pxchfon パッケージ

八登崇之 （Takayuki YATO; aka. “ZR”）

v2.1　[2024/08/11]

目次

1　概要

pLaTeX ／ upLaTeX + dvipdfmx のワークフローで PDF 文書を作る場合に、標準の和文フォント（明朝・ゴシック）に対して実際に使われる OpenType フォントをユーザが指定する機能を提供する。使用するフォントを LaTeX 文書中で指定するので、一度パッケージをインストールするだけで、任意の日本語フォント（ただし等幅に限る）を使うことができる。欧文部分を同じ日本語フォントで置き換えることも可能（ただしこれも半角幅の文字に限られる）である。japanese-otf パッケージ[*1]にも対応している。

また、(u)pLaTeX において広く行われているフォント設定（IPAex フォントの使用等）をパッケージオプション一つで適用する機能（プリセット指定）も備えている。

2　前提環境

- TeX フォーマット： LaTeX
- TeX エンジン：
  - pTeX（3.0 版以降）
  - upTeX（0.10 版以降）

  ※一部の機能は ε-TeX 拡張を必要とする。
- DVI ウェア： dvipdfmx （詳細は 3 節と付録 A を参照）
- 前提パッケージ：
  - atbegshi パッケージ（everypage オプション使用時）
  - pxufont パッケージ（unicode オプション使用時）

3　読込

プレアンブルにおいて、\usepackage を用いて読み込む。

```
\usepackage[<オプション>,...]{pxchfon}
```

オプションは次のものが用意されている。

[*1] japanese-otf パッケージ： https://www.ctan.org/pkg/japanese-otf

- **ドライバオプション**： dvipdfmx、dvips、dviout、xdvi が指定できる。[*2]ただし、本パッケージの主要機能である「フォントマップの文書内での指定」がサポートされるのは dvipdfmx **のみ**である。[*3]他の DVI ウェアにおける動作については付録 A を参照。ドライバオプションの既定値は dvipdfmx である。
- **プリセット指定オプション**（ms、haranoaji 等）名前に対応するプリセット指定を有効にする。
  ※詳細については 5 節を参照。
- **カスタムプリセット指定オプション**（user:**名前**）ユーザ定義のプリセット指定を有効にする。
  ※詳細については 8 節を参照。
- **ファイルプリセット指定オプション**（tl:⟨名前⟩ または file:⟨名前⟩[*4]）名前に対応するファイルプリセット指定を有効にする。
  ※詳細については 6 節を参照。
- use=⟨名前⟩： マップファイル読込指定。「本パッケージによるマップ設定に先行して指定のマップファイルを読み込む」ことを指示する。
  ※詳細については 7 節を参照。
- alphabet： 欧文フォントも（本パッケージの機能で）指定されたフォントの英数字部分で置き換える。（明朝が \rmfamily、ゴシックが \sffamily に適用される。）
  ※技術的制約のため[*5]、**半角等幅のフォント**しかサポートされないことに注意してほしい。つまり、この設定を使うと欧文が全て等幅になってしまう。「部分的に欧文フォントを和文フォントに合わせたい」という場合は、後述の relfont オプションの使用も検討されたい。
- noalphabet： alphabet の否定。欧文フォントは変更しない。
  ※ alphabet と noalphabet と後述の relfont の 3 つのオプションは排他である。プリセット指定オプション**使用**の場合は、このうち noalphabet が既定で有効になる。プリセット指定オプション**不使用**の場合は、以前の版では alphabet が既定になっていた。今でもこの動作は維持されているが、2.0 版以降はこの既定設定を利用することは**非推奨**[*6]となり、プリセット不使用の場合は 3 つのオプションの何れかを指定することが推奨される。
- otf（既定）： japanese-otf パッケージの使用時に、そのフォントも置換の対象とする。
- nootf： otf の否定。japanese-otf パッケージのフォントは置き換えない。
  ※ nootf を指定してかつ japanese-otf パッケージを用いる場合は、japanese-otf で noreplace を指定しない限り、標準の和文フォントは変化しない。
- jis2004 ／ prefer2004jis： pTEX ／ upTEX の標準和文フォントの CMap を「2004JIS 字形」指定のものに変更する。[*7]

[*2] 「上級者向け」の一覧にある nodvidriver と resetdvidriver もドライバオプションの一種である。
[*3] 現状では、dvipdfmx 以外のドライバ指定は全て nodvidriver と等価である。
[*4] 以前の版では **+名前** および ***名前** という形式であった。
[*5] プロポーショナル幅のフォントを使うためには、そのフォントの字幅に合わせた TEX 側の論理フォント（TFM）を事前に用意する必要があるため。和文が全角幅しか使えないのも同じ理由である。
[*6] プリセット指定オプションの有無により既定が変わる仕様は好ましくないため。2.1 版以降では警告が出る。
[*7] japanese-otf パッケージの和文フォントについては、japanese-otf パッケージの jis2004 オプションで 2004JIS 字形指定を行う仕様になっている。そのため、本パッケージの jis2004 の対象にはならない。ところが 0.7h 版以前の本パッケージでは、upTEX 上の japanese-otf の和文フォントにも jis2004 を適用していた。これは、昔の japanese-otf が upTEX 上では jis2004 オプションに未対応であったためであり、非公式の暫定仕様であった。現在では、この暫定仕様は廃止されている。特定のフォントが本パッケージと japanese-otf のどちらの jis2004 の設定対象であるかを判断するは難しいので、jis2004 をグローバルオプ

※グローバルオプション（\documentclass のオプション）に jis2004 を指定すると、japanese-otf と pxchfon の両方に適用される。

- nojis2004 ／ noprefer2004jis（既定）： jis2004 の否定。
- oneweight： japanese-otf パッケージを単ウェイトで使用する場合に、プリセット設定で使われるフォントの集合を pTEX 標準と同一にする。[*8]
- nooneweight（既定）： oneweight の否定。

■上級者向けオプション

- nodvidriver： 特殊なドライバオプションで、ドライバ依存動作の無効化を明示的に指示する。
- resetdvidriver： nodvidriver の否定。
  ※ドライバ指定は既定値である dvipdfmx に戻る。
- relfont： 指定された和文フォントの英数字部分を（pLATEX の標準機能である）**従属欧文フォント**として設定する。すなわち、既定では欧文フォントは置き換えないが、\selectfont で和文フォントを選択する際に予め \userelfont を実行しておくと欧文も和文と同じ書体になる。[*9]
  ※置換が設定されていないウェイトについては無効になる。
  ※ alphabet、noalphabet、relfont の 3 つのオプションは排他である。
- everypage： 出力 DVI の全ページにフォントマップ設定を書き込む。
  ※詳細については 10 節を参照。
- noeverypage（既定）： everypage の否定。出力 DVI の先頭ページにのみフォントマップ設定を書き込む。
- unicode[=⟨値⟩]： Unicode 直接指定。一部または全部のフォントについて、エンコーディング指定方式を“CMap 指定”から“Unicode 直接指定”に変更する。有効な値は false（既定）、UTF、simple、full の 4 つである。値を省略して「unicode」だけ書いた場合は full が指定される。
  ※詳細については 9 節を参照。
  ※一部のプリセット指定（sourcehan 等）は unicode の既定値を変更する。
  ※ Unicode 直接指定を有効にするオプションを指定しても、置換されて**いない**ウェイトは Unicode 直接指定にならない。（詳しくは 9.1 節の注意を参照。）
- usecmapforalphabet： alphabet または relfont を指定して日本語フォントの英数字部分を欧文フォントとして使う際に、そのエンコーディング指定方式を“CMap 指定”にする。
  ※半角英数字用の UniJIS-UCS2-HW-H という CMap が指定される。
  ※ AJ1 のグリフ集合をもつ OpenType フォントの場合、既定では[*10]英数字がプロポーショナル幅のものになるがこれはサポートされないので、このオプションを指定して半角幅のものを使う必要がある。
- nousecmapforalphabet（既定）： usecmapforalphabet の否定。

ションに指定することを推奨する。

[*8] 多ウェイト用プリセット設定の一部において、このオプションによりゴシックのウェイトが変更される。例えば、小塚フォントのプリセットでは、pTEX 標準のゴシック（jisg 等）には M ウェイトを割り当てる一方で、japanese-otf パッケージの 3 ウェイトのゴシックには R、B、H ウェイトを割り当てている。従って、japanese-otf を単ウェイトで用いる時に M ウェイトを使いたい場合には oneweight オプションを指定すればよい。

[*9] 特に「\userelfont\selectfont」だけ実行すると、欧文が現在の和文と同じ書体になる。

[*10] unicode オプションが指定されない場合。

※和文と異なり、欧文フォントでは実質的に Unicode 直接指定の方が既定となっている。
※一部のプリセット指定は usecmapforalphabet の指定を強制する。

