

2018年でもEPSを \TeX で 使う

鹿野 桂一郎
ラムダノート株式会社
k16.shikano@lambdanote.com
@golden_lucky

2018年11月10日
於 TeXConf 2018

**T_EX では、画像を
EPS で埋め込みます**

~~TeXでは、画像を
EPSで埋め込みます~~

そもそもEPS とはなにか

- Encapsulated **PostScript**
- **PostScript** は、印刷のためのデバイスに依存しない、プログラミング言語
- それだと実際の印刷に使いにくいので、**DSC** という規約がある

EPS は DSC 準拠のデータ交換形式

- **Document Structuring Convention**
- アプリケーションに依存する情報とか、そのファイルがどういう構造になっているとか、そういう情報を埋め込むときの決まりごと
- 他の **PostScript** ファイルから取り込めるような単ページの絵としてのファイル形式を、**DSC** に従って決めたものが、**EPS**

```
%!PS-Adobe-3.1 EPSF-3.0
:
%%BoundingBox: 0 0 300 100
```

T_EX は EPS を選んだ

- おそらく、ほかの選択肢が事実上なかった
 - Plain T_EX の epsf.sty マクロはクヌースの手がかかっている
- ただし、T_EX そのものは **PostScript** を「完全には」処理できない
 - DVI における外部画像の扱いは、「special」を使って後段のデバイスに丸投げ」が基本
 - pdfT_EX では、EPS は非サポート

Ghostscript

- ① **T_EX** エンジンが画像処理のたびに呼び出す
- ② **DVI** ウェアが画像処理のたびに呼び出す
- ③ だったら最初から画像をすべて **PDF** に変換して、それを **pdfT_EX** や **dvipdfmx** が直接 **PDF** に埋め込めるようにすべき、というのが最近の潮流

Ghostscript

- ① **T_EX** エンジンが画像処理のたびに呼び出す
- ② **DVI** ウェアが画像処理のたびに呼び出す
- ③ だったら最初から画像をすべて **PDF** に変換して、それを **pdfT_EX** や **dvipdfmx** が直接 **PDF** に埋め込めるようにすべき、というのが最近の潮流

「**T_EX** は **Ghostscript** から離れては生きていけないのよ」

**EPSは、いまやほんとうに
いらない子なのだろうか？**

それでもEPS を使いたいこともある

- **EPS ファイルをもらったけど Adobe Illustrator で開いたら微妙な状況になった**
- **画像をテキスト形式でバージョン管理したい**

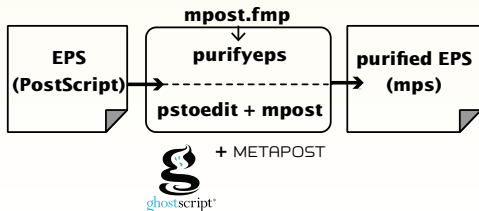
**T_EX（広義）だけで
PostScript を処理できれば
いいのに……**

METAPOST 由来のEPS

- METAPOST の EPS 出力は、複雑な PostScript コードを含まない (*purified EPS*)
- T_EX のエコシステムだけで処理できる、良い EPS
 - pdfT_EX と LuaT_EX は、EPS なら自分で処理する！
 - dvipdfmx も、METAPOST を処理できる！
 - いずれも .mps という拡張子にする必要あり
 - dvipdfmx には、.mps を直接 PDF にする -M オプションもある

EPS を purify する

- その名も purifyeps
- pstoeedit で METAPOST のソースに変換し、その結果を mpost にかけるだけの Perl スクリプト
 - もとの EPS の PostScript は、前段の pstoeedit が解釈してくれる
 - 後段の mpost で使うフォントマップを purifyeps に指定してくれる
 - METAPOST が知っているフォント名を適当に当てはめてあげる必要がある



pstoedit がすごい

- **PDF や EPS を、さまざまな画像ファイルに変換してしまう、グラフィックス界の **pandoc****
 - 実は **Ghostscript** のラッパー
 - **Ghostscript** 本体では非推奨になった **DELAYBIND** がデフォルトで有効という罠がある
 - 結果として、イラレなどで生成された **EPS** の多くは、素の **pstoedit** (したがって **purifyeps**) で変換しようとする
と意味不明な **PostScript** エラーになる
 - **DELAYBIND** を無効にするには、**purifyeps** のソースで **pstoedit** を呼んでいる部分で、**-nb** オプションを指定しなければならない

pstoedit がすごい

- **PDF や EPS を、さまざまな画像ファイルに変換してしまう、グラフィックス界の pandoc**
 - 実は **Ghostscript** のラッパー
 - **Ghostscript** 本体では非推奨になった **DELAYBIND** がデフォルトで有効という罠がある
 - 結果として、イラレなどで生成された **EPS** の多くは、素の pstoedit (したがって purifyeps) で変換しようとする
と意味不明な **PS エラー** になる
 - **DELAYBIND** を無効にするには、purifyeps のソースで pstoedit を呼んでいる部分で、**-nb** オプションを指定しなければならない

