

ClutT_EX マニュアル (バージョン 0.7.1)

ARATA Mizuki

2026 年 10 月 31 日

目次

第 1 章	Clut _T E _X の概要	2
第 2 章	Clut _T E _X の使い方	3
2.1	インストール	3
2.2	コマンドライン	3
2.3	Sync _T E _X	5
2.4	監視モード	5
2.5	MakeIndex や Bib _T E _X	5
2.6	大規模な文書を書く場合	5
2.7	Makefile と組み合わせる	6
2.8	出力ディレクトリについて	6
2.9	エイリアス	7
2.10	minted と epstopdf への対策	7
2.11	ドライバーファイルの検査	8
2.12	設定ファイルによるカスタマイズ	8

第 1 章

Clut_{TEX} の概要

Clut_{TEX} は、 $\text{\LaTeX}$ 処理の自動化ツールである。基本的な特徴として、

- 作業ディレクトリを `.aux` や `.log` 等の「余計な」ファイルで散らかさない
- (相互参照の解決などで) 複数回処理を行う必要がある場合に、必要な回数だけ自動で処理する
- 入力ファイルを監視し、変更があった場合に自動で再処理する (`--watch` オプション^{*1})
- `MakeIndex`, `BibTEX`, `Biber` 等のコマンドを自動で実行する (`--makeindex` オプション, `--bibtex` オプション, `--biber` オプション)
- `pTEX` 系列の処理系で PDF を生成する場合、別途 `dvipdfmx` を実行する必要がある (自動で `dvipdfmx` を実行する) もしも DVI ファイルが欲しいのであれば、`--output-format=dvi` を指定すれば良い。

などがある。

$\text{\LaTeX}$ 処理の自動化ツールとしては `latexmk` が普及している。そのような既存のツールに対する Clut_{TEX} の最大の差別化ポイントは「作業ディレクトリを散らかさない」ことである。

`.aux` 等の補助ファイルは「処理後に消す」のではなく、「隔離された場所に生成させる」。そのため、「相互参照を使う文書の処理に関して、Clut_{TEX} を使わない場合に比べて Clut_{TEX} を使う場合に実行回数が増える」ようなことは基本的にはない^{*2}。

^{*1} Unix 系 OS では、別途プログラムが必要。

^{*2} PC の再起動直後など、テンポラリディレクトリーが空の場合を除く。

第 2 章

ClutT_EX の使い方

2.1 インストール

ClutT_EX は最新の T_EX Live に収録されている。よって、T_EX Live を利用している方は、T_EX Live の更新（コマンドなら `tlmgr upgrade --all`）を行えば ClutT_EX がインストールされる。

何らかの理由により手動でインストールしたい場合は、GitHub^{*1}からアーカイブをダウンロードし、その中にある `bin/cluttex` か `bin/cluttex.bat` を PATH の通った場所にコピーする。

2.2 コマンドライン

基本的な使い方：

```
cluttex -e ENGINE OPTIONS [--] INPUT.tex
```

基本的なオプション：

- e, --engine=ENGINE 使用する T_EX エンジン・フォーマットを指定する。ENGINE は以下のいずれかを指定する：pdf_latex, pdf_tex, lua_latex, lua_tex, lua_jittex, xelatex, xetex, latex, etex, tex, platex, eptex, ptex, up_latex, eup_tex, uptex. 必須。
- o, --output=FILE 出力ファイル名を指定する。デフォルト：JOBNAME.FORMAT
- fresh 補助ファイルを削除してから処理を行う。--output-directory との併用はできない。
- max-iterations=N 相互参照の解決等のために最大何回処理を行うかを指定する。デフォルト：4
- watch[=ENGINE] 入力ファイルを監視する。別途、fswatch プログラムまたは inotifywait プログラムが必要となる場合がある。詳しくはセクション 2.4 を参照。
- color[=WHEN] ターミナルへの出力を色付けする。WHEN は always, auto, never のいずれかを指定する。--color 自体を省略した場合は auto, WHEN を省略した場合は always が使用される。
- includeonly=NAMEs \includeonly{NAMEs} を挿入する。
- make-depends=FILE Makefile 用の依存関係を FILE に書き出す。
- engine-executable=COMMAND 実際に使う T_EX コマンドを指定する。
- tex-option=OPTION, --tex-options=OPTIONS T_EX に追加のオプションを渡す。

