

pxjahyper パッケージ

八登崇之 (Takayuki YATO; aka. “ZR”)

v1.4 [2024/08/14]

目次

1	概要	1
2	パッケージの読み込み	2
3	機能	3
3.1	設定	3
3.2	Unicode 符号値による入力	4
3.3	japanese-otf パッケージの文字入力命令への対応	4
3.4	PDF 文字列用の文字命令のユーザ定義	5
3.5	PDF 文字列用中のテキスト装飾命令の無効化	6
4	hyperref の “unicode モード” での動作	7

1 概要

(u)pL^AT_EX + hyperref + dvipdfmx の組み合わせで日本語を含む **PDF 文字列** (“しおり” などの文書情報の文字列^{*1}) 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 L^AT_EX カーネル (`\a` や `\textsection` 等) や japanese-otf パッケージ (`\UTF` や `\ajMaru` 等) の文字出力用命令が “可能な限り” 正しく機能するようにする。
- T_EX の版面拡大機能が使われている (`\mag` 値が 1000 でない) 場合^{*2}には 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。
- hyperref が行う「テキスト装飾命令の無力化」の対象に、(u)pL^AT_EX 特有のいくつかの命令を追加する。

^{*1} 例えば、`\hypersetup` の `pdftitle` キーに指定する文字列は「PDF 文字列」の一種である。また、`\section` 命令の引数のテキストは「T_EX で組版する版面出力」と「PDF 文字列」の両方に使われる。後者のような箇所では「T_EX の版面出力」と「PDF 文字列」を別に指定したい場合は `\texorpdfstring` という命令が利用できる。

^{*2} 典型的には `jsclasses` の文書クラスで基底フォントサイズを 10pt 以外に設定している場合。

2 パッケージの読込

`\usepackage` で読み込む。

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

■**plautopatch との関係** plautopatch パッケージが事前に読み込まれている場合は、hyperref が読み込まれるとその直後に自動的に pxjahyper がオプション無しで読み込まれる。もし何らかのオプションを指定したい場合は、hyperref の読込より前に `\PassOptionsToPackage` でオプションを指定する必要がある。

```
\PassOptionsToPackage[nopapersize]{pxjahyper}
```

■使用可能なオプション

- **設定キー記述**： `\pxjahypersetup` 命令の引数に書く設定記述をパッケージオプションに書くことができる。例えば

```
\usepackage[fallback=delete]{pxjahyper}
```

と指定すると、表現不能文字を代替するゲタ文字が出力されなくなる。

- **ドライバオプション**： 以下の値が指定できる。
 - `dvipdfmx`
 - `dvips`
 - `resetdvidriver` (既定)： hyperref のドライバ指定から自動的に判定する。
※ 1.1 版において名前を `auto` から `resetdvidriver` に変更した。旧名の `auto`^{*3}は**非推奨**とする。
 - `nodvidriver`： ドライバ依存動作の無効化を明示的に指示する。ほとんどの動作が無効になる。^{*4}
- **tounicode** (既定)： 以下の 2 つの設定を行う。
 1. ドライバが `dvipdfmx` である場合は適切な「`tounicode special`」を発行する。
※この設定は `notounicode` オプションにより打ち消される。
 2. 「`tounicode special`」を前提にした出力を行う。
※この設定は `out2uni`・`convbkmk` オプションと排他である。
- **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 `convbkmk` が有効になる。これにより元の既定値 `tounicode` は実質的に^{*5}無効化される。
- **papersize** (既定)： `papersize special` の補正を有効にする。
- **nopapersize**： `papersize` の否定。