- dumpmap：「通常マップファイルダンプ出力」を有効にする。すなわち、本パッケージにより文書に設定されるマップ行を、⟨ジョブ名⟩.map の名のファイルに書き出す。
- nodumpmap（既定）：dumpmap の否定。
- dumpmaptl：「TeX Live マップファイルダンプ出力」を有効にする。すなわち、本パッケージによる設定を再現する kanji-config-updmap 用のマップファイルのセット（ptex-NAME.map、otf-NAME.map、uptex-NAME.map、otf-up-NAME.map の 4 つ、ただし NAME はジョブ名）を出力する。
※例えば、japanese-otf パッケージが使われない場合は japanese-otf パッケージ用のマップ行は適用されない。そのため、通常ダンプ出力はそのようなマップ行は書き出されない。これに対して、TeX Live 用ダンプ出力は「実際に適用されるか」は無関係で kanji-config-updmap の規則に従うため、japanese-otf パッケージ用のマップが otf-*.map に書き出される。
※ dumpmap と dumpmaptl は排他では**ない**。
- nodumpmaptl（既定）：dumpmaptl の否定。
- strictcsi： Identity-H/-V の CMap が指定されたマップ行について、CSI 指定を（仕様に厳密に従って）フォントが TrueType グリフの場合にのみ出力する。
※「CSI 指定」とはフォントファイル名の直後に書く“/AJ1”の類のことで、本来は（グリフ集合情報を持たない）TrueType グリフのフォントのためのものである。しかし、CFF グリフのフォントに対して CSI 指定があっても特に問題は起こらず、また、フォントのグリフ種別の判断する処理は少し時間がかかるため、既定では厳密な判定は行わない。
※ 2014/05/06 以降の版の ε-(u)pTeX が必要である。
- nostrictcsi（既定）：strictcsi の否定。Identity-H/-V に対する CSI 指定は常に出力される。
※さすがにファイルに出力されたマップ行に不備があるのは避けたいので、dumpmap(tl) が指定された場合は、既定が strictcsi に変更される。
- fullwidth： Unicode 直接指定が有効である場合に、全角幅のグリフを優先させる。
※全角幅想定の文字が使える可能性が増えるという利点がある一方で、半角幅想定の文字が使えなくなるという欠点もある。
- nofullwidth（既定）：fullwidth の否定。
- legacycode=⟨値⟩： Unicode 直接指定を全体に採用した場合[*11]にエンコーディングが Unicode ではない原メトリック和文 TFM に関するマップ行に対する取扱を規定する。有効な値は以下の通り：
  - apply（既定）：“CMap 指定”のマップ行を出力する。
  - ignore： マップ行を出力しない。
  - suppress： 当該の TFM の無効化を指示するマップ行を出力する。

  ※ legacycode-replace が真の場合は既定値が suppress に変更される。
- legacycode-replace=⟨真偽値⟩： pxufont パッケージを読み込んで、原メトリック和文 TFM についてエンコーディングが Unicode のもののみが利用されるようにする。既定値は偽だが、unicode オプションが full の場合は真に変更される。

[*11] つまり、unicode オプションの値が simple または full である場合。ただし現状では unicode の値がこれ以外の場合でも legacycode の値は設定可能であるが、恐らく既定の apply 以外の設定は無益であろう。

- `expert`（既定）： Unicode 直接指定が有効な場合に japanese-otf の `expert` オプションの機能を（可能な範囲で）エミュレートする。
  ※ japanese-otf の `expert` が指定されない場合は無意味。
- `noexpert`： `expert` の否定。Unicode 直接指定時には japanese-otf の `expert` は無効になる。
- `glyphid`： GID 指定入力（`\gid` 命令）の機能を有効にする。
  ※エンジンが upLaTeX でかつ Unicode 直接指定が有効の場合にのみ利用できる。
- `noglyphid`（既定）： `glyphid` の否定。
- `maybe-multiweight`： 「`deluxe` オプション付きの japanese-otf パッケージが読み込まれていないにもかかわらず多ウェイト用のフォント設定命令が使われた」場合に出る警告を抑止する。
- `nocheck-expert`： 「Unicode 直接指定と `expert` オプションが有効で、かつ置換されていないウェイトが存在する」場合に出る警告を抑止する。
  ※警告抑止オプションについては付録 D を参照。

4 機能

以下に該当する和文（CJK）用の論理フォント（原メトリック TFM）について、それに対応する物理フォント（OpenType フォント）をユーザ指定のものに置き換える。

- pTeX の標準のフォント — `rml*`/`gbm*`
- upTeX の日本語フォント — `urml*`/`ugbm*`/`uprml*`/`upgbm*`
- upTeX の中国語・韓国語フォント
- UTF パッケージのフォント — `hmr*`/`hkb*`/`unij*`/`cid*`
- japanese-otf パッケージの日本語フォント— `{,up}hmin*`/`{,up}hgoth*`/`otf-{u,c}j*`
- japanese-otf パッケージの中国語・韓国語フォント
- pxufont パッケージのフォント— `zur-?j*`

※中国語・韓国語フォントに対するサポートの詳細については付録 C を参照。

和文フォント置換は、dvipdfmx のマップ設定を文書内で（一時的に）変更するという方法で実現している。欧文フォントについては実現方法が少し異なる（11 節を参照）。

■単ウェイトの場合の設定 japanese-otf パッケージを `deluxe` オプション付きで用いている場合**以外**、すなわち明朝・ゴシックとも単ウェイトの場合、以下の命令を用いる。

- `\setminchofont[`⟨番号⟩`]{`⟨フォントファイル名⟩`}`： 明朝体（`\mcfamily`）のフォントを置き換えるフォントをファイル名で指定する。TTC 形式の場合の該当のフォントの番号を ⟨番号⟩ に指定する。
- `\setgothicfont[`⟨番号⟩`]{`⟨フォントファイル名⟩`}`： ゴシック体（`\gtfamily`[*12]）のフォントを置き換えるフォントをファイル名で指定する。⟨番号⟩ の意味は前項と同じ。
- 以上の 2 つの命令、および以降で紹介するフォント設定命令について、⟨フォントファイル名⟩ の値を `*` にするとフォント非埋込を指示する。また、この値を空にすると、以前に（当該の命令により）設定さ

[*12] 単ウェイト設定を用いる多くの場合、明朝体の太字（`\mcfamily\bfseries`）はゴシック体（`\gtfamily` と同じもの）で代替される。

れていた値を取り消して（dvipdfmx の）既定の設定に戻す。

以下に設定例を示す。

```
\setminchofont{ipam.ttf}        % 明朝体は"IPA 明朝"
\setgothicfont[0]{msgothic.ttc} % ゴシック体は"ＭＳ ゴシック"
\setminchofont{*}               % 明朝体は非埋込
\setgothicfont{}                % ゴシック体は既定設定に従う
```

■多ウェイトの場合の設定　japanese-otf パッケージを deluxe 付きで用いている場合は、明朝・ゴシックともに 3 ウェイトを使う。この時は、各ウェイト毎にフォント指定ができる。またこの場合、丸ゴシック（\mgfamily）が使用可能になるが、これに対して置き換えるフォントを指定することができる。

- \setlightminchofont[⟨番号⟩]{⟨フォントファイル名⟩}： 明朝・細ウェイト（\mcfamily\ltseries）
- \setmediumminchofont[⟨番号⟩]{⟨フォントファイル名⟩}： 明朝・中ウェイト（\mcfamily\mdseries）
- \setboldminchofont[⟨番号⟩]{⟨フォントファイル名⟩}： 明朝・太ウェイト（\mcfamily\bfseries）
- \setmediumgothicfont[⟨番号⟩]{⟨フォントファイル名⟩}：ゴシック・中ウェイト（\gtfamily\mdseries）
- \setboldgothicfont[⟨番号⟩]{⟨フォントファイル名⟩}： ゴシック・太ウェイト（\gtfamily\bfseries）
- \setxboldgothicfont[⟨番号⟩]{⟨フォントファイル名⟩}：ゴシック・極太ウェイト（\gtfamily\ebseries）
- \setmarugothicfont[⟨番号⟩]{⟨フォントファイル名⟩}： 丸ゴシック（\mgfamily）

さらに、この場合、\setminchofont と \setgothicfont は各々のファミリの 3 ウェイト全てを指定のフォントで置き換える。実質的に単ウェイトになってしまうようで無意味に思えるが、例えば明朝を実際には 2 ウェイトしか使わないという時に、

```
\setminchofont{minchoW3.otf}       % まず 3 ウェイト指定して
\setboldminchofont{minchoW6.otf}   % 太だけ再指定する
```

とする使い方が考えられる。特に、欧文フォントも置き換えたい場合は 3 ウェイトが全て置換されていないと有効にならないので、

```
\setmediumminchofont{minchoW3.otf}
\setboldminchofont{minchoW6.otf}
```

では思い通りにならないことになる。また、この仕様のため、deluxe 以外の場合（既定、bold、noreplace）は \setgothicfont で指定したものが確実にゴシック（単ウェイト）に反映される。