^{*1} <https://github.com/minoki/cluttex>

--dvipdfmx-option=OPTION, --dvipdfmx-options=OPTIONS dvipdfmx に追加のオプションを渡す。

--[no-]change-directory $\TeX$ の実行時に、出力ディレクトリに移動する。シェルエスケープするパッケージを利用する場合に有用となる可能性がある。

-h, --help

-v, --version

-V, --verbose

--print-output-directory --output-directory の値を標準出力に出力して、そのまま終了する。

--package-support=PKG1[,PKG2,...,PKGn] 外部コマンドを実行するパッケージ用の個別の対策を有効にする。現在のところ、minted と epstopdf および pdfx に対応している。

--check-driver=DRIVER いくつかのパッケージについて、正しいドライバーファイルが読み込まれていることを検査する。DRIVER は dvipdfmx, dvips, or dvisvgm のいずれかである。このオプションは --output-format=dvi が指定された場合にのみ指定できる。

--source-date-epoch=TIME このオプションを指定すると、環境変数 SOURCE_DATE_EPOCH を使用して PDF の作成日時を固定する。TIME は文字列 now もしくは、符号なし整数（大抵の環境では Unix 時間）を指定する。

--config-file=TOML 指定された設定ファイルを使用する。セクション 2.12 も参照。

補助コマンド実行用のオプション：

--makeindex=COMMAND MakeIndex を実行する。

--bibtex=COMMAND Bib $\TeX$ を実行する。

--biber[=COMMAND] Biber を実行する。COMMAND のデフォルト値：biber

--makeglossaries[=COMMAND] makeglossaries を実行する。このオプションは試験的なものである。

$\TeX$ 互換オプション：

--[no-]shell-escape

--shell-restricted

--synctex=NUMBER Sync $\TeX$ 用のファイルを生成する。注意点として、.synctex.gz ファイルは .pdf ファイルと同じディレクトリに生成される。詳しくはセクション 2.3 を参照。

--[no-]file-line-error デフォルト：Yes

--[no-]halt-on-error デフォルト：Yes

--interaction=STRING STRING は batchmode, nonstopmode, scrollmode, errorstopmode のいずれか。デフォルト：nonstopmode

--jobname=STRING

--fmt=FORMAT

--output-directory=DIR ($\TeX$ 処理系にとっての) 出力ディレクトリを指定する。補助ファイルはここで指定されたディレクトリに生成される。デフォルト：設定項目 temporary-directory に基づく。

--output-format=FORMAT 出力フォーマットを指定する。pdf または dvi を指定できる。デフォルト：pdf

長いオプションは基本的にハイフンを二つ必要とするが、 $\TeX$ 互換オプションに関してはハイフンが一つ

でも受理される（例：`-color` は受理されないが `-synctex=1` は受理される）。短いオプションを複数繋げる書き方には対応していない（例：`-Ve pdflatex` とは書けない）。

2.3 SyncTeX

`--synctex=1` オプションを使うと SyncTeX 用のファイルを生成させる。

ClutTeX のモットーは「作業ディレクトリを汚さない」であるが、`.synctex.gz` ファイルに関しては PDF ファイルと同じ場所に生成される。これは、`.synctex.gz` ファイルが PDF ファイルと同じ場所にないと SyncTeX が動作しないためである。

2.4 監視モード

ClutTeX に `--watch` オプションを指定して起動した場合、文書の処理後に**監視モード**に入る。

Windows 上では、ClutTeX 単体でファイルシステムの監視を行う。一方で、それ以外の OS（Unix 系）では、`fswatch`^{*2} プログラムまたは `inotifywait` プログラムが予めインストールされている必要がある。補助プログラムは自動検出されるが、ENGINE 引数で明示的に指定することもできる。

