Configuration options for LaTeX 2_ε

29 August 2001

Contents

Configuring LaTeX

Since one of the main aims of the new standard LaTeX is to give all users the freedom provided by a reliable document processing system linked to a highly portable document format, the number of configuration possibilities is strictly limited. The reasons for this are explained in more detail in the article *Modifying LaTeX* in the file `modguide.tex`. An important consequence of this is that any document that relies on any extension package must declare this package within the document file; this helps to ensure that the document will work at a different site, where the LaTeX system may be configured differently.

Local configuration options are, by convention, placed in 'configuration files', which have extension `.cfg`. This document describes the possibilities for configuration in this release of LaTeX; it also explains how to configure the font definition files to take advantage of the available fonts.

The last section considers briefly how to proceed if you require further customisation of the formatter.

System configuration

texsys.cfg

This is the only configuration file that *must* be present. During installation, if LaTeX cannot find a file with this name then a default file `texsys.cfg`, consisting entirely of comments, is written out and used. Note that, until this file has been read, LaTeX is not able to test reliably whether a given file exists on the system.

The contents of the file `texsys.cfg` allow LaTeX to cope with various differences between the behaviours of different TeX systems, mainly in relation to file handling. The default version of this file contains, in its comments, possible settings that may be needed for a range of TeX systems. For more information, typeset the file `ltdirchk.dtx`.

If you have copied your LaTeX installation from a computer that used a different operating system then you may well have a version of `texsys.cfg` that will make it difficult to install LaTeX on your system. If this happens then start the process again with an empty `texsys.cfg` file; this will produce an installation that should, at least, allow you to typeset the documentation. However, it is possible that LaTeX can still find only those files that are in the current directory; in this case you must set the macro `\input@path` correctly for your system.

Configuring the LaTeX format

There are four configuration files that enable personal preferences to be incorporated into the LaTeX format file `latex.fmt`. The range of preferences that can be configured by these files is strictly limited as this helps to ensure document portability.

All four files work in the same way: if the file ⟨*file*⟩`.cfg` is found, it will be input by iniTeX; otherwise a default file ⟨*file*⟩`.ltx` will be input; this is sometimes done via a minimal ⟨*file*⟩`.cfg` that simply inputs ⟨*file*⟩`.ltx`. Thus, providing your own version of any of these `.cfg` files can completely override any settings in the corresponding default standard `.ltx` file.

Font configuration

Before you even think about configuring the font declarations by producing a file `fontmath.cfg` or `fonttext.cfg`, you should read the documented file `fontdef.dtx`. This is the source file from which the default files `fonttext.ltx` and `fontmath.ltx` are produced; it contains information concerning the contents of the default files and what sort of customisation is possible. In particular, it describes in detail the effects of individual customisations on document portability including: which customisations can be made without endangering the ability to exchange documents with other sites (even if the formatting differs); and which things should be left untouched because they will make your system so different from others that the documents it produces will be non-portable.

WARNING Please note that use of either of these font configuration files has the following consequences.

- Since the content of the file `fontdef.dtx` *might* change in the future, anyone writing a font configuration file must be prepared to update it for use with future releases.
- Documents produced on your system are likely, at best, to be portable only in the sense of being processable at a different site—the actual formatting will not be the same if different fonts are used.
- The LaTeX3 project team will not be able to support you in diagnosing problems if these cannot be reproduced with a format that does not use any configuration files.

fonttext.cfg

The file `fonttext.cfg` can contain declarations relating to the use of fonts in text modes.

If it exists, it defines which font shapes, families and encodings are normally used in text mode, as well as the behavior of font attribute commands such as `\textbf` etc.

It could be used, for example, to produce a LaTeX format that, by default, typesets documents using Times fonts. Be warned, however, that such customisation can have unfortunate consequences; so please read carefully this section and the file `fontdef.dtx` below if you are thinking of doing this.

Please note carefully the above **warning**.

fontmath.cfg

The file `fontmath.cfg` can contain declarations relating to the use of fonts in math mode.

