The Debian TEX sub-policy

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

generated from $Id: Debian-TEX-Policy.sgml 334 2005-11-14 16:28:18Z frank $

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.

TDS The TEX Directory Structure, which describes file placement for TEX input files. 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

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 or `tex-common` (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. In the future, `/etc/texmf` might become a separate, additional *TEXMF* tree *TEXMFSYSCONFIG*; a TEX-related package should try to not rely on either setup.

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

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. Debian packages must not alter that file.

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

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

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, must 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` as detailed in 'Font configuration' on the next page. Packages that changed `language.dat` or `fmtutil.cnf` must call `fmtutil-sys` (see below). They must make sure to issue the `mktexlsr` command before 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 PostScript Type 1 fonts for TEX should be usable with any Basic TEX Package. The recommended way to implement the configuration scheme described below is to use the `debhelper` program `dh_installtexfonts` provided by `tex-common`. See `dh_installtexfonts(1)` for usage details.

For the rest of this section, we'll assume we are dealing with a package named *package* that installs PostScript Type 1 fonts for TEX. *package* should fulfill the following requirements:

1 It should depend on `tex-common` but not on any Basic TEX Package, unless needed for another task than simply installing the fonts for TEX.

2 It should install the necessary map files (`.map` extension) below `/etc/texmf/map/`. This directory (or relevant subdirectories thereof, such as `dvipdfm`, `dvips` and `pdftex`) is symlinked from *`TEXMFMAIN`*`/fonts/map` by the Basic TEX Packages.

The precise location under `/etc/texmf/map/` must conform to TDS version 1.1, except that when parts of *`TEXMFMAIN`* are accessible under `/etc` due to symbolic links installed by the Basic TEX Packages, the location under `/etc` should be used (this is the case when a directory is supposed to contain configuration files; for instance, don't install map files into `/usr/share/texmf/fonts/map/dvips/`, but use `/etc/texmf/map/dvips/` instead).

3 It should also obviously install other needed or useful files provided by upstream to use the fonts with TEX-related programs (`.pfb`, `.tfm`, `.enc`, `.fd`, `.sty`, documentation, etc.).

4 It should install one or more configuration files with names following the pattern `10`*`name`*`.cfg` into `/etc/texmf/updmap.d/`. Such files will be later merged by `update-updmap` to form `/var/lib/texmf/web2c/updmap.cfg`, the effective configuration file for `updmap-sys`.

Exactly what to put in these files is documented in `update-updmap(1)`. Basically, they should contain the pseudo-comment:

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

as well as the usual `Map` and/or `MixedMap` lines that *package* needs to add to `/var/lib/texmf/web2c/updmap.cfg`.

5 It should install a file named `/var/lib/tex-common/fontmap-cfg/`*`package`*`.list` that contains a reference to every `.cfg` file from the previous step, one per line. For instance, if *package* installs `10foo.cfg` and `10bar.cfg` into `/etc/texmf/updmap.d/`, the contents of `/var/lib/tex-common/fontmap-cfg/`*`package`*`.list` should be:

```
10foo
10bar
```

This *package*.list file must be shipped in the .deb, so that when *package* is removed (not necessarily purged), *package*.list disappears from /var/lib/tex-common/fontmap-cfg/.

6 It should run:
- in *package*.postinst;
- when *package*.postrm is called with remove or disappear as its first argument

the following commands in this order: update-updmap --quiet, mktexlsr and updmap-sys.

Since mktexlsr and updmap-sys are provided by the Basic TEX Packages, *package*.postinst has to ensure that they are only called when found in $PATH (unless *package* depends on the Basic TEX Packages for some reason). In *package*.postrm, the same considerations must be taken into account, with the addition that tex-common (that provides update-updmap) can be unconfigured or even uninstalled.

As long as tex-common is configured, it is expected that mktexlsr and updmap-sys can be safely run whenever available (even if the packages that provide them aren't configured).

A sample implementation of this scheme can be found in 'Sample code for font packages' on page 15, but don't forget that dh_installtexfonts can do the work for you.