2.5 MakeIndex や BibTeX

MakeIndex や BibTeX を使って処理を行う場合は、`--makeindex` や `--bibtex` 等のオプションを指定する。オプションの引数としては、実際に処理に使うコマンド名（`makeindex` や `mindex`）を指定する。

Biber を使って文献リストを処理する場合、使用すべきオプションは `--bibtex=biber` ではなく `--biber` である。

2.6 大規模な文書を書く場合

LaTeX で大きな文書を書く場合は `\include` コマンドによってファイル分割を行うことが多いだろう。この際に `\includeonly` コマンドを使うと、処理時に「一部のファイルしか処理しない」ようにできて、処理時間の削減ができる。しかし、`\includeonly` コマンドを TeX ソース中に記述していちいち切り替えるのは面倒である。

そこで、ClutTeX では `\includeonly` コマンドを `--includeonly` オプションによって指定できるようにした。使用例はセクション 2.7 を参照せよ。

Tips: `includeonly` を使用する際は、`--makeindex` 等のオプションは使用しない方が良い。

処理時間の削減方法として、`--max-iterations=1` を指定するという手もある。デフォルトでは ClutTeX は相互参照等を正しくするために TeX を複数回実行する。だが、大規模な文書であれば TeX を一回実行するのには数十秒や数分かかり、複数回実行すればその数倍の時間がかかる。作業中の文書に関してそれだけの時間をかけて相互参照等を正しくするのは時間の無駄であろう。であれば、作業中の文書に関しては `--max-iterations=1` を指定して TeX の実行回数を最小限に止めることが有効と考えられる。

^{*2} <http://emcrisostomo.github.io/fswatch/>

2.7 Makefile と組み合わせる

各プロジェクトに応じたコマンドを毎回打ち込むのは大変なので、Makefile と組み合わせると良いだろう。例：

```
main.pdf: main.tex chap1.tex chap2.tex
    cluttex -e lualatex -o $@ --makeindex=mendex $<

main-preview.pdf: main.tex chap1.tex chap2.tex
    cluttex -e lualatex -o $@ --makeindex=mendex --max-iterations=1 $<

chap1-preview.pdf: main.tex chap1.tex
    cluttex -e lualatex -o $@ --max-iterations=1 --includeonly=chap1 $<

chap2-preview.pdf: main.tex chap2.tex
    cluttex -e lualatex -o $@ --max-iterations=1 --includeonly=chap2 $<
```

--make-depends オプションを使うと、依存関係を Makefile のルールとしてファイルに書き出すことができる。これを使うと、main.tex, chap1.tex, chap2.tex の 3 つのファイルからなる文書を以下の Makefile で処理させることができる。この際、main.pdf の依存先に chap1.tex と chap2.tex を明示しなくても良い。

```
main.pdf: main.tex
    cluttex -e lualatex -o $@ --make-depends=main.pdf.dep $<

-include main.pdf.dep
```

ただし、--make-depends オプションはまだ実験的なものであり、--makeindex 等の他の機能との組み合わせがうまく動かなかったり、将来のバージョンで仕様変更されるかもしれない。

2.8 出力ディレクトリについて

デフォルトでは、.aux ファイル等の補助ファイルは、テンポラリディレクトリ以下の適当なディレクトリに生成される。このディレクトリ名は、以下の 3 要素に依存する：

- 入力ファイルの絶対パス
- --jobname オプション
- --engine オプション

一方、以下の要素はディレクトリ名に影響しない：

- --includeonly
- --makeindex, --bibtex, --biber, --makeglossaries

もし何らかの事情で自動生成された出力ディレクトリの位置を知りたいければ、ClutT_EX を `--print-output-directory` オプションを使うとよい。例えば、Makefile の `clean` ターゲットは次のように書ける：

```
clean:
    -rm -rf $(shell cluttex -e pdflatex --print-output-directory main.tex)
    -rm main.pdf
```

出力ディレクトリに生成された補助ファイルは、`--fresh` オプションを指定しない限り、ClutT_EX が消去することはない。一方、テンポラリディレクトリを使用するということは、PC の再起動時に補助ファイルが削除される可能性があるということでもある。

