The Debian TEX sub-policy

Frank Küster <frank@debian.org>
The Debian teTEX mailing List <debian-tetex-maint@lists.debian.org>

version 0.3, 2005-07-18

Abstract

This document provides a set of rules for the packaging of applications, fonts and input files related to TEX within the Debian GNU/Linux distribution.

Copyright Notice

Contents

Chapter 1

About this document

This document provides a set of rules for the packaging of applications, fonts and input files related to TEX within the Debian GNU/Linux distribution. It is still a in a draft state – some things might not yet be fully implemented, and others are advisable, but not strictly necessary. If in doubt, please ask on `debian-tetex-maint@lists.debian.org`.

The latest copy of this document can be found in the `Debian-TeX-policy` files in the `tex-common` package.

Chapter 2

Terms and Definitions

The following terms are used in this document:

TEX-related package Any Debian package that uses or provides parts of the TEX infrastructure, i.e. the TEX or METAFONT program or derivatives thereof, fonts or input files in a *TEXMF* tree, etc.

`tex-common` This package provides basic infrastructure and some configuration files for all TEX-related packages, including the `configuration update programs`.

Basic TEX packages A Basic TEX package is a Debian package that provides the basic infrastructure for TEX-related programs. It should provide sufficient functionality for typesetting most generated (La)TEX code, e.g. from docbook, debiandoc, or texinfo sources. Usually, the Basic TEX packages will be divided into an architecture-dependent and an architecture-independent package.

The arch-dependent package must provide at least one binary that is fully compatible with Donald E. Knuth's original TEX program, and it should provide the original TEX itself. The output formats `dvi`, PostScript and Adobe PDF must be available, either directly or by conversion of other output formats. The arch-independent package must provide at least the files necessary to create the formats for plain TEX and LATEX and the input files required by the LATEX distribution, as well as the Computer Modern fonts.

***TEXMF* tree** One directory tree, arranged according to the TEX Directory Structure. The latest version of the TDS is available at `http://www.tug.org/twg/tds/`.

configuration update programs The configuration information from files provided by different TEX-related packages must be merged and made available in appropriate form to the various programs. This is usually done by scripts that write files into the *TEXMFSYSVAR* tree.

Currently, the configuration update programs provided by `tex-common` are: `update-texmf`, `update-fmtutil`, `update-language`, `update-updmap`.

Chapter 3

File Placement

3.1 Path searching and `libkpathsea` / `libkpse`

The Basic TEX packages must provide a mechanism for searching through *TEXMF* trees that allows different files to be found depending on the invoking program and the specified file format. The only existing implementation is the `libkpathsea` library. Unfortunately, it was not originally designed for use as a dynamic shared library. A rewrite is under way to create a `libkpse` library with proper API specification and ABI compatibility. For the time being, the Basic TEX packages can provide a shared library, and program maintainers can decide to use it, or to link statically against their own copy of the code.

For use in scripts, the Basic TEX packages provide the utilities `kpsewhich`, `kpsepath`, `kpsexpand`, and `kpsestat`.

3.2 Directory trees

File locations must follow the TEX Directory Structure, TDS. It is a bug if a package only conforms to an outdated TDS version. It is a more severe bug, however, if it conforms to the current TDS version but does not make sure to depend on an appropriately recent version of the Basic TEX packages (that support this TDS version).

Configuration files must be placed below `/etc/texmf`, with symlinks pointing from the TDS locations to files or directories below `/etc/texmf`. The system-wide *TEXMFSYSCONFIG* tree, if defined, must be the same as the *TEXMFMAIN* tree; a TEX-related package must not change this.