The rest of this section explains the rationale behind the previous recommendations.

- The dependency on tex-common ensures that in *package*.postinst, update-updmap can be run and texmf.cnf is in a sane state, so that mktexlsr and updmap-sys can be run safely if present.
- The recommended order for running the programs update-updmap, mktexlsr and updmap-sys ensures that updmap-sys can locate the newly-installed files (in particular, the map files shipped by *package*), since mktexlsr is run before updmap-sys. It is also run after update-updmap, because /var/lib/texmf/web2c/updmap.cfg might have been created by update-updmap, although it more probably already existed. And since it would be of no use to call mktexlsr before update-updmap, we recommend to run it after, just in case.
- Now, about the "magic comments" in /etc/texmf/updmap.d/*.cfg and the *package*.list file in /var/lib/tex-common/fontmap-cfg/. Suppose that *package* is removed, but not purged. Its map files will stay in subdirectories of /etc/texmf/map/, but the actual font files below /usr/share/texmf/ will be removed, rendering the fonts unusable. Therefore, *package* has to make sure that its update-updmap configuration files in /etc/texmf/updmap.d/ are ignored when it is in this state. Besides, we want the /etc/texmf/updmap.d/*.cfg files to be conffiles (unless we really have no other choice), because then dpkg automatically handles upgrades while preserving user modifications for them. As a consequence, moving the .cfg files from *package* out of the way when it is removed is not an option. Moreover, the user would wonder where his configuration files have gone in such a case.

The solution we chose was to add a little bit of logic into update-updmap, so that whenever it sees a .cfg file (let's call it 10foo.cfg) that has the "magic comment", it actually includes its contents into updmap.cfg if, and only if 10foo appears on a line by itself in one of the .list files in /var/lib/tex-common/fontmap-cfg/. Additionally, that .list file should be named *package*.list if 10foo.cfg comes from *package*, for simple reasons of tidiness.

With this little mechanism in place, all the rest follows as expected:

- When *package* is removed, but not purged, *package*.list is first removed by dpkg from /var/lib/tex-common/fontmap-cfg/, thus disabling the the .cfg files shipped by *package* as far as update-updmap is concerned. Then, *package*.postrm calls update-updmap, mktexlsr and updmap-sys, with the result that *package*'s map files aren't listed anymore in the final map files (psfonts.map, pdftex.map...) generated by updmap-sys.
- If *package* is reinstalled later, *package*.list first reappears in /var/lib/tex-common/fontmap-cfg/. Then, *package*.postinst runs update-updmap, mktexlsr and updmap-sys, and the .cfg files shipped by *package* aren't ignored by update-updmap this time, since they are referenced in /var/lib/tex-common/fontmap-cfg/*package*.list. Thus, the map files shipped by *package* do end up in the final map files generated by updmap-sys.

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.

4.1.3 Format configuration

Packages that provide additional formats should put a configuration file according to `fmtutil.cnf(5)` into `/etc/texmf/fmt.d/`, run `update-fmtutil` and subsequently create the format with `fmtutil --byfmt format`. `fmtutil` will only try to create the format if it can find the corresponding *`format`*`.ini` file (the last argument in an `fmtutil.cnf` line). Therefore the *`format`*`.ini` file should not be a conffile.

If a package needs to create formats at runtime, it should use a local `fmtutil.cnf` with the appropriate entries and specifiy its location to `fmtutil` on the command line, using the `--cnffile` switch.

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 should 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. Maintainer scripts or programs in Debian packages should always use `fmtutil` or `fmtutil-sys` for format generation, and either add a `fmtutil.cnf` snippet in `/etc/texmf/fmt.d/` (with `fmtutil-sys`, for site-wide formats), or use `fmtutil` with the `--cnffile` option and an appropriate local `fmtutil.cnf` (for runtime programs)

Local administrators can override settings from `texmf.cnf` with environment variables; this has sometimes lead to errors in `postinst` scripts. It is recommended that `postinst` scripts unset relevant variables before format creation or other problematic tasks.

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).

