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

TEX 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かとか、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
```

TeX 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 TeX の `epsf.sty` マクロはクヌースの手がかかっている
- ただし、TeX 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
  - pdfTeX では、EPS は非サポート

Ghostscript

1. TEX 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 pdfTEX 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. TEX 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 pdfTEX 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

「TEX 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

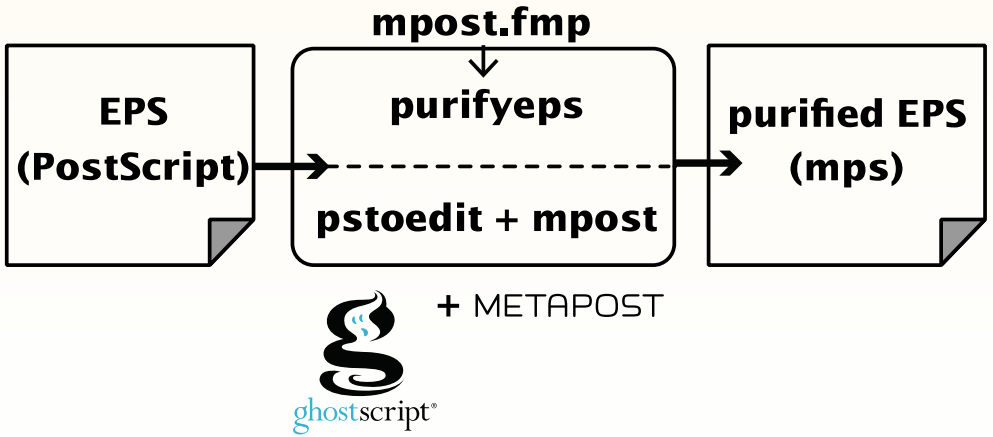
TEX（広義）だけで PostScript を処理できればいいのに……

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***）
- **TEX** のエコシステムだけで処理できる、**良い EPS**
  - **pdfTEX** と **LuaTEX** 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
- pstoedit 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 pstoedit が解釈してくれる
  - 後段の mpost で使うフォントマップを pstoedit 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

「TEX は Ghostscript から (ry

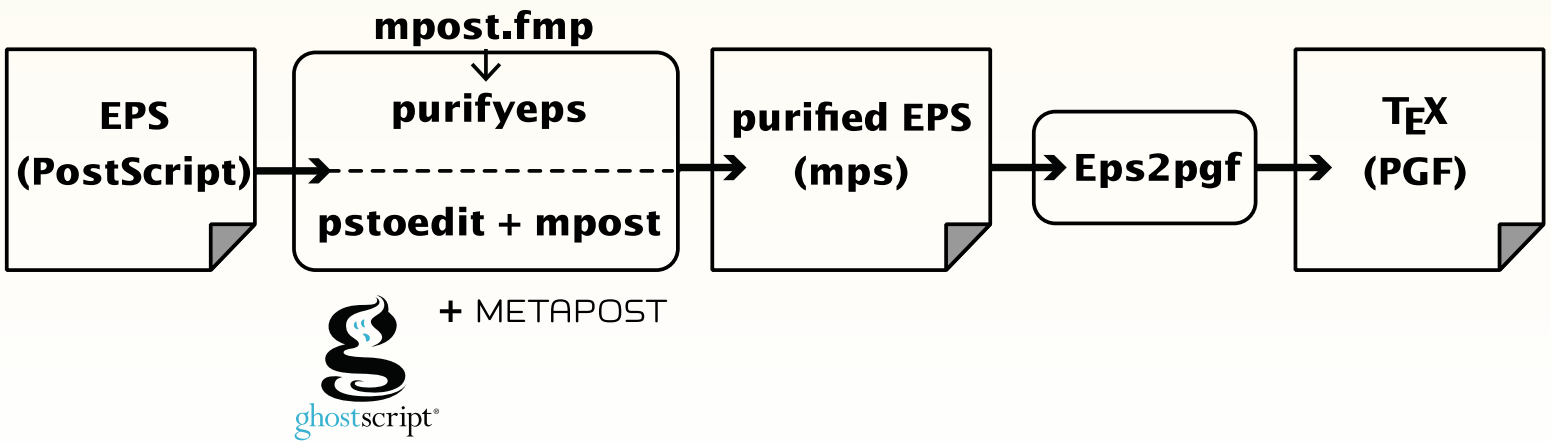
ここまできたら……

- purified EPS をネイティブの TEX 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
```

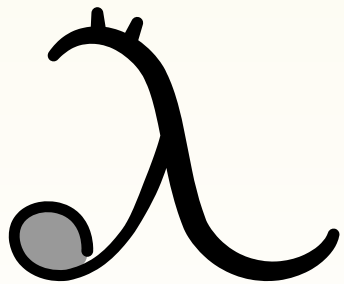


EPS を TEX として使う

- 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
- TEX 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
- TEX 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

Lambda Note 

参考資料

- **“The epsf package”,**
  http://tug.ctan.org/macros/generic/epsf/epsf-doc.pdf
  Plain TEX 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
  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
  pdfTEX のマニュアル。pdfTEX 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 **)。**
- **“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 **辞書を操作する** 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/noncvs/docs/pgf/pgfuserguide.pdf
  2004 年ころの PGF（非 TikZ）のマニュアル（全 25 ページ！）。