pLaTeX 2ε について

中野 賢 & 日本語 TeX 開発コミュニティ

作成日：2018/04/06

注意：

この文書では「コミュニティ版 pLaTeX 2ε」について簡単に説明します。株式会社アスキー（現アスキー・メディアワークス[1]）が配布している pLaTeX 2ε（以下、アスキー版 pLaTeX 2ε）とは異なる内容ですので、注意してください。

2010 年以降、アスキー pTeX は、国際的に広く使われている TeX Live というディストリビューションに取り込まれ、そこで独自の改良や仕様変更が加えられてきました。最近の TeX Live や W32TeX では、pLaTeX も元々の pTeX ではなく、その拡張版 ε-pTeX をエンジンとして用いるようになっています。また、pLaTeX のベースである LaTeX も更新が進められています。

こうした流れにあわせた新しい pLaTeX として、アスキー版から fork して日本語 TeX 開発コミュニティ (Japanese TeX Development Community) が配布しているものが、コミュニティ版 pLaTeX です。開発中の版は GitHub のリポジトリ[2]で管理しています。コミュニティ版 pLaTeX はアスキー版とは異なりますので、バグレポートはアスキー宛てではなく、日本語 TeX 開発コミュニティに報告してください。TeX Forum や GitHub の Issue システムが利用できます。

この文書 (platex.pdf) はコミュニティ版 pLaTeX の概要を説明したものですが、内容はアスキー版（1995 年頃）からほとんど変わっていませんので、今では歴史的な文書ということにしておきます。最近の pLaTeX の更新内容は pLaTeX ニュース（アスキー版：plnews*.pdf、コミュニティ版：plnewsc*.pdf）を参照してください。また、実際の pLaTeX のソースコードは pldoc.pdf で説明しています。

[1] アスキー日本語 TeX（pTeX）、http://ascii.asciimw.jp/pb/ptex/

[2] https://github.com/texjporg/platex

1　この文書について

この文書は pLaTeX 2$_\varepsilon$ の概要を示していますが、使い方のガイドではありません。pLaTeX 2$_\varepsilon$ の機能全般については、[1] を参照してください。また、[2] で説明されていた縦組向けの拡張コマンドについては、`pldoc.pdf` の中の `plext.dtx` の項目を参照してください。

日本語の組版処理については、pTeX（あるいはその前身の「日本語 TeX」）に関する文献 [3] や [4]（英語）, [5]（英語）も併せてご参照ください。

LaTeX の機能については、[7] や [9] などを参照してください。新しい機能については `usrguide.tex` を参照してください。

この文書の構成は次のようになっています。

第 1 節　この節です。この文書についての概要を述べています。

第 2 節　pLaTeX 2$_\varepsilon$ で拡張した機能についての概要です。付属のクラスファイルやパッケージファイルについても簡単に説明しています。

第 3 節　現在のバージョンの pLaTeX と旧バージョン、あるいは元となっている LaTeX との互換性について述べています。

付録 A　この文書ソース (platex.dtx) の DOCSTRIP のためのオプションについて述べています。

付録 B　pLaTeX 2$_\varepsilon$ の dtx ファイルをまとめて、一つのソースコード説明書にするための文書ファイルの説明をしています。

付録 C　付録 B で説明した文書ファイルを処理する sh スクリプト（手順）、DOCSTRIP 文書ファイル内の入れ子の対応を調べる perl スクリプトなどについて説明しています。

2　pLaTeX 2$_\varepsilon$ の機能について

pLaTeX 2$_\varepsilon$ が提供するファイルは、次の 3 種類に分類することができます。

- フォーマットファイル
- クラスファイル
- パッケージファイル

フォーマットファイルには、基本的な機能が定義されており、pLaTeX 2$_\varepsilon$ の核となるファイルです。このファイルに定義されているマクロは、実行時の速度を高めるために、あらかじめ TeX の内部形式の形で保存されています。

クラスファイルは文書のレイアウトを設定するファイル、パッケージファイルはマクロの拡張を定義するファイルです。前者は \documentclass コマンドを用いて読み込み、後者は \usepackage コマンドを用いて読み込みます。

古い *pLaTeX 2.09* ユーザへの注意：
クラスファイルとパッケージファイルは、従来、スタイルファイルと呼ばれていたものです。LaTeX 2ε ではそれらを、レイアウトに関するものをクラスファイルと呼び、マクロの拡張をするものをパッケージファイルと呼んで区別するようになりました。

2.1 フォーマットファイル

pLaTeX のフォーマットファイルを作成するには、ソースファイル “platex.ltx” を ε-pTeX の INI モードで処理します[3]。ただし、TeX Live や W32TeX ではこの処理を簡単にする fmtutil-sys あるいは fmtutil というプログラムが用意されています。以下を実行すれば、フォーマットファイル platex.fmt が作成されます。

```
fmtutil-sys --byfmt platex
```

