

pxjahyper パッケージ

八登崇之 (Takayuki YATO; aka. “ZR”)

v0.9b [2021/05/29]

1 概要

(u)p^ET_EX + hyperref + dvipdfmx の組み合わせで日本語を含む「しおり」をもつ PDF 文書を作成する場合に必要な機能を提供する。

- dvipdfmx 用の「tounicode special」について、内部漢字コードに応じて適切なものを出力する。
- PDF 文字列の中で LICR (`\a` や `\textsection` 等の文字出力の命令) が正しく機能するようにする。ただし、エンジンが p_ET_EX の場合は、out2uni/convbkmk を利用する場合を除き、JIS X 0208 にない文字は出力できない (hyperref の警告が出る)。
- T_EX の版面拡大機能が使われている (`\mag` が 1000 でない; 典型的には jsclasses の文書クラスで基底フォントサイズが 10 pt 以外である) 場合に、hyperref が出力する papersize special の紙面サイズの値が不正になるが、この場合に `\mag` 値を考慮して求めた正しいサイズによる papersize special を改めて出力する。

2 パッケージの読込

`\usepackage` で読み込む。

```
\usepackage[オプション,...]{pxjahyper}
```

使用可能なオプションは以下の通り。

- **ドライバオプション**: dvipdfmx、dvips、および nodvidriver (ドライバ依存動作の無効化を明示的に指示、現状ではほとんどの動作が無効になる) が指定できる。
※ 0.5 版より、nodvidriver の別名の none は非推奨の扱いとする。
- **auto** (既定): ドライバオプションを hyperref のドライバ指定から自動的に判定する。
- **tounicode** (既定): dvipdfmx 用の「tounicode special」を発行し、それを前提にした出力を行う。
- **notounicode**: dvipdfmx 用の「tounicode special」を発行しない。
- **out2uni**: out2uni フィルタを使うことを前提にした出力を行う。
- **convbkmk**: convbkmk フィルタを使うことを前提にした出力を行う。
※ ドライバが dvips の場合は既定で有効になる。
- **otfutf** (既定): japanese-otf パッケージの `\UTF` 命令を PDF 文字列中で使えるようにする。

※詳細は 3.2 節を参照。

- `nootfutf` : `otfutf` の否定。
- `otfcid` (既定) : `japanese-otf` パッケージの `\CID` 命令を PDF 文字列中で“可能な限り”使えるようにする。

※詳細は 3.2 節を参照。

※ `otfcid` の利用には、エンジンの ϵ -T_EX 拡張および `etoolbox` と `bxjatoucs` パッケージのインストールが必要。

- `nootfcid` : `otfcid` の否定。
- `disablecmds` (既定) : 「PDF 文字列中のテキスト装飾命令の無効化」を有効にする。

※詳細は 3.4 節を参照。

- `nodisablecmds` : `disablecmds` の否定。
- `otfmacros` (既定) : `japanese-otf` 付属の `ajmacros` パッケージが提供する文字入力命令 (`\ajMaru`、`\ajLig` 等) を PDF 文字列中で“可能な限り”使えるようにする。

※詳細は 3.2 節を参照。

※ `otfmacros` を有効にする場合は `otfcid` も有効にする必要がある。

- `nootfmacros` : `otfmacros` の否定。
- ※ 0.6 版より既定を `otfmacros` に変更した。
- `bigcode` (既定) : `upTEX` での ToUnicode CMap として既定の UTF8-UCS の代わりに UTF8-UTF16 を用いる。(当該のファイルが存在する必要がある。)
- `nobigcode` : `bigcode` の否定。
- ※ 0.3a 版より既定を `bigcode` に変更した。
- `papersize` (既定) : `papersize special` の補正を有効にする。
- `nopapersize` : `papersize` の否定。

3 機能

「概要」で述べた機能は (オプション設定に応じて) 自動的に実施される。

3.1 Unicode 符号値による入力

PDF 文字列入力中で、`\Ux` が以下の意味になる。PDF 文字列以外では `\Ux` は以前の定義 (または未定義) に戻る。^{*1}

- `\Ux{Unicode 符号値 16 進}` : その符号値の文字を出力する。具体的な動作は以下の通り :
 - エンジンが `upLATEX` のときは、当該の文字自体を書いたのと同等になる。
 - エンジンが `pLATEX` で `out2uni` または `convbkmk` が有効の場合は、エスケープ表記 (`\0xUUUU`) を出力する。

^{*1} `\Ux` という命令名は `bxbase` パッケージの Unicode 符号値入力用の命令が使っているものである。従って、`bxbase` パッケージを読み込んでいれば、「PDF 文字列と版面出力の両方に使われる」ようなテキストにおいて、`\Ux` で Unicode 符号値入力が可能になる。ただし、Unicode 符号値入力用の命令としては「`japanese-otf` パッケージの `\UTF` 命令」の方が有名であり、`pxjahyper` は `\UTF` も正しく扱えるので、こちらを使う方が無難かもしれない。

- それ以外の場合は、`\Ux` は定義されない。

3.2 japanese-otf パッケージの文字入力命令への対応

■**\UTF 命令** `\UTF` 命令は、PDF 文字列中では `out2uni` 用の出力を行うように設計されている。本パッケージで `otfutf` オプションを有効にした場合は、PDF 文字列中の動作が以下のように変更される。

- `\Ux` 命令 (3.1 節参照) が有効の場合は、`\Ux` と同じ動作になる。
- それ以外の場合 (エンジンが `pLATEX` で `tounicode` を前提にする場合) は、警告を出した上でゲタ記号を出力する。
※ `pLATEX` では `tounicode` の機能で Unicode 文字を出力できないため。