Appendix A

Sample code

This section contains sample code that implements the recommodations of this document.

A.1 Sample code for font packages

Sample postinst script:

```
#
# postinst-texfonts
#
# postinst snippets for installing fonts for TeX
#
# Author: Florent Rougon <f.rougon@free.fr>
#
update_fontmaps()
{
    update-updmap --quiet
    # mktexlsr is recommended now because updmap-sys relies heavily on
    # Kpathsea to locate updmap.cfg and the map files. Also, it is slightly
    # better not to specify a particular directory to refresh because
    # updmap.cfg is typically found in $TEXMFSYSVAR while the map files are i
    # $TEXMFMAIN.
    # According to the Debian TeX policy, running mktexlsr and updmap-sys
    # should work as long as tex-common is configured and these files are
    # available (general Debian policy wouldn't assure that without this
    # override from the Debian TeX policy).
    if which mktexlsr >/dev/null; then mktexlsr; fi
    if which updmap-sys >/dev/null; then
        printf "Running updmap-sys... "
        updmap-sys --quiet
```

```
        echo "done."
    fi

    return 0
}

case "$1" in
    configure|abort-upgrade|abort-remove|abort-deconfigure)
        update_fontmaps
    ;;

    *)
        echo "postinst called with unknown argument \`$1'" >&2
        exit 1
    ;;
esac
```

Sample postrm script:

```
#
# postrm-texfonts
#
# postrm snippets for installing fonts for TeX
#
# Author: Florent Rougon <f.rougon@free.fr>
#
tell_that_errors_are_ok()
{
    # Cheap option handling...
    if [ "$1" = -n ]; then
        prog="$2"
        endwith=' '
    else
        prog="$1"
        endwith='\n'
    fi

    # According to the Debian TeX policy, running mktexlsr and updmap-sys
    # should work as long as tex-common is configured and these files are
    # available (general Debian policy wouldn't assure that without this
    # override from the Debian TeX policy).
    printf "\
Trying to run '$prog' (error messages can be ignored if tex-common
is not configured)...$endwith"
```

```
    return 0
}

# The function name is *try_to*_update_fontmaps because the following
# scenario might happen:
#    1. this package is deconfigured
#    2. tex-common and tetex-bin are removed
#    3. this package is removed or purged
#
# (cf. Policy § 6.5, step 2, about a conflicting package being removed due
# to the installation of the package being discussed).
#
# In this case, update-updmap, mktexlsr and updmap-sys would all be gone once
# tex-common and tetex-bin are removed, so we must append "|| true" to their
# calls.
try_to_update_fontmaps()
{
    # Don't print alarming error messages if the programs aren't even
    # available.
    if which update-updmap >/dev/null; then
        tell_that_errors_are_ok -n update-updmap
        update-updmap --quiet || true
        echo "done."
    fi
    # mktexlsr is recommended now because updmap-sys relies heavily on
    # Kpathsea to locate updmap.cfg and the map files. Also, it is slightly
    # better not to specify a particular directory to refresh because
    # updmap.cfg is typically found in $TEXMFSYSVAR while the map files are i
    # $TEXMFMAIN.
    if which mktexlsr >/dev/null; then
        tell_that_errors_are_ok mktexlsr
        mktexlsr || true
        echo "done."
    fi

    if which updmap-sys >/dev/null; then
        tell_that_errors_are_ok -n updmap-sys
        updmap-sys --quiet || true
        echo "done."
    fi

    return 0
}

case "$1" in
    remove|disappear)
```

```
        try_to_update_fontmaps
    ;;

    purge)
        # Supposing updmap.cfg & Co are clean (which I think is a reasonable
        # assumption), we don't need to call try_to_update_fontmaps().
        # Calling it on remove _and_ on purge just for hypothetical users
        # who would break their config before purging this package seems to
        # be more annoying than useful (it takes a lot of time).
    ;;

    upgrade|failed-upgrade|abort-upgrade|abort-install)
    ;;

    *)
        echo "postrm called with unknown argument \`$1'" >&2
        exit 1
    ;;
esac
```