the upstream NNTP server, or localhost if the local machine is an NNTP server.

Other global files may be added as required for cross-package news configuration.

11.8 Programs for the X Window System

11.8.1 Providing X support and package priorities

Programs that can be configured with support for the X Window System must be configured to do so and must declare any package dependencies necessary to satisfy their runtime requirements when using the X Window System. If such a package is of higher priority than the X packages on which it depends, it is required that either the X-specific components be split into a separate package, or that an alternative version of the package, which includes X support, be provided, or that the package's priority be lowered.

11.8.2 Packages providing an X server

Packages that provide an X server that, directly or indirectly, communicates with real input and display hardware should declare in their `Provides` control field that they provide the virtual package `xserver`.[105]

11.8.3 Packages providing a terminal emulator

Packages that provide a terminal emulator for the X Window System which meet the criteria listed below should declare in their `Provides` control field that they provide the virtual package `x-terminal-emulator`. They should also register themselves as an alternative for `/usr/bin/x-terminal-emulator`, with a priority of 20. That alternative should have a slave alternative for `/usr/share/man/man1/x-terminal-emulator.1.gz` pointing to the corresponding manual page.

To be an `x-terminal-emulator`, a program must:

- Be able to emulate a DEC VT100 terminal, or a compatible terminal.
- Support the command-line option `-e` *`command`*, which creates a new terminal window[106] and runs the specified *command*, interpreting the entirety of the rest of the command line as a command to pass straight to exec, in the manner that `xterm` does.
- Support the command-line option `-T` *`title`*, which creates a new terminal window with the window title *title*.

11.8.4 Packages providing a window manager

Packages that provide a window manager should declare in their `Provides` control field that they provide the virtual package `x-window-manager`. They should also register themselves as an alternative for `/usr/bin/x-window-manager`, with a priority calculated as follows:

- Start with a priority of 20.
- If the window manager supports the Debian menu system, add 20 points if this support is available in the package's default configuration (i.e., no configuration files belonging to the system or user have to be edited to activate the feature); if configuration files must be modified, add only 10 points.
- If the window manager complies with The Window Manager Specification Project, written by the Free Desktop Group, add 40 points.
- If the window manager permits the X session to be restarted using a *different* window manager (without killing the X server) in its default configuration, add 10 points; otherwise add none.

That alternative should have a slave alternative for `/usr/share/man/man1/x-window-manager.1.gz` pointing to the corresponding manual page.

11.8.5 Packages providing fonts

Packages that provide fonts for the X Window System[107] must do a number of things to ensure that they are both available without modification of the X or font server configuration, and that they do not corrupt files used by other font packages to register information about themselves.

1. Fonts of any type supported by the X Window System must be in a separate binary package from any executables, libraries, or documentation (except that specific to the fonts shipped, such as their license information). If one or more of the fonts so packaged are necessary for proper operation of the package with which they are associated the font package may be Recommended; if the fonts merely provide an enhancement, a Suggests relationship may be used. Packages must not Depend on font packages.[108]

2. BDF fonts must be converted to PCF fonts with the `bdftopcf` utility (available in the `xfonts-utils` package, `gzip`ped, and placed in a directory that corresponds to their resolution:

    - 100 dpi fonts must be placed in `/usr/share/fonts/X11/100dpi/`.
    - 75 dpi fonts must be placed in `/usr/share/fonts/X11/75dpi/`.
    - Character-cell fonts, cursor fonts, and other low-resolution fonts must be placed in `/usr/share/fonts/X11/misc/`.

3. Type 1 fonts must be placed in `/usr/share/fonts/X11/Type1/`. If font metric files are available, they must be placed here as well.

4. Subdirectories of `/usr/share/fonts/X11/` other than those listed above must be neither created nor used. (The `PEX`, `CID`, `Speedo`, and `cyrillic` directories are excepted for historical reasons, but installation of files into these directories remains discouraged.)

5. Font packages may, instead of placing files directly in the X font directories listed above, provide symbolic links in that font directory pointing to the files' actual location in the filesystem. Such a location must comply with the FHS.

6. Font packages should not contain both 75dpi and 100dpi versions of a font. If both are available, they should be provided in separate binary packages with `-75dpi` or `-100dpi` appended to the names of the packages containing the corresponding fonts.

7. Fonts destined for the `misc` subdirectory should not be included in the same package as 75dpi or 100dpi fonts; instead, they should be provided in a separate package with `-misc` appended to its name.

8. Font packages must not provide the files `fonts.dir`, `fonts.alias`, or `fonts.scale` in a font directory:

    - `fonts.dir` files must not be provided at all.
    - `fonts.alias` and `fonts.scale` files, if needed, should be provided in the directory `/etc/X11/fonts/`*`fontdir`*`/`*`package`*`.`*`extension`*, where *fontdir* is the name of the subdirectory of `/usr/share/fonts/X11/` where the package's corresponding fonts are stored (e.g., `75dpi` or `misc`), *package* is the name of the package that provides these fonts, and *extension* is either `scale` or `alias`, whichever corresponds to the file contents.

