

Managing Multiple TDS Trees

Michael J Downes

American Mathematical Society

Providence, Rhode Island 02904

USA

mjd@ams.org

Abstract

Did you ever take on the task of editing a file named `texmf.cnf`? Did you find it as easy as falling off a log? Or did you find it about as easy as a log falling on your head—in the case of multiple TDS trees, more than one log? If so, you might like to hear about my own quest for a DWINE \TeX system in the labyrinth of TDS/ $\text{te}\TeX$ / kpathsea configuration. [Do What I Naively Expected]

The TDS specification provides a good structural framework for organizing a single TEXMF tree such as the one at the heart of Thomas Esser's $\text{te}\TeX$ distribution. In practice, however, because of the normal processes of local changes and ongoing upgrades it is often desirable to set up a more elaborate system of multiple TDS trees, one that organizes the pieces that have been added above and beyond the original base tree in a way that makes it easier to cope with continuing change.

Introduction

\TeX Directory Structure After \TeX had been in use for some years people started to notice that the set of files needed by \TeX when processing documents was growing rather large and unwieldy. In response a volunteer committee analyzed the categories of files involved and drew up a specification known as “TDS”: \TeX Directory Structure. (See <http://www.ctan.org/tex-archive/help/Catalogue/entries/tds.html>.)

The TDS spec gives a recommended hierarchical structure for collections of \TeX -related files in a generic form that is intended to be usable and compatible across the various kinds of computers in common use nowadays: PC/Windows, Macintosh, Unix, and so forth. A TDS tree is a tree of directories with each major subcategory of files residing in its own subtree—fonts, macros, and documentation, to name a few.

But a TDS tree has everything branching from a single root. In practice it may often be desirable to set up more than one TDS tree, as mentioned in the TDS documentation:

... we shall designate the root TDS directory by ‘texmf’ (for “ TeX and METAFONT”). We recommend using that name where possible, but the actual name of the directory is up to the installer. On PC networks, for example,

this could map to a logical drive specification such as ‘T:’.

...

A site may choose to have more than one TDS hierarchy installed (for example, when installing an upgrade).

\TeX systems: web2c , $\text{te}\TeX$, $\text{MiK}\TeX$, etc

Installing a \TeX system The \TeX Users Group has a good listing of \TeX systems for various platforms at <http://www.tug.org/interest.html>. I am going to place my discussion of TDS trees within the framework of a \TeX system called $\text{te}\TeX$, which is layered in turn on top of another system called **web2c** . One of the most important elements in **web2c** is a library called kpathsea , designed as a tool for other programs to use when they need to efficiently search a tree containing a large collection of files, and in particular to provide ways for users to specify search rules that reflect the needs of a given application. Thus, the versions of \TeX , METAFONT, dvips , bibtex , and other programs included in the **web2c** are all adapted to use the kpathsea library.

The use of the kpathsea library is the critical factor that determines the exact syntax and configuration methods one uses when making arrangements to have \TeX search multiple TDS trees. Other \TeX

systems such as MiKTeX or TrueTeX that support recursive searching through a TDS tree differ in various details, but many of the general principles will be the same.

The **web2c** guys, notably **Karl Berry & Olaf Weber** have pulled the source code for the programs TeX, Metafont, dvips, xdvi, into a coherent collection that compiles easily and plugs in the **kpathsea** path searching library where applicable. (Thank you, Karl and Olaf, for all your hard work!)

The teTeX guys, notably **Thomas Esser**, have built a big standard collection that includes **web2c** as a base, but (a) is designed to simplify installation for ordinary users who don't necessarily want to have all the sources and don't plan to make a practice of recompiling things; and (b) includes a rather comprehensive set of commonly needed runtime files, which makes the programs easier to use after they are installed. (Thank you, Thomas, for all your hard work!)

Maintaining a TeX System If you did not have any TeX stuff on your computer system, what would you do? You would look for something like teTeX that promises to be as easy as possible to install and contains a reasonably comprehensive selection of the files that most people consider useful. If the system that you install has everything you need, maintenance is easy: do nothing.