*3 1.0 版では `autodvidriver` としていたが、この旧名も**非推奨**とする。

*4 1.0 版において `nodvidriver` の非推奨の別名の `none` は**廃止**された。

*5 `tounicode` の説明中の項目 1 の機能は `dvips` では効果がなく、また項目 2 の機能は上書きされるため。

■上級者向けオプション

- `otfmacros` (既定^{*6}): `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 3.3 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。
- `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 3.3 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。
※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。
※詳細は 3.3 節を参照。
- `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 3.5 節を参照。
- `nodisablecmds`: `disablecmds` の否定。
- `charcmds` (既定): L^AT_EX 標準の文字命令を PDF 文字列中で使えるようにする。
- `nocharcmds`: `charcmds` の否定。
- `bigcode` (既定^{*7}): upTeX での ToUnicode CMap として UTF8-UTF16 を用いる。
- `nobigcode`: `bigcode` の否定。upTeX での ToUnicode CMap として UTF8-UCS2 を用いる。
- `jacommentline` (既定): `hyperref` が出力する `.out` ファイルの先頭に日本語の文字を含むコメント行を出力する。
※ pTeX 系エンジンの「入力漢字コード自動判定」に対する対策。
- `jacommentline`: `jacommentline` の否定。
※不具合が発生したときのために無効化できるようにしている。
- `force-unicode` (1.4 版で**非推奨**): このオプションは何もしない。^{*8}

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 設定

パッケージの動作の設定を `\pxjahypersetup` 命令で行える。

^{*6} 0.6 版より既定を `otfmacros` に変更した。

^{*7} 0.3a 版より既定を `bigcode` に変更した。

^{*8} pL^AT_EX での “unicode モード” が正式にサポートされたためこのオプションは不要になった。

`\pxjahypersetup{<キー>=<値>,...}`

有効な設定キーは以下の通り。

- `fallback=<値>`： PDF 文字列中で表現不能な文字についての代替出力の方法を指定する。^{*9}
 - `geta` (既定)： 表現不能な文字をゲタ記号 (■) に置き換える。
 - `delete`： 表現不能な文字を削除する。※以下、パッケージ動作の解説中で「ゲタ記号を出力」とある場合は実際にはこの設定に従う。
- `fallback-warn=<真偽値>`^{*10}： PDF 文字列中の表現不能な文字について「ゲタ記号を出力」する際に警告を出すか。既定値は真。
- `fallback-cidm=<真偽値>`： `japanese-otf` パッケージの AJ1 以外の CID 入力命令 (`\CIDC`・`\CIDK`・`\CIDT`) について、PDF 文字列中で「ゲタ記号を出力」する動作に置き換えるか。既定値は真。
※これらの命令をサポートするような (本パッケージとは別の) 何らかの機能を利用する場合には偽に設定する必要があるかもしれない。^{*11}

3.2 Unicode 符号値による入力

PDF 文字列入力中で、命令 `\Ux` が以下の意味に変更される。PDF 文字列以外では `\Ux` は以前の定義 (または未定義) に戻る。^{*12}

- `\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：
 - `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。
 - エンジンが `upLaTeX` の場合、あるいは `hyperref` の “unicode モード” (4 節) が有効の場合は、当該の Unicode 文字自体を書いたのと同等になる。
 - 上記以外で、`TeX Live 2022` 以降の `pLaTeX` の場合^{*13}は、当該の Unicode 文字に “対応” する JIS 符号系の文字を書いたのと同等になる。“対応” する文字がない場合は出力できないのでゲタ記号を出力する。
- どの条件にも当てはまらない場合は、`\Ux` は無効になる (定義されない)。

3.3 `japanese-otf` パッケージの文字入力命令への対応

■**\UTF 命令** `japanese-otf` パッケージの `\UTF` 命令は、PDF 文字列中では `out2uni` 用の出力を行う。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。

^{*9} 1.1 版ではパッケージオプション (`fallback-geta` / `fallback-delete`) として提供していたが、この方式は**非推奨**とする。

^{*10} 真偽値は `true` / `false` で指定する。

^{*11} ただし、本パッケージによる再定義は “優先度を下げて” いるので、設定が不要である場合もある。

^{*12} `\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` もサポートするので、こちらを使う方が無難かもしれない。

^{*13} 正確にいうと、`\uchar` と `\ucs` プリミティブをもつ `ε-(u)pTeX` エンジンである場合。

- `\Ux` 命令 (3.2 節参照) が有効の場合は、`\Ux` と同じ動作^{*14}になる。
- それ以外の場合は、常にゲタ記号を出力する。

※ `\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、なければゲタ記号を出力する。^{*15}
 ※例えば、`\CID{8226}` (ローマ数字 12) は Unicode 文字の U+217B に“対応”するので `\Ux{217B}` と同等になるが、`\CID{8297}` (ローマ数字 15) については“対応”する単独の Unicode 文字がないので、ゲタ記号に置き換えられる。
- ただし `\Ux` 命令 (3.2 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 (エラーになる可能性もある) より好ましいであろう。

