The file `ltxdoc.dtx` for use with LaTeX 2ε.[*]
It contains the code for `ltxdoc.cls`

David Carlisle

2021/12/07

This file is maintained by the LaTeX Project team.
Bug reports can be opened (category `latex`) at
https://latex-project.org/bugs.html.

1 Documentation of the LaTeX sources

This class file is designed for documenting the LaTeX source files. You may however find it generally useful as a class for typesetting the documentation of files produced in 'doc' format.

Each documented file in the standard distribution comes with extension `dtx`. The appropriate class package or initex file will be extracted from the source by the docstrip system. Each `dtx` file may be directly processed with LaTeX 2ε, for example

```
latex2e docclass.dtx
```

would produce the documentation of the Class and package interface.

Each file that is used in producing the LaTeX 2ε format (ie not including the standard class and packages) will be printed together in one document if you LaTeX the file `sources2e.tex`. This has the advantage that one can produce a full index of macro usage across all the source files.

If you need to customise the typesetting of any of these files, there are two options:

- You can use DOCSTRIP with the module 'driver' to extract a small LaTeX file that you may edit to use whatever class or package options you require, before inputting the source file.
- You can create a file `ltxdoc.cfg`. This configuration file will be read whenever the `ltxdoc` class is used, and so can be used to customise the typesetting of all the source files, without having to edit lots of small driver files.

The second option is usually more convenient. Various possibilities are discussed in the next section.

[*] This file has version number v2.1d, dated 2021/12/07.

2 Customisation

The simplest form of customisation is to pass more options to the `article` class which is loaded by `ltxdoc`. For instance if you wish all the documentation to be formatted for A4 paper, add the following line to `ltxdoc.cfg`:

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

All the source files are in two parts, separated by `\MaybeStop`. The first part (should) contain 'user' documentation. The second part is a full documented listing of the source code. The `doc` package provides the command `\OnlyDescription` which suppresses the code listings. This may also be used in the configuration file, but as the `doc` package is read later, you must delay the execution of `\OnlyDescription` until after the `doc` package has been read. The simplest way is to use `\AtBeginDocument`. Thus you could put the following in your `ltxdoc.cfg`.

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

If the full source listing `sources2e.tex` is processed, then an index and change history are produced by default, however indices are not normally produced for individual files.

As an example, consider `ltclass.dtx`, which contains the sources for the new class and package interface commands. With no `cfg` file, a 19 page document is produced. With the above configuration a slightly more readable document (4 pages) is produced.

Conversely, if you really want to read the source listings in detail, you will want to have an index. Again the index commands provided by the doc package may be used, but their execution must be delayed.

```
\AtBeginDocument{\CodelineIndex\EnableCrossrefs}
\AtEndDocument{\PrintIndex}
```

The `doc` package writes index files to be sorted using MakeIndex with the `gind` style, so one would then use a command such as

```
makeindex -s gind.ist ltclass.idx
```

and re-run LaTeX.

Similarly to print a Change history, you would add

```
\AtBeginDocument{\RecordChanges}
\AtEndDocument{\PrintChanges}
```

to `ltxdoc.cfg`, and use MakeIndex with a command such as

```
makeindex -s gglo.ist -o ltclass.gls ltclass.glo
```

Finally if you do not want to list all the sections of `source2e.tex`, you can use `\includeonly` in the `cfg` file:

```
\includeonly{ltvers,ltboxes}
```

3 Options

```
⟨*class⟩
```

```
\DeclareOption{a5paper}{\@latexerr{Option not supported}%
   {}}
```

Prevent loading of a config file.

```
\newif\ifltxdoc@load@cfg@ \ltxdoc@load@cfg@true
\DeclareOption{nocfg}{\ltxdoc@load@cfg@false}
```

```
\DeclareOption*{%
    \PassOptionsToClass  {\CurrentOption}{article}}
```

4 Option Processing

```
\ProcessOptions
```

5 Local configuration

Input a local configuration file, if it exists.

```
\ifltxdoc@load@cfg@
\InputIfFileExists{ltxdoc.cfg}
           {\typeout{*************************************^^J%
                     * Local config file ltxdoc.cfg used^^J%
                     *************************************}}
           {}
\else
   \typeout{*************************************^^J%
            * Local config file ignored^^J%
            *************************************}
\fi
```