次のリストが、platex.ltx の内容です。ただし、このバージョンでは、LaTeX から pLaTeX への拡張を plcore.ltx をロードすることで行ない、latex.ltx には直接、手を加えないようにしています。したがって platex.ltx はとても短いものとなっています。latex.ltx には LaTeX のコマンドが、plcore.ltx には pLaTeX で拡張したコマンドが定義されています。

```
⟨*plcore⟩
```

latex.ltx の末尾で使われている \dump をいったん無効化します。

```
\let\orgdump\dump
\let\dump\relax
```

latex.ltx を読み込みます。TeX Live の標準的インストールでは、この中で Babel 由来のハイフネーション・パターン hyphen.cfg が読み込まれるはずです。

```
\input latex.ltx
```

plcore.ltx を読み込みます。

```
\typeout{**************************^^J%
         *^^J%
         * making pLaTeX format^^J%
         *^^J%
         **************************}
\makeatletter
\input plcore.ltx
```

[3] 2016 年以前は pTeX と ε-pTeX のどちらでもフォーマットを作成することができましたが、2017 年に LaTeX が ε-pTeX 必須となったことに伴い、pLaTeX も ε-pTeX が必須となりました。

フォント関連のデフォルト設定ファイルである、pldefs.ltx を読み込みます。TEX の入力ファイル検索パスに設定されているディレクトリに pldefs.cfg ファイルがある場合は、そのファイルを使います。

```
\InputIfFileExists{pldefs.cfg}
          {\typeout{*************************************^^J%
                    * Local config file pldefs.cfg used^^J%
                    *************************************}}%
          {\input{pldefs.ltx}}
```

フォーマット作成時に pLATEX のバージョンがわかるように、端末に表示します。

```
\the\everyjob
```

pLATEX 2_ε の起動時に platex.cfg がある場合、それを読み込むようにします。バージョン 2016/07/01 ではコードを plcore.ltx に入れていましたが、platex.ltx へ移動しました。

```
\everyjob\expandafter{%
  \the\everyjob
  \IfFileExists{platex.cfg}{%
    \typeout{************************^^J%
             * Loading platex.cfg.^^J%
             ************************}%
    \input{platex.cfg}}{}%
}
```

フォーマットファイルにダンプします。

```
\let\dump\orgdump
\let\orgdump\@undefined
\makeatother
\dump
%\endinput
```

⟨/plcore⟩

実際に pLATEX 2_ε への拡張を行なっている plcore.ltx は、DOCSTRIP プログラムによって、次のファイルの断片が連結されたものです。

- plvers.dtx は、pLATEX 2_ε のフォーマットバージョンを定義しています。
- plfonts.dtx は、NFSS2 を拡張しています。
- plcore.dtx は、上記以外のコマンドでフォーマットファイルに格納されるコマンドを定義しています。

また、プリロードフォントや組版パラメータなどのデフォルト設定は、platex.ltx の中で pldefs.ltx をロードすることにより行います[4]。このファイル pldefs.ltx も plfonts.dtx から生成されます。

[4]アスキー版では plcore.ltx の中でロードしていましたが、2018 年以降の新しいコミュニティ版 pLATEX では platex.ltx から読み込むことにしました。

注意：

このファイルに記述されている設定を変更すれば pLaTeX 2$_\varepsilon$ をカスタマイズすることができますが、その場合は pldefs.ltx を直接修正するのではなく、いったん pldefs.cfg という名前でコピーして、そのファイルを編集してください。フォーマット作成時に pldefs.cfg が存在した場合は、そちらが pldefs.ltx の代わりに読み込まれます。

2.1.1 バージョン

pLaTeX 2$_\varepsilon$ のバージョンやフォーマットファイル名は、plvers.dtx で定義しています。

2.1.2 NFSS2 コマンド

LaTeX 2$_\varepsilon$ では、フォント選択機構として NFSS2 を用いています。pLaTeX 2$_\varepsilon$ では、オリジナルの NFSS2 と同様のインターフェイスで、和文フォントを選択できるように、plfonts.dtx で NFSS2 を拡張しています。

pLaTeX 2$_\varepsilon$ の NFSS2 は、フォントを切替えるコマンドを指定するときに、それが欧文書体か和文書体のいずれかを対象とするものかを、できるだけ意識しないようにする方向で拡張しています。いいかえれば、コマンドが（可能な限りの）判断をします。したがって数多くある英語版のクラスファイルやパッケージファイルなどで書体の変更を行っている箇所を修正する必要はあまりありません。