■上級者向け機能

- \pxchfonsetup{⟨設定⟩,…}： 設定を変更する。設定可能なパラメタは以下の通り。
  ※真偽値をとるパラメタでは、値を省略した場合は true と等価となる。
  ※残念ながら、実装の都合のためパッケージ読込後に変更可能な設定項目はかなり少ない。
  - jis2004[=⟨真偽値⟩]： オプションの (no)jis2004 と同じ。
- \usecmapforalphabet： usecmapforalphabet オプションの設定に切り替える。（3 節を参照。）
- \nousecmapforalphabet： nousecmapforalphabet オプションの設定に切り替える。
- \setnewglyphcmapprefix{⟨文字列⟩}： 2004JIS 用の JIS コード系の CMap の名前の接頭辞を指定

する。そのような CMap は、pTEX の標準和文フォントについて 2004JIS 字形を選択（jis2004 指定）した時に必要となるが、Adobe が配布している CMap ファイルには該当するものがないので、それを適宜用意してそのファイル名をこの命令で指定する必要がある。引数に与えるのは最後の 1 文字（書字方向の「H」「V」）を除いた部分の文字列である。
※ CMap 名接頭辞の既定値は「2004-」で、これは最近の TEX Live に含まれている「2004-H」等の CMap ファイルを用いることを意味する。[*13]

- \usefontmapfile{⟨マップファイル名⟩}： 指定の dvipdfmx 用のマップファイルの読込を指示する。
※ pdfTEX の \pdfmapfile に相当する機能。
- \usefontmapline{⟨マップ行⟩}： dvipdfmx のマップ行を直接指定して、その読込を指示する。
※ pdfTEX の \pdfmapline に相当する機能。
- \diruni： 現在の和文フォントを“Unicode 直接入力”（フォントマップを Unicode 直接指定にした上でさらに和文 VF をバイパスする）の状態に切り替える（宣言型命令）。これにより、実際のフォントに存在する任意の全角幅の文字が出力可能となる。その代わり、この状態では、約物の周りの空き調整が無効になる。
※現在のウェイトについて Unicode 直接指定が有効になっている必要がある。
※この命令自体は単に NFSS のシェープ値を diruni という値に変えているだけであり、このシェープに“Unicode 直接入力”のフォントが予め設定されているわけである。
- \textdiruni{⟨テキスト⟩}： \diruni に対応する引数型命令。
- \gid{⟨整数⟩}： 現在の和文フォントで、指定の値の GID をもつグリフを出力する。
※全角幅のグリフでないと正常に出力されない。
※エンジンが upLATEX であり、かつ、現在のウェイトについて Unicode 直接指定が有効になっている必要がある。

5 プリセット指定

このパッケージの元々の意図は、標準のフォントを普段使っているものと全く別の書体に変えることであったが、例えば「普段使う設定が複数ありそれを簡単に切り替えたい」という場合にも有用であることがわかった。そこで、(u)pLATEX において広く行われている設定をパッケージ内に組み込んで、パッケージオプションでそれを呼び出すという機能が後になって追加された。[*14]

パッケージオプションにプリセット名を指定すると予め決められたフォントファイル名が \setminchofont 等の命令で設定される。例えば、

```
\usepackage[ipa]{pxchfon}
```

は以下の記述と同等になる。

```
\usepackage[noalphabet]{pxchfon}
\setminchofont{ipam.ttf}
\setgothicfont{ipag.ttf}
```

[*13] 引数に * を与えた場合は JISX0213-2004- が指定されたと見なされる（歴史的理由から）。
[*14] 元々は pxjafont という別のパッケージで提供されていた機能であるが、0.5 版からこのパッケージに組み入れることにした。詳しくは付録 B を参照。

注意として、プリセット指定を用いた場合は、欧文フォントの置換について noalphabet（無効）が既定になる。プリセット指定の場合は和文が「普通の」明朝・ゴシックのフォントとなるので欧文フォントを変更しない場合が多いと考えられるためである。

5.1 単ウェイト用のプリセット設定

後述の「多ウェイト用の設定」で述べられた設定以外で使う場合に使用する。

- noembed： フォントを埋め込まない。
```
\setminchofont{*} % 非埋込
\setgothicfont{*} % 非埋込
```
- ms： MS フォント。
```
\setminchofont[0]{msmincho.ttc} % ＭＳ 明朝
\setgothicfont[0]{msgothic.ttc} % ＭＳ ゴシック
```
- ipa： IPA フォント。
```
\setminchofont{ipam.ttf} % IPA 明朝
\setgothicfont{ipag.ttf} % IPA ゴシック
```
- ipaex： IPAex フォント。
```
\setminchofont{ipaexm.ttf} % IPAex 明朝
\setgothicfont{ipaexg.ttf} % IPAex ゴシック
```

5.2 多ウェイト用のプリセット設定

japanese-otf パッケージの deluxe オプション使用時に有効になる。明朝 3 ウェイト、ゴシック 3 ウェイト、丸ゴシック 1 ウェイトを設定する。

- ms-hg： MS フォント + HG フォント。
  ※ HG フォント = Microsoft Office 付属の日本語フォント
  ※「HG 丸ｺﾞｼｯｸ M-PRO」は欧文が等幅でないので alphabet オプション指定とともに使うことができない。（後掲の ipa-hg、ipaex-hg についても同様。）
```
\setminchofont[0]{msmincho.ttc}    % ＭＳ 明朝
\setboldminchofont[0]{hgrme.ttc}   % HG 明朝 E
\setgothicfont[0]{msgothic.ttc}    % ＭＳ ゴシック
\setmediumgothicfont[0]{hgrgm.ttc} % HG ｺﾞｼｯｸ M
\setboldgothicfont[0]{hgrge.ttc}   % HG ｺﾞｼｯｸ E
\setxboldgothicfont[0]{hgrsgu.ttc} % HG 創英角ｺﾞｼｯｸ UB
\setmarugothic{hgrsmp.ttf}         % HG 丸ｺﾞｼｯｸ M-PRO
```
- ipa-hg： IPA フォント + HG フォント。
```
\setminchofont{ipam.ttf}           % IPA 明朝
\setboldminchofont[0]{hgrme.ttc}   % HG 明朝 E
\setgothicfont{ipag.ttf}           % IPA ゴシック
\setmediumgothicfont[0]{hgrgm.ttc} % HG ｺﾞｼｯｸ M
\setboldgothicfont[0]{hgrge.ttc}   % HG ｺﾞｼｯｸ E
\setxboldgothicfont[0]{hgrsgu.ttc} % HG 創英角ｺﾞｼｯｸ UB
\setmarugothic{hgrsmp.ttf}         % HG 丸ｺﾞｼｯｸ M-PRO
```

- ipaex-hg： IPAex フォント + HG フォント。

```
\setminchofont{ipaexm.ttf}          % IPAex 明朝
\setboldminchofont[0]{hgrme.ttc}    % HG 明朝 E
\setgothicfont{ipaexg.ttf}          % IPAex ゴシック
\setmediumgothicfont[0]{hgrgm.ttc} % HG ｺﾞｼｯｸ M
\setboldgothicfont[0]{hgrge.ttc}    % HG ｺﾞｼｯｸ E
\setxboldgothicfont[0]{hgrsgu.ttc} % HG 創英角ｺﾞｼｯｸ UB
\setmarugothic{hgrsmp.ttf}          % HG 丸ｺﾞｼｯｸ M-PRO
```

- moga-mobo： Moga フォント + Mobo フォント。
  ※「丸ゴシック」ファミリに MoboGothic を充てている。
  ※ Moga/Mobo フォントは AJ1 非対応であるが、フォント実体を変えることで jis2004 オプションに対応させている。

  jis2004 非指定時

```
\setminchofont[3]{mogam.ttc}        % Moga90Mincho
\setboldminchofont[3]{mogamb.ttc}  % Moga90Mincho Bold
\setgothicfont[2]{mogag.ttc}        % Moga90Gothic
\setboldgothicfont[2]{mogagb.ttc}  % Moga90Gothic Bold
\setxboldgothicfont[2]{mogagb.ttc} % Moga90Gothic Bold
\setmarugothic[2]{mobog.ttc}        % Mobo90Gothic
```

  jis2004 指定時

```
\setminchofont[0]{mogam.ttc}        % MogaMincho
\setboldminchofont[0]{mogamb.ttc}  % MogaMincho Bold
\setgothicfont[0]{mogag.ttc}        % MogaGothic
\setboldgothicfont[0]{mogagb.ttc}  % MogaGothic Bold
\setxboldgothicfont[0]{mogagb.ttc} % MogaGothic Bold
\setmarugothic[0]{mobog.ttc}        % MoboGothic
```