If you have any special interests, however, you may well find that you need to obtain some additional packages or fonts that were not included in the default installation of your TeX system.

Or suppose that you develop a set of favored abbreviations and put them in a file **potter.sty** so that you can call it from a L^ATeX document with **\usepackage**. Where can you put **potter.sty** so that you can use it readily in any of your documents—i.e., without putting a copy in each subdirectory that you might run L^ATeX from.

Or suppose that you develop a special package for your company that needs to be usable not just by you but by anyone in the company?

Case 1. Making it available to all users on a multi-user system.

Case 2. Making it available on multiple systems.

If you want to automatically propagate changes to other systems, keeping a separate “additions and upgrades” tree makes it easier to identify what needs to be copied and what doesn't. This can also be accomplished with other tools such as **rsync**—but only if the target system is one that is network-connected and capable of running **rsync**.

Documentation

Much of the most useful documentation for **web2c** and **kpathsea** is in “Info” form, which can be read with Emacs or with the standalone **info** application.

web2c [web2c.info](#)

kpathsea [kpathsea.info](#)

teTeX See TETEXDOC.* and teTeX-FAQ:

TEXMF/doc/tetex/teTeX-FAQ

TEXMF/doc/tetex/TETEXDOC.dvi

dvips [dvips.info](#)

xdvi [man page](#)

If your TeX system is teTeX, you can use the command

texdoc TETEXDOC

to automatically find the .dvi file and start up a viewer (probably xdvi).

When I tried **texdoc teTeX-FAQ**, however, it didn't work. You may have better luck. My attempt was with teTeX 1.0.

How many TEXMF trees?

The number of TEXMF trees that you will want to use depends on your circumstances, of course. The simplest approach is of course to have only one TEXMF tree, the one that came with your TeX system, and dump all upgrades and added packages into it. This takes less work ... provided that you plan to kick the bucket before you ever have to upgrade your TeX system.

However, if you were perfectly happy with such an approach you probably wouldn't be reading this article now. So supposing that you are convinced that maintaining more than one TDS tree is probably a good idea, I will go on to suggest two main strategies, one for a single user scenario and one for sys-admin types who have to keep TeX in good working order on a multi-user system.

Single user setup For a typical single user situation, I suggest three trees: MAIN, PUBLIC (added-public), and PRIVATE (added-private), and using file-database lookup only for the MAIN tree. The distinction between public and private is simply this: put a file into the PRIVATE tree if it is not publicly released somewhere on the World-Wide Web for all to use. This could include

- packages or fonts that you created yourself
- packages or fonts that were sent to you privately by a colleague
- old versions of packages that are no longer available from CTAN (but if this becomes a major

component of your system you will probably want to add another OLDVERSIONS tree; see Old Versions.

So suppose you install version 1.2 of teTeX . The MAIN tree will be the TDS tree as given in the teTeX distribution, out of the box, with absolutely no changes. The PUBLIC TDS tree will be for any packages that you fetch from CTAN as needed, i.e., whenever you go to use a package and find that it is not present in the MAIN tree. The PRIVATE TDS tree is for any others that are not readily obtainable from the WWW.

If the name of your primary installed TEXMF tree turns out to be `/usr/local/teTeX/texmf`, then put a double bang in front of that tree (and that tree only) when defining `$TEXMF`. Run `mktexlsr` once and for all. (In older versions of teTeX , `mktexlsr` was known as `texhash`.)

I.e., suppose you list the three trees in the `texmf.cnf` file as:

```
TEXMFMAIN = /usr/local/teTeX/share/texmf
TEXMFPUBLIC = /usr/local/texmf/public
TEXMFPRIVATE = /usr/local/texmf/private
```

Then you would define `$TEXMF` as

```
TEXMF =
{ $TEXMFPRIVATE; $TEXMFPUBLIC; !!$TEXMFMAIN }
```

The capabilities of other systems in this respect can vary, of course. The impression I have from looking at the MiKTeX documentation is that multiple TDS trees are supported but at the present time you do not have the option of turning the database lookup on or off for different trees. If this is true then you would have to run the MiKTeX command that updates the database every time you add a new file, regardless of which tree it goes into. (See <http://www.miktex.org/>.)

