BXjscls パッケージ
(BXJS 文書クラス集)
ユーザマニュアル

八登崇之 (Takayuki YATO; aka. “ZR”)

v2.9e　[2026/03/29]

概要

本パッケージに含まれる文書クラス集は、奥村晴彦氏および“日本語 TEX 開発コミュニティ”により作製された jsclasses パッケージの文書クラス集の拡張版に相当する。元の jsclasses のクラスが pLATEX と upLATEX のみをサポートするのに対して、本パッケージのクラスは主要エンジンの全てをサポートする。

目次

注意

BXJS 文書クラスについては、サイト “TEX Wiki” 中の記事、[*1]およびそこからたどれる情報も併せて参照してほしい。

1 本パッケージの目的

本パッケージに含まれる文書クラス集（以下では **BXJS（文書）クラス**と呼ぶ）は、奥村晴彦氏および “日本語 TEX 開発コミュニティ” により作製された jsclasses パッケージの文書クラス集（以下では **JS（文書）クラス**と呼ぶ）の拡張版に相当する。JS クラスのレイアウトデザインと機能をほぼ踏襲しているが、以下の点で改良が加えられている。

- JS クラスは pLATEX と upLATEX のみをサポートするが、BXJS クラスはこれらに加えて (pdf)LATEX と XeLATEX と LuaLATEX をサポートしており主要エンジンの全てで使用可能である。
- (u)pLATEX 以外では各々のエンジンの日本語処理パッケージを利用するが、“標準設定” を用いることで、それらのパッケージの設定を書かずに済ませられるので、pLATEX 並に簡単に日本語の文書を書き始めることができる。
- JS クラスでは、フォントのオプティカルサイズを最適にするため、基底フォントサイズが 10 pt 以外の時に TEX の版面拡大（mag）機能を利用しているが、これが他のパッケージと衝突して不具合を起こすことがある。BXJS クラスでは mag 機能を使う他に別の方式を選べるようにしている。
  ※ JS クラスについても新しい（2016/07/11 以降の）版では同様の機能が提供されている。
- 用紙サイズや基底フォントサイズについて、任意の値を指定することができる。

[*1] https://texwiki.texjp.org/?BXjscls

- 文書形式変換ツール Pandoc を用いた日本語 LaTeX 文書生成のために調整された設定である「Pandoc モード」が利用できる。

2 最も基本的な使い方

ここでは、BXJS クラスを“標準設定”（standard 和文ドライバ）で用いる場合について解説する。この場合、`\documentclass` 命令を次のように書く。[*2]

```
\documentclass[⟨エンジン⟩,⟨ドライバ⟩,ja=standard,jafont=⟨フォント指定⟩,⟨他オプション⟩]
    {⟨クラス名⟩}
```

- ⟨エンジン⟩ の指定は必須で、実際に使っている「LaTeX のコマンド名」を書く。`platex`、`uplatex`、`pdflatex`、`xelatex`、`lualatex` 等が指定できる。
- DVI 出力のエンジンを使う場合は、⟨ドライバ⟩ の指定が必須で、これは実際に使っている「DVI ウェアの名前」を書く。`dvips`、`dvipdfmx`、`dviout`、`xdvi` が指定できる。PDF 出力のエンジンの場合は ⟨ドライバ⟩ の指定は不要である。
- “標準設定”を適用するので `ja=standard` を指定する。
- 既定以外のフォント設定を利用する場合は、⟨フォント指定⟩ にその名前を書く。既定の設定を用いる場合は `jafont=...` 自体を省略する。
- その他のクラスオプション（a4paper 等）については、多くの場合 JS クラスと同じものが使える。
- BXJS クラスについて、⟨クラス名⟩ は以下のものがある。
  - bxjsarticle：章のないレポート（jsarticle に相当する）
  - bxjsreport：章のあるレポート（jsreport に相当する）
  - bxjsbook：書籍（jsbook に相当する）
  - bxjsslide：スライド（jsarticle＋`slide` に相当する）

XeLaTeX で bxjsarticle クラスを用いた完全な文書ソースの例を示す。[*3]

```
\documentclass[a4paper,xelatex,ja=standard]{bxjsarticle}
\usepackage[unicode,colorlinks,
    pdftitle={いきなり日本語}]{hyperref}
\title{いきなり日本語}
\author{七篠 権兵衛}
\begin{document}
\maketitle

\section{日本語で{\LaTeX}する}
中身はまだない。

\end{document}
```

以下では各エンジンについて、挙動を少し詳しく説明する。

[*2] もちろんクラスオプションの順序は任意である。
[*3] 組版結果における日付の出力は JS クラスと同様の「2015 年 7 月 3 日」の形式になる。

2.1 pLaTeX の場合

例えば次の設定は：

```
\documentclass[a4paper,platex,dvipdfmx,ja=standard]{bxjsarticle}
```

対応する JS クラスを用いた次の設定とほぼ等価になる[*4]：

```
\documentclass[a4paper,dvipdfmx]{jsarticle}
```

次のように jafont を指定した場合は：

```
\documentclass[a4paper,platex,dvipdfmx,ja=standard,jafont=ms]{bxjsarticle}
```

jafont の値をプリセットオプションとして pxchfon が読み込まれる：

```
\documentclass[a4paper,dvipdfmx]{jsarticle}
\usepackage[ms]{pxchfon}
```

2.2 upLaTeX の場合

例えば次の設定は：

```
\documentclass[a4paper,uplatex,dvipdfmx,ja=standard]{bxjsarticle}
```

次の設定とほぼ等価になる[*5]：

```
\documentclass[uplatex,a4paper,dvipdfmx]{jsarticle}
```

jafont オプションの扱いは pLaTeX の場合と同じである。

2.3 pdfLaTeX の場合

エンジン指定が pdflatex の場合、日本語処理パッケージとして bxcjkjatype（これ自体は内部で CJK パッケージを読み込む）を利用する。

例えば次の設定は：

```
\documentclass[a4paper,pdflatex,ja=standard]{bxjsarticle}
```

次の設定と大体同じであるが、ただし文書レイアウトは article でなく jsarticle とほぼ同じになっている：

```
\documentclass[a4paper]{article}% ただしレイアウトは jsarticle 相当
\usepackage[whole,autotilde]{bxcjkjatype}
```

jafont を指定した場合は：

[*4] すなわち、論理フォントは明朝が jis、ゴシックが jisg が使われる。なお、BXJS では mingoth 等の論理フォント変更のオプションはサポートされていない。

[*5] 論理フォントについては、従来のもの（明朝が upjisr-h、ゴシックが upjisg-h）に代わって、BMP 外の文字に対応したもの（明朝が upjpnrm-h、ゴシックが upjpngt-h）を採用した。組み方は従来のものと変わらない。

```
\documentclass[a4paper,pdflatex,ja=standard,jafont=ipaex]{bxjsarticle}
```

その値が bxcjkjatype のフォントプリセットになる。

```
\documentclass[a4paper]{article}% ただしレイアウトは jsarticle 相当
\usepackage[whole,autotilde,ipaex]{bxcjkjatype}
```

※補足：

- 自動的に文書本体が CJK* 環境[*6]で囲まれかつ \CJKtilde が有効な状態になっている。従っていきなり日本語を書き始めることができる。ただし和欧文間空白（四分空き）は手動で ~ を入れる必要がある。[*7]日本語出力の挙動の詳細については bxcjkjatype のマニュアルを参照してほしい。以下に完全な文書ソースの例を示す：

```
\documentclass[a4paper,pdflatex,ja=standard]{bxjsarticle}
\begin{document}
日本語で~pdf{\LaTeX}~するテスト。
\end{document}
```

- bxcjkjatype パッケージにおけるフォントの既定設定は「Type1 形式の IPAex フォント」（ipaex-type1 パッケージ）である。一方、ipaex プリセットを指定した場合は「TrueType 形式の IPAex フォント」が使われるので、両者の出力は見かけは同じであるが、PDF データとしては異なる。[*8]

2.4 XeLaTeX の場合

エンジン指定が xelatex の場合、日本語処理パッケージとして zxjatype（これ自体は内部で xeCJK パッケージを読み込む）を利用する。

例えば次の設定は：

```
\documentclass[a4paper,twocolumn,xelatex,ja=standard]{bxjsarticle}
```

次の設定と大体同じであるが、ただし文書レイアウトは jsarticle とほぼ同じになっている：

```
\documentclass[a4paper,twocolumn]{article}% ただしレイアウトは jsarticle 相当
\usepackage{zxjatype}
\setCJKmainfont[BoldFont=HaranoAjiGothic-Medium.otf]{HaranoAjiMincho-Regular.otf}
\setCJKsansfont[BoldFont=HaranoAjiGothic-Medium.otf]{HaranoAjiGothic-Medium.otf}
```

※ 2.0 版より、既定の和文フォントが「IPAex フォント」から「原ノ味フォント」に変更された。

jafont を指定した場合は：

```
\documentclass[a4paper,xelatex,ja=standard,jafont=ms]{bxjsarticle}
```

その値が zxjafont のプリセットになる。

```
\usepackage{zxjatype}
```

[*6] CJKspace パッケージが読み込まれた下での CJK* 環境である。

[*7] CJK パッケージには自動で和欧文間空白を入れる機能はない。

[*8] ちなみに、bxcjkjatype には ipaex-type1 というオプションもあるが、この設定と既定設定（オプション無し）も動作は異なる。BXJS クラスが用いるのは既定設定の方である。

```
\usepackage[ms]{zxjafont}
```