「**T_EX** は Ghostscript から (ry

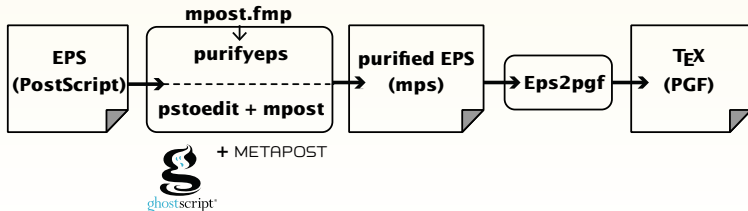
ここまでできたら……

- **purified EPS** をネイティブの **T_EX** ソースに変換できないか？
- **TikZ** のインタフェースでは **PostScript** のシンタックスと違いすぎる
- それなら **PGF**
 - いまではドキュメントでも **TikZ** と同じインタフェースのように扱われているが、**PGF** 独自のインタフェースはかなり **PostScript** っぽい

Eps2pgf

- すでにあった
- **PostScript** 処理系ではなく、**METAPOST** の出力した **EPS** から **PGF** へのコンバーター
- Java 製でメンテもされてなさそう
- **Sourceforge** にポストされているがソースがない
- 不安材料はあるけど、とにかく動く

```
$ java -jar eps2pgf.jar image.eps -o image.tex
```

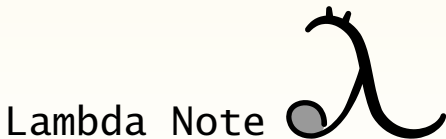


EPS を T_EX として使う

- 本文と図のフォントが当然のように一致する。画期的
- もちろんテキストなのでバージョン管理も簡単
- 座標で位置あわせを数値指定できる
- 複数の画像ファイルでパーツを一気に差し替える、みたいなことも可能
 - パーツだけを PGF 内で `\includegraphics` すればいい
- T_EX にした図中では、日本語もまともな組版で使える
 - 元の EPS を日本語で作ってしまうと `pstoedit` によってアウトライン化されてしまうけど

まとめ

- このスライドはすべて `.tex` だけで作られています
- **T_EX** は **Ghostscript** から離れては生きていけない
- METAPOST は福音かも
- ラムダノート株式会社は出版を中心として技術文書まわりのお手伝いをいろいろする会社です
 - <https://lambdanote.com>



参考資料

- **“The epsf package”,**
<http://tug.ctan.org/macros/generic/epsf/epsf-doc.pdf>
Plain T_EX で **EPS** を取り込むのに使われる **epsf** パッケージのマニュアル。ク
ヌースの関与もわかる。
- **“The Dvipdfmx User’ s Manual”** [http://www.tug.org/texlive/
/devsrc/Master/texmf-dist/doc/dvipdfmx/dvipdfmx.pdf](http://www.tug.org/texlive/devsrc/Master/texmf-dist/doc/dvipdfmx/dvipdfmx.pdf)
dvipdfmx のマニュアル。**dvipdfmx** における画像の扱いの考え方がわかる。
- **“The pdfTEX user manual”,**
<http://texdoc.net/texmf-dist/doc/pdftex/manual/pdftex-a.pdf>
pdfT_EX のマニュアル。**pdfT_EX** における画像の扱いの考え方がわかる。**Heiko
Oberdiek** 氏による **epstopdf** パッケージのドキュメントもよい資料 (**“The
epstopdf package”,** [http://mirrors.ctan.org/macros/latex/
contrib/oberdiek/epstopdf.pdf](http://mirrors.ctan.org/macros/latex/contrib/oberdiek/epstopdf.pdf))。
- **“METAPOST, a user’s manual”,**
<https://www.tug.org/docs/metapost/mpman.pdf>
METAPOST のマニュアル

参考資料（つづき）

- **“Ghostscript and the PostScript Language”,**
<https://www.ghostscript.com/doc/9.20/Language.htm>
PostScript の **bind** を **Ghostscript** では **.bind** として再定義していました、ということが書いてある。
- **“PS interpreter - remove superexec from systemdict”,**
<http://git.ghostscript.com/?p=ghostpdl.git;a=commitdiff;h=8556b698892e4706aa0b9d996bec82fed645eaa5>
DELAYBIND は、**PostScript** の **bind** コマンドの動作をちょっと変えることで、標準ライブラリのコマンド名を上書きしているような **ps** ファイルを扱えるようにするための **Ghostscript** 独自の仕掛け。**Adobe Distiller** が隠し持っている **internaldict** 辞書进行操作する **superexec** というコマンドが **systemdict** にあったのを取り除いたときに、副作用があるので除去された。そのときのコミット。
- **“PostScript Language Reference Manual”**
第 2 版がアドビシステムズジャパン監訳で翻訳されている。幸い、**DSC** と **EPS** については第 3 版より第 2 版のほうがわかりやすい。
- **“Eps2pgf”** <https://sourceforge.net/projects/eps2pgf/>
SourceForge の **Eps2pgf** の配布サイト。
- **“User’s Guide to the PGF Package, Version 0.61”**
<https://www.tuteurs.ens.fr/noncvts/docs/pgf/pgfuserguide.pdf>
2004 年ころの **PGF** (非 **TikZ**) のマニュアル (全 25 ページ!)。