NFSS2 についての詳細は、LaTeX 2$_\varepsilon$ に付属の fntguide.tex を参照してください。

2.1.3 出力ルーチンとフロート

plcore.dtx は、次の項目に関するコマンドを日本語処理用に修正や拡張をしています。

- プリアンブルコマンド
- 改ページ
- 改行
- オブジェクトの出力順序
- トンボ
- 脚注マクロ
- 相互参照

- 疑似タイプ入力

2.2 クラスファイルとパッケージファイル

pLaTeX 2$_\varepsilon$ が提供をするクラスファイルやパッケージファイルは、オリジナルのファイルを基にしています。

pLaTeX 2$_\varepsilon$ に付属のクラスファイルは、次のとおりです。

- jarticle.cls, jbook.cls, jreport.cls

  横組用の標準クラスファイル。`jclasses.dtx` から作成される。

- tarticle.cls, tbook.cls, treport.cls

  縦組用の標準クラスファイル。`jclasses.dtx` から作成される。

- jltxdoc.cls

  日本語の `.dtx` ファイルを組版するためのクラスファイル。`jltxdoc.dtx` から作成される。

また、pLaTeX 2$_\varepsilon$ に付属のパッケージファイルは、次のとおりです。

- plext.sty

  縦組用の拡張コマンドなどが定義されているファイル。`plext.dtx` から作成される。

- ptrace.sty

  LaTeX でフォント選択コマンドのトレースに使う `tracefnt.sty` が再定義してしまう NFSS2 コマンドを、pLaTeX 2$_\varepsilon$ 用に再々定義するためのパッケージ。`plfonts.dtx` から作成される。

- pfltrace.sty

  LaTeX でフロート関連コマンドのトレースに使う `fltrace.sty` [5]が再定義してしまうコマンドを、pLaTeX 2$_\varepsilon$ 用に再々定義するためのパッケージ。`plcore.dtx` から作成される。

- oldpfont.sty

  pLaTeX 2.09 のフォントコマンドを提供するパッケージ。`pl209.dtx` から作成される。

なお、以前のバージョンに同梱していた ascmac パッケージと nidanfloat パッケージは、別のバンドルとして独立させました。

[5] LaTeX 2$_\varepsilon$ 2014/05/01 で追加されました。参考：LaTeX 2$_\varepsilon$ News Issue 21 (ltnews21.tex)

3　他のフォーマット・旧バージョンとの互換性

ここでは、この pLaTeX 2$_\varepsilon$ のバージョンと以前のバージョン、あるいは LaTeX 2$_\varepsilon$ との互換性について説明をしています。

3.1　LaTeX 2$_\varepsilon$ との互換性

pLaTeX 2$_\varepsilon$ は、LaTeX 2$_\varepsilon$ の上位互換という形を取っていますが、いくつかの命令の定義やパラメータなども変更しています。したがって英文書など、LaTeX 2$_\varepsilon$ でも処理できるファイルを pLaTeX 2$_\varepsilon$ で処理しても、完全に同じ結果になるとは限りません。

LaTeX 2$_\varepsilon$ 向けに書かれた多くのクラスファイルやパッケージファイルは、そのまま使えると思います。ただし、それらが pLaTeX 2$_\varepsilon$ で拡張しているコマンドと同じ名前のコマンドを再定義している場合は、その拡張の仕方によってはエラーになることもあります。用いようとしているクラスファイルやパッケージファイルがうまく動くかどうかを、完全に確かめる方法は残念ながらありません。一番簡単なのは、動かしてみることです。不幸にもうまく動かない場合は、ログファイルや付属の文書ファイルを参考に原因を調べてください。

3.2　pLaTeX 2.09 との互換性

pLaTeX 2$_\varepsilon$ では、文書が使用するクラスを、プリアンブルで `\documentclass` コマンドにより指定します。ここで `\documentclass` の代わりに `\documentstyle` を用いると、pLaTeX 2$_\varepsilon$ は自動的に *2.09* **互換モード**に入ります。これは LaTeX 2$_\varepsilon$ が LaTeX 2.09 互換モードに入るのと同様で、互換モードは古い文書を組版するためだけに作られています。新しく文書を作成する場合は、`\documentclass` コマンドを用いてください。

互換モードでは (p)LaTeX 2$_\varepsilon$ の新しい機能を利用できず、また古いネイティブな pLaTeX 2.09 環境と微妙に異なる結果になる可能性もあるという点は、英語版の LaTeX 2$_\varepsilon$ でも同じです。詳細は、LaTeX 2$_\varepsilon$ に付属の `usrguide.tex` を参照してください。

3.3　latexrelease パッケージへの対応

LaTeX `<2015/01/01>`で導入された latexrelease パッケージをもとに、新しい pLaTeX では platexrelease パッケージを用意しました。platexrelease パッケージを用いると、過去の pLaTeX をエミュレートしたり、フォーマットを作り直すことなく新しい pLaTeX を試したりすることができます。詳細は platexrelease のドキュメントを参照してください。