If it exists, it defines which fonts in which sizes are used in math mode, and how they are used. It also defines all the math mode commands that 'are likely to' depend on the choice of math fonts used (e.g. commands that depend on the position of a glyph in a math font).

The main reason for the existence of this file is to provide for future updates when a standard math font encoding is available. Right now we do *not* encourage the use of this configuration file other than for special applications. Writing a proper configuration file for math mode needs expert knowledge!

Please note carefully the above **warning**.

preload.cfg

The contents of the file `preload.cfg` can control the preloading of commonly used fonts. Preloading fonts speeds up the processing of documents but, because fonts cannot be 'unloaded', you should not preload too many; otherwise you may be unable to process documents requiring unusual font families.

The default file `preload.ltx` is produced from `preload.dtx`. It loads only a few fonts and these are a good choice if you normally use documents at the default, 10 pt, size. If you normally use 11 pt or 12 pt then the time for LaTeX to startup may be noticeably decreased if you preload the corresponding fonts for the sizes you use. Similarly, if you normally use a different font family, for example Times Roman (`ptm`) then you may want to preload fonts in this family rather than the default Computer Modern fonts.

Hyphenation configuration

hyphen.cfg

In order to hyphenate text, TeX must have hyphenation patterns and, since these patterns can be loaded only by iniTeX, the choice of which patterns to load must be made when the format is created.

The hyphenation patterns for American English are stored in the file named `hyphen.tex`; LaTeX 2.09 always loaded this file when its format was made.

With LaTeX 2_ε it is possible to configure which hyphenation patterns are to be loaded into the format. When iniTeX is processing `latex.ltx`, it looks for a file called `hyphen.cfg`; this file can be used to control which hyphenation patterns are loaded. If a file `hyphen.cfg` cannot be found then iniTeX will load the file `hyphen.ltx`.

The file `hyphen.ltx` loads the file `hyphen.tex` if it can find it; otherwise it stops with an error since a format with no hyphenation patterns is not very useful. It then sets `\language=0` and it sets the values `\lefthyphenmin=2` and `\righthyphenmin=3`, which are needed for American English.

Thus, if you want any other patterns to be loaded then you should create a file `hyphen.cfg`. For each language for which you wish to load hyphenation patterns this file should:

- set `\language=`⟨*number*⟩;
- load the file which contains the hyphenation patterns for that language.

If the patterns you use require some definitions or assignments then a group should be used to keep such changes local to their file.

Note. The hyphenation files that are read in should *only* set the hyphenation tables for the language, using the commands `\hyphenation` and `\patterns`. In particular they should make no assignments to the lowercase/uppercase tables (`\lccode` and `\uccode`) and should not make any global command definitions to be used after the file has been read. Unfortunately some older hyphenation files do contain such settings; thus they are *incompatible* with the mechanisms LaTeX uses to ensure independence of input and output encodings.

After this the file `hyphen.cfg` should:

- set `\language` to its default value;
- set `\lefthyphenmin` and `\righthyphenmin` to the correct values for this default language.

There are packages available, such as 'french', that can help you with this configuration. The 'babel' collection contains many examples of setting up a multilingual LaTeX format. The documentation in `lthyphen.dtx` (the source file for `hyphen.ltx`) also contains some useful examples.

[We intend in a future release of LaTeX to provide a set of standard commands for use in configuring hyphenation.]

Configuring the font definition files

If you have special fonts available (or if some fonts are unavailable) at your site then you may need to produce customised versions of the font definition files; these have extension `.fd` and are read by LaTeX to obtain information about the font files installed at your system and when to load them.

Although we do not encourage such customisation, you will find information about the content of these files and its syntax in the documented source file `cmfonts.fdd` and *LaTeX 2ε font selection* in the file `fntguide.tex`. [We hope to be able to provide further information and examples on this subject at some time in the future.]

Please note that the use of customised font definition files has the following consequences.