Notice that this MAIN/PUBLIC/PRIVATE strategy can easily be adapted to a situation where you are working on a multi-user system but you do not have sys-admin privileges to add fonts or packages except in your own personal area. Suppose that the name of your home directory is something like `/home/abc` (because your name is Abercrombie B. Cutpurse, so your initials are abc).

Then the easiest approach is to make your own personal cnf file, `/home/abc/texmf/web2c/texmf.cnf`, containing everything in the standard cnf file but defining the TEXMF trees as follows:

```
TEXMFMAIN = /usr/local/teTeX/share/texmf
TEXMFPUBLIC = /home/abc/texmf/public
TEXMFPRIVATE = /home/abc/texmf/private
```

To ensure that your cnf file will be read instead of the system-wide cnf file, define the environment vari-

able TEXMFCNF in your shell startup file (`.cshrc`, `.profile`, or the equivalent):

```
# for .cshrc:
setenv TEXMFCNF /home/abc/texmf/web2c
```

Multi-user setup For a typical multi-user situation, I suggest a set of TDS trees that is basically analogous to the single-user situation but with two additional considerations. If your company's name is Bingle Publishing Enterprises, then use MAIN and PUBLIC trees as before but define a BINGLE tree instead of PRIVATE. You may also want to consider using a VAR tree and, possibly, an OLD tree.

MAIN The basic TDS tree provided by teTeX

PUBLIC Additional packages not found in MAIN but needed for Bingle work.

BINGLE Additional packages needed for Bingle work but not publicly released (items created in-house or otherwise privately obtained).

VAR This is the VARTEXMF tree alluded to in the `texmf.cnf` file, which is intended to hold format files and font files generated at your site. If you use this tree it is also a good place to put the Bingle version of `texmf.cnf`.

OLD A tree to hold obsolete versions of packages (see **Old Versions**) that may still need to be available for some of your documents.

SELFAUTOPARENT or SELFAUTODIR?

If you look at a real `texmf.cnf` file you will see that the definition of TEXMFMAIN normally refers to some puzzling variables SELFAUTOPARENT or SELFAUTODIR. There are two ideas here:

1. When TeX or another web2c program starts up, it should be able to decide where to look for the cnf file by checking its own location—i.e., if the full path for the `dvips` program is `/usr/local/teTeX/bin/dvips`, then by default `dvips` will look for the cnf file in `/usr/local/teTeX/texmf/web2c`.
2. The full path of programs such as `dvips` may vary according to an installation option: whether to include an extra level that indicates the architecture and operating system for which the programs were compiled.

So look at the location of `kpsewhich`. If it has an architecture/OS specific part after “bin”, use SELFAUTOPARENT, but if it looks like

```
/usr/local/teTeX/bin/kpsewhich
```

then you should be able to use SELFAUTODIR:

```
%TEXMFMAIN = $SELFAUTOPARENT/share/texmf
TEXMFMAIN = $SELFAUTODIR/share/texmf
```

VARTEXMF? VARTEXFONTS?

The option of generating PK files on demand, which has been around for some years now, brings with it some questions about where to put the files that are generated. On a multi-user system, ordinary users cannot normally create files in the main TEXMF tree. The idea of setting up a VARTEXFONTS area for the generated files, which was the earliest approach, was later generalized to the idea of a VARTEXMF tree to hold several kinds of locally generated files: pk files, tfm files, T_EX format files, and configuration files.

The evolution and documentation of this idea was not particularly easy for me to follow, and I believe that if you have a PRIVATE or BINGLE tree it can serve as the location of all the files that would otherwise go into the VARTEXMF tree. But I have not tested this hypothesis extensively enough to rule out the possibility that the name “VARTEXMF” gets special treatment either by the kpathsea library or by some of the t_EX configuration scripts.

