BXjscls パッケージ
(BXJS 文書クラス集)
ソースコード説明書

八登崇之 (Takayuki YATO; aka. "ZR"

v2.9b  [2024/01/22]

この文書はソースコード説明書です。一般の文書作成者向けの解説については、ユーザマニュアル bxjscls-manual.pdf を参照してください。

目次

1  はじめに  4
2  オプション  13
3  和文フォントの変更  44
4  フォントサイズ  54
5  レイアウト  60
  5.1 ページレイアウト  61
6  改ページ（日本語 TeX 開発コミュニティ版のみ）  76
7  ページスタイル  77
8  文書のマークアップ  80
  8.1 表題  80
  8.2 章・節  86
  8.3 リスト環境  98
  8.4 パラメータの設定  105
  8.5 フロート  107
  8.6 キャプション  108
9  フォントコマンド  109
付録 F 補助パッケージ一覧 ☃

付録 G 補助パッケージ：bxjscompat ☃
G.1 準備 ................................................. 186
G.2 8bit 欧文 TEX ........................................ 187
G.3 XeTeX .............................................. 187
G.4 LuaTeX ............................................. 188
G.5 完了 .............................................. 189

付録 H 補助パッケージ：bxjcjkcat ☃
H.1 準備 ................................................. 190
H.2 和文カテゴリコードの設定 ................................... 191
H.3 ギリシャ・キリル文字の扱い ................................... 191
H.4 初期設定 ........................................... 199
H.5 完了 .............................................. 199

付録 I 補助パッケージ：bxjspandoc ☃
I.1 準備 ................................................. 199
I.2 パッケージオプション .................................... 200
I.3 パッケージ読込の阻止 .................................... 200
I.4 fixltx2e パッケージ .................................... 200
I.5 cmap パッケージ ...................................... 201
I.6 microtype パッケージ ................................... 201
I.7 Unicode 文字変換対策 ................................... 201
I.8 PandLa モジュール .................................... 202
I.9 完了 .............................................. 203
１ はじめに

この文書は「BXJS ドキュメントクラス」の DocStrip 形式のソースである。BXJS ドキュメントクラス（以降では「BXJS クラス」と略称する）は奥村晴彦氏および日本語 TeX 開発コミュニティによる「pLaTeX 新ドキュメントクラス」（以降では「JS クラス」と呼ぶ）に改変を加えたものである。

BXJS クラスに関する解説と原版著者による原版に対する解説を区別するために以下の規則を設ける。

- 見出しに "☃" 印が付いている節・小節・段落の記述は BXJS クラスのものである。
- この形式の枠の中の記述は BXJS クラスのものである。

インストール時のモジュール指定は以下のものが用意されている。

<table>
<thead>
<tr>
<th>モジュール名</th>
<th>ファイル名</th>
<th>説明</th>
</tr>
</thead>
<tbody>
<tr>
<td>article</td>
<td>bxjsarticle.cls</td>
<td>短いレポート（章なし）のクラス</td>
</tr>
<tr>
<td>report</td>
<td>bxjsreport.cls</td>
<td>長いレポート（章あり）のクラス</td>
</tr>
<tr>
<td>book</td>
<td>bxjsbook.cls</td>
<td>書籍用のクラス</td>
</tr>
<tr>
<td>slide</td>
<td>bxjsslide.cls</td>
<td>スライド用のクラス</td>
</tr>
<tr>
<td>minimal</td>
<td>bxjsja-minimal.def</td>
<td>minimal 和文ドライバ</td>
</tr>
<tr>
<td>standard</td>
<td>bxjsja-standard.def</td>
<td>standard 和文ドライバ</td>
</tr>
<tr>
<td>modern</td>
<td>bxjsja-modern.def</td>
<td>modern 和文ドライバ（未公開）</td>
</tr>
<tr>
<td>pandoc</td>
<td>bxjsja-pandoc.def</td>
<td>pandoc 和文ドライバ</td>
</tr>
<tr>
<td>compat</td>
<td>bxjscompat.sty</td>
<td>古いやつをどうにかする補助パッケージ</td>
</tr>
<tr>
<td>cjkcator</td>
<td>bxjsjkcator.sty</td>
<td>modern ドライバ用の補助パッケージ</td>
</tr>
<tr>
<td>ancpandoc</td>
<td>bxjspandoc.sty</td>
<td>Pandoc 用の補助パッケージ</td>
</tr>
</tbody>
</table>

※このソースには jsclasses.dtx との差分を抑制するために "jspf"・"kiyou"・"minijs" のモジュール指定を残しているが、これらの指定が行われることは想定していない。

