

pL^AT_EX 2_ε について

中野 賢 & 日本語 T_EX 開発コミュニティ

作成日：2018/02/18

注意：

この文書では「コミュニティ版 pL^AT_EX 2_ε」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 pL^AT_EX 2_ε（以下、アスキー版 pL^AT_EX 2_ε）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 pT_EX は、国際的に広く使われている T_EX 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 T_EX Live や W32T_EX では、pL^AT_EX も元々の pT_EX ではなく、その拡張版 ϵ -pT_EX をエンジンとして用いるようになっています。また、pL^AT_EX のベースである L^AT_EX 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 pL^AT_EX として、アスキー版から fork して日本語 T_EX 開発コミュニティ (Japanese T_EX Development Community) が配布しているものが、コミュニティ版 pL^AT_EX 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 pL^AT_EX 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 T_EX 開発コミュニティに報告してください。T_EX 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 pL^AT_EX の概要を説明したのですが、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 pL^AT_EX の更新内容は pL^AT_EX 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 pL^AT_EX 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 T_EX (pT_EX)、<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 p \LaTeX 2 ϵ 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。p \LaTeX 2 ϵ 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、p \LaTeX 2 ϵ の中の `plex \LaTeX 2 ϵ .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、p \LaTeX (あるいはその前身の「日本語 \LaTeX 」) 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 p \LaTeX 2 ϵ 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 p \LaTeX 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 p \LaTeX 2 ϵ 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 p \LaTeX 2 ϵ の機能について

p \LaTeX 2 ϵ 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、p \LaTeX 2 ϵ 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 \LaTeX 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 *p*LaTeX 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 2_ε 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。

```
1 <{*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。

```
2 \let\orgdump\dump
3 \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が。

```
4 \input latex.ltx
```

`plcore.ltx` を読み込みます。

```
5 \typeout{*****~^J%
6      *~^J%
7      * making pLaTeX format~^J%
8      *~^J%
9      *****}
10 \makeatletter
11 \input plcore.ltx
```

³2016 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。

```
12 \InputIfFileExists{pldefs.cfg}
13     {\typeout{*****^J%
14         * Local config file pldefs.cfg used^J%
15         *****}}%
16     {\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。

```
17 \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。

```
18 \everyjob\expandafter{%
19   \the\everyjob
20   \IfFileExists{platex.cfg}{%
21     \typeout{*****^J%
22         * Loading platex.cfg.^J%
23         *****}%
24     \input{platex.cfg}}{%
25 }
```

フォーマットファイルにダンプします。

```
26 \let\dump\orgdump
27 \let\orgdump\@undefined
28 \makeatother
29 \dump
30 %\endinput
31 </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 $\text{p}\text{L}\text{A}\text{T}\text{E}\text{X}2_{\epsilon}$ 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

$\text{p}\text{L}\text{A}\text{T}\text{E}\text{X}2_{\epsilon}$ 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

$\text{L}\text{A}\text{T}\text{E}\text{X}2_{\epsilon}$ では、フォント選択機構として NFSS2 を用いています。 $\text{p}\text{L}\text{A}\text{T}\text{E}\text{X}2_{\epsilon}$ 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。

$\text{p}\text{L}\text{A}\text{T}\text{E}\text{X}2_{\epsilon}$ 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、 $\text{L}\text{A}\text{T}\text{E}\text{X}2_{\epsilon}$ に付属の `fntguide.tex` を参照してください。

2.1.3 出カ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

p \LaTeX 2 ϵ 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。

p \LaTeX 2 ϵ 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、p \LaTeX 2 ϵ に付属のパッケージファイルは、次のとおりです。

- plect.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 tracefmt.sty が再定義してしまう NFSS2 コマンドを、p \LaTeX 2 ϵ 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.sty⁵が再定義してしまうコマンドを、p \LaTeX 2 ϵ 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
p \LaTeX 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。

⁵p \LaTeX 2 ϵ 2014/05/01 で追加されました。参考： \LaTeX 2 ϵ 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 p \LaTeX 2 ϵ のバージョンと以前のバージョン、あるいは \LaTeX 2 ϵ との互換性について説明をしています。

3.1 \LaTeX 2 ϵ との互換性

p \LaTeX 2 ϵ は、 \LaTeX 2 ϵ 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 ϵ でも処理できるファイルを p \LaTeX 2 ϵ 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 ϵ 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 p \LaTeX 2 ϵ 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 p \LaTeX 2.09 との互換性

p \LaTeX 2 ϵ 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、p \LaTeX 2 ϵ は自動的に 2.09 互換モードに入ります。これは \LaTeX 2 ϵ 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 ϵ の新しい機能を利用できず、また古いネイティブな p \LaTeX 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 ϵ でも同じです。詳細は、 \LaTeX 2 ϵ 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 p \LaTeX 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 p \LaTeX 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 p \LaTeX 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
<code>plcore</code>	フォーマットファイルを作るためのファイルを生成
<code>pldoc</code>	pL ^A T _Ε X 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 (<code>pldoc.tex</code>) を生成
<code>shprog</code>	上記のファイルを作成するための sh スクリプトを生成
<code>plprog</code>	入れ子構造を調べる簡単な perl スクリプトを生成
<code>Xins</code>	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 (<code>Xins.ins</code>) 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 p_ΛT_EX 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。

