The Debian TEX sub-policy

The Debian TEX mailing list <debian-tex-maint@lists.debian.org>

2015-09-03 (incomplete)

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-tex-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.

TDS The TEX Directory Structure, which describes file placement for TEX input files. The current version of the TDS is installed with this document as tds.pdf (`file:///usr/share/doc/tex-common/tds.pdf`) and tds.html (`file:///usr/share/doc/tex-common/tds.html`). The latest version of the TDS is available at `http://www.tug.org/twg/tds/`.

***TEXMF* tree** One directory tree, arranged according to the TDS

TEX input file A file that is meant to be used by a TEX-related program; technically any file that can be found by the `/kpathsea/kpse` library. This includes e.g. Type1 font files.

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

TEX packages for the impatient

- A package that only installs TEX input files, e.g. a new LATEX package, should install them in the *TEXMFDEBIAN* tree (`/usr/share/texmf/`) at the place indicated by the TDS, see tds.html (`file:///usr/share/doc/tex-common/tds.html`) and 'File searching and `libkpathsea` / `libkpse`' on page 9, and register them in the maintainer scripts, usually by calling `dh_installtex` in `debian/rules`
- Packages that add fonts, hyphenation patterns or formats, or want to change the basic configuration in `texmf.cnf`, need to follow the rules in 'Configuration update programs' on page 11 in addition to that.

Chapter 4

Meta-packages and dependencies

The TEX Live collection of basic and add-on TEX packages provides some meta-packages for the convenience of users.

Depending on the `texlive-*` metapackages is only acceptable for editors, IDEs and other tools which handle user-generated code. TEX add-on packages, as well as automated input generators etc., must instead depend on a list of individual texlive packages which are actually used. [1]

[1] This is, for example, required to be able to adapt dependencies of metapackages according to the users' needs.

Chapter 5

File Placement

This chapter describes the placement of TEX input files, so that they can be found by programs. Files that are not input files for TEX or related programs must not be put in a TEXMF tree (put them into `/usr/share/`*`package`* instead). As an exception, documentation files in plain text may be used inside a *TEXMF* tree, e.g. to explain the purpose of an otherwise empty directory.

5.1 File searching and `libkpathsea` / `libkpse`

File locations must follow the TEX Directory Structure, TDS. The TDS specification is available as tds.pdf (`file:///usr/share/doc/tex-common/tds.pdf`) and tds.html (`file:///usr/share/doc/tex-common/tds.html`), and the latest version of the TDS is available at `http://www.tug.org/twg/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 or `tex-common` (that supports this TDS version).

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`.

5.2 Directory trees

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

1 `/usr/share/texlive/texmf-dist/`, referenced as *TEXMFDIST*

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

3 `/usr/share/texlive/texmf/`, referenced as *TEXMFMAIN*

4 `/usr/share/texmf/`, referenced as *TEXMFDEBIAN*

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

6 `/etc/texmf/`, referenced as *TEXMFSYSCONFIG*

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

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

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

The role of the trees *TEXMFMAIN* and *TEXMFDIST* in Debian parallels the usage in upstream TEX Live. Upstream uses *TEXMFMAIN* for the files that have to match the binary executables and *TEXMFDIST* for other TEX input files that are

replaced when a new texmf tarball appears; *TEXMFDEBIAN* is an additional tree where TEX add-on packages can put their files.

Debian packages generally install files in *TEXMFDEBIAN*, and may ship or create empty directories in the other trees, in accordance with Debian Policy. Configuration file handling in *TEXMFSYSCONFIG* is described below in 'Configuration files' on the facing page. Packages should take care to ignore *TEXMFHOME* in their maintainer scripts.

5.3 Generated files

Generated files should be created below *TEXMFSYSVAR* (or the user-specific variable directories, *TEXMFVAR*), with the subdirectory structure conforming to the TDS. Generated font files will either be created in each user's *TEXMFVAR* tree, or in the *VARTEXFONTS* tree[1]

An exception is the generated file `/etc/texmf/web2c/texmf.cnf`. Local administrators should not edit this file, as manual changes will be overwritten later on. Instead, configuration file snippets in `/etc/texmf/texmf.d` must be used.

5.4 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.

There are two exception to this rule:

1 Basic TEX packages install their files into their *TEXMFDIST* directory and will usually contain files that are also in other basic TEX packages.

2 Packages that need newer versions of a file than already supplied by a basic TEX package and installed in *TEXMFDIST* can place them into *TEXMFDEBIAN*. Thus, the outdated file will be shadowed, and the new one is in effect.