これは lаLaTeX3 Project の classes.dtx と株式会社アスキーの jclasses.dtx に基づいてもと奥村晴彦により作成されたものです。現在は日本語 TeX 開発コミュニティにより GitHub で管理されています。

https://github.com/texjporg/jsclasses

[2002-12-19] いろいろなものに収録していただく際にライセンスを明確にする必要が生じてきました。アスキーのものが最新は modified BSD ライセンスになっていますので、私のものもそれに準じて modified BSD とすることにします。

[2016-07-13] 日本語 TeX 開発コミュニティによる管理に移行しました。

[2009-02-22] 田中琢爾氏による uplаLaTeX 対応パッチを取り込みました。
ここでは次のドキュメントクラス（スタイルファイル）を作ります。
2017-02-13 forum:2121 の議論を機に，jsreport クラスを新設しました。従来の jsbook の report オプションと比べると，abstract 環境の使い方および挙動がアスキーの jreport に近づきました。

以下では実際のコードに即して説明します。minijs は，jsclasses に似た設定を行うパッケージです。

■環境検査

\jsDocClass 〔トークン〕文書クラスの種別。以下の定値トークンの何れかと同値： \jsArticle = bxjsarticle, \jsBook = bxjsbook, \jsReport = bxjsreport, \jsSlide = bxjsslide。
制御綴の意味が同名のプリミティブである場合にのみ＜コード＞を実行する。

\def\bxjs@test@engine#1#2{\edef\bxjs@tmpa{\string#1}\edef\bxjs@tmpb{\meaning#1}\ifx\bxjs@tmpa\bxjs@tmpb #2\fi}

\let\jsEngine=n
\bxjs@test@engine\kanjiskip{\let\jsEngine=j}
\bxjs@test@engine\intercharskip{\let\jsEngine=j}
\bxjs@test@engine\Omegaversion{\let\jsEngine=O}
\bxjs@test@engine\XeTeXversion{\let\jsEngine=x}
\bxjs@test@engine\pdftexversion{\let\jsEngine=p}
\bxjs@test@engine\luatexversion{\let\jsEngine=l}

現状での処理系バージョン要件は以下の通りである。

- \TeX: 0.997版（2007年）以上

\TeX: 3.0版 [1990/03] 以上
- \p\TeX: 2.0版 [1995/03] 以上
- \up\TeX: 0.10版 [2007/07] 以上
- pdf\TeX: 1.40版 [2007/01] 以上
- \Lua\TeX: 0.60版 [2010/04] 以上
- Xq\TeX: 0.9994版 [2009/06] 以上

※ OmegaとNTT\jTeXは“公式にはサポート外”の扱い（動作は何も保証されない）であるが、クラス本体では処理系の種類は扱いため検査しないこととする。
※クラス本体での要件は扱えたのであるが、実質的にはそれらの要件を満たすことが求められる。

\TeX処理系のバージョンがサポート対象であるかを検査する。
\let\tempswatrue
非サポートのバージョン場合は強制終了させる。

非サポートのバージョン場合は強制終了させる。

万が一「互換モード」になっていた場合は、これ以上進むと危険なので強制終了させる。

※ 2017/04/15 版より前のカーネルの@ifpackagelater は非読込の場合に実行するとエラーになることに注意。

※ 2017/04/15 版より前のカーネルの@ifpackagelater は非読込の場合に実行するとエラーになることに注意。

@ifwithupTeX [スイッチ] エンジンが「内部漢字コードが Unicode の upTEX」であるか。

※ 2023 年 6 月に pMTEX の TEX 処理系が「e-pTEX」から「内部漢字コードが非 Unicode の e-upTEX」に変わったが、これによる影響はない。
エンジンが pTeX-ng であるか。

エンジンが ε-TeX 拡張をもつか。
※ XeTeX と LuaTeX は ε-TeX 拡張をもつ版のみがあり、NTT jTeX はもたない版のみがある。その他のエンジンは両方の版がある。

pdfTeX・LuaTeX が PDF モードで動作しているか。

expl3 がカーネルに組み込まれているか。
※ 2020/02/02 版以降のカーネルには組み込まれている。

オプション中の波括弧の使用にカーネルが対応しているか。
※ 正確に言うと、2021/06/01 版以降のカーネルでは「未使用オプション判定」の処理で = 以降のトークン列（key-value の value の部分）を無視するので、この部分には波括弧を含めることができることができる。
※ \@removeelement と \in@ の実装は変更されておらず、これらのマクロの第 1 引数には波括弧を含むトークン列を指定できない。

LATEX の既定のフォントエンコーディングが TU であるか。
※ 2017/01/01 以降の LATEX カーネルにおいて「Unicode を表す LATEX 公式のフォントエンコーディング」である “TU” が導入され、これ以降の LATEX を XeTeX または LuaTeX で動かしている場合は、既定のエンコーディングが TU になる。それ以外の場合は、既定のエンコーディングは OT1 である。