```
32 \begin{filecontents}{pldoc.dic}
33 西暦      せいいき
34 和暦      われき
35 \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。

```
37 \documentclass{jltxdoc}
38 \usepackage{plext}
39 \listfiles
40
```

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

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

```

54 \DoNotIndex{\hsize,\vsize,\hskip,\vskip,\kern,\hfil,\hfill,\hss,\vss,\unskip}
55 \DoNotIndex{\m@ne,\z@,\z@skip,\@ne,\tw@,\p@,\@minus,\@plus}
56 \DoNotIndex{\dp,\wd,\ht,\setlength,\addtolength}
57 \DoNotIndex{\newcommand,\renewcommand}
58

```

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。

```

59 \ifJAPANESE
60 \IndexPrologue{\part*{索引}}%
61         \markboth{索引}{索引}%
62         \addcontentsline{toc}{part}{索引}%
63 イタリアック体の数字は、その項目が説明されているページを示しています。
64 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。
65 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。}
66 \else
67 \IndexPrologue{\part*{Index}}%
68         \markboth{Index}{Index}%
69         \addcontentsline{toc}{part}{Index}%
70 The italic numbers denote the pages where the corresponding entry
71 is described, numbers underlined point to the definition,
72 all others indicate the places where it is used.}
73 \fi
74 %
75 \ifJAPANESE
76 \GlossaryPrologue{\part*{変更履歴}}%
77         \markboth{変更履歴}{変更履歴}%
78         \addcontentsline{toc}{part}{変更履歴}}
79 \else
80 \GlossaryPrologue{\part*{Change History}}%
81         \markboth{Change History}{Change History}%
82         \addcontentsline{toc}{part}{Change History}}
83 \fi
84

```

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。

```

85 \makeatletter
86 \def\changes@#1#2#3{%
87   \let\protect\@unexpandable@protect
88   \edef\@tempa{\noexpand\glossary{#2\space\currentfile\space#1\levelchar
89     \ifx\saved@macroname\@empty
90       \space\actualchar\generalname
91     \else
92       \expandafter\@gobble
93       \saved@macroname\actualchar
94       \string\verb\quotechar*%
95       \verbatimchar\saved@macroname
96       \verbatimchar
97     \fi
98     :\levelchar #3}}%
99   \@tempa\endgroup\@esphack}

```

```

100 \makeatother

変更履歴と 2 段組の索引を作成します。

101 \RecordChanges
102 \CodelineIndex
103 \EnableCrossrefs
104 \setcounter{IndexColumns}{2}
105 \settowidth\MacroIndent{\ttfamily\scriptsize 000\ }

ここからが本文ページとなります。

106 \begin{document}
107 \title{The \pLaTeX\ Sources}
108 \author{Ken Nakano \& Japanese \TeX\ Development Community}
109
110 % Get the date and patch level from plvers.dtx
111 \makeatletter
112 \let\patchdate=\@empty
113 \begin{group}
114   \def\ProvidesFile#1\pfmtversion#2#3\ppatch@level#4{%
115     \date{#2}\xdef\patchdate{#4}\endinput}
116   \input{plvers.dtx}
117 \endgroup
118
119 % Add the patch version if available.
120 \def\Xpatch{0}
121 \ifx\patchdate\Xpatch\else
122 % number is assumed
123 \ifnum\patchdate>0
124   \edef\@date{\@date\space Patch level\space\patchdate}
125 \else
126   \edef\@date{\@date\space Pre-Release\patchdate}
127 \fi\fi
128 \makeatother
129
130 \pagenumbering{roman}
131 \maketitle
132 \renewcommand\maketitle{}
133 \tableofcontents
134 \clearpage
135 \pagenumbering{arabic}
136
137 \DocInclude{plvers} % pLaTeX version
138
139 \DocInclude{plfonts} % NFSS2 commands
140
141 \DocInclude{plcore} % kernel commands
142
143 \DocInclude{plext} % external commands
144
145 \DocInclude{pl209} % 2.09 compatibility mode commands