The maintainer of the basic TEX package should be made aware of the problem [2] The package maintainer must make sure to follow new releases of the basic TEX packages and not continue shadowing a file that is newer than the version provided by the shadowing package.

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).

Installing more than two versions of a file will most likely lead to confusion. Therefore, the possibility to shadow a file once 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.

5.5 Documentation

Packages should make documentation available to `texdoc`. This can be done be either installing the files below `/usr/share/texmf/doc`, or by providing symlinks from subdirectories of that location to the actual documentation files. To allow partial parallel installation of different basic TEX packages, these always install their documentation files into `/usr/share/doc/`*`packagename`* and put symlinks into their respective *TEXMFDIST*.

Note that the previous location `/usr/share/doc/texmf` is obsolete and should not be used. Files installed there will not be found by `texdoc`.

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 [3].

[1] Per default, this tree is located in the world-writeable directory `/tmp/texfonts/`, in order to allow automatic package builds to work without user directories. On multi user systems, the admin might want to change this to a persistent directory and set up proper permissions

[2] A wishlist bug on the shadowing package, blocked by an other wishlist bug on the basic TEX package, can help tracking these issues.

[3] 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 6

Configuration

6.1 Configuration files

Files that are used to modify the behavior of executables must be treated as any other configuration file in a Debian package. However, files that are used to control the typeset output - the appearance of documents - need not be treated as configuration files. It is up to the maintainer of the package to decide which files make sense to be used for site-wide (as opposed to per-project or per-document) customization.

A typical case for a site-wide configuration file is a file that must be changed if a style file should use additional modules (installed, for example, into TEXMFLOCAL). Options that only control document output are rather used for a particular document or documentation project and should usually not be installed as a configuration file.

Note that `/etc/texmf/` is a usual TDS tree. Files can be put into appropriate TDS-conforming subdirectories (e.g. `/etc/texmf/fonts/map/`), but directories not specified in TDS (or added Debian-specifically in `tex-common`'s files in `/etc/texmf/texmf.d/`) are generally not searched for TEX input files and can be used by packages for configuration files that are not TEX input files (e.g. the files in subdirectories `fmt.d` or `hyphen.d`).

6.2 Configuration update programs

Configuration files in the TEX world come in two classes: stackable and unstackable. The first class means that the respective programs read *all* configuration files found, while in the later case only the top or first configuration file is used.

Stackable configuration files in TEX are `TEXMFTREE/web2c/texmf.cnf` (central configuration for TEX applications), `TEXMFTREE/web2c/updmap.cfg` (font configuration), and `TEXMFTREE/web2c/fmtutil.cnf` (for format definitions). Unstackable configuration files are `TEXMFTREE/tex/generic/config/language.dat` (language support/hyphenation patterns for latex based formats), `TEXMFTREE/tex/generic/config/language.def` (the same for etex based formats), and `TEXMFTREE/tex/generic/config/language.dat.lua` (the same for luatex based formats).

In Debian, by default the respective configuration files of the following trees are used: For `texmf.cnf`: TEXMFDEBIAN (the texmf.cnf file is a link to the one in TEXMFMAIN). For `updmap.cfg`:TEXMFDIST, TEXMFDEBIAN. For `updmap.cfg`:TEXMFDIST, TEXMFDEBIAN. For the unstackable configuration files the respective copies in TEXMFSYSVAR are used.

The stackable configuration files are either static (texmf.cnf) or generated automatically in the background without any need for configuration, since changes can be included in a higher order configuration file.

The non stackable configuration files plus the file `/etc/texmf/web2c/texmf.cnf` are generated by configuration update programs from configuration files in subdirectories of `/etc/texmf`. For all of them this is the only method of configuration.

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].

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.

[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.

All package configuration related to TEX files can be done using `tex-common`'s trigger mechanism. That means that packages that changed `updmap.cfg` (via `update-updmap`) must either call `update-texmf-config map` which will pass the configuration work to `tex-common`, or call `updmap-sys`.

In a similar way, packages that changed `language.dat` or `fmtutil.cnf` must either call `update-texmf-config hyphen` (for `language.dat`) or `update-texmf-config format` (for `fmtutil.cnf`), which will pass the configuration work to `tex-common`, or call `fmtutil-sys` (see below).

The recommended way to implement the configuration scheme necessary is to use the `debhelper` program `dh_installtex` provided by `tex-common`. See `dh_installtex(1)` for usage details.