- moga-mobo-ex： MogaEx フォント + MoboEx フォント。
  ※「丸ゴシック」ファミリに MoboExGothic を充てている。
  ※フォント実体を変えることで jis2004 オプションに対応させている。

  jis2004 非指定時

```
\setminchofont[4]{mogam.ttc}        % MogaEx90Mincho
\setboldminchofont[4]{mogamb.ttc}  % MogaEx90Mincho Bold
\setgothicfont[3]{mogag.ttc}        % MogaEx90Gothic
\setboldgothicfont[3]{mogagb.ttc}  % MogaEx90Gothic Bold
\setxboldgothicfont[3]{mogagb.ttc} % MogaEx90Gothic Bold
\setmarugothic[3]{mobog.ttc}        % MoboEx90Gothic
```

  jis2004 指定時

```
\setminchofont[1]{mogam.ttc}        % MogaExMincho
\setboldminchofont[1]{mogamb.ttc}  % MogaExMincho Bold
\setgothicfont[1]{mogag.ttc}        % MogaExGothic
\setboldgothicfont[1]{mogagb.ttc}  % MogaExGothic Bold
\setxboldgothicfont[1]{mogagb.ttc} % MogaExGothic Bold
\setmarugothic[1]{mobog.ttc}        % MoboExGothic
```

- moga-maruberi： Moga フォント + モトヤ L マルベリ 3 等幅。

※ moga-mobo と以下を除いて同じ。

```
\setmarugothic{MTLmr3m.ttf}        % モトヤ L マルベリ 3 等幅
```

- ume： 梅フォント。

```
\setminchofont{ume-tmo3.ttf}       % 梅明朝
\setgothicfont{ume-tgo5.ttf}       % 梅ゴシック O5
\setmediumgothicfont{ume-tgo4.ttf} % 梅ゴシック
\setmarugothic{ume-tgo5.ttf}       % 梅ゴシック O5
```

- kozuka-pro： 小塚フォント（Pro 版）。

```
\setminchofont{KozMinPro-Regular.otf}      % 小塚明朝 Pro R
\setlightminchofont{KozMinPro-Light.otf}   % 小塚明朝 Pro L
\setboldminchofont{KozMinPro-Bold.otf}     % 小塚明朝 Pro B
\setgothicfont{KozGoPro-Medium.otf}        % 小塚ゴシック Pro M
\setmediumgothicfont{KozGoPro-Regular.otf} % 小塚ゴシック Pro R
\setboldgothicfont{KozGoPro-Bold.otf}      % 小塚ゴシック Pro B
\setxboldgothicfont{KozGoPro-Heavy.otf}    % 小塚ゴシック Pro H
\setmarugothicfont{KozGoPro-Heavy.otf}     % 小塚ゴシック Pro H
```

- kozuka-pr6： 小塚フォント（Pr6 版）。

```
\setminchofont{KozMinProVI-Regular.otf}      % 小塚明朝 Pro-VI R
\setlightminchofont{KozMinProVI-Light.otf}   % 小塚明朝 Pro-VI L
\setboldminchofont{KozMinProVI-Bold.otf}     % 小塚明朝 Pro-VI B
\setgothicfont{KozGoProVI-Medium.otf}        % 小塚ゴシック Pro-VI M
\setmediumgothicfont{KozGoProVI-Regular.otf} % 小塚ゴシック Pro-VI R
\setboldgothicfont{KozGoProVI-Bold.otf}      % 小塚ゴシック Pro-VI B
\setxboldgothicfont{KozGoProVI-Heavy.otf}    % 小塚ゴシック Pro-VI H
\setmarugothicfont{KozGoProVI-Heavy.otf}     % 小塚ゴシック Pro-VI H
```

- kozuka-pr6n： 小塚フォント（Pr6N 版）。

```
\setminchofont{KozMinPr6N-Regular.otf}      % 小塚明朝 Pr6N R
\setlightminchofont{KozMinPr6N-Light.otf}   % 小塚明朝 Pr6N L
\setboldminchofont{KozMinPr6N-Bold.otf}     % 小塚明朝 Pr6N B
\setgothicfont{KozGoPr6N-Medium.otf}        % 小塚ゴシック Pr6N M
\setmediumgothicfont{KozGoPr6N-Regular.otf} % 小塚ゴシック Pr6N R
\setboldgothicfont{KozGoPr6N-Bold.otf}      % 小塚ゴシック Pr6N B
\setxboldgothicfont{KozGoPr6N-Heavy.otf}    % 小塚ゴシック Pr6N H
\setmarugothicfont{KozGoPr6N-Heavy.otf}     % 小塚ゴシック Pr6N H
```

- hiragino-pro： ヒラギノフォント基本 6 書体セット（Pro/Std 版） + 明朝 W2。

```
\setminchofont{HiraMinPro-W3.otf}       % ヒラギノ明朝 Pro W3
\setlightminchofont{HiraMinPro-W2.otf}  % ヒラギノ明朝 Pro W2
\setboldminchofont{HiraMinPro-W6.otf}   % ヒラギノ明朝 Pro W6
\setgothicfont{HiraKakuPro-W3.otf}      % ヒラギノ角ゴ Pro W3
\setboldgothicfont{HiraKakuPro-W6.otf}  % ヒラギノ角ゴ Pro W6
\setxboldgothicfont{HiraKakuStd-W8.otf} % ヒラギノ角ゴ Std W8
\setmarugothicfont{HiraMaruPro-W4.otf}  % ヒラギノ丸ゴ Pro W4
```

- hiragino-pron： ヒラギノフォント基本 6 書体セット（ProN/StdN 版） + 明朝 W2。

```
\setminchofont{HiraMinProN-W3.otf}       % ヒラギノ明朝 ProN W3
\setlightminchofont{HiraMinProN-W2.otf}  % ヒラギノ明朝 ProN W2
\setboldminchofont{HiraMinProN-W6.otf}   % ヒラギノ明朝 ProN W6
\setgothicfont{HiraKakuProN-W3.otf}      % ヒラギノ角ゴ ProN W3
```

```
\setboldgothicfont{HiraKakuProN-W6.otf}  % ヒラギノ角ゴ ProN W6
\setxboldgothicfont{HiraKakuStdN-W8.otf} % ヒラギノ角ゴ StdN W8
\setmarugothicfont{HiraMaruProN-W4.otf}  % ヒラギノ丸ゴ ProN W4
```

- hiragino-elcapitan-pro： ヒラギノフォント（Mac OS X El Capitan 搭載；Pro/Std 版）＋明朝 W2。

```
\setminchofont[1]{HiraginoSerif-W3.ttc}
\setlightminchofont{HiraMinPro-W2.otf}
\setboldminchofont[1]{HiraginoSerif-W6.ttc}
\setgothicfont[3]{HiraginoSans-W3.ttc}
\setboldgothicfont[3]{HiraginoSans-W6.ttc}
\setxboldgothicfont[2]{HiraginoSans-W8.ttc}
\setmarugothicfont[0]{HiraginoSansR-W4.ttc}
```

- hiragino-elcapitan-pron： ヒラギノフォント（Mac OS X El Capitan 搭載；ProN/StdN 版）＋明朝 W2。

```
\setminchofont[0]{HiraginoSerif-W3.ttc}
\setlightminchofont{HiraMinProN-W2.otf}
\setboldminchofont[0]{HiraginoSerif-W6.ttc}
\setgothicfont[2]{HiraginoSans-W3.ttc}
\setboldgothicfont[2]{HiraginoSans-W6.ttc}
\setxboldgothicfont[3]{HiraginoSans-W8.ttc}
\setmarugothicfont[1]{HiraginoSansR-W4.ttc}
```

※「新しい macOS のヒラギノフォント」用のプリセットについては 5.4 節を参照されたい。

- morisawa-pro： モリサワフォント基本 7 書体（Pro 版）。

```
\setminchofont{A-OTF-RyuminPro-Light.otf}         % A-OTF リュウミン Pro L-KL
\setboldminchofont{A-OTF-FutoMinA101Pro-Bold.otf} % A-OTF 太ミン A101 Pro
\setgothicfont{A-OTF-GothicBBBPro-Medium.otf}     % A-OTF 中ゴシック BBB Pro
\setboldgothicfont{A-OTF-FutoGoB101Pro-Bold.otf}  % A-OTF 太ゴ B101 Pro
\setxboldgothicfont{A-OTF-MidashiGoPro-MB31.otf}  % A-OTF 見出ゴ MB31 Pro
\setmarugothicfont{A-OTF-Jun101Pro-Light.otf}     % A-OTF じゅん Pro 101
```

- morisawa-pr6n： モリサワフォント基本 7 書体（Pr6N 版[*15]）。

```
\setminchofont{A-OTF-RyuminPr6N-Light.otf}         % A-OTF リュウミン Pr6N L-KL
\setboldminchofont{A-OTF-FutoMinA101Pr6N-Bold.otf} % A-OTF 太ミン A101 Pr6N
\setgothicfont{A-OTF-GothicBBBPr6N-Medium.otf}     % A-OTF 中ゴシック BBB Pr6N
\setboldgothicfont{A-OTF-FutoGoB101Pr6N-Bold.otf}  % A-OTF 太ゴ B101 Pr6N
\setxboldgothicfont{A-OTF-MidashiGoPr6N-MB31.otf}  % A-OTF 見出ゴ MB31 Pr6N
\setmarugothicfont{A-OTF-Jun101Pro-Light.otf}      % A-OTF じゅん Pro 101
```