```


C おまけプログラム

C.1 シェルスクリプト mkpldoc.sh

pL^AT_EX 2_ε 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。

```
177 <shprog>
178 for f in pldoc.toc pldoc.idx pldoc.glo ; do
179 if [ -e $f ]; then rm $f; fi
180 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。

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

そして、pldoc.tex を処理します。

```
182 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、L^AT_EX 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。

```
183 mendex -s gind.ist -d pldoc.dic -o pldoc.ind pldoc.idx
184 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

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。

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。

```
185 echo "\includeonly{" > ltxdoc.cfg
186 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。

```
187 echo "" > ltxdoc.cfg
188 platex pldoc.tex
189 # EOT
190 </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うがちです。 \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。

```
191 <*plprog>
192 ##
193 ## DOCSTRIP 文書内の環境や条件の入れ子を調べる perl スクリプト
194 ##
```

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。

```

195 push(@dst,"DUMMY"); push(@dst,"000");
196 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。

```

197 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。

```

198   if (/^<\*(\[>]+\>/) { # check conditions
199       push(@dst,$1);
200       push(@dst,$.);

```

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。

```

201   } elsif (/^<\[/([>]+\>/) {
202       $linenum = pop(@dst);
203       $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。

```

204   if ($1 ne $conditions) {
205       if ($conditions eq "DUMMY") {
206           print "$ARGV: '</$1>' (l.$.) is not started.\n";
207           push(@dst,"DUMMY");
208           push(@dst,"000");
209       } else {
210           print "$ARGV: '<*$conditions>' (l.$linenum) is ended ";
211           print "by '<*$1>' (l.$.)\n";
212       }
213   }
214 }

```

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。

```

215   if (/^% *\\begin\{verbatim\}/) { # check environments
216       while(<>) {
217           last if (/^% *\\end\{verbatim\}/);
218       }
219   } elsif (/^% *\\begin\{([~}]+)\}\{(.*)\}/) {
220       push(@env,$1);
221       push(@env,$.);
222   } elsif (/^% *\\begin\{([~}]+)\}\{([~}]+)\}/) {
223       push(@env,$1);
224       push(@env,$.);
225   } elsif (/^% *\\end\{([~}]+)\}\{([~}]+)\}/) {

```

```

226     $linenum = pop(@env);
227     $environment = pop(@env);
228     if ($1 ne $environment) {
229         if ($environment eq "DUMMY") {
230             print "ARGV: '\\end{$1}' (l.$.) is not started.\n";
231             push(@env, "DUMMY");
232             push(@env, "000");
233         } else {
234             print "ARGV: '\\begin{$environment}' (l.$linenum) is ended ";
235             print "by '\\end{$1}' (l.$.)\n";
236         }
237     }
238 }

```

ここまでの、最初の while ループです。

```

239 }

```

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。

```

240 $linenum = pop(@dst);
241 $conditions = pop(@dst);
242 while ($conditions ne "DUMMY") {
243     print "ARGV: '<*$conditions>' (l.$linenum) is not ended.\n";
244     $linenum = pop(@dst);
245     $conditions = pop(@dst);
246 }

```

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。

```

247 $linenum = pop(@env);
248 $environment = pop(@env);
249 while ($environment ne "DUMMY") {
250     print "ARGV: '\\begin{$environment}' (l.$linenum) is not ended.\n";
251     $linenum = pop(@env);
252     $environment = pop(@env);
253 }
254 exit;
255 </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。


```

256 <*Xins>
257 \input docstrip
258 \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。

```

259 {\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。