A　DOCSTRIP プログラムのためのオプション

この文書のソース (platex.dtx) を DOCSTRIP プログラムで処理することによって、いくつかの異なるファイルを生成することができます。DOCSTRIP プログラムの詳細は、docstrip.dtx を参照してください。

この文書の DOCSTRIP プログラムのためのオプションは、次のとおりです。

オプション	意味
plcore	フォーマットファイルを作るためのファイルを生成
pldoc	pLaTeX 2_ε のソースファイルをまとめて組版するための文書ファイル (pldoc.tex) を生成
shprog	上記のファイルを作成するための sh スクリプトを生成
plprog	入れ子構造を調べる簡単な perl スクリプトを生成
Xins	上記の sh スクリプトや perl スクリプトを取り出すための DOCSTRIP バッチファイル (Xins.ins) を生成

A.1　ファイルの取り出し方

たとえば、この文書の “plcore” の部分を “platex.ltx” というファイルにするときの手順はつぎのようになります。

1. platex docstrip
2. 入力ファイルの拡張子（dtx）を入力する。
3. 出力ファイルの拡張子（ltx）を入力する。
4. DOCSTRIP オプション（plcore）を入力する。
5. 入力ファイル名（platex）を入力する。
6. platex.ltx が存在する場合は、確認を求めてくるので、“y” を入力する。
7. 別の処理を行なうかを問われるので、“n” を入力する。

これで、platex.ltx が作られます。

あるいは、次のような内容のファイル fmt.ins を作成し、platex fmt.ins することでも platex.ltx を作ることができます。

```
\def\batchfile{fmt.ins}
\input docstrip.tex
\generateFile{platex.ltx}{t}{\from{platex.dtx}{plcore}}
```

B　文書ファイル

ここでは、このパッケージに含まれている dtx ファイルをまとめて組版し、ソースコード説明書を得るための文書ファイル pldoc.tex について説明をしています。個別に処理した場合と異なり、変更履歴や索引も付きます。全体で、およそ 200 ページ程度になります。

デフォルトではソースコードの説明が日本語で書かれます。もし英語の説明書を読みたい場合は、

```
\newif\ifJAPANESE
```

という内容の platex.cfg を予め用意してから pldoc.tex を処理してください（2016 年 7 月 1 日以降のコミュニティ版 pLaTeX 2_ε が必要）。

filecontents 環境は、引数に指定されたファイルが存在するときは何もしませんが、存在しないときは、環境内の内容でファイルを作成します。pldoc.dic ファイルは、mendex プログラムで索引を処理するときに \ 西暦, \ 和暦に対する「読み」を付けるために必要です。

```
⟨*pldoc⟩
\begin{filecontents}{pldoc.dic}
西暦　　せいれき
和暦　　われき
\end{filecontents}
```

文書クラスには、jltxdoc クラスを用います。plext.dtx の中でサンプルを組み立てていますので、plext パッケージが必要です。

```
\documentclass{jltxdoc}
\usepackage{plext}
\listfiles

```

いくつかの TeX プリミティブと plain TeX コマンドを索引に出力しないようにします。

```
\DoNotIndex{\def,\long,\edef,\xdef,\gdef,\let,\global}
\DoNotIndex{\if,\ifnum,\ifdim,\ifcat,\ifmmode,\ifvmode,\ifhmode,%
            \iftrue,\iffalse,\ifvoid,\ifx,\ifeof,\ifcase,\else,\or,\fi}
\DoNotIndex{\box,\copy,\setbox,\unvbox,\unhbox,\hbox,%
            \vbox,\vtop,\vcenter}
\DoNotIndex{\@empty,\immediate,\write}
\DoNotIndex{\egroup,\bgroup,\expandafter,\begingroup,\endgroup}
\DoNotIndex{\divide,\advance,\multiply,\count,\dimen}
\DoNotIndex{\relax,\space,\string}
\DoNotIndex{\csname,\endcsname,\@spaces,\openin,\openout,%
            \closein,\closeout}
\DoNotIndex{\catcode,\endinput}
\DoNotIndex{\jobname,\message,\read,\the,\m@ne,\noexpand}
```

```
\DoNotIndex{\hsize,\vsize,\hskip,\vskip,\kern,\hfil,\hfill,\hss,\vss,\unskip}
\DoNotIndex{\m@ne,\z@,\z@skip,\@ne,\tw@,\p@,\@minus,\@plus}
\DoNotIndex{\dp,\wd,\ht,\setlength,\addtolength}
\DoNotIndex{\newcommand, \renewcommand}