- yu-win： 游書体（Windows 8.1 搭載版）。

```
\setminchofont{yumin.ttf}         % 游明朝 Regular
\setlightminchofont{yuminl.ttf}   % 游明朝 Light
\setboldminchofont{yumindb.ttf}   % 游明朝 Demibold
\setgothicfont{yugothic.ttf}      % 游ゴシック Regular
\setboldgothicfont{yugothib.ttf}  % 游ゴシック Bold
\setxboldgothicfont{yugothib.ttf} % 游ゴシック Bold
\setmarugothicfont{yugothic.ttf}  % 游ゴシック Regular
```

- yu-win10： 游書体（Windows 10/11 搭載版）。

[*15] 「じゅん」は Pr6N 版が存在しないため Pro 版が使われる。

※フォントの性質のため、この設定では横組の和文クオート“”‘’の出力が不正になる。この不具合は unicode オプションを指定することで解決できる。詳細については 9 節を参照されたい。

```
\setminchofont{yumin.ttf}
\setlightminchofont{yuminl.ttf}
\setboldminchofont{yumindb.ttf}
\setgothicfont[0]{YuGothM.ttc}
\setmediumgothicfont[0]{YuGothR.ttc}
\setboldgothicfont[0]{YuGothB.ttc}
\setxboldgothicfont[0]{YuGothB.ttc}
\setmarugothicfont[0]{YuGothM.ttc}
```

- yu-osx： 游書体（Mac OS X 搭載版）。

```
\setminchofont{YuMin-Medium.otf}       % 游明朝体 ミディアム
\setboldminchofont{YuMin-Demibold.ttf} % 游明朝体 デミボールド
\setgothicfont{YuGo-Medium.otf}        % 游ゴシック体 ミディアム
\setboldgothicfont{YuGo-Bold.otf}      % 游ゴシック体 ボールド
\setxboldgothicfont{YuGo-Bold.otf}     % 游ゴシック体 ボールド
\setmarugothicfont{YuGo-Medium.otf}    % 游ゴシック体 ミディアム
```

- sourcehan-otc： Source Han Serif（源ノ明朝）+ Source Han Sans（源ノ角ゴシック）、OTC 版。

```
\setminchofont[0]{SourceHanSerif-Regular.ttc}
\setlightminchofont[0]{SourceHanSerif-Light.ttc}
\setboldminchofont[0]{SourceHanSerif-Bold.ttc}
\setgothicfont[0]{SourceHanSans-Medium.ttc}
\setmediumgothicfont[0]{SourceHanSans-Regular.ttc}
\setboldgothicfont[0]{SourceHanSans-Bold.ttc}
\setxboldgothicfont[0]{SourceHanSans-Heavy.ttc}
\setmarugothicfont[0]{SourceHanSans-Medium.ttc}
```

- sourcehan： Source Han Serif（源ノ明朝）+ Source Han Sans（源ノ角ゴシック）、言語別 OTF 版。

```
\setminchofont{SourceHanSerif-Regular.otf}
\setlightminchofont{SourceHanSerif-Light.otf}
\setboldminchofont{SourceHanSerif-Bold.otf}
\setgothicfont{SourceHanSans-Medium.otf}
\setmediumgothicfont{SourceHanSans-Regular.otf}
\setboldgothicfont{SourceHanSans-Bold.otf}
\setxboldgothicfont{SourceHanSans-Heavy.otf}
\setmarugothicfont{SourceHanSans-Medium.otf}
```

- sourcehan-jp： Source Han Serif JP（源ノ明朝）+ Source Han Sans JP（源ノ角ゴシック）、地域別サブセット OTF 版。

```
\setminchofont{SourceHanSerifJP-Regular.otf}
\setlightminchofont{SourceHanSerifJP-Light.otf}
\setboldminchofont{SourceHanSerifJP-Bold.otf}
\setgothicfont{SourceHanSansJP-Medium.otf}
\setmediumgothicfont{SourceHanSansJP-Regular.otf}
\setboldgothicfont{SourceHanSansJP-Bold.otf}
\setxboldgothicfont{SourceHanSansJP-Heavy.otf}
\setmarugothicfont{SourceHanSansJP-Medium.otf}
```

- noto-otc： Noto Serif CJK JP + Noto Sans CJK JP、OTC 版。

```
\setminchofont[0]{NotoSerifCJK-Regular.ttc}
```

```
\setlightminchofont[0]{NotoSerifCJK-Light.ttc}
\setboldminchofont[0]{NotoSerifCJK-Bold.ttc}
\setgothicfont[0]{NotoSansCJK-Medium.ttc}
\setmediumgothicfont[0]{NotoSansCJK-Regular.ttc}
\setboldgothicfont[0]{NotoSansCJK-Bold.ttc}
\setxboldgothicfont[0]{NotoSansCJK-Black.ttc}
\setmarugothicfont[0]{NotoSansCJK-Medium.ttc}
```

- `noto`： Noto Serif CJK JP + Noto Sans CJK JP、言語別 OTF 版。

```
\setminchofont{NotoSerifCJKjp-Regular.otf}
\setlightminchofont{NotoSerifCJKjp-Light.otf}
\setboldminchofont{NotoSerifCJKjp-Bold.otf}
\setgothicfont{NotoSansCJKjp-Medium.otf}
\setmediumgothicfont{NotoSansCJKjp-Regular.otf}
\setboldgothicfont{NotoSansCJKjp-Bold.otf}
\setxboldgothicfont{NotoSansCJKjp-Black.otf}
\setmarugothicfont{NotoSansCJKjp-Medium.otf}
```

- `noto-jp`： Noto Serif JP + Noto Sans JP、地域別サブセット OTF 版。

```
\setminchofont{NotoSerifJP-Regular.otf}
\setlightminchofont{NotoSerifJP-Light.otf}
\setboldminchofont{NotoSerifJP-Bold.otf}
\setgothicfont{NotoSansJP-Medium.otf}
\setmediumgothicfont{NotoSansJP-Regular.otf}
\setboldgothicfont{NotoSansJP-Bold.otf}
\setxboldgothicfont{NotoSansJP-Black.otf}
\setmarugothicfont{NotoSansJP-Medium.otf}
```

- `haranoaji`： 原ノ味フォント。

```
\setminchofont{HaranoAjiMincho-Regular.otf}
\setlightminchofont{HaranoAjiMincho-Light.otf}
\setboldminchofont{HaranoAjiMincho-Bold.otf}
\setgothicfont{HaranoAjiGothic-Medium.otf}
\setmediumgothicfont{HaranoAjiGothic-Regular.otf}
\setboldgothicfont{HaranoAjiGothic-Bold.otf}
\setxboldgothicfont{HaranoAjiGothic-Heavy.otf}
\setmarugothicfont{HaranoAjiGothic-Medium.otf}
```

5.3 ptex-fontmaps 互換のプリセット設定

ptex-fontmaps のプリセット名を別名として用意した。

- `noEmbed`： `noembed` の別名。
- `kozuka`： `kozuka-pro` の別名。
- `hiragino`： `hiragino-pro` の別名。
- `hiragino-elcapitan`： `hiragino-elcapitan-pro` の別名。
- `morisawa`： `morisawa-pro` の別名。

5.4 pxchfon-extras パッケージ

本パッケージの方針として、「プロプライエタリなフォントを利用するためのプリセット設定」は、1.0 版以降は追加しないことにしている。代わりに、そのようなプリセット設定を提供するために別に pxchfon-extras **パッケージ**[*16]を用意している。

例えば「ptex-fontmaps の `hiragino-highsierra-pron` に相当するプリセット」は pxchfon-extras パッケージで提供されているので、必要な人は使用を検討してほしい。

※ pxchfon-extras パッケージをインストールするだけで pxchfon で `hiragino-highsierra-pron` 等の追加プリセットが使用可能になる。

5.5 廃止されたプリセット設定

以下に挙げるのは、0.5 版以降で非推奨となっていたプリセット設定である。これらは 1.0 版において**廃止**されたため、現在は指定するとエラーが発生する。

- `ipa-otf`： 「拡張子が `.otf` の」IPA フォント。
  ※代替のプリセットはない。
- `ipa-otf-dx`： 「拡張子が `.otf` の」IPA フォント + HG フォント。
  ※代替のプリセットはない。
- `kozuka4`： 小塚フォント（Pro 版）の単ウェイト使用。
  ※ `kozuka-pro` + `oneweight` オプションで代替可能。
- `kozuka6`： 小塚フォント（Pr6 版）の単ウェイト使用。
  ※ `kozuka-pr6` + `oneweight` オプションで代替可能。
- `kozuka6n`： 小塚フォント（Pr6n 版）の単ウェイト使用。
  ※ `kozuka-pr6n` + `oneweight` オプションで代替可能。
- `hiragino`： ヒラギノフォントの単ウェイト使用。
  ※ `hiragino-pro` + `oneweight` オプションで代替可能。
  ※ 1.2a 版以降で、`hiragino-pro` の別名として再定義された。
- `ms-dx`： `ms-hg` の別名。
- `ipa-ttf`： `ipa` の別名。
- `ipa-ttf-dx`： `ipa-hg` の別名。
- `ipav2`： `ipa` の別名。
- `ipav2-dx`： `ipa-hg` の別名。
- `ipa-dx`： `ipa-hg` の別名。
- `hiragino-dx`： `hiragino-pro` の別名。