2.5 LuaLaTeX の場合

エンジン指定が lualatex の場合、日本語処理パッケージとして LuaTeX-ja を利用する。
例えば次の設定は：

```
\documentclass[b5paper,9pt,lualatex,ja=standard]{bxjsarticle}
```

次の設定とほぼ等価になる（ただし luatexja-preset は実際には読み込まれない）：

```
\documentclass[b5paper,9pt]{ltjsarticle}
\usepackage[haranoaji]{luatexja-preset}% 実際にはパッケージは読み込まれない
```

※ 2.0 版より、既定の和文フォントが「IPAex フォント」から「原ノ味フォント」に変更された。
jafont を指定した場合は：

```
\documentclass[b5paper,lualatex,ja=standard,jafont=ms]{bxjsarticle}
```

次の設定とほぼ等価になる：

```
\documentclass[b5paper]{ltjsarticle}
\usepackage[ms]{luatexja-preset}% 実際にパッケージが読み込まれる
```

※補足：

- luatexja-preset パッケージの読込が行われるのは jafont を指定した場合に限られる。

2.6 注意事項

主に JS クラスとの違いについての注意。

- ページレイアウトについて、JS クラスの設計思想を受け継いでいるが、全く同じになるわけではない。
- JS クラスの一部のオプションで、BXJS クラスでは使用不可能なものがある。（4.3 節参照。）
- BXJS クラスではページレイアウトを設定するために内部で geometry パッケージを読み込んでいる。そのため、後からユーザが geometry を読み込むことはできない。ページレイアウトを変更する場合は、BXJS クラスが用意している再設定用の命令（7.1 節参照）か、または geometry パッケージが提供する再設定用命令（\geometry 等）を利用する。
- papersize オプションは既定で有効になっていて、出力用紙サイズはクラスオプションで指定したものに自動的に設定される。この処理を無効にするには nopapersize オプションを指定すればよい。
- papersize オプションの処理は geometry パッケージの機能により行われる。hyperref パッケージや（最近の）graphics／color パッケージがもつ出力用紙サイズ設定の機能はこれと干渉する可能性があるので、BXJS クラスにおいては自動的に無効化される。
- hyperref パッケージにおける“PDF の文字コード”の設定はエンジンごとに適切な値が異なっていて複雑であり、これが不適切であるために PDF 文書情報（しおり等）が文字化けしてしまう事例が数知

れない。そこで、文書クラス側でエンジン毎に適切な設定を予め行うようにした。[*9]（ただし文書クラスが hyperref を読み込むわけではない。）

- pdfLaTeX 上で hyperref で pdftitle 等の文書情報に和文文字を含めたい場合は、\hypersetup 命令を通常通り使うことができる。[*10]

```
\documentclass[pdflatex,a4paper,ja=standard]{bxjsarticle}
\usepackage[colorlinks]{hyperref}
\hypersetup{pdftitle={日本語タイトル}}
```

※ hyperref のパッケージオプションで和文文字を含む文書情報を指定することはできない。（hyperref の制限仕様。）

- (u)pLaTeX において、jafont が無い場合の既定のフォント設定は「何も指定しない状態」（JS クラスと同様）である。すなわち実際に使われる物理フォントの選択は DVI ウェアの設定に委ねられている。
- 1.2 版より、\bf や \it 等の LaTeX 2.09 方式のフォント選択命令の使用が非推奨となり、これらの命令を使うと警告が出るようになった。詳細は 9 節を参照。

3 Pandoc モード

「Pandoc モード」は文書形式変換ツールである Pandoc[*11]を用いた日本語 LaTeX 文書生成（および LaTeX 経由の PDF 生成）のために調整された設定である。特に「Pandoc（2.0 版以降）の既定の LaTeX 出力用テンプレートをそのまま用いても日本語文書として適切な出力が得られる」ことを主要な目的としている。

現状の Pandoc モードでは以下の調整が行われる。

※これらの一部についてはユーザによる設定変更が可能である。詳細は 6.2 節、7.3 節を参照。

- Pandoc の Strong（重要）要素の出力を \strong 命令に合わせるため、\textbf 命令を \strong 命令に移譲させる。
- \texttt 命令・\verb 命令と和文との間に入る和欧文間空白を調節している。
- Pandoc は入力文書中の三点リーダ“…”を \ldots 命令に変換するが、このせいで欧文扱いになるのを回避するため、\ldots を一定の条件下において和文で出力させる。
- 全角空白文字の入力を（文字ではなく）空きの挿入と解釈する。
- Pandoc の一部の設定は \paragraph の見出しの体裁を変更するが、この場合に連動して paragraph-mark の既定値を空に変更する。
- Pandoc の一部の設定はプレアンブルでの geometry の読込を発生させるが、この場合に重複読込のエラーになるのを回避するため、geometry の読込を \setpagelayout* の呼出に振り替える。
- 入力文書が言語指定を含む場合には Babel パッケージが読み込まれるが、この際に発生する可能性がある不整合を回避する。
- hyperref パッケージの unicode オプションの有無について適切な選択（エンジンにより決まる）を強制する。((u)pLaTeX をサポートするための措置。)

[*9] 従って、(u)pLaTeX において、ほとんどの場合に pxjahyper パッケージを読み込む必要がない。ただし読み込んでも構わないし、必要な場合もある。

[*10] ちなみに、普通に CJKutf8 パッケージを用いた場合は、この方法では失敗してしまう。恐らく \hypersetup 命令全体を CJK* 環境で囲う必要があるのだと思われる。

[*11] http://pandoc.org/

3.1 Pandoc モードの使い方

Pandoc モードを使う場合はクラスオプションを以下のように指定する。

```
\documentclass[pandoc,⟨ドライバ⟩,jafont=⟨フォント指定⟩,⟨他オプション⟩]{⟨クラス名⟩}
```

- クラスオプションに pandoc を指定し、代わりに「エンジン」と「和文ドライバ（ja）」のオプションを省く。[*12]
- エンジンが DVI 出力である場合のドライバの既定値が dvipdfmx になる。ただし明示的にドライバオプションを与えることで dvips などに変更できる。
- 「和文フォント（jafont）」および他のクラスオプションは従来通り使用できる。

3.2 Pandoc の設定の例

BXJS クラスを用いる場合の Pandoc の設定について、以下で例を示す。
※なお、以下の記述は Pandoc 2.x 版に従う。（3,x 版でも通用すると思われる。）Pandoc のオプション体系は 2.x 版と 1.x 版とでかなり異なることに注意されたい。

■bxjsarticle の例　以下の設定で bxjsarticle クラスを使用したい。

- XeLaTeX 経由
- 用紙サイズは A4 判
- 和文フォント設定は ipaex

この場合のコマンド行は次のようになる：[*13]

```
pandoc ⟨入力ファイル名⟩ -o ⟨出力ファイル名⟩.pdf -t latex-smart
  --pdf-engine=xelatex -V papersize=a4 -V documentclass=bxjsarticle
  -V classoption=pandoc -V classoption=jafont=ipaex
```

なおこの場合、途中で生成される LaTeX 文書のクラス指定は以下のようになる：

```
\documentclass[a4paper,jafont=ipaex,pandoc]{bxjsarticle}
```

※注意事項：

- Pandoc の latex 出力の smart 機能は普通の日本語文書においては悪影響を及ぼすので、変換先書式（-t）の値は latex でなく latex-smart（smart 機能を無効化）とすべきである。

■bxjsbook の例　以下の設定で bxjsbook クラスを使用したい。

- 節番号を出力する

[*12] pandoc が指定された場合は、エンジンオプションの値は autodetect-engine、和文ドライバの値は pandoc に固定される。
[*13] もちろん、実際には改行を含めず 1 行で書く。以降の例でも同様。

- LuaLaTeX 経由
- 用紙サイズは JIS B5 判
- 和文の基底フォントサイズは 11 Q
- 和文フォント設定は `haranoaji`
- 欧文フォントを Pandoc の機能で設定

この場合のコマンド行は次のようになる：

```
pandoc ⟨入力ファイル名⟩ -o ⟨出力ファイル名⟩.pdf -N -t latex-smart
  --top-level-division=chapter --pdf-engine=luslatex
  -V papersize=b5 -V documentclass=bxjsbook -V classoption=pandoc
  -V classoption=jbase=11Q -V classoption=jafont=haranoaji
  -V mainfont="TeX Gyre Termes" -V sansfont="TeX Gyre Heros"
```

※注意事項：

- bxjsbook クラスは「章（`\chapter`）」をもつクラスなので、Pandoc で `--top-level-division=chapter` の指定が必要。
- 既定テンプレート使用時は、`papersize=〜` の指定は「クラスオプションに `〜paper` を指定する」という形で反映される。BXJS クラスでは `b〜paper` は JIS の B 列を表すので、`papersize=b5` で JIS B5 判に設定される。出力形式を LaTeX 以外にする（あるいは他の文書クラスを利用する）場合は `papersize=b5` で ISO B5 判に設定される可能性があることに注意。

■(u)pLaTeX の使用　Pandoc で LaTeX 経由で PDF を出力する場合、エンジン指定（`--pdf-engine`）は `pdflatex`、`xelatex`、`lualatex` のみがサポートされる。これに対して、Pandoc の出力を「単体の LaTeX 文書」（`-s` 指定）とすると Pandoc が LaTeX を実行しなくなり、この場合は LaTeX エンジンに (u)pLaTeX を使用することができる。出力された LaTeX 文書は通常の方法で PDF や PostScript 形式に変換できる。

```
pandoc mydoc.md -o mydoc.tex -s -V documentclass=bxjsreport -V classoption=pandoc ...
uplatex mydoc
uplatex mydoc
dvipdfmx mydoc
```