```

索引と変更履歴の見出しに \part を用いるように設定をします。

```
\ifJAPANESE
\IndexPrologue{\part*{索 引}%
                  \markboth{索 引}{索 引}%
                  \addcontentsline{toc}{part}{索 引}%
イタリック体の数字は、その項目が説明されているページを示しています。
下線の引かれた数字は、定義されているページを示しています。
その他の数字は、その項目が使われているページを示しています。}
\else
\IndexPrologue{\part*{Index}%
                  \markboth{Index}{Index}%
                  \addcontentsline{toc}{part}{Index}%
The italic numbers denote the pages where the corresponding entry
is described, numbers underlined point to the definition,
all others indicate the places where it is used.}
\fi
%
\ifJAPANESE
\GlossaryPrologue{\part*{変更履歴}%
                  \markboth{変更履歴}{変更履歴}%
                  \addcontentsline{toc}{part}{変更履歴}}
\else
\GlossaryPrologue{\part*{Change History}%
                  \markboth{Change History}{Change History}%
                  \addcontentsline{toc}{part}{Change History}}
\fi

```

標準の \changes コマンドを、複数ファイルの文書に合うように修正しています。

```
\makeatletter
\def\changes@#1#2#3{%
  \let\protect\@unexpandable@protect
  \edef\@tempa{\noexpand\glossary{#2\space
                \currentfile\space#1\levelchar
                \ifx\saved@macroname\@empty
                   \space\actualchar\generalname
                \else
                   \expandafter\@gobble
                   \saved@macroname\actualchar
                   \string\verb\quotechar*%
                   \verbatimchar\saved@macroname
                   \verbatimchar
                \fi
                :\levelchar #3}}%
```

```
  \@tempa\endgroup\@esphack}
```

コード行では、少しの Overfull を警告無しに許容します。

```
\renewcommand*\MacroFont{\fontencoding\encodingdefault
                   \fontfamily\ttdefault
                   \fontseries\mddefault
                   \fontshape\updefault
                   \small
                   \hfuzz 6pt\relax}
```

章番号の桁数が多い場合を考慮し、目次でのスペースを少し増やします。

```
\renewcommand*\l@subsection{\@dottedtocline{2}{1.5em}{2.8em}}
\renewcommand*\l@subsubsection{\@dottedtocline{3}{3.8em}{3.4em}}
\makeatother
```

変更履歴と 2 段組の索引を作成します。

```
\RecordChanges
\CodelineIndex
\EnableCrossrefs
\setcounter{IndexColumns}{2}
\settowidth\MacroIndent{\ttfamily\scriptsize 000\ }
```

ここからが本文ページとなります。

```
\begin{document}
 \title{The \pLaTeXe\ Sources}
 \author{Ken Nakano \& Japanese \TeX\ Development Community}

% Get the date and patch level from plvers.dtx
\makeatletter
\let\patchdate=\@empty
\begingroup
   \def\ProvidesFile#1\pfmtversion#2#3\ppatch@level#4{%
      \date{#2}\xdef\patchdate{#4}\endinput}
   \input{plvers.dtx}
\endgroup

% Add the patch version if available.
\def\Xpatch{0}
\ifx\patchdate\Xpatch\else
% number is assumed
\ifnum\patchdate>0
  \edef\@date{\@date\space Patch level\space\patchdate}
\else
  \edef\@date{\@date\space Pre-Release\patchdate}
\fi\fi
\makeatother

\pagenumbering{roman}
\maketitle
\renewcommand\maketitle{}
```

```
\tableofcontents
\clearpage
\pagenumbering{arabic}

\DocInclude{plvers}   % pLaTeX version

\DocInclude{plfonts}  % NFSS2 commands

\DocInclude{plcore}   % kernel commands

\DocInclude{plext}    % external commands

\DocInclude{pl209}    % 2.09 compatibility mode commands

\DocInclude{kinsoku}  % kinsoku parameter

\DocInclude{jclasses} % Standard class

\DocInclude{jltxdoc}  % dtx documents class

```

ltxdoc.cfg に \AtEndOfClass{\OnlyDescription}が指定されている場合は、ここで終了します。

```
\StopEventually{\end{document}}

```

変更履歴と索引を組版します。変更履歴ファイルと索引の作り方の詳細については、おまけ C.1 を参照してください。

```
\clearpage
\pagestyle{headings}
% Make TeX shut up.
\hbadness=10000
\newcount\hbadness
\hfuzz=\maxdimen
%
\PrintChanges
\clearpage
%
\begingroup
  \def\endash{--}
  \catcode`\-\active
  \def-{\futurelet\temp\indexdash}
  \def\indexdash{\ifx\temp-\endash\fi}

  \PrintIndex
\endgroup
```

ltxdoc.cfg に 2 度目の \PrintIndex が指定されているかもしれません。そこで、最後に、変更履歴や索引が 2 度組版されないように \PrintChanges および