2.9 エイリアス

Unix 用コマンドの中には、自身の名前によって挙動を変えるものがある。つまり、あるコマンドに対してシンボリックリンクによって別名をつけると、元のコマンドと別名によって挙動を変える。T_EX Live でも、

- `extractbb`, `dvipdfmx` は `xdvipdfmx` へのエイリアス
- `repstopdf` は `epstopdf` へのエイリアス

という例がある。

`cluttex` が `cl<エンジン名>` として呼び出された場合、使用されるエンジン名 (`--engine` オプション) がそれに指定される。

例えば、`cllualatex` は `cluttex --engine lualatex` の別名であり、`clxelatex` は `cluttex --engine xelatex` の別名である。

2.10 minted と epstopdf への対策

一般に、外部コマンド実行 (シェルエスケープ) を行うパッケージは `-output-directory` を指定した際に正常に動作しない。したがって、ClutT_EX の下ではそういうパッケージはうまく動かない。

一方で、パッケージによっては `-output-directory` の値を指示するためのパッケージオプションを持っているものがある。例えば、`minted` の `outputdir` オプション、`epstopdf` の `outdir` オプションがそれである。

ClutT_EX からこれらのパッケージオプションを指定することはできるが、そのためには使用するパッケージを ClutT_EX が事前に知っておかねばならない。使用するパッケージを ClutT_EX に知らせるには、`--package-support` オプションを使う。

例えば、`minted` を使う文書进行处理の場合は次のように実行すれば良い：

```
cluttex -e pdflatex --shell-escape --package-support=minted document.tex
```

2.11 ドライバーファイルの検査

Clut_TE_X は、いくつかのパッケージについて正しいドライバーファイルが読み込まれていることを検査することができる。現在のバージョンで対応しているパッケージは `graphics(x)`, `color`, `expl3`, `hyperref`, `xy` である。

PDF モードの場合、ドライバーの検査は常に行われる。DVI モードで検査を有効にするには、`--check-driver` オプションを使用する。

2.12 設定ファイルによるカスタマイズ

Clut_TE_X のいくつかの挙動は、設定ファイルによりカスタマイズできる。

Clut_TE_X の設定ファイルは、以下の順序で探索される：

1. `--config-file` コマンドラインオプション。
2. `CLUTTEX_CONFIG_FILE` 環境変数。
3. (Unix のみ) `$XDG_CONFIG_HOME/cluttex/config.toml`
4. (Unix のみ) `$HOME/.config/cluttex/config.toml`
5. (Windows のみ) `%APPDATA%\cluttex\config.toml`

Clut_TE_X の設定ファイルは TOML^{*3}形式に従う。

設定可能なキーは以下の通りである：

`temporary-directory` (文字列) 補助ファイルの作成先。 `--output-directory` のデフォルト値は `temporary-directory` 以下のどこかとなる。省略された場合は、テンポラリディレクトリ以下のどこかが使用される。

`color.{type,execute,warning,diagnostic,information}` (テーブル) Clut_TE_X によるメッセージの色を指定する。それぞれのテーブルは以下のキーを持てる：`fore` (色を表す文字列)、`back` (色を表す文字列)、`bold` (真理値)、`dim` (真理値)、`underline` (真理値)、`blink` (真理値)、`reverse` (真理値)、`italic` (真理値)、`strike` (真理値)。色を表す文字列は以下のいずれかである：`default`、`black`、`red`、`green`、`yellow`、`blue`、`magenta`、`cyan`、`white`、`brightblack`、`brightred`、`brightgreen`、`brightyellow`、`brightblue`、`brightmagenta`、`brightcyan`、`brightwhite`。

設定ファイルの例は以下のようになる：

```
# 実際の出力ディレクトリは
# "/home/user/.cache/cluttex/cluttex-<hash>"
# のようになる
temporary-directory = "/home/user/.cache/cluttex"
```

[color]

^{*3} <https://toml.io/en/>

```
type = { reverse = true }  
error = { fore = "brightred", back = "brightwhite" }
```