```

260 \declarepreamble\thispre
261 \endpreamble
262 \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。

```

263 \declarepostamble\thispost
264 \endpostamble
265 \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。

```

266 \generate{
267   \file{dstcheck.pl}{\from{platex.dtx}{plprog}}
268   \file{mkpldoc.sh}{\from{platex.dtx}{shprog}}
269 }
270 \endbatchfile
271 </Xins>

```

参考文献

- [1] 中野 賢『日本語 L^AT_EX 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 T_EX』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 T_EX テクニカルブック I』アスキー, 1990.
- [4] Haruhiko Okumura, pT_EX and Japanese Typesetting The Asian Journal of T_EX, Volume 2, No. 1, 2008.
(<http://ajt.ktug.org/2008/0201okumura.pdf>)
- [5] Hisato Hamano, Vertical Typesetting with T_EX. TUGboat issue 11:3, 1990.
(<https://tug.org/TUGboat/tb11-3/tb29hamano.pdf>)
- [6] Donald E. Knuth. “*The T_EXbook*”. Addison-Wesley, 1984. (邦訳：斎藤信男監修, 鷺谷好輝訳, T_EX ブック 改訂新版, アスキー出版局, 1989)
- [7] Laslie Lamport. “*L^AT_EX: A Document Preparation System*”. Addison-Wesley, second edition, 1994.
- [8] Laslie Lamport. “*L^AT_EX: 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 L^AT_EX, アスキー, 1990)
- [9] Michel Goossens, Frank Mittelbach, Alexander Samarin. “*The L^AT_EX 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	2016/05/12 v1.0i
・最初のバージョン 2	・一時コマンド <code>\orgdump</code> を最終的に未定義へ 4
1995/08/25 v1.0a	2016/05/20 v1.0j
・互換性について、DOCSTRIP の使い方、参考文献を追加 2	・ <code>pfttrace</code> の説明を追加 6
1996/02/01 v1.0b	2016/05/21 v1.0k
・ <code>omake-sh.ins</code> , <code>omake-pl.ins</code> を DOCSTRIP の変更にもなう変更をした 16	・変更履歴も出力するようにした . . . 1
1997/01/23 v1.0c	2016/06/19 v1.0l
・DOCSTRIP にもなう変更 16	・パッチレベルを <code>plvers.dtx</code> から取得 11
・ <code>gind.ist</code> と <code>gglo.ist</code> を <code>\$TEXMF/tex/latex2e/base</code> ディレクトリからコピーしないようにした 13	2016/08/26 v1.0m
1997/01/25 v1.0c	・ <code>platex.cfg</code> の読み込みを <code>plcore.ltx</code> から <code>platex.ltx</code> へ移動 4
・ <code>pldoc.dic</code> を <code>filecontents</code> 環境により作成 9	2016/09/14 v1.0n
1997/01/29 v1.0c	・ \LaTeX のバナーの保存しかたを改良 3
・ <code>pltpatch.ltx</code> を <code>plpatch.ltx</code> に名称変更 11	2017/09/24 v1.0o
2016/01/27 v1.0d	・パッチレベルが負の数の場合を <code>pre-release</code> 扱いへ 11
・ \LaTeX 2 _ε に付属するファイルの説明を更新 6	2017/11/11 v1.0p
・ <code>rm</code> コマンド実行前に存在確認するようにした 13	・ \LaTeX のバナーを保存するコードを <code>platex.ltx</code> から <code>plcore.ltx</code> へ移動 3
2016/02/16 v1.0e	2017/11/29 v1.0q
・ <code>platexrelease</code> の説明を追加 7	・英語版ドキュメントを追加 1
2016/04/12 v1.0f	2017/12/02 v1.0r
・ドキュメントを更新 1	・英語の参考文献も追加 2
2016/05/07 v1.0g	2017/12/05 v1.0s
・フォーマット作成時に \LaTeX のバナーを一旦保存 3	・デフォルト設定ファイルの読み込みを <code>plcore.ltx</code> から <code>platex.ltx</code> へ移動 4
2016/05/08 v1.0h	2018/02/07 v1.0t
・ドキュメントから <code>plpatch.ltx</code> を除外 11	・ <code>ascmac</code> パッケージを独立させた . . 6
	2018/02/18 v1.0u
	・ <code>nidanfloat</code> パッケージを独立させた 6