4　クラスオプション

一般的な値の書式についての注意：

- 真偽値は、`true`（真）または `false`（偽）で指定する。
- 長さ値および整数値は calc パッケージの式（`2cm+3pt` 等）で指定できる。
- ⟨長さ*⟩ のように“*”のついた長さパラメタについては、calc の長さ式の代わりに、和文用単位（`Q`、`H`、`zw`、`zh`）付きの実数値で指定することもできる。[*14]

[*14] この機能は `units` 和文パラメタとは無関係である。和文用単位を含む長さ式は（(u)pLaTeX 以外では）使えない。なお、ここで使われる `zw` は常に（(u)pLaTeX でも）“規約上の全角幅”（`\jsZw`）を指す（和文フォント設定が未完了のため）。

- 真偽値を引数にもつオプションについて、引数を省略してキー名のみを指定した場合は、引数に `true` を与えたものと見なされる。例えば、`use-zw` というオプション指定は `use-zw=true` と等価になる。

4.1 BXJS クラスに特有のオプション

JS クラスには無く BXJS クラスで追加されたクラスオプション。

- エンジンオプション：実際に使用する LaTeX エンジン（実行コマンド名）を指定する。有効な値は `latex`、`platex`、`uplatex`、`pdflatex`、`xelatex`、`lualatex`、`platex-ng` である。エンジンオプション（と次項の `autodetect-engine` の何れか）の指定は必須である。
  ※ `platex-ng` は pTeX-ng エンジン（別名 ApTeX、Asiatic pTeX）のためのオプションである。現時点では `platex-ng` 指定時の仕様は流動的であり詳細を述べることは避けるが、大体は `uplatex` と同様になる。pTeX-ng は PDF 出力を行うので、ドライバオプションは指定しない。[*15]
- `autodetect-engine`：使用しているエンジンを判定して、自動的に適切なエンジンオプションを設定する。[*16]
  ※ BXJS クラスの設計の思想としては、「LaTeX 文書がどのエンジンでコンパイルすべきものかはソース中に明示されるべき」と考えていて、従って、人間が“普通に”文書を作る際にはこのオプションの使用は推奨されない。このオプションは「LaTeX ソースの自動生成」が絡む処理を念頭において用意されている。
- ドライバオプション：DVI 出力のエンジンを用いる場合に、実際に使用する DVI ウェアの名前を指定する。有効な値は `dvips`、`dvipdfmx`、`dviout`、`xdvi` である。DVI 出力時はドライバオプション（および次項の `nodvidriver` と `dvi` の何れか）の指定は必須である。[*17]
- `nodvidriver` ／ `class-nodvidriver`：特殊なドライバオプションの一種で、BXJS クラスが持つドライバ依存の機能を全て抑止することを指定する。[*18]
  ※ BXJS クラス内では `nodvidriver` と `class-nodvidriver` は等価であるが、「何がグローバルオプションにあるか」という観点で両者は異なる。現状で `nodvidriver` オプションを認識するパッケージがあり、`nodvidriver` の方はそれらのパッケージについても有効になる。
- `dvi=`⟨ドライバオプション名⟩：エンジンが DVI 出力の場合に限り、指定のドライバオプションを有効にする。[*19]`autodetect-engine` と一緒に使うことが想定されている。
- `pandoc`：「Pandoc モード」（3 節参照）を指定する。以下の設定が行われる：
  - エンジンオプションが `autodetect-engine` に**固定**される。
  - 和文ドライバが `pandoc` に**固定**される。
  - ドライバオプションについて `dvi=dvipdfmx` が既定になる（明示指定で上書可能）。

[*15] 現状の実装では、自動的に `dvipdfmx` がグローバルオプションに追加される。また、`platex-ng` オプションの変種に `platex-ng*` があり、こちらは `dvipdfmx` の自動追加を行わない。

[*16] 実はエンジンの判定は常に行っていて、エンジンオプションが指定された場合はそれが正しいかを検査して、誤りの場合はエラーを出すようにしている。

[*17] ただし現状では、ドライバオプションが無い場合にはエラーではなく警告が出る。

[*18] 従って、別にドライバ依存の機能を使用しなければ、出力される DVI ファイルは特定の DVI ウェアに依存しないものになる。ただし最近の LaTeX カーネルはそれ自身がドライバ依存をもつことに注意。

[*19] 実際にドライバオプションが有効化された場合は、そのオプションがグローバルオプションとしても働く。例えば、DVI 出力のエンジンで `dvi=dvipdfmx` が指定された場合は、`dvipdfmx` がグローバルオプションに追加される。

- `nopapersize`：出力用紙サイズ設定（DVI 出力の場合は所謂“papersize special 出力”）を抑止する。
  ※ JS クラスの `papersize` オプションの否定に相当する。JS クラスとは異なり、`papersize` は既定で有効である。
  ※出力用紙サイズ設定を行う他のパッケージとの干渉に対する対策。
- `ja=`⟨名前⟩：使用する「和文ドライバ」の名前を指定する。（詳細は 5 節を参照。）標準で提供されている和文ドライバには `minimal`、`standard`、`pandoc` がある。エンジンが `platex` か `uplatex` の時は `ja` の既定値は `standard` である。それ以外の場合は、一応 `minimal` となるが、2.0 版以降ではこの場合の `ja` の省略は**非推奨**であり、3.0 版で**禁止**される予定である。
- `jafont=`⟨名前⟩：「和文フォントプリセット指定」の名前を設定する。既定値は空。
  ※詳細については 5.2 節を参照。
  プリセット指定の値が丸括弧を含む（例えば“`sourcehan(-otc)`”）場合、以下のように解釈される。
  - エンジンが XeLaTeX／LuaLaTeX の場合は、丸括弧囲いの部分を削除した文字列（“`sourcehan`”）が使われる。
  - それ以外の場合は、丸括弧だけ削除した文字列（“`sourcehan-otc`”）が使われる。
- `japaram={`⟨キー⟩`=`⟨値⟩`,...}`：「和文ドライバパラメタ」の値を設定する。既定値は空。
  ※詳細については 6 節を参照。
  ※ `japaram` を複数回指定した場合は各々の指定内容（キーと値の組の列）が累積する。
  ※ `jafont` と `japaram` の値がどのように解釈されるかは和文ドライバの仕様次第である。
- `ja:`⟨キー⟩`=`⟨値⟩：和文ドライバパラメタ設定の代替書式で、`japaram={`⟨キー⟩`=`⟨値⟩`}` と書いたのと同値である。
  ※ Pandoc で波括弧の扱いが面倒になる場合があることへの対策。
- `base=`⟨長さ*⟩：基底フォントサイズ（`\normalsize` のサイズ）を指定する。JS クラスの `10pt`、`11pt` 等と同じ役割で、任意の長さ値を指定できる。基底フォントサイズの既定値は 10 pt（`bxjsslide` クラスのみ 36 pt）である。
  ※ `##pt` の形のオプションには名前と実際に設定される値がずれているものが多く、例えば `11pt` は 10.95 pt、`14pt` は 14.4 pt が実際の設定値である。[*20] これに対して `base=14pt` は文字通り基底フォントサイズに 14 pt を設定する。
- `jbase=`⟨長さ*⟩：和文を基準にして基底フォントサイズを指定する。すなわち、和文フォントの `\normalsize` のサイズを指定の長さとする。[*21]
  ※ 1.8 版から、`base` と `jbase` の両方の指定が可能になった。この場合、和文スケール値がそれらに基づいて決定される（`scale` 指定は無効になる）。
- `scale=`⟨実数⟩：和文スケール値（和文の全角幅÷要求フォントサイズ）を設定する。既定値は 0.924715（= 13 Q/10 pt）である。[*22]
- 用紙サイズオプション：用紙サイズ名のオプションは JS クラスで定義されているもの（4.2 節参照）の他に以下のものが使える。
  - `iso-bsize`：`bNpaper` の名前のオプションを ISO B 判の指定と見なす。既定では `bNpaper` は JIS B 判の指定と見なされる。

[*20] これは昔の LaTeX の“magstep”の習慣に由来する。
[*21] この場合に決定される mag 値は和文スケール値にも依存することに注意。
[*22] これは JS クラスの設計に基づく値である。ただし実装の都合で、JS クラスの実際のスケール値はこれから僅かだけずれている。

– geometry パッケージ（5.x 版）の用紙サイズ名のオプション：以下に挙げるもの[*23]：

a0paper〜a6paper、b0paper〜b6paper、c0paper〜c6paper、b0j〜b6j、ansiapaper〜ansiepaper、letterpaper、executivepaper、legalpaper、screen

※ bNpaper は geometry では ISO B 判を意味するが、BXJS での意味は iso-bsize の有無に従う（つまり既定では JIS B 判）。bNj は geometry と同じく常に JIS B 判である。

– 末尾が 〜paper でない名前のオプションについて、末尾に 〜paper を付けたもの。以下のものが該当する[*24]：

a4varpaper、b5varpaper、screenpaper

※ Pandoc で -V papersize=a4var 等の指定を可能にするため。

– custompaper：実際には何もしないダミーの用紙サイズ指定オプション。

※外部ツールと連携する場合に、「何らかの用紙サイズの名前が要求されるが、用紙サイズは別の方法（paper オプション等）で指定したい」という場合のダミーとして利用する。