The following *TEXMF* trees are defined, as outlined below:

1 `/usr/share/texmf/`, referenced as *TEXMFMAIN* [1]

[1] The separation between a *TEXMFMAIN* tree (for the files that have to match the binary executables) and a *TEXMFDIST* tree (for other TEX input files) is not made in Debian, because it is not necessary on a system with a decent package managment system

2 `/var/lib/texmf/`, referenced as *TEXMFSYSVAR*

3 `/usr/share/texmf-site/`, referenced as *TEXMFSITE*

4 `/usr/local/share/texmf/`, referenced as *TEXMFLOCAL*

5 optionally user-specific directories for configuration files (*TEXMFCONFIG)* and generated files (*TEXMFVAR*)

6 Any directories listed in the *TEXMFHOME* configuration variable in `texmf.cnf` or as an environment variable,

The search order is from bottom up (files in *TEXMFHOME* taking precedence over files in *TEXMFMAIN).*

Debian packages generally install files in *TEXMFMAIN* exclusively (but see 'Filenames and installation of alternative files' on this page), and may ship or create empty directories in the other trees, in accordance with Debian Policy. Packages should take care to ignore *TEXMFHOME* in their maintainer scripts.

3.3 Generated files

Generated font files must be put in subdirectories of `/var/cache/fonts`, all other generated files should be below `/var/lib/texmf` (or the user-specific variable directories), with the subdirectory structure conforming to the TDS. If necessary, symbolic links can point from static *TEXMF* trees to files below `/var/`.

An exception is the generated file `/etc/texmf/texmf.cnf`. It is not intended that local administrators edit that file, but if they do, the configuration update programs must respect these changes.

3.4 Filenames and installation of alternative files

Packages may not install files with the same name as a file already installed in a *TEXMF* tree, unless both files are in subdirectories where they will only be found by different applications, as determined by the `--progname` or `--format` switches to kpsewhich.

As an exception to this rule, packages that need newer versions of a file than already supplied by an other package and installed in *TEXMFMAIN* can place them into *TEXMFSITE.* The package must make sure that the newer version is backward-compatible, meaning it must not break compilation of any TEX document, and it should not change the output file. A change of the output file may be acceptable if an obviously buggy behavior is corrected, **and** if it had previously not been possible to easily fix this behavior in user's documents (or if the updated package and a possible fix in the document combined lead to a correct document).

Packages that install files in *TEXMFSITE* must make sure to follow not only their own upstream development, but also that of the package(s) that install the files in *TEXMFMAIN*, and make sure not to get outdated with respect to the files in *TEXMFMAIN*

Installing more than two versions of a file will most likely lead to confusion. Therefore, the possibility to shadow a file once using *TEXMFSITE* should be enough, and the usage of `dpkg-divert` is discouraged.

It is also discouraged to use a file other than from the canonical source for that file, usually the CTAN network.

3.5 Documentation

Packages should make documentation available to `texdoc`. This can be done be either installing the files below `/usr/share/doc/texmf`, or by providing symlinks from subdirectories of that location to the actual documentation files.

The entry points for documentation should have names that indicate what they document. Names like `manual.pdf` or `index.html` should be avoided, even if the directory name is unmistakable [2].

[2] This allows users to say `texdoc` *`packagename`* directly. Otherwise they will first have to find the right command line (e.g. `texdoc` *`packagename`*`/user.dvi`) using `texdoc -s` *`keyword`*

Chapter 4

Configuration

4.1 Configuration update programs

The central configuration file for TEX applications is `/etc/texmf/texmf.cnf`, the central font configuration file is `/var/lib/texmf/web2c/updmap.cfg`, the central language/hyphenation configuration `/var/lib/texmf/tex/generic/config/language.dat`, and format generation is determinded by `/var/lib/texmf/web2c/fmtutil.cnf`. All four files are generated by configuration update programs from configuration files in subdirectories of `/etc/texmf`. For `updmap.cfg`, `language.dat` and `fmtutil.cnf`, this is the only method of configuration. `texmf.cnf` can be edited manually by local system administrators, and changes will be handled by ucf. Package installation scripts, however, should not change this file, but use the `update-texmf` mechanism. Local administrators are encouraged to use the `update-texmf` mechanism, too.

Packages are free to add configuration items to the common configuration files, but they should not try to override configuration items that are supplied by other packages. Rather, shared configuration items should be supplied by the Basic TEX packages or any other package on which all involved packages depend, with a setting appropriate for all. If this is impractical, the involved packages must at least agree on the way different packages override other's settings[1].