以下に挙げるプリセット設定は TeX Live 2017（dvipdfmx 20170318 版）のために用意された特殊な設定である。1.5 版においてこれらのプリセットは**非推奨**となり、2.0 版で廃止された。これらのプリセットを利用していた場合は、**TeX システムを更新**（dvipdfmx を 20170918 版以降に）した上で、“本来の正しい設定”

[*16] pxchfon-extras パッケージ： https://github.com/zr-tex8r/PXchfon-extras

に移行することを推奨する。

- `sourcehan+`、`sourcehan-otc+`、`noto+`、`noto-otc+`：
  →代わりに + 無しの名前（`sourcehan` 等）を指定する。
  ※ `unicode` は自動的に有効になる。
  ※日本語版以外のフォントはもはや不要である。
- `yu-win10+`：
  →代わりに `yu-win10` と `unicode` を指定する。

6 ファイルプリセット機能

ファイルプリセット機能を利用すると、既存の dvipdfmx 用のマップファイルの読込を文書内で指定することが可能になる。パッケージオプションに次の何れかの形式の文字列を指定すると、ファイルプリセットの指定と見なされる。

- `tl:`**名前**： TEX Live 用ファイルプリセット。
- `file:`**名前**： 単純ファイルプリセット。

※ 1.6a 版以前では、TEX Live 用が「**+名前**」、通常が「***名前**」という形式であった。これらの古い形式も当面の間はサポートされるが、新しい形式の使用を推奨する。

6.1 TEX Live 用ファイルプリセット機能

TEX Live では (u)pLATEX のフォントの設定を kanji-config-updmap というユーティリティで行うことができる。そこでは、決まった形式のファイル名をもつ dvipdfmx 用のマップファイルを用意していて、ユーザが要求したプリセット名に対応したファイルを updmap の機構を用いて有効化することで、dvipdfmx の既定の設定を切り替えている。

パッケージオプションとして `tl:` で始まる文字列（仮に `tl:NAME` とする）を与えると、kanji-config-updmap 用のマップファイルの読込が指示される。具体的には、以下の名前のマップファイルが読み込まれる。

- pLATEX の場合：
  - `ptex-NAME.map`
  - `otf-NAME.map`
- upLATEX の場合、**上記のものに加えて**以下のもの：
  - `uptex-NAME.map`
  - `otf-up-NAME.map`

例えば、pLATEX 文書において以下のようにパッケージを読み込んだとする。

```
\usepackage[tl:yu-win]{pxchfon}
```

この場合、`ptex-yu-win.map` と `otf-yu-win.map` の 2 つのマップファイルが dvipdfmx 実行時に読み込

まれる。

6.2 単純ファイルプリセット機能

パッケージオプションとして file: で始まる文字列（仮に file:NAME とする）を与えると、NAME.map という名前のマップファイルの読込が指示される。

例えば、以下のようにパッケージを読み込んだとする。

```
\usepackage[file:yu]{pxchfon}
```

この場合、yu.map というマップファイル[*17]が dvipdfmx 実行時に読み込まれる。

7 マップファイル読込機能

既存の dvipdfmx 用のマップファイルの読込を文書内で指定ための機能として、従来の「ファイルプリセット」機能に加えて 1.9 版から新たに「マップファイル読込」機能がサポートされる。

両機能の違いは以下の通りである。

- 「ファイルプリセット」の指定は“プリセット指定の一種”と見なされるのに対し、「マップファイル読込」ではプリセット指定とは無関係に自由にマップファイルを読み込める。
- 「ファイルプリセット」はプリセットの一種であるため、高々 1 つしか指定できず、また（想定仕様上は[*18]）「本パッケージの設定対象である日本語用の論理フォント」に対するマップファイルのみを対象とする。対して、「マップファイル読込」は何回でも使用できて、また任意の dvipdfmx 用マップファイルを対象とする。
- 「マップファイル読込」によるマップ設定は、プリセット（「ファイルプリセット」も含む）やフォント指定命令（\setminchofont 等）によるマップ設定に先行して行われるため、優先度が低い。
  ※このため、「マップファイル読込」を“中国語・韓国語フォントに対するファイルプリセット指定”の代わりに使うことができる。

「マップファイル読込」は use オプションで指定する。

- use=⟨指定⟩： マップファイル読込を指示する。⟨指定⟩ は以下の何れかの形式（NAME は文字列）。
  - file:NAME： NAME.map を読み込む。（単純ファイルプリセットと同じ。）
  - tl:NAME または tl-ja:NAME：「TeX Live 用ファイルプリセット」と同じ規則で決まるファイル群（ptex-NAME.map 等）を読み込む。
  - tl-LL:NAME（※ LL は ko・sc・tc の何れか）： kanji-config-updmap 向けの中国語・韓国語用のマップファイルを読み込む設定。具体的な規則は以下の通り。
    - upLaTeX の場合は、uptex-LL-NAME.map を読み込む。
    - japanese-otf 使用時は、otf-LL-NAME.map を読み込む。
  - “:”を含まない文字列：「⟨指定⟩.map」を読み込む。（file: が付いている場合と同じ。）

[*17] 例えば W32TeX では yu.map というマップファイルが用意されている。
[*18] 実際にマップファイルの中身を検証しているわけではない。

※参考：マップ設定の優先度は以下のようになっている（後のものほど優先度が高い）。

- 「マップファイル読込」機能によるマップファイル読込。
- プリセット（「ファイルプリセット」を含む）による設定。
- フォント指定命令（\setminchofont 等）による設定。
- \usefontmapfile・\usefontmapline 命令による設定。

8 カスタムプリセット機能

1.7 版で新設された「カスタムプリセット機能」とは、ユーザが独自にプリセットを用意するためのものである。ユーザが定義したプリセット（**カスタムプリセット**）を「pxchfon.cfg」という名前の**カスタムプリセット定義ファイル**に記述してそのファイルを TEX が読める場所に置くと、pxchfon パッケージ読込時にオプションとして指定することでカスタムプリセットを利用できるようになる。

カスタムプリセットを定義する命令は以下の通りである。これらの命令は pxchfon.cfg の中でのみ利用できる。

- \pxchfonDeclareOneWeightPreset{⟨名前⟩}{⟨明朝⟩}{⟨ゴシック⟩}：単ウェイトのカスタムプリセットを定義する。プリセットの名前は ⟨名前⟩ の文字列の前に「user:」を前置したものになる。第 2 引数以降は使用するフォントファイル名（TTC 形式の場合は番号も含めて：**番号: ファイル名** の形式）を指定する。
- \pxchfonDeclareMultiWeightPreset{⟨名前⟩}{⟨明朝・細⟩}{⟨明朝・中⟩}{⟨明朝・太⟩}{⟨ゴシック・中⟩}{⟨ゴシック・太⟩}{⟨ゴシック・極太⟩}{⟨ゴシック・単⟩}{⟨丸ゴシック⟩}：多ウェイトのカスタムプリセットを定義する。引数の意味は前項と同様である。⟨ゴシック・単⟩ は「oneweight オプション指定時のゴシック体」を表す。

例えば、全ての和文フォントを（Microsoft Office 付属の）「HG 創英角ポップ体」に置き換えるような単ウェイトのカスタムプリセット user:soeikakupoptai を定義するには次の命令を実行する。

```
\pxchfonDeclareOneWeightPreset{soeikakupoptai}
  {:0:HGRPP1.TTC}{:0:HGRPP1.TTC}
```

このカスタムプリセットを利用したい場合はパッケージ読込を以下のようにする。

```
\usepackage[user:soeikakupoptai,...]{pxchfon}
```

9 Unicode 直接指定

dvipdfmx のフォントマップ設定において、和文フォントのエンコーディングを指定する方法は“CMap 指定”と“Unicode 直接指定”の 2 種類がある。[*19]かつては、Unicode で包摂されている異体字を区別するためには CMap 指定の利用が必須であったため、慣習的に、dvipdfmx のフォントマップ設定においては CMap 指定が主に用いられてきた。

[*19] 詳細については dvipdfmx のマニュアルを参照されたい。

しかしこの CMap 指定は、Adobe-Japan1（AJ1）[*20]のグリフ集合に対応した OpenType フォントにしか適用できない、という欠点がある。近年は、“AJ1 でない”OpenType フォント[*21]が普及しつつあり、そのようなフォントでは異体字の切替などの付加機能を専ら「OpenType 属性の指定」により行うことを想定している。これに対応するため、dvipdfmx のマップ指定において OpenType 属性の指定がサポートされるようになった。

pxchfon では和文フォントのエンコーディングに対する Unicode 直接指定をサポートしている。特に 1.0 版から、新しい dvipdfmx の OpenType 属性の指定を積極的に利用することで、“AJ1 でない”フォントを使用した場合でも、CMap 指定の場合の機能性を可能な限り保つことを目指している。
※ Unicode 直接指定に対するサポートは発展途上であるため、過渡的な要素が多く混ざっていてやや煩雑になっていることに注意してほしい。