- paper={⟨横幅⟩}{⟨縦幅⟩}：長さ値の直接指定による用紙サイズ設定。用紙サイズの既定値は A4 縦（210 mm × 294 mm）である。
- paper=⟨横幅⟩*⟨縦幅⟩：用紙サイズ指定の別形式で、paper={⟨横幅⟩}{⟨縦幅⟩} と等価。
  ※ Pandoc で波括弧の扱いが面倒になる場合があることへの対策。
  ※用紙サイズ指定にはさらに jlreq 互換の別形式 paper={⟨横幅⟩,⟨縦幅⟩} もある。
- enablejfam=⟨値⟩：数式中の和文出力をサポートするか否か。値は true（有効）／false（無効）／default（既定値に従う）の何れかである。エンジンや和文ドライバの種類により、既定値が有効・無効の何れになるかは異なり、また、そもそも有効・無効の一方しか選択できない場合もある。
  ※詳細については 8 節を参照。
- textwidth=⟨長さ⟩：行長（本文領域の横幅；\textwidth）を指定する。
  ※実際には全角の整数倍に丸めた値が使われる（7.1 節参照）。
  – bxjsbook 以外では、geometry で textwidth を指定したのと同値。この場合、既定では textwidth を指定しない。
  – bxjsbook では、geometry の textwidth は実際には \fullwidth（ヘッダ部分の横幅）の指定とみなされる。従って、\textwidth はこのオプションで指定する必要がある。この場合の既定値は 40 zw である。
    ※\textwidth が \fullwidth を超えている場合は \fullwidth と同じと見なされる。
- number-of-lines=⟨整数⟩：1 ページあたりの行数を指定する。geometry で lines を指定したのと同値。既定では geometry の lines は指定されない。

■上級者向けのオプション

- use-zw=⟨真偽値⟩：\jsZw と等価な命令として \zw を定義するか。既定値は真。
  ※つまり \zw の命令名が衝突する場合に、use-zw=false を指定する。
- disguise-js=⟨真偽値⟩：継承元の JS クラス（例えば bxjsbook の場合は jsbook）が読込済であるように振舞うか否か。既定値は真。

[*23] JS クラスにあるオプションも含めている。
[*24] aNj・bNj は aNpaper・bNpaper で代用できるので除いている。

※つまり「JS クラスの一種である」と判定されると不都合な場合に、disguise-js=false を指定する。

- bigcode（既定[*25]）：upTEX エンジンと dvipdfmx の組合せで hyperref パッケージを利用する時に適用される ToUnicode CMap として UTF8-UTF16 を指定する。PDF の文書情報の文字列に BMP 外の文字が含まれる場合にはこの指定が必要である。UTF8-UTF16 のファイルがインストールされていないと、dvipdfmx の処理が失敗する。
- nobigcode：bigcode の否定。ToUnicode CMap として UTF8-UCS2 を指定する。この場合は文書情報の文字列に BMP 外の文字を使用できない。
- precise-text=⟨真偽値⟩：XeTEX エンジンにおいて、「ActualText 生成機能」を有効化する[*26]か否か。既定値は偽。
- simple-ja-setup=⟨真偽値⟩：XeTEX エンジン自体の行組版機能（\XeTeXlinebreaklocale 等）を利用した、簡易的な日本語用組版設定を行うか否か。既定値は真（だが多くの場合に無効化される）。
  ※ XeTEX エンジン以外では無効である。また、xeCJK や zhspacing 等の日本語（CJK）組版用パッケージが読み込まれた場合も無効化される。特に、和文ドライバが standard である場合は、必ず xeCJK が読み込まれるため、このオプションは無意味である。
- plautopatch=⟨真偽値⟩：エンジンが (u)pLaTEX である場合に（十分に早い時点で）plautopatch パッケージを読み込む。既定値は偽。
  ※ただし、Pandoc モードの場合は既定値が真になる。
- mag=⟨整数⟩：版面拡大率（mag 値）の直接設定。既定は基底フォントサイズから算出する。
  ※ mag 値が n の場合、版面が $n/1000$ 倍に拡大される。
- magstyle=⟨値⟩：“版面拡大”の実現方法を指定する。有効な値は usemag、nomag、nomag* の何れか。
  ※詳細については 4.5 節を参照。
- geometry=⟨値⟩：geometry パッケージの読込に対する制御。
  - class（既定）：通常通り、文書クラスが geometry パッケージを読み込む。ユーザは geometry を後から読み込むことはできない。
  - user：文書クラスによる geometry パッケージの読込をスキップする。この場合、ユーザが自分で geometry を読み込むことが想定される。[*27]
    ※「自動読込の際に指定するはずのオプション列」が \jsGeometryOptions に保存されている。

  ※「どうしても geometry パッケージを自分で読み込みたい」という人のための設定。
- oldfontcommands：\bf 等の“二文字フォント命令”の使用を許容する。
- nooldfontcommands（既定）：“二文字フォント命令”の使用に対して警告を出す。
  ※詳細については 9 節を参照。
- fancyhdr=⟨真偽値⟩：fancyhdr パッケージの機能に対する補正を行うか。真の場合、以下の補正が行われる。既定値は真。
  - ヘッダ・フッタ書式の既定値に含まれる“二文字フォント命令”を除去する。
  - bxjsbook クラスでヘッダ・フッタの横幅を（\textwidth ではなく）\fullwidth に一致させる。

[*25] 2.0 版より常に bigcode が既定になる。昔の版では TEX エンジンの版に応じて既定値を変えていた。
[*26] つまり、\XeTeXgenerateactualtext=1 を行う。ActualText 生成機能と日本語処理は相性が悪いため、これを使うと出力 PDF のサイズが増大する（1.5〜2 倍）ことに注意。
[*27] geometry の読込は必須ではなく、ページレイアウトのパラメタを自分で設定しても構わない。ただし geometry の読込が強く推奨される。（geometry 非読込時の動作テストはほとんど行っていない。）

- paragraph-mark=⟨文字 1 つ⟩：パラグラフ（\paragraph）の見出し先頭に付く記号。既定値は“■”。※“paragraph-mark=”のように値を空にするのも可能。
- whole-zw-lines=⟨真偽値⟩：ページレイアウト策定において「行長を全角幅の整数倍に丸める処理」を行うか否か。既定値は真。
- hyperref-enc=⟨真偽値⟩：hyperref パッケージについて「PDF 文字列の文字コード設定を補正する処理」を行うか否か。既定値は真。
- jaspace-cmd=⟨真偽値⟩：以下に挙げる和文空白命令（7.3 節を参照）を定義するか否か。
  \jaenspace、\jathinspace、\>、\␣（\＋全角空白）、\jaspace
  既定値は真。
- fix-at-cmd=⟨真偽値⟩：\@ 命令に対して JS クラスと同様の拡張を施すか否か。既定値は真。
- label-section=⟨値⟩：節番号の書式出力、特に「\pre/postsectionname」「bxjsarticle での付録部における \appendixname」の語句の付加の方法を選択する。
  - compat（既定）：JS クラスと同じ実装を用いる。この場合、節のカウンタの書式（\thesection）そのものに語句が付加されるため、\thesection を参照する他のカウンタ書式が奇妙になる可能性がある。
  - modern：節のカウンタの書式（\thesection）には語句の付加を行わず、実際にそれが節番号として出力される際に語句を付加する。
  - none：節番号に対する語句の付加を抑止する。欧文・和文の標準文書クラスと同等になる。
- layout=⟨値⟩：レイアウトの変種を選択する。現状では、過去の版との互換性を維持するために用いられている。有効な値は以下の通り。
  - bxjsbook クラスの場合：
    - v2（既定）：現版の既定のレイアウト。
    - v1：1.2a 版以前の（本来は不適切な）水平マージンの設定を適用する。（詳細は 7.1 節を参照。）
  - bxjsreport クラスの場合：
    - v2（既定[*28]）：JS クラスに新設された jsreport クラスのレイアウトを継承する。
    - v1：従来の「jsbook クラス＋report オプション」のレイアウトを継承する。
  - それ以外のクラスでは、本オプションは無効である。

■jlreq 文書クラスとの互換用のオプション

- paper={⟨横幅⟩,⟨縦幅⟩}：paper={⟨横幅⟩}{⟨縦幅⟩} と同値（用紙サイズ設定）。
- fontsize=⟨長さ⟩：base=⟨長さ⟩ と同値（基底フォントサイズ）。
- jafontsize=⟨長さ⟩：jbase=⟨長さ⟩ と同値（和文基底フォントサイズ）。
- line_length=⟨長さ⟩：textwidth=⟨長さ⟩ と同値（行長指定）。
- number_of_lines=⟨整数⟩：number-of-lines=⟨整数⟩ と同値（行数指定）。