- Documents produced on your system will, at best, to be portable only in the sense of being processable at a different site—the actual formatting will not be the same if different fonts are used.
- The LaTeX3 project team will not be able to support you in diagnosing problems if these cannot be reproduced with a format that does not use any customised font definition files.

Note to system administrators

If you install a version of LaTeX with a locally configured font set-up then this system is likely to produce documents that are no longer 'formatting compatible'; for example, the use of different default fonts will most likely produce different line and page breaks. If you do install, on a multi-user system, a system that is configured in such a way that it is not 'formatting compatible' then you should consider carefully the needs of users who need to create portable documents. A good way to provide for their needs is to make available, in addition, a standard form of LaTeX without any 'formatting incompatible' customisations.

Configuring compatibility mode

When processing documents that begin with `\documentstyle`, LaTeX 2ε tries to emulate the old LaTeX 2.09 system as far as possible.

latex209.cfg

Whenever a LaTeX document starts with `\documentstyle`, rather than with `\documentclass`, LaTeX assumes that it is a LaTeX 2.09 document and therefore processes it in 'compatibility mode'. This does the following:

- sets the flag `\@compatibilitytrue`;
- inputs the file `latex209.def`;
- inputs the file `latex209.cfg` if it exists.

The LaTeX 2.09 set-up allowed the format itself to be customised. When making the format with iniTeX, the process ended with this request:

```
Input any local modifications here.
```

If your site did input any modifications at that point then the LaTeX 2_ε 'compatibility mode' will not fully emulate LaTeX 2.09 *as installed at your site*. In this case you should put all these 'local modifications' into a file called `latex209.cfg` and put this file in the default input path at your site. These 'local modifications', although not stored in the format, will then be loaded before any old-style document is processed. This should ensure that you can continue to process any old documents that made use of this local customisation.

Configuration files for standard packages and classes

Most of the packages in the distribution do not have any associated configuration files. The exceptions are listed here.

sfonts.cfg

The file `sfonts.cfg` can contain declarations relating to the use of fonts in the slides class. If it exists, it is read instead of the file `sfonts.def`.

Please note that use of this configuration file has the following consequences.

- Since the font set-up for slides has not yet been revised to fit modern usage, the content of this file should be completely updated sometime. Thus anyone writing such a configuration file must be prepared to update it for use with future releases.
- Documents are portable only in the sense of being processable at a different site—the actual formatting will not be the same if different fonts are used.
- The LaTeX3 project team will not be able to support you in diagnosing problems if these cannot be reproduced with a format that does not use this configuration file.

ltnews.cfg

The file `ltnews.cfg` can be used to customise some aspects of the behaviour of the ltnews class; this class is used to typeset the the one page newsletter accompanying every LaTeX distribution. If this file is present then it is read in at the beginning of the file `ltnews.cls`.

ltxdoc.cfg

The file `ltxdoc.cfg` can be used to customise some aspects of the behaviour of the ltxdoc class; this class is used to typeset the documented code in the `.dtx` files. If this file is present then it is read in at the beginning of the file `ltxdoc.cls`.

As this file is read before the article class is loaded, you may pass options to article. For example the following line might be added to `ltxdoc.cfg` to format the documentation for A4 paper instead of the default US letter paper size.

```
\PassOptionsToClass{a4paper}{article}
```

You should note however, that even if paper size options are specified, the ltxdoc class always sets the `\textwidth` parameter to 355 pt, to enable 72 columns of text to appear in the verbatim code listings. If you really need to over-ride this you could use:

```
\AtEndOfClass{\setlength{\textwidth}{ ...}}
```

To set the `\textwidth` to your desired value at the end of the ltxdoc class.

By default, most of the `.dtx` documented code files in the distribution will produce a 'description' section followed by full source listing of the package. If you wish to suppress the source listings you may add the following line to `ltxdoc.cfg`:

```
\AtBeginDocument{\OnlyDescription}
```

The documentation of the ltxdoc package, which may be typeset from the file `ltxdoc.dtx`, contains more examples of the use of this configuration file.

ltxguide.cfg

The class ltxguide is used by the 'guide' documents, such as this document, in the LaTeX distribution. A configuration file `ltxguide.cfg` may be used with this class in a way very similar to the customisation of the ltxdoc class described in the previous section.