For some related information see also the “Filesystem Hierarchy Standard” at <http://www.pathname.com/fhs/>.

Upgrades

Upgrading t_EX Suppose that you hear t_EX 2.0 has been released and you are eager to install it on your system to replace your current version, 1.4. What will your plan of action be?

Why of course you will want to install version 2.0 in an entirely separate tree where it can be tested for a while, so that you feel reasonably sure that t_EX 2.0 will produce the same results (or, possibly, better) with your existing document base.

Searching

This is for a recent t_EX (1.x). Some options might not be searched for earlier versions. For a program such as L^AT_EX, the search path is taken from the first of the following possibilities that is discovered to be non-null:

\$TEXINPUTS_{latex} (env variable)

\$TEXINPUTS (env variable)

TEXINPUTS_{latex} (cnf file)

TEXINPUTS (cnf file)

One way to temporarily change the search path of a given command is thus (for those using sh):

```
TEXINPUTSlatex=$HOME/foo: latex xyz
```

To find out what the order is (for those using sh):

```
KPATHSEA_DEBUG=122 latex xyz
```

MikT_EX

To give a flavor of the differences between T_EX systems when it comes to search paths, here is an example from the MikT_EX documentation: the search path for L^AT_EX.

[L^AT_EX]

```
Input Dirs=.;c:\users\me//
;R\tex\latex//;R\tex\generic//
```

The direct t_EX equivalent would be

```
TEXINPUTSlatex=!!/users/me//
;!!$TEXMF/tex/latex//;!!$TEXMF/tex/generic//
```

although the latter two parts would more usually be combined using brace notation:

```
TEXINPUTSlatex=!!/users/me//
;!!$TEXMF/tex/{latex,generic}//
```

Some miscellaneous notes from <http://www.miktex.org/faq/index.html>:

4.1 Which is the best directory to keep .sty files where MiK_T_EX can find them?

Each new L^AT_EX package should be installed in the folder `tex\latex\package` relative to the local TEXMF folder (usually

`C:\localtexmf\`).

Example: suppose you want to install the package `thesis-ex` which consists of the style file `thesis-es.sty`:

Create a new folder:

```
C:\> mkdir C:\localtexmf\tex\latex\thesis-ex
C:\>
```

Copy all files (only one in our case) to the new folder:

```
C:\> copy thesis-ex.sty
C:\localtexmf\tex\latex\thesis-ex
C:\>
```

Refresh the file name database so that MiK_T_EX finds the new files:

```
C:\> initexmf -u
C:\>
```

The latest release of MiK_T_EX (2.1) appears to have a capability for automatically fetching upgrades from a remote server:

You use MiK_T_EX Update Wizard to update the MiK_T_EX installation.

How It Works

MiK_T_EX Update Wizard opens a remote package repository and reads the package database.

The time-stamps of local packages and remote packages are compared. If a remote package is newer than the corresponding local package, then that package is added to the update list.

FROM THE texmf.cnf FILE:

```
% Earlier entries (in the same or another file) override later ones,
and
% an environment variable foo overrides any texmf.cnf definition of
foo.
%
% All definitions are read before anything is expanded, so you can use
% variables before they are defined.
%
% If a variable assignment is qualified with '.PROGRAM', it is ignored
% unless the current executable (last filename component of argv[0])
is
% named PROGRAM. This foo.PROGRAM construct is not recognized on the
% right-hand side. For environment variables, use FOO_PROGRAM.
%
% Which file formats use which paths for searches is described in the
% various programs' and the kpathsea documentation.
%
% // means to search subdirectories (recursively).
% A leading !! means to look only in the ls-R db, never on the disk.
% A leading/trailing/doubled ; in the paths will be expanded into the
% compile-time default. Probably not what you want.
%
% You can use brace notation, for example: /usr/local/{mytex:othertex}
% expands to /usr/local/mytex:/usr/local/othertex. Instead of the
path
% separator you can use a comma: /usr/local/{mytex,othertex} also
expands
% to /usr/local/mytex:/usr/local/othertex. However, the use of the
comma
% instead of the path separator is deprecated.
%
% The text above assumes the path separator is a colon (:). Non-UNIX
% systems use different path separators, like the semicolon (;).
```

MiKTeX Update Wizard goes through the update list and updates the packages on demand.