■旧版との互換用のオプション

2.0 版以降では、これらのオプションの使用は**非推奨**である。

[*28] 1.6 版より既定値が v2 に変更された。

※ (no)zw、(no)js、jadriver、noscale については、3.0 版で**廃止**する予定。

- zw：use-zw=true と同値。
- nozw：use-zw=false と同値。
- js：disguise-js=true と同値。
- nojs：disguise-js=false と同値。
- precisetext：precise-text=true と同値。
- noprecisetext：precise-text=false と同値。
- simplejasetup：simple-ja-setup=true と同値。
- nosimplejasetup：simple-ja-setup=false と同値。
  ※以上の 8 個は 1.9 版より前で使われた。
- textwidth-limit=⟨整数⟩：textwidth=⟨整数⟩zw と同値。
  ※ 1.8 版より前で使われた。
- dvipdfmx-if-dvi：dvi=dvipdfmx と同値。
  ※ 1.2 版より前で使われた。
- magstyle=mag/real/xreal：それぞれ magstyle=usemag/nomag/nomag* と同値。
  ※ 1.1f 版より前で使われた。
- jadriver=⟨名前⟩：ja=⟨名前⟩ と同値（和文ドライバ指定）。
  ※ 1.0 版より前で使われた。
- noscale：scale=1 と同値。
  ※ 0.9 版より前で使われた。

4.2 JS クラスのオプションで使用可能なもの

これらについては名前だけ列挙するに留める。ただし、“JS クラス特有”（標準クラスに無い）オプションの一部については解説を加える。

■用紙サイズ指定 a3paper、a4paper、a5paper、a6paper、b4paper、b5paper、b6paper、a4j、a5j、b4j、b5j、a4var、b5var、letterpaper、legalpaper、executivepaper。
※ a4var は A4 変判（210 mm × 283 mm）、b5var は B5 変判（182 mm × 230 mm）。
※ JS／BXJS クラスでは a4j は a4paper と全く等価である。（他の b4j 等も同様。）

■横置き landscape。

■基底フォントサイズ 8pt、9pt、10pt、11pt、12pt、14pt、17pt、20pt、21pt、25pt、30pt、36pt、43pt、12Q、14Q、10ptj、10.5ptj、11ptj、12ptj。
※ 10pt（10 pt）、11pt（10.95 pt）、12pt（12 pt）、14pt（14.4 pt）、17pt（17.28 pt）、21pt（20.74 pt）、.25pt（24.88 pt）、30pt（29.86 pt）、36pt（35.83 pt）、43pt（43 pt）はそれぞれ magstep の 0、0.5、1、2、3、4、5、6、7、8 に相当し、括弧内が実際の値である。これ以外の 8pt、9pt、20pt は文字通りの値。##Q／##ptj は jbase=##Q／jbase=##pt と等価（つまり和文規準）。[*29]

[*29] ちなみに JS クラスの（固定の）和文スケール値に従うと 10pt が jbase=13Q に相当するので 13Q というオプションは無い。

■両面用レイアウト　oneside、twoside、vartwoside。
※ vartwoside は twoside と同様だが傍注が常に右側余白に出力される。

■段組み　onecolumn、twocolumn。

■表題ページ　titlepage、notitlepage。

■起こし　openright、openleft、openany。
※ jsreport と jsbook にのみ存在するオプション。
※ openleft は部・章の開始を見開き左側のページ（偶数ページ）に強制する（左起こし）。

■数式配置　leqno、fleqn。

■オーバーフル警告　final、draft。

■papersize special 出力　papersize。
※ BXJS クラスでは papersize は既定で有効。

■英語化　english。

■エンジン種別　uplatex、autodetect-engine。
※既に 4.1 節で述べた通り。

■magstyle 指定　usemag、nomag、nomag*。
※ BXJS クラスでは、これらは“magstyle=”を前置したものと同等に扱われる。詳細は 4.5 節を参照。

■和文数式ファミリ不使用　disablejfam。
※ BXJS クラスでは enablejfam=false と同値。詳細は 8 節を参照。

■ロゴ命令パッケージの読込　jslogo、nojslogo。
※ BXJS クラスでは nojslogo が既定値である。

4.3　JS クラスのオプションで使用不可能なもの

- クラス変種指定：report、slide。
  ※ report 相当は bxjsreport、slide 相当は bxjsslide と別クラスになっている。
- トンボ出力：tombow、tombo、mentuke
  ※これは pLaTeX のカーネル命令を利用しているのでとりあえず除外。
- 和文フォントメトリック指定：jis、mingoth。
  ※異なるエンジンで汎用的に扱うのが難しい。

4.4　クラスオプション設定の既定値

- BXJS クラス共通：a4paper、onecolumn、final、その他は各オプションの解説（4.1 節）を参照。
- bxjsarticle：10pt、oneside、notitlepage

- bxjsreport：10pt、oneside、titlepage、openany
- bxjsbook：10pt、twoside、titlepage、openright
- bxjsslide：36pt、oneside、notitlepage

4.5 magstyle オプション

JS クラスにおけるページレイアウト決定の過程では、基底フォントサイズが 10 pt 以外の場合に、「版面を拡大縮小する」という処理を採用している。[*30] これには、「フォントのオプティカルサイズの選択を最適にするため[*31]」という理由があり、またこれにより、多種の基底フォントサイズへの対応が容易になるという利点もある。[*32] ところがここで、JS クラスではこの“版面拡大”を実現するために TEX エンジンが持つ版面拡大機能（仮に「mag 設定」と呼称する）を用いていて、これについて批判されることが多い。また、現実問題として、mag 設定が LATEX で用いられる機会は少ないため、実際に用いられた時にそれを想定していないパッケージが誤動作するという問題もある。

これらの問題を緩和するため、BXJS クラスでは“版面拡大”について他の実現方法を提供している。それを選択するのが以下に挙げる「magstyle オプション」である。[*33]

- usemag：JS クラスと同様に、“版面拡大”のために mag 設定を用いる。
- nomag：mag 設定を一切用いず、代わりに、全てのページレイアウトのパラメタの値をスケールさせる。\normalsize や \large 等の高位フォントサイズ命令で指定されるフォントサイズもスケールさせるが、“オプティカルサイズの調整”は行わない。いわゆる「基本 35 書体」のようなオプティカルサイズでない[*34] フォントのみを用いるのであれば、この設定が最も適切である。
- nomag*：nomag と同様に、全てのページレイアウトのパラメタの値をスケールさせる。さらに、“オプティカルサイズの調整”を実現するために、NFSS の実装コードにパッチを当てる。[*35] この場合、mag 設定による不具合は起こらなくなるが、当然、NFSS のパッチのせいで別の不具合が起こる可能性はある。

※ LuaTEX の 0.87 版以降では「PDF 出力時の mag 設定」の機能が廃止されている。そのため、そのようなエンジンでは usemag はサポートされない（エラーになる）。
※ magstyle オプションの既定値は usemag である。ただし例外として、usemag がサポートされないエンジンでは nomag* が既定値となる。

[*30] 例えば、基底フォントサイズが 20 pt だとすると、まずは指定されたものの半分の縦横幅をもつ用紙に対して基底フォントサイズが 10 pt としてレイアウトを決定し、それを縦横 2 倍に拡大する、という過程をとっている。
[*31] LATEX の既定の欧文フォントである Computer Modern フォントがオプティカルサイズの性質をもつことは有名であるが、少々癖が強くて、本文を 10 pt（cmr10）で組んだ場合と 12 pt（cmr12）で組んだ場合でかなり異なった印象を受ける場合がある。JS クラスではそれを嫌って、本文（\normalsize のフォント）が必ず「cmr10 を拡大縮小したもの」で組まれることを企図しているのである。
[*32] BXJS で「任意の」基底フォントサイズが設定できるのもこの利点があるため。
[*33]「magstyle オプション」の値は、magstyle をキー名にした keyval 形式（例えば magstyle=nomag*）で書くこともできる。
[*34] 或いは、オプティカルサイズに“変な癖”のない。
[*35] 要するに、OT1/cmr/m/n/12 が要求された時に、cmr12 でなくて cmr10 at 12pt が選ばれるようにする。

5　和文ドライバ

BXJS クラスでは様々なエンジンについて、そのエンジンおよびそれに対応するパッケージが提供する日本語処理機能を活用することで、日本語用の文書クラスとしての機能を実現している。そこでの汎用性を確保するため、“日本語処理機能と連携する部分”の実装をモジュールとして分離していて、これを**和文ドライバ**と呼ぶ。[*36]BXjscls のバンドルでは以下に挙げる和文ドライバを提供している。

- standard 和文ドライバ：各エンジンについて、最も一般的に用いられる特定の“日本語処理機能”（例えば `lualatex` なら LuaTEX-ja）を連携対象とした和文ドライバ。(u)pLATEX 上の JS クラスと同じくらい容易に日本語が書き始められることを目指している。
- minimal 和文ドライバ：“何も実装されていない”和文ドライバ。上級ユーザがプレアンブルや自作パッケージ等にアドホックな連携コードを書いて、好きな“日本語処理機能”との連携を実現するために用いることを想定している。
- pandoc 和文ドライバ：「Pandoc モード」で使用される和文ドライバ。standard ドライバの機能を継承するが、「Pandoc の既定の `latex` テンプレート」が使われることを前提として、それと BXJS の設定を整合させるための措置を加えている。
  ※本マニュアルで「standard 和文ドライバの場合」という場合、特に明示されない限りは pandoc 和文ドライバも含まれる。

和文ドライバは自分で作製することも可能である。[*37]`bxjsja-XXX.def`（`XXX` は任意の文字列[*38]）の名前のファイルに実装コードを書いてそのファイルを配置すると、`ja=XXX` のオプション指定でその和文ドライバを利用できる。

なお、和文ドライバの指定（`ja` オプション）は**必須**である。現状では `minimal` が既定値となっているが、2.0 版より省略は**非推奨**の扱いであり、3.0 版で**禁止**される予定である。
※ただし、(u)pLATEX については、日本語処理機能がエンジン自体に備わっていて不可分なため例外的な扱いになっていて、[*39]和文ドライバ指定は省略可能で `standard` が既定値になっている。