\PrintIndex コマンドを何も実行しないようにします。

```
\let\PrintChanges\relax
\let\PrintIndex\relax
\end{document}
⟨/pldoc⟩
```

C　おまけプログラム

C.1　シェルスクリプト mkpldoc.sh

pLaTeX 2ε のマクロ定義ファイルをまとめて組版し、変更履歴と索引も付けるときに便利なシェルスクリプトです。このシェルスクリプト[6]の使用方法は次のとおりです。

```
sh mkpldoc.sh
```

C.1.1　mkpldoc.sh の内容

まず、以前に pldoc.tex を処理したときに作成された、目次ファイルや索引ファイルなどを削除します。

```
⟨*shprog⟩
for f in pldoc.toc pldoc.idx pldoc.glo ; do
if [ -e $f ]; then rm $f; fi
done
```

そして、ltxdoc.cfg を空にします。このファイルは、jltxdoc.cls の定義を変更するものですが、ここでは、変更されたくありません。

```
echo "" > ltxdoc.cfg
```

そして、pldoc.tex を処理します。

```
platex pldoc.tex
```

索引と変更履歴を作成します。このスクリプトでは、変更履歴や索引を生成するのに mendex プログラムを用いています。mendex は makeindex の上位互換のファイル整形コマンドで、索引語の読みを自動的に付けるなどの機能があります。

-s オプションは、索引ファイルを整形するためのスタイルオプションです。索引用の gind.ist と変更履歴用の gglo.ist は、LaTeX のディストリビューションに付属しています。

-o は、出力するファイル名を指定するオプションです。

-f は、項目に “読み” がなくてもエラーとしないオプションです。makeindex コマンドには、このオプションがありません。

[6] このシェルスクリプトは UNIX 用です。しかし rm コマンドを delete コマンドにするなどすれば、簡単に DOS などのバッチファイルに修正することができます。

```
mendex -s gind.ist -d pldoc.dic -o pldoc.ind pldoc.idx
mendex -f -s gglo.ist -o pldoc.gls pldoc.glo
```

ltxdoc.cfg の内容を \includeonly{}にし、pldoc.tex を処理します。このコマンドは、引数に指定されたファイルだけを “\include” するためのコマンドですが、ここでは何も \include したくないので、引数には何も指定をしません。しかし、\input で指定されているファイルは読み込まれます。したがって、目次や索引や変更履歴のファイルが処理されます。この処理は、主に、これらでエラーが出るかどうかの確認です。

```
echo "\includeonly{}" > ltxdoc.cfg
platex pldoc.tex
```

最後に、再び ltxdoc.cfg を空にして、pldoc.tex を処理をします。本文を 1 ページから開始していますので、この後、もう一度処理をする必要はありません。

```
echo "" > ltxdoc.cfg
platex pldoc.tex
# EOT
```
⟨/shprog⟩

C.2 Perl スクリプト dstcheck.pl

DOCSTRIP 文書ファイルは、LaTeX のソースとその文書を同時に管理する方法として、とてもすぐれていると思います。しかし、たとえば jclasses.dtx のように、条件が多くなると、入れ子構造がわからなくなってしまいがちです。LaTeX で処理すれば、エラーによってわかりますが、文書ファイルが大きくなると面倒です。

ここでは、DOCSTRIP 文書ファイルの入れ子構造を調べるのに便利な、perl スクリプトについて説明をしています。

この perl スクリプトの使用方法は次のとおりです。

```
perl dstcheck.pl <file-name>
```

C.2.1 dstcheck.pl の内容

最初に、この perl スクリプトが何をするのかを簡単に記述したコメントを付けます。

⟨*plprog⟩
```
##
## DOCSTRIP 文書内の環境や条件の入れ子を調べる perl スクリプト
##
```

このスクリプトは、入れ子の対応を調べるために、次のスタックを用います。⟨ 条件 ⟩ あるいは ⟨ 環境 ⟩ を開始するコードが現れたときに、それらはスタックにプッシュされ、終了するコードでポップされます。したがって、現在の ⟨ 条件 ⟩ あるいは ⟨ 環境 ⟩ と、スタックから取り出した ⟨ 条件 ⟩ あるいは ⟨ 環境 ⟩ と一致すれば、対応が取れているといえます。そうでなければエラーです。

@dst スタックには、⟨ 条件 ⟩ が入ります。条件の開始は、“%<*⟨ 条件 ⟩>” です。条件の終了は、“%</⟨ 条件 ⟩>” です。⟨ 条件 ⟩ には、>文字が含まれません。@env スタックには、⟨ 環境 ⟩ が入ります。