6 Loading article and doc

```
\LoadClass{article}
```

```
\RequirePackage{doc}
```

Make | be a ‘short verb’ character, but not in the document preamble, where an active character may interfere with packages that are loaded.

```
\AtBeginDocument{\MakeShortVerb{\|}}
```

As ‘doc’ documents tend to have a lot of monospaced material, Set up some `tt` substitutions to occur silently.

```
\DeclareFontShape{OT1}{cmtt}{bx}{n}{<-> ssub * cmtt/m/n}{}
\DeclareFontFamily{OMS}{cmtt}{\skewchar\font 48}  % ’60
\DeclareFontShape{OMS}{cmtt}{m}{n}{<-> ssub * cmsy/m/n}{}
\DeclareFontShape{OMS}{cmtt}{bx}{n}{<-> ssub * cmsy/b/n}{}
```

This substitution is in the standard fd file, but not silent.

```
\DeclareFontShape{OT1}{cmss}{m}{it}{<->ssub*cmss/m/sl}{}
```

```
\CodelineNumbered
\DisableCrossrefs
```

Increase the text width slightly so that with the standard fonts 72 columns of code may appear in a `macrocode` environment.

```
\setlength{\textwidth}{355pt}
```

Increase the marginpar width slightly, for long command names. And increase the left margin by a similar amount

```
\addtolength\marginparwidth{30pt}
\addtolength\oddsidemargin{20pt}
\addtolength\evensidemargin{20pt}
```

```
\setcounter{StandardModuleDepth}{1}
```

7 Useful abbreviations

`\cmd{\foo}` Prints `\foo` verbatim. It may be used inside moving arguments. It can *not* be use to record commands that are defined as "`\outer`" nor is it possible to use it on conditionals such as `\iftrue` or defined by `\newif`. `\cs{foo}` already available with the `doc` package also prints `\foo`, for those who prefer that syntax. (This second form can be used to record all types of command so the above restrictions do not apply.

`\cmd`
`\cs`

```
\def\cmd#1{\cs{\expandafter\cmd@to@cs\string#1}}
\def\cmd@to@cs#1#2{\char\number`#2\relax}
%\DeclareRobustCommand\cs[1]{\texttt{\char`\\#1}}     % already part of doc.sty
```

`\marg` `\marg{text}` prints `{`⟨*text*⟩`}`, 'mandatory argument'.

```
\providecommand\marg[1]{%
  {\ttfamily\char`\{}\meta{#1}{\ttfamily\char`\}}}
```

`\oarg` `\oarg{text}` prints `[`⟨*text*⟩`]`, 'optional argument'.

```
\providecommand\oarg[1]{%
  {\ttfamily[}\meta{#1}{\ttfamily]}}
```

`\parg` `\parg{te,xt}` prints `(`⟨*te,xt*⟩`)`, 'picture mode argument'.

```
\providecommand\parg[1]{%
  {\ttfamily(}\meta{#1}{\ttfamily)}}
```

8 Old Comments

The LaTeX 2$_\varepsilon$ sources contain a lot of code inherited from LaTeX2.09. The comments in this code were not designed to be typeset, and do not contain the necessary LaTeX markup. The `oldcomments` environment typesets these comments, automatically sensing when any control sequence appears, and implicitly adding the `\verb`. This procedure does not produce particularly beautiful pages, but it allows us to fully document new sections, and have some form of typeset comments on all the old code.