5.1　standard ドライバで用いられる日本語処理機構

- pLATEX、upLATEX（および pLATEX-ng）の場合：エンジン自体が日本語処理の能力を持っているのでそれが常に用いられる。
- (pdf)LATEX の場合：bxcjkjatype パッケージが `whole` と `autotilde` のオプション付で読み込まれる。
- XeLATEX の場合：zxjatype パッケージが読み込まれる。
- LuaLATEX の場合：luatexja パッケージが読み込まれる。

[*36] graphicx パッケージ等の「ドライバ」と類似した概念のためこの名称を用いた。
[*37] 和文ドライバの実装に必要な連携仕様の情報については、ソースコード説明書（bxjscls.pdf）の付録 A を参照してほしい。
[*38] LATEX の非特殊文字（TEX 言語でいうとカテゴリコードが 11 または 12 の文字）からなる必要がある。
[*39] JS クラスの実装から分離した「日本語処理関連」のコードを minimal に配置している。

5.2 和文フォント設定

和文フォントプリセット（jafont オプション）が指定されない場合は、JS クラスと同様の「明朝・ゴシックで各 1 ウェイトのみを用いて、明朝の太字がゴシックになる」という設定が適用される。

明朝・ゴシックのファミリとして用いられる物理フォントの割当（マッピング）は以下のようになる：

- pLaTeX、upLaTeX の場合：クラスでは何もマッピングを設定しない。従って、DVI ウェアでの設定が適用される。
- pdfLaTeX の場合：bxcjkjatype パッケージの既定設定となり、従って、Type1 形式の IPAex フォント（ipaex-type1 で提供される）が使用される。
- XeLaTeX の場合：「原ノ味フォント」が使用される。zxjafont パッケージの haranoaji プリセットの単ウェイト使用と同等であるが、パッケージが読み込まれるわけではない。
- LuaLaTeX の場合：「原ノ味フォント」が使用される。luatexja-preset パッケージの haranoaji プリセットの単ウェイト使用と同等であるが、パッケージが読み込まれるわけではない。

※ DVI ウェアにおける既定の和文フォントマッピング（すなわち (u)pLaTeX における既定の和文フォント）の設定は、昔は「IPAex フォント」であったが TeX Live 2020 において「原ノ味フォント」に変更された。これに追随する形で、BXJS クラスの XeLaTeX・LuaLaTeX での既定設定についても、2.0 版より「原ノ味フォント」に変更された。ただし Type1 形式の原ノ味フォントは存在しないため、(pdf)LaTeX では ipaex-type1 のフォントが引き続き使用される。

和文フォントプリセット（jafont）を指定した場合、具体的には以下のように処理される：

- pLaTeX、upLaTeX の場合：「jafont の値」をオプションにして pxchfon パッケージを読み込む。
- (pdf)LaTeX の場合：bxcjkjatype のオプションに「jafont の値」を追加する。
- XeLaTeX の場合：「jafont の値」をオプションにして zxjafont パッケージを読み込む。
- LuaLaTeX の場合：「jafont の値」をオプションにして luatexja-preset パッケージを読み込む。

ただし、jafont の値として auto は特別な意味をもち、「TeX Live の updmap-kanji-config で指定した和文フォント設定を全エンジンに[*40]適用する」という動作を行う。具体的には以下のように処理される[*41]：

- pLaTeX、upLaTeX の場合：何もマッピングを設定しない。（実際に updmap-kanji-config の設定を活かせばよいため。）
- それ以外の場合：「updmap-kanji-config で指定されたファミリ名」が jafont の値に指定されたと見なされる。[*42]

[*40] updmap-kanji-config は DVI ウェアのフォントマップ設定を行うものであるため、そこでの設定は本来は (u)pLaTeX にしか影響しない。

[*41] 当然であるが、jafont=auto の指定は TeX Live 以外の TeX システムではサポートされず、また敢えてその指定を行った場合の動作は未定義である。

[*42] updmap-kanji-config の --jis2004 オプションにより 2004JIS 字形が選択されている場合は、さらに jis2004 和文パラメタを有効にする。

6　和文ドライバパラメタ

和文ドライバパラメタ（japaram オプション）は和文ドライバに依存する設定を指定するために用意されている。japaram オプションの値はそれ自身がキー値ペアのリストであり、一般的に次のような形式で指定される。

```
\usepackage[...,japaram={key=value,...},...]{bxjsarticle}
```

minimal 和文ドライバには有効な和文ドライバパラメタは存在しない。

■和文ドライバパラメタの代替表記　Pandoc（の LaTeX 出力の既定テンプレート）では、classoption というパラメタで任意のクラスオプションを指定できるが、この際に波括弧 {…} の扱いがしばしば厄介なことになる。例えば和文パラメタ指定の

```
japaram={jis2004}
```

という値を YAML メタデータブロックで指定したいとする。この場合、単純に

```
classoption: japaram={jis2004}
```

と書いたのでは、{jis2004} の部分が \{jis2004\} のように“エスケープ”されてしまって失敗する。所望の指定をするには代わりに

```
classoption: "`japaram={jis2004}`{=latex}"
```

のような複雑怪奇な書き方をする必要がある。[*43]

この問題に対処するため、2.9 版において「和文ドライバパラメタ代替表記」をサポートすることにした。クラスオプションに ja: で始まる文字列を書くと、以下のように解釈される。

- ja:XXX と書くと、japaram={XXX} を指定したことになる。
- ja:XXX=YYY と書くと、japaram={XXX=YYY} を指定したことになる。

代替表記を利用すると、先ほどの例は以下のように書くことができる。

```
classoption: ja:jis2004
```

6.1　standard 和文ドライバの場合

以下の和文ドライバパラメタが存在する。

- jis2004=⟨真偽値⟩：漢字の字形について「2004JIS 字形を優先させる」処理を行うか。真の場合、以下の処理が行われる。既定値は偽。

[*43] YAML メタデータブロックで指定された文字列（YAML の文字列型の値）について、Pandoc では「Markdown ソースとして解釈される」という奇妙な仕様を（歴史的経緯のため）採用している。このため、{jis2004} は「Markdown → LaTeX の変換」が適用されて \{jis2004\} となってしまう。余計な変換を回避するためには Pandoc Markdown の RawInLine 要素を利用して `japaram={jis2004}`{=latex} と書く必要がある。さらに、YAML の文法規則として、文字列が ` で始まる場合には "…" で全体を囲わなければならない。

  - グローバルオプションに jis2004 を追加する。[*44]
  - エンジンが (u)pLaTeX の場合、pxchfon と pxbabel パッケージに予め prefer2004jis オプションが渡される。
- units=⟨真偽値⟩：これを真にすると、LaTeX における長さ指定において、pTeX の和文用の単位（zw、zh、(true)Q、(true)H）を全てのエンジンで使用できるようにする。既定値は偽。
  ※ bxcalc パッケージを読み込む。
  ※厳密にいうと、「calc の長さ数式が使える箇所」において和文用の単位が使用できるようになる。
- font={⟨文字列⟩}：standard 和文ドライバが利用する和文フォントパッケージについて追加のオプションを指定する。
- strong-cmd=⟨真偽値⟩：fontspec パッケージと互換の \strong 命令と strongenv 環境を定義するか。既定値は真。

6.2 pandoc 和文ドライバの場合

standard 用のものに加えて、以下の和文ドライバパラメタが存在する。

- strong=⟨値⟩：\strong 命令・strongenv 環境で実際に適用される書体（フォント変更命令）を選択する。
  - bold（既定）：太字（\bfseries）。
  - sans：サンセリフ（\sffamily）。
  - boldsans：太字サンセリフ（\sffamily\bfseries）。

  ※ Pandoc モードの既定では Pandoc の重要（Strong）要素は \strong 命令で実現されるので、このパラメタの設定に追随する。
- fix-strong=⟨真偽値⟩：\textbf 命令の動作を \strong と同等にするか。既定値は真。
  ※ Pandoc の重要（Strong）要素は LaTeX への変換において \textbf 命令が適用される。和文フォントを単ウェイトで用いている場合、生の \textbf は体裁が悪い（ゴシックの和文とセリフの欧文の組み合わせになる）ため、\strong に移譲して調整ができるようにしている。[*45]
  ※\bfseries 命令の動作は変わらない。
- fix-code=⟨真偽値⟩：\texttt 命令と周囲の和文の間に和欧文間空白が常に入るようにするか。既定値は真。
  ※ Pandoc のインラインコード（Code）要素は LaTeX への変換において \texttt 命令が適用される。
  ※真である場合、\texttt の出力の周囲に欧文ゴーストが挿入される。

7 ユーザ用命令

原則として、BXJS クラスで追加されたものだけを説明する。

[*44] japanese-otf や luatexja-preset などのパッケージが jis2004 オプションを利用する。
[*45] \strong に適用される書体は strong パラメタで選択できるが、\strongfontdeclare 命令（7.2 節）でも設定できる。