先頭を明示的に示すために、ダミーの値を初期値として用います。スタックは、⟨ 条件 ⟩ あるいは ⟨ 環境 ⟩ の名前と、その行番号をペアにして操作をします。

```
204 push(@dst,"DUMMY"); push(@dst,"000");
205 push(@env,"DUMMY"); push(@env,"000");
```

この while ループの中のスクリプトは、文書ファイルの 1 行ごとに実行をします。

```
206 while (<>) {
```

入力行が条件を開始する行なのかを調べます。条件の開始行ならば、@dst スタックに ⟨ 条件 ⟩ と行番号をプッシュします。

```
207   if (/^%<\*([^>]+)>/) { # check conditions
208     push(@dst,$1);
209     push(@dst,$.);
```

そうでなければ、条件の終了行なのかを調べます。現在行が条件の終了を示している場合は、@dst スタックをポップします。

```
210   } elsif (/^%<\/([^>]+)>/) {
211     $linenum = pop(@dst);
212     $conditions = pop(@dst);
```

現在行の ⟨ 条件 ⟩ と、スタックから取り出した ⟨ 条件 ⟩ が一致しない場合、その旨のメッセージを出力します。

なお、DUMMY と一致した場合は、一番外側のループが合っていないということを示しています。このとき、これらのダミー値をスタックに戻します。いつでもスタックの先頭をダミー値にするためです。

```
213     if ($1 ne $conditions) {
214       if ($conditions eq "DUMMY") {
215         print "$ARGV: `</$1>' (l.$.) is not started.\n";
216         push(@dst,"DUMMY");
217         push(@dst,"000");
218       } else {
219         print "$ARGV: `<*$conditions>' (l.$linenum) is ended ";
220         print "by `<*$1>' (l.$.)\n";
221       }
222     }
223   }
```

環境の入れ子も条件と同じように調べます。

verbatim 環境のときに、その内側をスキップしていることに注意をしてください。

```
224   if (/^% *\\begin\{verbatim\}/) { # check environments
225     while(<>) {
226         last if (/^% *\\end\{verbatim\}/);
227     }
```

```
228   } elsif (/^% *\\begin\{([^{}]+)\}\{(.*)\}/) {
229     push(@env,$1);
230     push(@env,$.);
231   } elsif (/^% *\\begin\{([^{}]+)\}/) {
232     push(@env,$1);
233     push(@env,$.);
234   } elsif (/^% *\\end\{([^{}]+)\}/) {
235     $linenum = pop(@env);
236     $environment = pop(@env);
237     if ($1 ne $environment) {
238       if ($environment eq "DUMMY") {
239         print "$ARGV: `\\end{$1}' (l.$.) is not started.\n";
240         push(@env,"DUMMY");
241         push(@env,"000");
242       } else {
243         print "$ARGV: \\begin{$environement} (l.$linenum) is ended ";
244         print "by \\end{$1} (l.$.)\n";
245       }
246     }
247   }
```

ここまでが、最初の `while` ループです。

```
248 }
```

文書ファイルを読み込んだ後、終了していない条件があるかどうかを確認します。すべての条件の対応がとれていれば、この時点での`@dst` スタックにはダミー値しか入っていません。したがって、対応が取れている場合は、最初の２つのポップによって、ダミー値が設定されます。ダミー値でなければ、ダミー値になるまで、取り出した値を出力します。

```
249 $linenum = pop(@dst);
250 $conditions = pop(@dst);
251 while ($conditions ne "DUMMY") {
252     print "$ARGV: `<*$conditions>' (l.$linenum) is not ended.\n";
253     $linenum = pop(@dst);
254     $conditions = pop(@dst);
255 }
```

環境の入れ子についても、条件の入れ子と同様に確認をします。

```
256 $linenum = pop(@env);
257 $environment = pop(@env);
258 while ($environment ne "DUMMY") {
259     print "$ARGV: `\\begin{$environment}' (l.$linenum) is not ended.\n";
260     $linenum = pop(@env);
261     $environment = pop(@env);
262 }
263 exit;
264 ⟨/plprog⟩
```

C.3 DOCSTRIP バッチファイル

ここでは、付録 C.1 と付録 C.2 で説明をした二つのスクリプトを、このファイルから取り出すための DOCSTRIP バッチファイルについて説明をしています。

まず、DOCSTRIP パッケージをロードします。また、実行経過のメッセージを出力しないようにしています。

```
⟨*Xins⟩
\input docstrip
\keepsilent
```

DOCSTRIP プログラムは、連続する二つのパーセント記号 (%%) ではじまる行をメタコメントとみなし、条件によらず出力をします。しかし、“%” は TEX ではコメントであっても、sh や perl にとってはコメントではありません。そこで、メタコメントとして出力する文字を “##” と変更します。

```
{\catcode`#=12 \gdef\MetaPrefix{## }}
```