Old Versions

The Comprehensive T_EX Archive Network does not attempt to systematically archive old versions of software. At any given time, you may see the previous version and the current version of a particular package, or the current version and a beta version of the next release.

Therefore, if you have a need to run different documents with different versions of a given package, you will need to take it on yourself to keep a copy of all the old versions that you want.

How do you arrange that a particular document gets the appropriate version of a particular package?

Well, if you thought things were complicated before, you're about to see what "really really complicated" looks like.

If the number of different package/version combinations remains relatively small, you can use additional TDS trees for this purpose. Let's suppose, for example, that in addition to the latest release of L^AT_EX you would like to be able to run older releases on demand, one for each of the years from 1996 through 2000.

Set up a separate TDS tree for each release of L^AT_EX:

```
/usr/local/texmf.latex1996
/usr/local/texmf.latex1997
/usr/local/texmf.latex1998
/usr/local/texmf.latex1999
/usr/local/texmf.latex2000
```

Make links in the `teTeX` bin directory so that `latex1996`, etc., can be used from the command line. Then in the `texmf.cnf` file put

```
TEXMFOLD=/usr/local/texmf
TEXINPUTS.latex1996=
    $TEXMFOLD.latex1996/tex/latex//
    :$TEXMF/tex/{latex,generic}//
```

and similarly for the others.

Appendix Q: Questions

1. Where do I put locally generated `.pk` files so that they are accessible to all local users?
2. I made my own Metafont font, where do I put it so that it will be used by `TeX`, preview properly, and print properly?
3. My colleague sent me a copy of his package for doing xyz diagrams, where do I put it so that I can easily use it? So that other people on the same computer system can also use it?
4. I want to upgrade my copy of `LATeX` but without upgrading my entire `TeX` system, what's the best way to go about it?
5. If I want to compound the `.cnf` files, do later settings override earlier, or vice versa?

Appendix R: Remarks

- Programs that use `kpathsea`: `tex`, `metapost`, `bibtex`, `dvips` [`dvipsk`], `xdvi` [`xdvik`], and quite a few others.
- Some programs that don't use `kpathsea`: `ghostscript`, `ghostview`, `latex2html`, `info`, `man`.
- Drawback of fine-toothed TDS tree: It's harder to do total-group operations across the boundaries. Advantage: It's easier to do operations on the natural subsets.
- I like to put the `.tex`, `x.tex`, etc., files from `latex` into the generic tree.
- `kpathsea` changes the generic `;` path separator into the Unix equivalent `:` for output purposes, if running on a Unix system.
- If a directory is listed in `TEXMFDBS` in the `texmf.cnf` then the disk is never searched—I think. Need to be wary of complications here.

Appendix X: `xdvi`

xdvi search paths The preliminary setup:

TEXMFCNF to find `texmf.cnf` and read search path info

TEXMFDBS to search for `ls-R` database files

MIMELIBDIR for mime types

MAILCAPDIR for mailcap file

For dealing with a particular font (e.g., `cmtt10`):

VFFONTS search for `cmtt10.vf`

MKTEXPK use `mktexpk` script? 0 false, 1 true

PKFONTS search for `cmtt10.300pk`

Generating PK files The config files that affect the generation of PK files are

```
...teTeX/texmf/web2c/texmf.cnf
```

and

```
...teTeX/fontname/*
```

(Note that the latter is outside of the `texmf` hierarchy.) In particular, if the MF sources were found in one `texmf` tree, the PK files are written into that same `texmf` tree. If that `TEXMF` tree is not writable, the generated fonts go into `VARTEX-FONTS`. The file in the `fontname` directory control which subdirectory of the PK tree the `fPK` files go into.