

ClutT<sub>E</sub>X マニュアル  
(バージョン 0.3)

ARATA Mizuki

2019 年 4 月 30 日

# 目次

第 1 章	Clut <sub>T</sub> E <sub>X</sub> の概要	2
第 2 章	Clut <sub>T</sub> E <sub>X</sub> 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 <sub>T</sub> E <sub>X</sub> . . . . .	4
2.4	監視モード . . . . .	5
2.5	MakeIndex や Bib <sub>T</sub> E <sub>X</sub> . . . . .	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 . . . . .	5
2.8	出力ディレクトリについて . . . . .	6
2.9	エイリアス . . . . .	6

## 第 1 章

# ClutT<sub>E</sub>X の概要

ClutT<sub>E</sub>X は、L<sup>A</sup>T<sub>E</sub>X 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<sup>\*1</sup>)
- MakeIndex, BibT<sub>E</sub>X, 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)
- pT<sub>E</sub>X 系列の処理系で PDF を生成する場合、別途 `dvipdfmx` を実行する必要がない (自動で `dvipdfmx` を実行する)

などがある。

L<sup>A</sup>T<sub>E</sub>X 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 ClutT<sub>E</sub>X 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、ClutT<sub>E</sub>X を使わない場合に比べて ClutT<sub>E</sub>X 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<sup>\*2</sup>。

---

<sup>\*1</sup> Unix 系 OS では、別途プログラムが必要。

<sup>\*2</sup> 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<sub>E</sub>X の使い方

### 2.1 インストール

ClutT<sub>E</sub>X は最新の T<sub>E</sub>X Live に収録されている。よって、T<sub>E</sub>X Live を利用している方は、T<sub>E</sub>X Live の更新（コマンドなら `tlmgr upgrade --all`）を行えば ClutT<sub>E</sub>X 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<sup>\*1</sup>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<sub>E</sub>X 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：`pdflatex`, `pdftex`, `lualatex`, `luatex`, `luajitte`, `xelatex`, `xetex`, `latex`, `etex`, `tex`, `platex`, `eptex`, `ptex`, `uplatex`, `euptex`, `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：3
- watch 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。
- tex-option=OPTION, --tex-options=OPTIONS T<sub>E</sub>X に追加のオプションを渡す。
- dvi-pdfmx-option=OPTION, --dvi-pdfmx-options=OPTIONS dvi-pdfmx に追加のオプションを渡す。

---

<sup>\*1</sup> <https://github.com/minoki/cluttex>

`--[no-]change-directory`  $\mathrm{T}_{\mathrm{E}}\mathrm{X}$  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`

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` `BibTEX` 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。

$\mathrm{T}_{\mathrm{E}}\mathrm{X}$  互換オプション：

`--[no-]shell-escape`

`--shell-restricted`

`--synctex=NUMBER` `SyncTEX` 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` ( $\mathrm{T}_{\mathrm{E}}\mathrm{X}$  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：テンポラリディレクトリのどこか

`--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、 $\mathrm{T}_{\mathrm{E}}\mathrm{X}$  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 SyncT<sub>E</sub>X

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

`ClutEX` 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

Clut $\text{\TeX}$  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、Clut $\text{\TeX}$  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`<sup>\*2</sup> 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。

## 2.5 MakeIndex や Bib $\text{\TeX}$

MakeIndex や Bib $\text{\TeX}$  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 `mendex`) 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

L $\text{\TeX}$  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  $\text{\TeX}$  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、Clut $\text{\TeX}$  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よい。

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 Clut $\text{\TeX}$  は相互参照等を正しくするために  $\text{\TeX}$  を複数回実行する。だが、大規模な文書であれば  $\text{\TeX}$  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  $\text{\TeX}$  の実行回数を最小限に止めることが有効と考えられる。

## 2.7 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 $<
```

---

<sup>\*2</sup> <http://emcrisostomo.github.io/fswatch/>

```
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が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`

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、Clut $\text{\TeX}$  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。 $\text{\TeX}$  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。