Scan control names and put them in tt. Will actually (incorrectly) scan past `\\` but this does not matter as this is almost never followed by a letter in practice. (ie `\\foo`) would put `foo` in `\ttfamily`.

```
\def\oc@scan#1{%
  \ifx\oc@bslash#1%
                        \egroup\let\next\oc@bslash\else
  \ifcat a\noexpand#1%
```

```
                      #1\let\next\oc@scan\else
  \ifx\oc@percent#1%
                      \def\next{\char`\%\egroup}%
  \else
                      #1\let\next\egroup
  \fi\fi\fi\next}
\def\oc@bslash{\bgroup\oc@ttf\char`\\\oc@scan}%
\def\oc@verb#1{%
  \catcode`#1\active
  \uccode`\~`#1%
  \uppercase{\def~{{\oc@ttf\char`#1}}}}
\begingroup
  \obeyspaces%
  \catcode`\/=\catcode`\\
  /catcode`/\/active
  /catcode`<=/catcode`{%
  /catcode`>=/catcode`}%
  /catcode`/{/active%
  /catcode`/}/active%
  /gdef/oldc< \end{oldcomments}>%
  /gdef/begmac<    \begin{macrocode}>%
  /gdef/obs</def <</oc@ttf/ >>>%
/endgroup%
\begingroup
  \catcode`\/=\catcode`\\
  \catcode`\\=13
  /catcode`/|=/catcode`/%
  /catcode`/%=13
  /gdef/oldcomments{|
    /makeatletter
    /let/do/oc@verb/dospecials
    /frenchspacing/@vobeyspaces/obs
    /raggedright
    /oc@verb/>|
    /oc@verb/<|
    /let\/oc@bslash
    /let%/oc@percent
    /obeylines
    /parindent/z@
    /ttfamily/expandafter/let/expandafter/oc@ttf/the/font
    /rmfamily
    /textit{Historical /LaTeX/,2.09 comments (not necessarily accurate any more):}
    /hfuzz/maxdimen
    }
/endgroup
\begingroup
  \sloppy%
  \obeylines%
  \gdef\oc@percent#1^^M{%
    \ifvmode%
    \def\commentline{#1}%
    \ifx\commentline\oldc%
    \textit{End of historical \LaTeX\,2.09 comments.}
    \end{oldcomments}%
    \else%
    \ifx\commentline\begmac%
    \begin{macrocode}%
    \else%
    \leavevmode%
    #1^^M%
    \fi\fi%
    \else%
    {\oc@ttf\char`\%}#1^^M%
    \fi}%
\endgroup%
```

9 DocInclude

```
\@addtoreset{CodelineNo}{part}
```

\DocInclude More or less exactly the same as \include, but uses \DocInput on a dtx file, not \input on a tex file.

```
\def\partname{File}
```

```
\newcommand*{\DocInclude}[1]{%
  \relax
  \clearpage
  \docincludeaux
  \IfFileExists{#1.fdd}%
    {\def\currentfile{#1.fdd}}%
    {\def\currentfile{#1.dtx}}%
  \ifnum\@auxout=\@partaux
    \@latexerr{\string\include\space cannot be nested}\@eha
  \else
    \set@curr@file{#1}%
    \edef\@curr@file{\@strip@tex@ext\@curr@file}%
    \expandafter\@docinclude\expandafter{\@curr@file}
 \fi}
\def\@docinclude#1 {\clearpage
```

```
\if@filesw \immediate\write\@mainaux{\string\@input{#1.aux}}\fi
\@tempswatrue\if@partsw \@tempswafalse\edef\@tempb{#1}\@for
\@tempa:=\@partlist\do{\ifx\@tempa\@tempb\@tempswatrue\fi}\fi
\if@tempswa \let\@auxout\@partaux \if@filesw
\immediate\openout\@partaux "#1.aux"
\immediate\write\@partaux{\relax}\fi
\@filehook@set@CurrentFile
```

We need to save (and later restore) various index-related commands which might be changed by the included file.

```
\let\@ltxdoc@PrintIndex\PrintIndex
\let\PrintIndex\relax
\let\@ltxdoc@PrintChanges\PrintChanges
\let\PrintChanges\relax
\let\@ltxdoc@theglossary\theglossary
\let\@ltxdoc@endtheglossary\endtheglossary
\part{\currentfile}%
```

```
  {\let\ttfamily\relax
  \xdef\filekey{\filekey, \thepart={\ttfamily\currentfile}}}%
\DocInput{\currentfile}%
\let\PrintIndex\@ltxdoc@PrintIndex
\let\PrintChanges\@ltxdoc@PrintChanges
\let\theglossary\@ltxdoc@theglossary
\let\endtheglossary\@ltxdoc@endtheglossary
\clearpage
\@writeckpt{#1}\if@filesw \immediate\closeout\@partaux \fi
\else\@nameuse{cp@#1}\fi\let\@auxout\@mainaux}
```

```
\gdef\codeline@wrindex#1{\if@filesw
```

Set `\protect` to a suitable value in the index entries (we can't use `\set@display@protect` as that would result in different number of spaces after a command depending on the number of expansion happening prior to writing the index).

```
      \begingroup
        \let\protect\noexpand
        \immediate\write\@indexfile
            {\string\indexentry{#1}%
            {\filesep\number\c@CodelineNo}}%
      \endgroup\fi}
```

```
\let\filesep\@empty
```