Configuration for other supported packages

The 'graphics' bundle of packages needs two configuration files, primarily to specify the driver used to process the `.dvi` file that LaTeX produces. More documentation on these files comes with the graphics bundle but we mention them here for completeness.

graphics.cfg

Normally this file just specifies a default option, by calling `\ExecuteOptions`, for example `\ExecuteOptions{dvips}` or `\ExecuteOptions{textures}`.

This file is read by the graphics package, and so affects all the packages in the bundle that are based on graphics: graphicx, epsfig, lscape.

color.cfg

Normally this file is identical to `graphics.cfg`. It specifies the default driver option for the color package.

Non-standard versions

If you feel the need to make a version of LaTeX that differs from the standard version in ways that are not possible using the above configuration possibilities, then you should first read *Modifying LaTeX* in the file `modguide.tex`; this will probably make you realise that you do not have any such need.

Thus we are sure that you will never need to create a non-standard version and, even if you do create one, we hope that you will not distribute such a version. Nevertheless, you are permitted to do this provided you take great care to do the following:

- respect the conditions in legal.txt and individual files regarding modification of files and changing the name;
- change all the relevant '`\typeout` banners': i.e. those produced by all the non-standard files in your version and by the format;
- ensure that the method used to run your version is clearly distinguished from that used to run standard LaTeX; e.g. by using a command name or menu entry that is clearly different from `latex` (or `LaTeX` etc).

Examples

Since we have been prompted, despite our misgivings, to document how to do this by members of the League for Programming Freedom, it seems appropriate to describe here a possible modification of LaTeX to produce a system called fsfTeX.

To do this, you should create a file called `fsftex.tex` and then run it using iniTeX and the standard LaTeX format.

The contents of the file `fsftex.tex` should be as shown on page 11. The particular changes to the LaTeX kernel that you wish to make need to be added to the file at the position indicated. You can also choose the extensions you want to use for the class and package files in your system.

```
% fsftex.tex
%
% iniTEX Source code to make a ‘fsftex’ format.
%
% To make this format on Unix:
%
%   initex \&latex fsftex
%
% Then to run the format on file.tex:
%
%   tex &fsftex file
%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% *** VERY IMPORTANT!!! ***
% Change the typeout banner so users know that they
%       are NOT running Standard LaTeX.
\everyjob{\typeout{fsfTeX 1.0 based on LaTeX2e \fmtversion}}
\makeatletter

% fsfTeX changes some LaTeX internals:
%   ... put what you like here ...
\def \fsf@xxxx {Some arbitrary \emph{freely modifiable} code goes here}

% fsfTeX class files have extension .fcl (this week):
\def \@clsextension {fcl}

% fsfTeX package files have extension .fsy:
\def \@pkgextension {fsy}

% Change the file handling so that when a fsfTeX package or class
% is not available, the standard LaTeX file will be read.
%
% For example, \documentclass{article} will load article.fcl if such
% a file exists, but article.cls otherwise.  This allows arbitrary
% processing on ‘article’ documents without changing the standard
% article.cls file.

\let\fsf@missingfileerror\@missingfileerror

\def\@missingfileerror#1#2{%
  \ifx #2\@clsextension
    \InputIfFileExists {#1.cls}%
      {\wlog {fsfTeX: loading #1.cls rather than #1.#2.}}%
      {\fsf@missingfileerror {#1}{#2}}%
  \else
    \ifx #2\@pkgextension
      \InputIfFileExists {#1.sty}%
        {\wlog {fsfTeX: loading #1.sty rather than #1.#2.}}%
        {\fsf@missingfileerror {#1}{#2}}%
    \else
      \fsf@missingfileerror {#1}{#2}%
    \fi
  \fi
}

\makeatother
\dump
```