■Unicode 直接指定オプション　`unicode` オプションを指定することで Unicode 直接指定の有効・無効を切り替えられる。

- `unicode`[`=full`]： 全般的に Unicode 直接指定を利用する。最も理想的な設定であるが、**TeX Live 2018 以降の** dvipdfmx（20170918 版以降）が必要である。
- `unicode=all`： 全般的に Unicode 直接指定を利用するが、古い（TeX Live 2016 以前の）dvipdfmx に対応するために、OpenType 属性の指定を全く行わない。つまり、入力の Unicode 文字に対する既定のグリフが常に出力され、異体字の区別は全て無効になる。
- `unicode=UTF`： japanese-otf の`\UTF` 入力のフォントに限って Unicode 直接指定を利用する。`all` と同じく OpenType 属性の指定を全く行わない。
  ※前述の通り `all` はデメリットが強いため、適用範囲を限定したもの。
- `unicode=false`： Unicode 直接指定を利用せず、全般的に CMap 指定を適用する。

※単に `unicode` と指定するのは `unicode=full` と等価になる。
※以前は `unicode=all` は `directunicode*`、`unicode=UTF` は `directunicode` という指定方法であったが、極めて解りにくいので 2.0 版で「key-value 型の `unicode` オプション」に統一された。従来の形式も当面はサポートされるが新しい `unicode` の使用を推奨する。[*22]

TeX Live 2017 の dvipdfmx（20170318 版）専用の**過渡的**な設定である `unicode*` は 2.0 版で**廃止**された。これを使っている場合は、**TeX システムを更新**（dvipdfmx を 20170918 版以降に）した上で、代わりに `unicode` を指定することを推奨[*23]する。

■Unicode 直接指定専用プリセット　以下に挙げるプリセット設定は“AJ1 でない”OpenType フォントを利用するものである。そのため、これらのプリセットを指定した場合は、自動的に `unicode` が（既定として）指定される[*24]。

[*20] または各々の CJK 言語の“Adobe 標準”のグリフ集合、例えば簡体字中国語なら Adobe-GB1。
[*21] 例えば、Adobe 開発のフリーフォントの「Source Han Serif（源ノ明朝）」など。
[*22] なお「単独の `unicode`」も従来の動作を維持しているが、これは古い形式ではなく「新しい `unicode` の指定法の一種」として扱う。
[*23] `unicode=all` は TeX Live 2017 でも使えるが、出力は元々の `unicode*` より劣化する。
[*24] 1.0〜1.1b 版では TeX Live 2017 のための暫定措置として「`+` 付の特殊プリセットへの自動振替」が行われていたが、1.2 版から本来の仕様が適用される。

- sourcehan-otc
- sourcehan
- sourcehan-jp
- noto-otc
- noto
- noto-jp

9.1 Unicode 直接指定に関する注意事項

- プリセット指定または \set...font 命令でフォントを置換していないウェイトについては、Unicode 直接指定は無効になる。従って、そのようなウェイトは想定する出力にはならず、また Unicode 直接指定を必要とする機能（expert や glyphid の各オプション）も使えない。
- 特に、expert については、非置換のウェイトの動作が非常に複雑で解りにくい。従って、expert を指定する場合は、全ファミリ・ウェイトについて置換を指定することを推奨する（これに反する場合は警告が出る）。
- 非埋込を指定したウェイトについては Unicode 直接指定は無効になる。この動作も非常に解りにくいため、Unicode 直接指定を用いる場合は、非埋込の指定をしないことを推奨する（これに反する場合は警告が出る）。

10 dvipdfmx のページ抜粋処理への対応

dvipdfmx には元の DVI 文書の一部のページだけを抜粋して PDF 文書に変換する機能がある（-s オプション）。ところが、本パッケージではユーザが指定したフォントマップ情報を DVI の先頭ページに書き出すという処理方法をとっている（すなわち「ページ独立性」を保っていない）ため、先頭ページを含まない抜粋を行った場合は、フォント置換が効かないという不具合が発生する。

この問題を解決するのが everypage パッケージオプションである。このオプションが指定された場合は、DVI 文書の全てのページにフォントマップ情報を書き出すので、ページ抜粋を行っても確実にフォント置換が有効になる。ただし、このオプションを指定する場合は atbegshi パッケージが必要である。

11 欧文フォントの置換の原理

指定された和文フォントの半角部分からなる欧文フォントファミリとして cfjar（明朝）、cfjas（ゴシック）、cfjam（丸ゴシック）の 3 つ（以下では **CFJA ファミリ**と総称する）を定義している。その上で、CFJA ファミリに対するマップ指定を和文と同じ方法で行っている。

■alphabet オプション指定時　オプション alphabet を指定した場合、CFJA ファミリを既定の欧文ファミリとして設定する（cfjar →\rmdefault；cfjas →\sfdefault）。従って、例えば一時的に従来の CM フォントを使いたい場合は、適宜ファミリの変更（\fontfamily{cmr}等）を行えばよい。

■relfont オプション指定時　オプション relfont を指定した場合、CFJA ファミリを和文ファミリの従属欧文フォントに設定する（cfjar →\mcfamily；cfjas →\gtfamily；cfjam →\mgfamily）。

■利用可能な欧文エンコーディング　現状で、CFJA ファミリが対応している欧文エンコーディングは OT1、T1、TS1 の 3 つである[*25]。このため、欧文フォントの置換（alphabet や relfont オプション）を利用する文書は、欧文エンコーディングが OT1 か T1 である必要がある[*26]。

様々な理由により、利用できる文字の種類が限られている。具体的には、以下で挙げられている文字（でかつ当該の欧文エンコーディングに含まれるもの）が使用可能である。

1. 欧文フォント（CFJA ファミリ）が“CMap 指定”である場合は以下の文字。
   ※ AJ1 対応フォント用のプリセット（kozuka-pr6n 等）を指定した場合は既定で欧文が“CMap 指定”になる。
   a) ASCII 文字[*27]および en-ダッシュ（–）。
      ※これらは半角幅である。
   b) クオート（“ ” ‘ ’）。
      ※これらは和文用の全角幅の字形を、空きを詰めて半角幅で出力する。
   c) em-ダッシュ（—）。
      ※これらは U+2015 の全角幅の字形をそのまま全角幅で出力する。
2. 欧文フォント（CFJA ファミリ）が“Unicode 直接指定”である場合は以下の文字。
   - 1 項の a、b、c の文字はフォントが正しい幅（a は**半角**、b と c は**全角**）の字形を持っていれば使える。
   - それ以外でも、Unicode 文字の大部分は、フォントが**半角幅**の字形を持っていれば使える。
     ※半角幅のまま出力される。
   - unicode オプションを指定すると、正しい幅の字形を優先して使おうとするため、既定の場合よりも多くの文字が使える。

12　注意事項

- 指定できるフォントは等幅のものに限られる。実際に使われるメトリックは置換前と変わらない。（例えば jsarticle の標準設定なら JIS メトリック）
- 欧文部分を置き換えた場合、残念ながら欧文も等幅（半角幅）になってしまう。
- japanese-otf パッケージ使用時に \UTF や \CID で指定した文字が出力されるかは、指定したフォントがその文字を持っているかに依存する。
- deluxe 付きの japanese-otf パッケージと alphabet 付きの pxchfon を同時に使う場合には、japanese-otf パッケージを先に読み込む必要がある。（これに反した場合はエラーになる。）
- 単ウェイトの場合は、明朝の太字はゴシックになるという一般的な設定に欧文フォントの置換の際にも

[*25] 1.4 版で T1 に“暫定対応”し、1.5 版で T1 と TS1 に正式に対応した。
[*26] TS1 は記号用のエンコーディングであり、本文の既定のエンコーディングとして使うものではない。
[*27] 出力される文字の話であることに注意。例えば、OT1 や T1 で ' （U+0027）を入力したときに出力される文字は ’（U+2019）であり、これは ASCII 文字ではない。なお、TS1 は U+0027 を含む。

従っているが、明朝のみが置換されている場合は、明朝の置換フォントが太字にも適用される。

付録 A　dvipdfmx 以外の DVI ウェアでの使用

本パッケージの核心の機能である「使用フォントを文書中で指定する」ことの実現には dvipdfmx の拡張機能を利用している。従って、dvipdfmx の利用が必須となるのだが、（プレビュー等の目的で）文書中で指定したフォントが反映されなくてもよいのなら、他の DVI ウェアでも本パッケージを利用した DVI 文書を扱える可能性がある。