`\aalph` Special form of `\alph` as currently `source2e.tex` includes more than 26 files .

```
\def\aalph#1{\@aalph{\csname c@#1\endcsname}}
\def\@aalph#1{%
  \ifcase#1\or a\or b\or c\or d\or e\or f\or g\or h\or i\or
         j\or k\or l\or m\or n\or o\or p\or q\or r\or s\or
         t\or u\or v\or w\or x\or y\or z\or A\or B\or C\or
         D\or E\or F\or G\or H\or I\or J\or K\or L\or M\or
         N\or O\or P\or Q\or R\or S\or T\or U\or V\or W\or
         X\or Y\or Z\else\@ctrerr\fi}
```

`\docincludeaux`

```
\def\docincludeaux{%
  \def\thepart{\aalph{part}}\def\filesep{\thepart-}%
  \let\filekey\@gobble
  \g@addto@macro\index@prologue{%
    \gdef\@oddfoot{\parbox[t]{\textwidth}{\strut\footnotesize
       \raggedright{\bfseries File Key:} \filekey}}%
    \let\@evenfoot\@oddfoot}%
  \global\let\docincludeaux\relax
 \gdef\@oddfoot{%
   \expandafter\ifx\csname ver@\currentfile\endcsname\relax
    File \thepart: {\ttfamily\currentfile} %
   \else
    \GetFileInfo{\currentfile}%
    File \thepart: {\ttfamily\filename} %
    Date: \filedate\ %
    Version \fileversion
    \fi
    \hfill\thepage}%
 \let\@evenfoot\@oddfoot}%
```

\MaintainedByLaTeXTeam Generate boilerplate reference to bug database.

```
\def\MaintainedBy#1{\gdef\@maintainedby{#1}}
```

```
\let\@maintainedby\@empty
```

```
\def\MaintainedByLaTeXTeam#1{%
{\gdef\@maintainedby{%
This file is maintained by the \LaTeX{} Project team.\\%
Bug reports can be opened (category \texttt{#1}) at\\%
\url{https://latex-project.org/bugs.html}.}}}
```

```
\def\@maketitle{%
  \newpage
  \null
  \vskip 2em%
  \begin{center}%
  \let \footnote \thanks
    {\LARGE \@title \par}%
    \vskip 1.5em%
    {\large
      \lineskip .5em%
      \begin{tabular}[t]{c}%
        \@author
      \end{tabular}\par}%
    \vskip 1em%
    {\large \@date}%
    \ifx\@maintainedby\@empty
    \else
    \vskip 1em%
    \fbox{\fbox{\begin{tabular}{@{}l@{}}\@maintainedby\end{tabular}}}%
    \fi
  \end{center}%
  \par
  \vskip 1.5em}
```

\url

```
\providecommand\url{\texttt}
```

```
\def\task#1#2{}
⟨/class⟩
```

10 Configuration file

```
⟨*cfg⟩
%
% This is the ltxdoc configuration file we use to format the LaTeX
% kernel sources.
%
%
% Copyright 2006, 2007, 2011 Heiko Oberdiek
% Copyright 2014-2021 The LaTeX Project
%

\ProvidesFile{ltxdoc.cfg}%
  [2021/12/07 v2.0c ltxdoc.cls configuration (LaTeX Project)]
```

```
\PassOptionsToClass{a4paper}{article}

% hyperref and hypdoc are now loaded late (or by the user) so we have to wait
% with any adjustments until that has happened

\AddToHook{package/hyperref/after}{%
%%  \RequirePackage{hypdoc}%       % this is now triggered by doc
  \RequirePackage{pdftexcmds}\relax
  \ifnum\pdf@strcmp{\jobname}{inputenc}=0 %
    \hypersetup{pdfencoding=auto}%
    \pdfstringdefDisableCommands{%
      \def\meta#1{% inputenc.dtx
        \9060\010#1\9060\011%
      }%
    }%
  \else
  \fi
  \pdfstringdefDisableCommands{%
    \let\env\relax % longtable.dtx
    \let\mytt\relax % tabularx.dtx
  }%
}

% This should work well for documentation of packages outside the
% LaTeX kernel, but if not, you can prevent the loading with the
% option "nocfg", i.e.,
%
%   \documentclass[nocfg]{ltxdoc}
%
% or by providing your own config file

\endinput
⟨/cfg⟩
```