9. Font packages must declare a dependency on `xfonts-utils` in their `Depends` or `Pre-Depends` control field.

10. Font packages that provide one or more `fonts.scale` files as described above must invoke `update-fonts-scale` on each directory into which they installed fonts *before* invoking `update-fonts-dir` on that directory. This invocation must occur in both the `postinst` (for all arguments) and `postrm` (for all arguments except `upgrade`) scripts.

11. Font packages that provide one or more `fonts.alias` files as described above must invoke `update-fonts-alias` on each directory into which they installed fonts. This invocation must occur in both the `postinst` (for all arguments) and `postrm` (for all arguments except `upgrade`) scripts.

12. Font packages must invoke `update-fonts-dir` on each directory into which they installed fonts. This invocation must occur in both the `postinst` (for all arguments) and `postrm` (for all arguments except `upgrade`) scripts.

13. Font packages must not provide alias names for the fonts they include which collide with alias names already in use by fonts already packaged.

14. Font packages must not provide fonts with the same XLFD registry name as another font already packaged.

11.8.6 Application defaults files

Application defaults files must be installed in the directory `/etc/X11/app-defaults/` (use of a localized subdirectory of `/etc/X11/` as described in the *X Toolkit Intrinsics - C Language Interface* manual is also permitted). They must be registered as `conffiles` or handled as configuration files.

Customization of programs' X resources may also be supported with the provision of a file with the same name as that of the package placed in the `/etc/X11/Xresources/` directory, which must be registered as a `conffile` or handled as a configuration file.[109]

11.8.7 Installation directory issues

Historically, packages using the X Window System used a separate set of installation directories from other packages. This practice has been discontinued and packages using the X Window System should now generally be installed in the same directories as any other package. Specifically, packages must not install files under the `/usr/X11R6/` directory and the `/usr/X11R6/` directory hierarchy should be regarded as obsolete.

Include files previously installed under `/usr/X11R6/include/X11/` should be installed into `/usr/include/X11/`. For files previously installed into subdirectories of `/usr/X11R6/lib/X11/`, package maintainers should determine if subdirectories of `/usr/lib/` and `/usr/share/` can be used. If not, a subdirectory of `/usr/lib/X11/` should be used.

Configuration files for window, display, or session managers or other applications that are tightly integrated with the X Window System may be placed in a subdirectory of `/etc/X11/` corresponding to the package name. Other X Window System applications should use the `/etc/` directory unless otherwise mandated by policy (such as for Application defaults files, Section 11.8.6).

11.9 Perl programs and modules

Perl programs and modules should follow the current Perl policy.

The Perl policy can be found in the `perl-policy` files in the `debian-policy` package. It is also available from the Debian web mirrors at `/doc/packaging-manuals/perl-policy/`.

11.10 Emacs lisp programs

Please refer to the "Debian Emacs Policy" for details of how to package emacs lisp programs.

The Emacs policy is available in `debian-emacs-policy.gz` of the `emacsen-common` package. It is also available from the Debian web mirrors at `/doc/packaging-manuals/debian-emacs-policy`.

11.11 Games

The permissions on `/var/games` are mode 755, owner `root` and group `root`.

Each game decides on its own security policy.

Games which require protected, privileged access to high-score files, saved games, etc., may be made set-*group*-id (mode 2755) and owned by `root:games`, and use files and directories with appropriate permissions (770 `root:games`, for example). They must not be made set-*user*-id, as this causes security problems. (If an attacker can subvert any set-user-id game they can overwrite the executable of any other, causing other players of these games to run a Trojan horse program. With a set-group-id game the attacker only gets access to less important game data, and if they can get at the other players' accounts at all it will take considerably more effort.)

Some packages, for example some fortune cookie programs, are configured by the upstream authors to install with their data files or other static information made unreadable so that they can only be accessed through set-id programs provided. You should not do this in a Debian package: anyone can download the `.deb` file and read the data from it, so there is no point making the files unreadable. Not making the files unreadable also means that you don't have to make so many programs set-id, which reduces the risk of a security hole.

As described in the FHS, binaries of games should be installed in the directory `/usr/games`. This also applies to games that use the X Window System. Manual pages for games (X and non-X games) should be installed in `/usr/share/man/man6`.

[previous] [Contents] [1] [2] [3] [4] [5] [6] [7] [8] [9] [10] [11] [12] [A] [B] [C] [D] [E] [F] [G] [next]

Debian Policy Manual

version 3.9.8.0, 2016-03-30

The Debian Policy Mailing List