A.1　和文フォントだけを置き換えた場合（noalphabet 指定時）

この設定で生成される DVI ファイルを dvipdfmx 以外の DVI ウェアに読ませた場合、フォント置換が無視され、そのソフトウェアで設定されたフォントで出力されるはずである。

A.2　欧文フォントも置き換えた場合（alphabet 指定時）

欧文フォントを置き換えた DVI ファイルは、独自の欧文フォント（r-cfja* という形式の名前）を含んでいるので、少なくともそれに関する設定をしない限りは dvipdfmx 以外の DVI ウェアで処理することができない。さらに、このフォントを扱うためには DVI ウェアがサブフォント（sfd）に対応している必要がある。文書中での設定を dvipdfmx 以外の DVI ウェアで活かすことはできない。しかし、「独自部分の欧文フォントを常に特定の代替フォントで表示させる」ということは、sfd 対応の DVI ウェアであれば可能である。

以下に、ttf2pk について、「常に MS フォントで代替する」ための設定を掲げておく。この記述を ttf2pk のマップファイル（ttfonts.map）に加えると、例えば、dviout で本パッケージ使用の DVI ファイルを閲覧できるようになる。

```
r-cfjar-l-@PXcjk0@     msmincho.ttc FontIndex=0
r-cfjar-r-@PXcjk0@     msmincho.ttc FontIndex=0
r-cfjar-b-@PXcjk0@     msmincho.ttc FontIndex=0
r-cfjas-r-@PXcjk0@     msgothic.ttc FontIndex=0
r-cfjas-b-@PXcjk0@     msgothic.ttc FontIndex=0
r-cfjas-x-@PXcjk0@     msgothic.ttc FontIndex=0
r-cfjam-r-@PXcjk0@     msgothic.ttc FontIndex=0
```

付録 B　pxjafont パッケージ

現在の版の pxchfon パッケージは旧来の pxjafont の機能を取り込んでいるため、pxjafont は不要である。古い環境との互換性のため pxjafont を残していたが、2.0 版において**廃止**された。

pxjafont を利用していたユーザは、5 節を参照して現在の pxchfon 用の適切な設定に書き直す必要がある。※古いプリセット名の中に廃止されたものがあるので注意。

付録 C　中国語・韓国語フォントへの対応

0.7c 版で japanese-otf パッケージ（multi オプション指定）および upTEX 標準の中国語・韓国語フォントについてのサポートを始めた。

■単ウェイトの場合の設定　以下の命令が用意されている。

- \setkoreanminchofont[⟨番号⟩]{⟨フォントファイル名⟩}： 韓国語・明朝体。
- \setkoreangothicfont[⟨番号⟩]{⟨フォントファイル名⟩}： 韓国語・ゴシック体。
- \setschineseminchofont[⟨番号⟩]{⟨フォントファイル名⟩}： 簡体字中国語・明朝体（宋体）。
- \setschinesegothicfont[⟨番号⟩]{⟨フォントファイル名⟩}： 簡体字中国語・ゴシック体（黒体）。
- \settchineseminchofont[⟨番号⟩]{⟨フォントファイル名⟩}： 繁体字中国語・明朝体（明体）。
- \settchinesegothicfont[⟨番号⟩]{⟨フォントファイル名⟩}： 繁体字中国語・ゴシック体（黒体）。

■多ウェイトの場合の設定　japanese-otf パッケージ（upTEX 対応版）の 0.26 版から、中国語・韓国語のフォント[*28]についても多ウェイトがサポートされるようになった。

本パッケージの 1.9 版から中国語・韓国語の多ウェイト設定をサポートする。以下の命令が用意されている。
※引数の書式は全て\setminchofont と同じなので省略する。
※日本語用の\setminchofont 等と同様に、多ウェイト環境において単ウェイト用の命令を用いた場合は、それは全ウェイトに対する設定と見なされる。

- \setkoreanlightminchofont： 韓国語・明朝体・細ウェイト。
- \setkoreanmediumminchofont： 韓国語・明朝体・中ウェイト。
- \setkoreanboldminchofont： 韓国語・明朝体・太ウェイト。
- \setkoreanmediumgothicfont： 韓国語・ゴシック体・中ウェイト。
- \setkoreanboldgothicfont： 韓国語・ゴシック体・太ウェイト。
- \setkoreanxboldgothicfont： 韓国語・ゴシック体・極太ウェイト。
- \setkoreanmarugothicfont： 韓国語・丸ゴシック体。
- \setschinese lightminchofont： 簡体字中国語・明朝体（宋体）・細ウェイト。
- \setschinesemediumminchofont： 簡体字中国語・明朝体（宋体）・中ウェイト。
- \setschineseboldminchofont： 簡体字中国語・明朝体（宋体）・太ウェイト。
- \setschinesemediumgothicfont： 簡体字中国語・ゴシック体（黒体）・中ウェイト。
- \setschineseboldgothicfont： 簡体字中国語・ゴシック体（黒体）・太ウェイト。
- \setschinesexboldgothicfont： 簡体字中国語・ゴシック体（黒体）・極太ウェイト。
- \setschinesegothicfont： 簡体字中国語・丸ゴシック体。
- \settchinese lightminchofont： 繁体字中国語・明朝体（明体）・細ウェイト。
- \settchineseboldminchofont： 繁体字中国語・明朝体（明体）・中ウェイト。
- \settchinesexboldminchofont： 繁体字中国語・明朝体（明体）・太ウェイト。

[*28] ただし、japanese-otf パッケージにおける中国語・韓国語入力は飽くまでコード値入力（\UTFC 等）であり、直接入力はサポート外であることに注意。

- \settchinesemediumgothicfont： 繁体字中国語・ゴシック体（黒体）・中ウェイト。
- \settchineseboldgothicfont： 繁体字中国語・ゴシック体（黒体）・太ウェイト。
- \settchinesexboldgothicfont： 繁体字中国語・ゴシック体（黒体）・極太ウェイト。
- \settchinesegothicfont： 繁体字中国語・丸ゴシック体。

C.1 注意事項

- プリセット指定は中国語・韓国語のフォントについては何も指定しない。従って、上記の命令を用いない場合は、これらのフォントのマップ再設定が行われることはない。
- マップファイル読込機能（7 節参照）を利用してマップファイルを読み込むことで、中国語・韓国語フォントのマップを設定することも可能である。
  ※「ファイルプリセット機能」の方はプリセット設定の一種であるため、日本語用のフォントについて用いられることが想定されている。
- 「Unicode 直接指定オプション」は中国語・韓国語のフォントに対しても有効である。unicode=UTF の場合は「japanese-otf パッケージの Unicode 入力命令」（\UTFK、\UTFM、等）が対象となり、それ以外はこれに加えて upTEX 標準のフォントも対象になる。

付録 D 警告抑止用のオプション・開発者用フラグ

本パッケージのオプション設定は極めて複雑なため、使用法によっては「動作が極めて解りにくい状態」や「ユーザ命令の範囲ではサポートされない状態」といった“奇妙な”状態に移行していまうことがある。一方で「そういう“奇妙な”設定を解った上で敢えて利用している」「本パッケージ以外の手段で生成したフォントマップ行を併用している」などの理由で、“奇妙な”設定が正当である可能性もある。このため、現状の方針としては「“奇妙な”状態になった場合は（エラーでなく[*29]）**警告を出す**」ことにしている。

しかし、正当に利用している人にとってはこの警告は煩わしいはずなので、特定の警告を抑止するための手段を用意する。警告抑止の方法は「オプション」と「開発者用フラグ」の 2 種類がある。

■開発者用フラグ 開発者用フラグ[*30]とは、例えば \pxchfonNoCheckMultiweight のように、本パッケージが指定する特定の制御綴のことである。制御綴が**定義済である**場合にフラグが有効であると判断する。従って、本パッケージの機能と連携するパッケージの開発者は、

```
\let\pxchfonNoCheckMultiweight=t
```

のようなコードを実行することで、不要な警告を抑止できる。
※定義済であれば意味は何でもよいが、上記コードのように「文字トークンの t」を使うことを推奨する。

■オプション・開発者用フラグの一覧 以下のものが用意されている。

- maybe-multiweight オプション／\pxchfonNoCheckMultiweight フラグ：「deluxe オプション付きの japanese-otf パッケージが読み込まれていないにもかかわらず多ウェイト用のフォント設定命令が

[*29] 将来的に、警告抑止の手段が十分に周知された段階で、エラーに移行することも考えられる。
[*30] 開発者用の機能であるため、ここでは TEX 言語の知識を仮定する。

使われた」場合に出る警告を抑止する。
※例えば「単ウェイト・多ウェイト兼用を意図したフォント設定命令をテンプレートに記述している」「japanese-otf 以外でその和文 TFM を流用している」などの状況を想定している。
※有効化した場合、たとえ japanese-otf が読み込まれていなくても japanese-otf 用のフォントマップ行を書き込む。

- `nocheck-expert` オプション／`\pxchfonNoCheckExpert` フラグ：「Unicode 直接指定と `expert` オプションが有効で、かつ置換されていないウェイトが存在する」場合に出る警告を抑止する。