`\CID` 命令の多言語版、すなわち `\CIDC`・`\CIDK`・`\CIDT` 命令についてはサポートされないので、常にゲタ記号を出力する。

※ `otfcid` の利用には、エンジンの ϵ -`TeX` 拡張および `etoolbox` と `bxjatoucs` パッケージが必要。

■ **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。

- 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。(`\Ux` が無効の場合はゲタ記号になる。) 代替表現の場合は「普通の文字の出力に置き換えられる」可能性がある。
 ※例えば、`\ajLig{ドル}` (“ドル”の組文字) は Unicode 文字の U+3326 に“対応”するので `\Ux{3326}` と同等になるが、`\ajLig{ウルシ}` (“ウルシ”の組文字) は Unicode に“対応”する文字がないため単に“ウルシ”と書いたのと同等になる。

※ `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。) `otfcid` が無効な場合は `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) で `\ajMaru` 等の命令を使いたい場合は、命令の前に `\protect` を付ける必要がある。^{*16}

3.4 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。)

^{*14} もし `\Ux` の出力がゲタ記号になる場合は、`\UTF` もゲタ記号になる。

^{*15} 旧版では削除していたが、他の同様の場合と合わせるため 1.0 版よりゲタ記号を出力する仕様を変更した。

^{*16} 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。

- `\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。Unicode 文字の出力が可能な状況（エンジンが `upLATEX` の場合は常に該当する）では「JIS 符号値」は使われないので省略して（空にして）もよい^{*17}（或いはそもそも JIS X 0208 にない文字の場合は省略する）。

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

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

3.5 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^{*18}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。

- `hyperref` での無効化の対象である「フォント選択命令」の和文版に相当するもの。例えば、`\textmc` `\gtfamily` `\kanjifamily` `\romanshape` `\usekanji` `\useroman` `\userelfont` 等が該当する。
- `pLATEX` カーネル命令: `\<`
- `pTEX` プリミティブ: `\(dis)inhibitglue` `\(no)autospadding` `\(no)autoxspacing`
- `plext` の命令: `\bou` `\kasen` `\rensuji`
- `japanese-otf` の命令: `\textmg` `\mgfamily` `\ltseries` `\ebseries` `\propshape`
- `jsclasses` のクラスの命令: `\maybeblue` `\HUGE`
- `jlreq` クラスの命令: `\jafontsize` `\tatechuyoko` `\jidori`

^{*17} 一応、「JIS 符号値」が使われることが確実な状況では「Unicode 符号値」も省略可能であるが、そのような状況であるかの判断は困難であるため、「Unicode 符号値」の省略は推奨されない。

^{*18} 例えば、“`\textit{text}`”や“`{\large text}`”は単に“`text`”と書いたものと見なされる。

4 hyperref の “unicode モード” での動作

hyperref パッケージの `unicode` オプションが有効である場合（これを “unicode モード” と呼ぶことにする^{*19}）で動作している場合は、PDF 文字列の Unicode への変換は（DVI ドライバ等でなく）hyperref 自身により行われる。hyperref が “unicode モード” である場合には、`pxjahyper` はそれを自動的に検知してそれに適応した動作に切り替える。

- `\Ux` 命令は hyperref の `\unichar` 命令を利用して出力する。このため、`pLATEX` でも Unicode 文字の出力が可能になる。
- PDF 文字列中の和文文字や LICR 命令の処理は hyperref の側に任せられる。
※ただし現状では「`tounicode special`」の発行は（特に害はないため）無効化されない。

ただし、現状での “unicode モード” 対応動作には以下の制限がある。

- hyperref の（`pdftitle` 等の）パッケージオプション中での和文文字の処理は失敗する。このため、文書情報は `\hypersetup` 命令で指定する必要がある。

^{*19} “unicode モード” を有効にする方法は `unicode(=true)` の指定以外にも存在する。また、hyperref の 7.00g 版 [2021-02-04] より、`pLATEX` 以外のエンジン（`upLATEX` も含む）について “unicode モード” は既定で有効になっていることに注意。