7.1 レイアウト設定関連

BXJS クラスではページレイアウトの設定に geometry パッケージを用いて次の手順で行っている。

1. 基底フォントサイズにより決定された mag 値を実際に設定する。
2. geometry で次のパラメタを設定する。
   (a) クラスオプションで指定された用紙サイズ、および `truedimen` とドライバ指定。
   (b) bxjsarticle／bxjsreport の場合は次のパラメタ値。
   ```
   headheight=10pt, footskip=0.03367\paperheight,
   headsep=\footskip-\topskip, includeheadfoot,
   hscale=0.76, hmarginratio=1:1, vscale=0.83, vmarginratio=1:1
   ```
   (c) bxjsbook の場合は次のパラメタ値。
   ```
   headheight=10pt, headsep=6mm, nofoot, includeheadfoot,
   hmargin=18mm, vscale=0.83, vmarginratio=1:1
   ```
   ※ bxjsbook の 1.2a 版以前では、この設定の中の“`hmargin=18mm`”の代わりに“`hmargin=36mm, hmarginratio=1:1`”を用いていた。これでは jsbook の水平マージン設定と同等にならないため 1.3 版で現在の設定に修正された。もし 1.2a 版以前との互換性を保ちたい場合は、クラスオプションに `layout=v1` を指定してほしい。
   (d) bxjsslide の場合は次のパラメタ値。
   ```
   noheadfoot, hscale=0.9, hmarginratio=1:1,
   vscale=0.944, vmarginratio=1:1
   ```
3. 後処理を行う。以下の処理が含まれる。
   - `textwidth` を全角幅の整数倍に、`textheight` を整数行分の自然長になるように丸める。
   - `marginpar` 関連の設定を行う。

   ※ただし、横幅の全角整数倍への丸めは、`whole-zw-lines=false` 指定時には抑止される。

ページレイアウトの再設定のために次の命令が用意されている。

- `\setpagelayout{`⟨設定⟩`}`：現在のページレイアウトの設定の一部を修正する。⟨設定⟩ は geometry のパラメタの記述であり、現在の設定に追記して geometry が再設定を行った後、再び 3 の後処理が行われる。
- `\setpagelayout*{`⟨設定⟩`}`：用紙以外の設定をリセットして改めてページレイアウトの設定を行う。具体的には、まず geometry の `reset` オプションで初期化し、その後 2a と ⟨設定⟩ の内容を用いて再設定を行った後、再び 3 の後処理が行われる。
- `\setpagelayout+{`⟨設定⟩`}`：`\setpagelayout*` の変種で、「本文領域のサイズと位置」の設定のみをリセットして改めてページレイアウトの設定を行う。すなわち、前項の“再設定”の際に、2b〜2d の内容のうち「本文領域」以外のものが追加される。

なお、`\geometry` 命令を直接呼び出すことも可能である。当然この場合は 3 の後処理は行われない。

7.2 構造マークアップ関連

- `\subtitle{`⟨テキスト⟩`}`：サブタイトルを設定する。

※\maketitle の出力にサブタイトルが含まれるようになる。

- chapterabstract 環境：jsbook クラスの abstract 環境[*46]と等価な環境で、「各章の初めにちょっとしたことを書く」用途を想定したもの。（ただし使用可能な場所に特に制限はない。）
  なお、abstract 環境の機能については継承元の JS クラスと同一になるため以下のようになる。
  - bxjsarticle および layout=v2 指定の bxjsreport：jsarticle／jsreport クラスと同じ、つまり「文書要旨」のための環境。
  - bxjsbook および layout=v1 指定の bxjsreport：jsbook クラスと同じ、つまり chapterabstract 環境と等価。
- \Seireki ／\Wareki：**\西暦**／**\和暦** と等価な英字名の命令。
- \jayear：和暦表示（**\和暦**）が有効な時の \today のテキスト中の年号（「年」より前の部分、例えば「平成 28」）の部分を表すマクロ。

label-section=modern の場合に限り、以下の命令が利用できる。

- \labelsection：節見出し（\section）における節番号の表示形式を表すマクロ。通常、\thesection（section カウンタの出力形式）に必要な装飾（“節”や“§”など）を加えたテキストを指定する。
  既定値は以下に示すものと同等になっている。
  - 文書開始時は“\presectionname\thesection\postsectionname”。
  - 付録開始時（\appendix 実行時）に“\appendixname\thesection”に切り替わる。[*47]

  ※ label-section=compat の場合は（JS クラスと同様に）付録開始時に「\presectionname を \appendixname で上書きする」という動作が発生するが、label-section=modern ではこれは起こらない。
- \labelsubsection／\labelsubsubsection： これらのマクロが定義されている場合は、各々の内容のテキストが小節見出しおよび小々節見出しの表示形式として利用される。未定義の場合は \thesub(sub)section で代用される。

7.2.1 standard 和文ドライバで追加される命令

- \strong{⟨テキスト⟩}：引数のテキストに、重要性を表す書体変更を適用する。
- strongenv 環境：\strong の環境版。
- \strongfontdeclare{⟨書体変更命令列⟩,...}：\strong で実際に適用される書体変更（宣言型命令の列）を指定する。コンマ区切りで複数の値が指定可能で、n 重の \strong が適用されたテキストに対して n 番目の書体変更命令列が実行される。
  ※既定値は“\bfseries”であり、\strong は太字で出力される。
  ※以上の 3 つの機能は fontspec パッケージとの互換を目的としたものである。strong-cmd=false 指定時はこれらの機能は無効になる。ただし、fontspec が読み込まれている場合は、常にそちらの定義が維持される。

[*46] article 系や report 系のクラスにあるような（文書要旨のための）abstract 環境は、book 系クラスでは用意されていないことが多いことに注意。

[*47] \labelsection を実際に再定義するわけではないので、ユーザが独自の \labelsection を設定している場合は切替は発生しなくなる。

7.3 和文用設定関連

- \jsZw：和文の全角幅を表す長さ命令。例えば 2\jsZw が pLaTeX の 2zw に相当する。
- \zw：\jsZw の別名。[*48]ただし use-zw=false 指定時は定義されない。
- \zwspace：全角（\jsZw）幅の水平空き。
- \␣（\＋全角空白）：\zwspace の別名。全角幅の水平空き。
  ※ jlreq クラスと互換の命令。ただし BXJS クラスは JLREQ の和字間隔の規定とは無関係であり、この命令は単純に \hspace{\jsZw} と等価である。
- \jaenspace：半角（0.5 \jsZw）幅の水平空き。
- \jathinspace：和欧文間空白を挿入する。
- \>：（非数式モードで[*49]）和欧文間空白を挿入する。
- \jaspace{⟨値⟩}：引数の値に応じて水平空きを挿入する。[*50]可能な値は以下の通り：
  - zenkaku：全角幅。（\zwspace と同じ。）
  - nibu：半角幅。（\jaenspace と同じ。）
  - shibu：四分幅（0.25 \jsZw）。[*51]

  ※ただし jaspace-cmd=false 指定時は \␣・\jaenspace・\jathinspace・\>・\jaspace は定義されない。[*52]
- \allowoldfontcommands：これ以降に実行される二文字フォント命令を警告の対象にしない。
- \disallowoldfontcommands：これ以降に実行される二文字フォント命令を警告の対象にする。
  ※詳細については 9 節を参照。
- \jQ、\jH、\trueQ、\trueH：それぞれ pLaTeX の単位 Q、H、true Q、true H に相当する長さ命令。
- \ascQ：1 true Q を和文スケール値で割った長さを表す長さ命令。[*53]
  ※例えば、\fontsize{10\ascQ}{16\trueH} で和文のサイズが 10 Q になる。[*54]
- \ascpt：1 true pt を和文スケール値で割った長さを表す長さ命令。
  ※例えば、\fontsize{9\ascpt}{13truept} で和文のサイズが 9 ポイントになる。
- pLaTeX カーネルで規定される「基底フォントサイズパラメタ」の長さ命令 \Cwd／\Cht／\Cdp／\Cvs／\Chs が全てのエンジンで利用できる。また「和文スケールの値を実数値マクロ \Cjascale に設定する」という規約にも従っている。

[*48] LuaTeX-ja では「実際の全角幅」を表す長さ命令 \zw（pLaTeX の zw と本当に等価）が規定されている。lualatex エンジン指定かつ和文ドライバが standard の場合はこの \zw の定義がそのまま使われる。（従って use-zw は実質的に無意味である。）なお、\jsZw は「規約上の全角幅」であり、「実際の全角幅」と本来は一致するはずだが、実際には計算誤差のせいで僅かに値が異なる。

[*49] \> という命令名は、plain TeX では数式中の空白（LaTeX の \: と同等）を表す。このため、実際には LaTeX でも \> は \: と同等の命令として定義されている。この「plain 互換の \>」を利用したコードが影響を受けるのを避けるため、数式中では \> は従来通り \: と同等の動作を行う。

[*50] jlreq クラスに同様の命令が実装されているが、なぜか undocumented になっている。

[*51] 和欧文間空白（\jathinspace）とは必ずしも一致**しない**ことに注意。

[*52] ただし \> については注意が必要で、standard 和文ドライバで自動的に読み込まれる zxjatype や bxcjkjatype のパッケージは、それ自体が同様の機能の \> を提供する。

[*53] 命令名は“anti-scaled Q”の略。

[*54] 1.5c 版で導入された \jafontsize 命令を用いて \jafontsize{10trueQ}{16trueH} としてもよい。

7.3.1 standard 和文ドライバで追加される命令

standard 和文ドライバでは和文に関連する文書ソース記述をエンジンに依らずに共通になることを目指している。従って、和文関連の組版パラメタの設定についても「共通の命令」が提供される。