そして、プリアンブルに出力されるメッセージを宣言します。ここでは、とくに何も指定していませんが、宣言をしないとデフォルトの記述が ‘%%’ 付きで出力されてしまうため、それを抑制する目的で使用しています。

```
\declarepreamble\thispre
\endpreamble
\usepreamble\thispre
```

ポストアンブルも同様に、宣言をしないと ‘\endinput’ が出力されます。

```
\declarepostamble\thispost
\endpostamble
\usepostamble\thispost
```

\generate コマンドで、どのファイルに、どのファイルのどの部分を出力するのかを指定します。

```
\generate{
   \file{dstcheck.pl}{\from{platex.dtx}{plprog}}
   \file{mkpldoc.sh}{\from{platex.dtx}{shprog}}
}
\endbatchfile
⟨/Xins⟩
```

参考文献

[1] 中野 賢『日本語 LaTeX 2_ε ブック』アスキー, 1996.

[2] インプレス・ラボ監修, アスキー書籍編集部編『縦組対応 パーソナル日本語 TeX』アスキー出版局, 1994

[3] アスキー出版技術部責任編集『日本語 TeX テクニカルブック I』アスキー, 1990.

[4] Haruhiko Okumura, pTeX and Japanese Typesetting The Asian Journal of TeX, Volume 2, No. 1, 2008.
(`http://ajt.ktug.org/2008/0201okumura.pdf`)

[5] Hisato Hamano, Vertical Typesetting with TeX. TUGboat issue 11:3, 1990.
(`https://tug.org/TUGboat/tb11-3/tb29hamano.pdf`)

[6] Donald E. Knuth. "*The TeXbook*". Addison-Wesley, 1984. (邦訳：斎藤信男監修, 鷺谷好輝訳, TeX ブック 改訂新版, アスキー出版局, 1989)

[7] Laslie Lamport. "*LaTeX: A Document Preparation System*". Addison-Wesley, second edition, 1994.

[8] Laslie Lamport. "*LaTeX: A Document Preparation System*". Addison-Wesley, 1986. (邦訳：倉沢良一監修, 大野俊治・小暮博通・藤浦はる美訳, 文書処理システム LaTeX, アスキー, 1990)

[9] Michel Goossens, Frank Mittelbach, Alexander Samarin. "*The LaTeX Companion*". Addison-Wesley, 1994.

[10] 河野 真治『入門 Perl』アスキー出版局, 1994

変更履歴

1995/05/08 v1.0
・最初のバージョン 2
1995/08/25 v1.0a
・互換性について、DOCSTRIP の使い方、参考文献を追加 2
1996/02/01 v1.0b
・omake-sh.ins, omake-pl.ins を DOCSTRIP の変更にともなう変更をした 17
1997/01/23 v1.0c
・DOCSTRIP にともなう変更 17
・gind.ist と gglo.ist を $TEXMF/tex/latex2e/base ディレクトリからコピーしないようにした 13
1997/01/25 v1.0c
・pldoc.dic を filecontents 環境により作成 9
1997/01/29 v1.0c
・pltpatch.ltx を plpatch.ltx に名称変更 11
2016/01/27 v1.0d
・pLaTeX 2ε に付属するファイルの説明を更新 6
・rm コマンド実行前に存在確認するようにした 13
2016/02/16 v1.0e
・platexrelease の説明を追加 7
2016/04/12 v1.0f
・ドキュメントを更新 1
2016/05/07 v1.0g
・フォーマット作成時に LaTeX のバナーを一旦保存 3
2016/05/08 v1.0h
・ドキュメントから plpatch.ltx を除外 11
2016/05/12 v1.0i
・一時コマンド \orgdump を最終的に未定義へ 4
2016/05/20 v1.0j
・pfltrace の説明を追加 6
2016/05/21 v1.0k
・変更履歴も出力するようにした . . 1
2016/06/19 v1.0l
・パッチレベルを plvers.dtx から取得 11
2016/08/26 v1.0m
・platex.cfg の読み込みを plcore.ltx から platex.ltx へ移動 4
2016/09/14 v1.0n
・LaTeX のバナーの保存しかたを改良 3
2017/09/24 v1.0o
・パッチレベルが負の数の場合を pre-release 扱いへ 11
2017/11/11 v1.0p
・LaTeX のバナーを保存するコードを platex.ltx から plcore.ltx へ移動 3
2017/11/29 v1.0q
・英語版ドキュメントを追加 1
2017/12/02 v1.0r
・英語の参考文献も追加 2
2017/12/05 v1.0s
・デフォルト設定ファイルの読み込みを plcore.ltx から platex.ltx へ移動 4
2018/02/07 v1.0t
・ascmac パッケージを独立させた . 6
2018/02/18 v1.0u
・nidanfloat パッケージを独立させた 6
2018/04/06 v1.0v
・最新の source2e への追随 11