※`\UTF` 命令の多言語版、すなわち `\UTFC`・`\UTFK`・`\UTFM`・`\UTFT` 命令も `\UTF` と同じ扱いになる。

■**\CID 命令** `japanese-otf` パッケージの仕様では `\CID` 命令は、PDF 文字列中ではサポートされない (未定義動作となる)。本パッケージで `otfcid` オプションを有効にした場合は、PDF 文字列中で `\CID` が“可能な限り”使えるようにする。具体的な仕様は以下の通り。

- 当該の AJ1 のグリフに“対応”する**単独の** Unicode 文字があればそれを出力し、なければ (警告を出した上で) 削除する。^{*2}
- ただし `\Ux` 命令 (3.1 節参照) が無効になる場合は、そもそも Unicode 文字も出力できないため、常に (警告を出した上で) ゲタ記号を出力する。結局情報は欠落するが、それでも未定義動作 (エラーになりえる) よりは好ましいであろう。

※ `otfcid` の利用には、エンジンの `ε-TEX` 拡張および `etoolbox` と `bxjatoucs` パッケージのインストールが必要。

※`\CID` 命令の多言語版はサポートされない。

■**ajmacros パッケージの命令** 本パッケージで `otfmacros` オプションを有効にした場合は、`japanese-otf` 付属の `ajmacros` パッケージが提供する文字入力命令 (`\ajMaru`、`\ajLig` 等) を PDF 文字列中で“可能な限り”使えるようにする。具体的な仕様は以下の通り。

- 現状では、`\Ux` 命令が有効になる場合のみがサポートされる。
※それ以外の場合は `otfmacros` オプションは無効になる。
- Unicode 文字で表現可能であればそれを出力し、なければ代替表現を出力する。
- Unicode 文字を出力する場合の仕様は `\CID` と同じ。代替表現の場合は文字自体が出力されることもある。

※ `otfmacros` を有効にする場合は `otfcid` も有効にする必要がある。(従って `otfcid` と同じ前提条件が課される。) `nootfcid` を指定した場合は `otfmacros` も無効になる。

※ `ajmacros` パッケージの多くの命令は“脆弱 (fragile)”である。そのため、節見出し (`\section` 等の引数)

^{*2} 他の同様の場合と合わせてゲタ記号を出力するようにしたい。1.0 版において仕様を変更する予定。

で `\ajMaru` 等の命令を使いたい場合は、命令の前に `\protect` 付ける必要がある。^{*3}

3.3 PDF 文字列用の文字命令のユーザ定義

以下の命令が提供される。(プリアンブルでのみ使用可能。)

- `\pxDeclarePdfTextCommand{\制御綴}{<JIS 符号値>}{<Unicode 符号値>}`: PDF 文字列中の `\制御綴` の動作として、指定した符号値の文字を出力することを指定する。
- `\pxDeclarePdfTextComposite{\制御綴}{<引数>}{<JIS 符号値>}{<Unicode 符号値>}`: PDF 文字列中の `\制御綴` (アクセント命令) + `<引数>` の動作として、指定した符号値の文字を出力することを指定する。

これらの命令において、符号値は 16 進数で指定する。「JIS 符号値」は `upLaTeX` では使われないので省略して (空にして) もよい (或いはそもそも JIS X 0208 にない文字の場合は省略する)。逆に「Unicode 符号値」は `pLaTeX` の動作でかつ「JIS 符号値」が指定されている場合は省略してよい。

例えば、以下のように定義しておくと、PDF 文字列中で `\textschwa` (schwa 記号) や `\d{t}` (`t̥`) が使えるようになる。

```
\pxDeclarePdfTextCommand{\textschwa}{}{0259}  
\pxDeclarePdfTextComposite{\d}{t}{}{1E6D}
```

3.4 PDF 文字列用中のテキスト装飾命令の無効化

PDF 文字列は単なる Unicode 文字列として扱われるものなので、`\textit` や `\large` 等のテキスト装飾用の命令は意味をなさず、またそれらの命令の実装は PDF 文字列の解釈中は正常に処理できない。PDF 文字列と版面出力の両方に使われるテキスト (節見出し等) についてテキスト装飾命令が支障なく使えるように、`hyperref` では基本的なテキスト装飾命令 (多くは `LaTeX` カーネルが提供するもの) について、「PDF 文字列として扱う場合は自動的に無力化^{*4}する」機構を実装している。これにより、例えば節見出しのテキストに “`\textit{text}`” が含まれていたとすると、版面に出力する場合には “*text*” のように装飾が施され、一方で、PDF 文字列としては “`text`” と解釈されることになる。

0.5 版以降の `pxjahyper` では、この無効化の対象に「和文用のテキスト装飾命令 (およびそれに準じるもの)」を追加するようになった。以下の命令が対象になる。

- 和文のフォント選択命令: `\textmc \gtfamily \kanjifamily \useroman \userelfont` など
- 次の `pLaTeX` カーネル命令: `\<`
- 次の `pTeX` プリミティブ: `\inhibitglue \noautospace \noautoxspace`
- 次の `plext` の命令: `\bou \kasen \rensuji`
- 次の `japanese-otf` の命令: `\textmg \mgfamily \ltseries \ebseries \propshape`

^{*3} ちなみに、引数が PDF 文字列として解釈される場合には、`\protect` は全く結果に影響しない。

^{*4} 例えば、“`\textit{text}`” や “`{\large text}`” は単に “`text`” と書いたものと見なされる。