- 和文ファミリ変更命令：pLaTeX と同様に、\mcfamily で「明朝」、\gtfamily で「ゴシック」に変更される。\textmc、\textgt も使える。
- 欧文ファミリ変更命令での和文の連動：JS クラスと同様[*55]に、\rmfamily で和文が「明朝」、\sffamily および \ttfamily で和文が「ゴシック」に変更される。
- \setxkanjiskip{⟨長さ⟩}：和欧文間空白の量を指定する。pLaTeX での \setlength{\xkanjiskip}{⟨長さ⟩} に相当する。
- \getxkanjiskip：現在の和欧文間空白の量を表す文字列に展開される。pLaTeX での \xkanjiskip の読出[*56]に相当する。
- \autoxspacing／\noautoxspacing：和欧文間空白の挿入を有効／無効にする。pLaTeX の同名の命令と同等。
- \setkanjiskip{⟨長さ⟩}：和文間空白の量を指定する。pLaTeX での \setlength{\kanjiskip}{⟨長さ⟩} に相当する。
- \getkanjiskip：現在の和文間空白の量を表す文字列に展開される。pLaTeX での \kanjiskip の読出[*57]に相当する。
- \autospacing／\noautospacing：和文間空白の挿入を有効／無効にする。pLaTeX の同名の命令と同等。
- \jachar{⟨文字 1 つ⟩}：指定の文字を和文文字として（現在の和文フォントで）出力する。
- \jafontsize{⟨フォントサイズ*⟩}{⟨行送り*⟩}：“和文規準”でフォントサイズを指定する。すなわち、和文の 1 zw が ⟨フォントサイズ⟩ と等しくなるようにフォントサイズを設定する。
- 和文数式フォント命令：JS クラスと同様に、\mathmc で「明朝」、\mathgt で「ゴシック」の和文数式フォントを指定する。
- 欧文数式フォント命令での和文の連動：\mathrm で和文が「明朝」、\mathsf および \mathtt で和文が「ゴシック」に指定される。[*58]
  ※ JS クラスとは異なり、連動の組合せはテキストと同一であることに注意。

例えば、pLaTeX において、次のように「和文間空白」を利用して均等割りを行うという技法が知られている。

```
%% \kintouwari{<整数 n>}{<テキスト>}
% n 全角の幅にテキストを均等割りで出力する.
\newcommand\kintouwari[2]{{%
  \setlength{\kanjiskip}{\fill}%
```

[*55] ちなみに、(u)pLaTeX の既定ではこの連動は起こらない。
[*56] TeX 言語でいうと \the\xkanjiskip。
[*57] TeX 言語でいうと \the\kanjiskip。
[*58] 和文連動の対象となる欧文数式フォント命令を bm パッケージの \bm 命令の中で使う（例えば \bm{\mathsf{A}} 等）と正常に動作しないことが判明している。暫定対処として、2.9b 版以降では bm パッケージが読み込まれている場合には欧文数式フォント命令の和文連動が無効になる。

```
\makebox[#1zw][s]{#2}}
```

これと同等のものを、次のようにエンジン非依存な形で書くことができる。

```
\newcommand\kintouwari[2]{{%
  \setkanjiskip{\fill}%
  \makebox[#1\zw][s]{#2}}}
```

7.3.2 pandoc 和文ドライバで追加される命令

Pandoc モードにおける出力の調整の対象となる要素について、以下の命令を再定義することで調整方法をカスタマイズできる。例えば

```
\renewcommand{\pandocZWSpace}{\jachar{　}}
```

（ここで 「　」 は全角空白文字）と再定義することで、「全角空白文字の入力を空きに変換する」調整を実質的に無効化できる。

- \pandocZWSpace：全角空白文字が入力されたときに実際に実行される命令。
  ※\pandocZWSpace を再定義する際に、その定義内容の中で全角空白文字を直接使うと無限ループになってしまうため、代わりに \jachar{　} と書く必要がある。
- \pandocLdots：非数式で \ldots を実行したときに実際に実行される命令。
  ※一般に Pandoc の LaTeX 出力においては入力文書中の “…” が \ldots 命令に変換される。
  ※\pandocLdots を再定義する際に、その定義内容の中で \ldots を使うと無限ループになってしまうため、代わりに \textellipsis 等を適宜使う必要がある。ただし特別な規約として
  ```
  \renewcommand{\pandocLdots}{\ldots}
  ```
  と再定義した場合は、\ldots に対する調整自体が無効になる。

8 数式中の和文出力について

minimal 和文ドライバは数式中の和文出力の機能を何も提供しない。従って、そのような機能を提供する他のパッケージを併用するのでない限り、数式中に和文を書いたときの挙動は未定義である。

standard 和文ドライバ（およびそれを継承する和文ドライバ）における数式中の和文出力の扱いは、エンジンと enablejfam オプションの値の組合せにより異なり、表 1 に示すようになる。以下でこの表に関する補足説明を行う。

- この表にある以外のエンジンと enablejfam 値の組合せは許容されない。この場合、警告が出て、enablejfam が可能な値に自動的に変更される。
- 「直書き」が「可」の場合、数式フォント命令（\mathXX{}）の外に書いた和文文字は明朝体で出力される。「不可」の場合、そのような和文文字の扱いは未定義である。
- 「\mathmc/gt」が「サポート有り」の場合、これらの命令は “本物” の数式フォント命令として働く。「フォールバック」の場合は、これらの命令は内部で一旦テキストモードに切り替えて非数式として出力される。このフォールバック機能を実用したい場合は、amstext（または amsmath）パッケージの併

表 1　standard 和文ドライバにおける数式中の和文出力のサポート

エンジン	enablejfam	直書き	\mathmc/gt	和欧文連動
(u)platex	true（既定）	可	サポート有り	有り
	false	不可	フォールバック	—
lualatex	true	可	サポート有り	有り
xelatex	true	可	フォールバック	無し
	false（既定）	不可	フォールバック	—
pdflatex	false	不可	フォールバック	—

用が望ましい。[*59]

9　“二文字フォント命令”に対する警告

ここでいう“二文字フォント命令”というのは、\bf や \it 等の LaTeX 2.09 で標準であったフォント選択命令のことである。[*60] LaTeX 2_ε においては、これらに代わって、\bfseries 等の NFSS 方式の新しい命令群が標準となり、古い二文字フォント命令はカーネルではもはサポートされなくなった。しかし同時に、二文字フォント命令を利用したパッケージを動作させるための“当面の”[*61]互換性対策として、「標準の文書クラス（article、book 等）で二文字フォント命令のサポートを行う」という方針がとられた。これに倣って、他の文書クラスの多くもクラスのレベルで二文字フォント命令をサポートしていて、BXJS クラスもその例に含まれる。

ところが最近になって、一部の文書クラス（KOMA-Script クラス群や memoir クラス等）において、二文字フォント命令を明示的に非推奨の扱いにした上で、その使用に制限を設ける（警告を出す、オプションを指定しないと使えない、等）という措置が取られるようになっている。

これに合わせて、BXJS クラスでは 1.2 版より二文字フォント命令を非推奨とし、また、既定でその使用に対して警告を出すようにした。

9.1　警告の内容

文書中で \bf などの二文字フォント命令が呼び出された場合、コンパイルの最後に（一度だけ）以下の警告メッセージが表示される。

```
Class bxjsarticle Warning: Some old font commands were used in text:
(bxjsarticle)                \bf \it
(bxjsarticle)              You should note, that since 1994 LaTeX2e provides a
(bxjsarticle)              new font selection scheme called NFSS2 with several
(bxjsarticle)              new, combinable font commands. The class provides
(bxjsarticle)              the old font commands only for compatibility.
```

[*59] amstext を読み込まない場合、添字中で \mathmc/gt を用いたときに文字サイズが非添字のものに戻ってしまうという不具合が出る。

[*60] なお、\em は「二文字の名前のフォント命令」であるが、これは LaTeX 2_ε でも標準命令であり、“二文字フォント命令”には含まれない。

[*61] ちなみに、LaTeX 2_ε が最初にリリースされたのは 1994 年のことである。

なお、この警告は、パッケージの機能の実装として用いられたものも含めて全ての二文字フォント命令の呼出が対象になる。ただし例外として、thebibliography 環境の内部で呼び出されたものだけは対象から除外される。BibTeX の文献スタイルファイル（.bst）や文献データベース（.bib）のファイルは（パッケージと比較しても）極めて古いものが割と普通に使い続けられることが多い。そういった極めて古いファイルに由来する二文字フォント命令を警告したとしても、多くの場合、ユーザ側には対処する方法が存在しない。これが文献リスト環境中で警告を抑止する理由である。

9.2 警告の制御

二文字フォント命令に対する警告の有無はクラスオプションで制御できる。

- oldfontcommands：二文字フォント命令を警告の対象にしない。
- nooldfontcommands（既定）：二文字フォント命令を警告の対象にする。

また、以下の命令により、文書中で一時的に警告の設定を変えられる。

- \alloweldfontcommands：これ以降に実行される二文字フォント命令を警告の対象にしない。
- \disalloweldfontcommands：これ以降に実行される二文字フォント命令を警告の対象にする。

9.3 将来の二文字フォント命令の扱い

現在の版で存在する和文ドライバを使用する場合には、将来にわたって以下の方針が維持される。

- 二文字フォント命令に対する警告の様式は、今後変更される可能性がある。
- しかし、将来に二文字フォント命令のサポートが廃止されることはない。
- oldfontcommands オプションおよび \alloweldfontcommands 命令は継続して提供され、これらの機能を用いた場合は、二文字フォント命令に関する警告が端末に表示されることは一切無い。