Maintainer scripts should call `update-updmap` with the option `--quiet`. Besides that, the configuration update programs should be called without any options to allow for internal changes, e.g. of the directories where the generated files are placed.

Packages that changed `updmap.cfg` must call `updmap-sys`, packages that changed `language.dat` or `fmtutil.cnf` must call `fmtutil-sys`. They must make sure to issue the necessary `mktexlsr` commands before and after this.

[1] Note that in `texmf.cnf`, as well as in the sequence of multiple `texmf.cnf` files that are read, earlier entries override later ones.

4.1.1 Font configuration

A package that provides additional Postscript Type1 fonts for TEX should put the necessary `map` files below `/etc/texmf/map/` (which must be symlinked from *`TEXMFMAIN`*`/fonts/map` by the Basic TEX packages), and have them registered by putting a configuration file with extension `.cfg` into `/etc/texmf/updmap.d/` and calling `update-updmap --quiet`. The file contents will then be incorporated into `/var/lib/texmf/web2c/updmap.cfg`, the effective configuration file for `updmap`. `updmap` has to be called afterwards.

If a package is removed, but not purged, its `map` files will stay in subdirectories of `/etc/texmf`, but the actual font files below `/usr/share/texmf` are removed, and the font is unusable. Therefore it must make sure that the `updmap` configuration file in `/etc/texmf/updmap.d` is ignored when the package is in this state. This can be achieved by putting the "magic comment"

```
# -_- DebPkgProvidedMaps -_-
```

in the file, and installing a file in `/var/lib/tex-common/fontmap-cfg/` with the name *`package`*`.list` and the content

```
10foo
12bar
```

if the packages installs the files `10foo.cfg` and `12bar.cfg` in `/etc/texmf/updmap.d`. When the package is removed, but not purged, this file will be removed, thus disabling the `updmap` configuration files.

4.1.2 Language/Hyphenation configuration

A package that provides additional hyphenation patterns for TEX should put the actual hyphenation file into the respective places in *TEXMFMAIN*, and have them registered by putting a configuration file with extension `.cnf` into `/etc/texmf/language.d` and calling `update-language`. The file contents will then be incorporated into `/var/lib/texmf/tex/generic/config/language.dat`, the effective configuration file for TEX and friends' hyphenations.

Hyphenation patterns present the same problem as described in the previous section for font configuration files: If the package is removed, but not purged, the patterns are deleted, but the configuration information is still in `/etc/texmf/language.d/`, and the format generation would fail if they would be included in `language.dat`. Therefore, an analogous mechanism has been implemented as described for `update-updmap`: If a file in `/etc/texmf/language.d/` contains the "magic comment"

```
# -_- DebPkgProvidedMaps -_-
```

it will only be used as long it is listed in a file in `/var/lib/tex-common/language-cnf/` which should have the name *`package`*`.list`.

4.2 Best practices for packages that `build-depend` on the TEX system:

If packages that build-depend on the TEX system need a changed configuration, they should not try to provide it statically. If settings in any other configuration file are inappropriate for a package to build, this is (usually) a bug in the package that provides the file. It should be fixed in this package, not circumvented by a workaround in the build process. Such workarounds have proven to be problematic, because they might stop working after changes in the depended-on package, and such failure cannot be foreseen by its maintainers. If a change is still necessary, the package should use the configuration update programs with the `--outputdir` and `--add-file` options.

4.3 Command execution and format files

If TEX formats need to be generated before execution, this must be done in the post-installation script. Packages that depend on an executable can thus simply declare `Depends:` on the package providing the executable, and *only* do that. Any additional checks, e.g. for the existence of format files, is unnecessary and harmful, causing internal changes (e.g. of format file extensions) to break the depending package that does this check.

4.4 The Dpkg Post-Invoke Mechanism

To be done...

Packages should be able to delay running of mktexlsr, updmap and perhaps even "fmtutil –all" until all TEX-related packages that want to do this are configured. Thus, it would be unnecessary to call the programs multiple times. Coding this is easy, however it is unclear how it can be made sure that failures get attributed to the correct program (even updmap has recently been reported to fail).