LATEX の新しいフック管理システムが未導入であるか。
※カーネルの 2020/10/01 版で導入された。

\newif\ifbxjs@old@hook@system
\bxjs@if@format@at@least{2020/10/01}{}{\bxjs@old@hook@systemtrue}

■依存パッケージ読込
長さ値の指定で式を利用可能にするため calc を読み込む。
\RequirePackage{calc}
クラスオプションで key-value 形式を使用するため keyval を読み込む。
\RequirePackage{keyval}
PDF モードの判定を L\TeX 公式のパッケージに任せたいので、もし「iftex の \ifpdf」が利用できるならば、jsInPdfMode スイッチをその値に一致させる。
※ iftex で \ifpdf が利用できるのは 1.0 版 [2019/10/24] から。
\IfFileExists{iftex.sty}{{\RequirePackage{iftex}}}{%}
\begingroup\expandafter\endgroup
\expandafter\ifx\csname ifpdf\endcsname\@undefined\else
\expandafter\let\csname ifjsInPdfMode\expandafter\endcsname\csname ifpdf\endcsname\fi
クラスの本体ではこの他に以下のパッケージが読み込まれる。
geometry
また状況によっては以下のパッケージが読み込まれる可能性がある。
bxwareki、jslogo、plautopatch、type1cm
※和文ドライバがさらにパッケージを読み込むこともある。
\jsAtEndOfClass  このクラスの読込終了時に対するフック。(補助パッケージ中で用いられる。)
\def\jsAtEndOfClass{%}
\expandafter\@addto@macro\csname\bxjs@clsname.cls-hook\endcsname{}}
互換性のための補助パッケージを読み込む。
\IfFileExists{bxjscompat.sty}{{\RequirePackage{bxjscompat}}}{%}
\begin{itemize}
\item BXJS クラス特有の設定

Lua\TeX の場合、本クラス用の Lua モジュールを用意する。
\ifx 1\jsEngine
\directlua{ bxjs = {} }
\fi
\bxjs@protected \TeX の拡張が有効な場合にのみ protected の効果をもつ。
無引数の頑強な命令を定義する。
\def\bxjs@robust@def\protect\def

\bxjs@CGHN は LATEX カーネルの 2021/11/15 版の改修で「要素の順が変わった」フック名について、「新仕様において正しい名前」を「使用中の LATEX において正しい名前」に変換する。例えば、\bxjs@CGHN{package/PKG/after} は旧仕様の LATEX では "package/after/PKG" に展開される。

\def\bxjs@CGHN\if\if\format\at\least\((2021/11/15)\}\%\else\\fi{\bxjs@CGHN\if\else\bxjs@CGHN\a\fi//}{\bxjs@CGHN\a//}{\bxjs@CGHN\a#1/}{\bxjs@CGHN\a#1/#2/#3/}{\bxjs@CGHN\a#1/#2/#3/#2)}

\bxjs@cond は LATEX の if 文（ifXXX……\fi）を末尾呼出形式に変換するためのマクロ。
\def\bxjs@cond\if\if\bxjs@cond\fi\%\else\bxjs@cond\fi{\bxjs@cond\fi\%\fi}

\bxjs@expanded を定義する。
\def\bxjs@cslet\bxjs@cslet\bxjs@cslet\bxjs@cslet\csname\endcsname

\bxjs@csletcs は \bxjs@csletcs\bxjs@csletcs\bxjs@csletcs\csname\endcsname

\bxjs@catopt は \bxjs@catopt\bxjs@catopt\bxjs@catopt\csname\endcsname

\bxjs@ifplus は \if\star\bxjs@ifplus\bxjs@ifplus\bxjs@ifplus\csname\endcsname

\bxjs@trim は \bxjs@trim\bxjs@trim\CS

TODO: 2.9 \bxjs@expanded を定義する。
\def\bxjs@trim#1{\expandafter\bxjs@trim@a#1\@nil#1}
\def\bxjs@trim@a{\futurelet\bxjs@tmpb\bxjs@trim@b}
\def\bxjs@trim@b{\bxjs@cond\ifx\bxjs@tmpb\@sptoken\fi \bxjs@trim@c\bxjs@trim@a}{\bxjs@trim@d}
\def\bxjs@trim@c#1 {#1}
\def\bxjs@trim@d#1\@nil{\bxjs@trim@e#1\@nil\@nnil}
\def\bxjs@trim@e#1\@nil#2\@nnil{\bxjs@cond\ifx\@nnil#2\@nnil\fi \bxjs@trim@f#1\@nnil}{\bxjs@trim@e#1\@nil\@nnil\@nnil}
\def\bxjs@trim@f#1\@nil#2\@nnil#3{\def#3{#1}}

\bxjs@set@array@from@clist \bxjs@set@array@from@clist{⟨配列名接頭辞⟩}{⟨コンマ区切りリスト⟩}: コンマ区切りの値のリストから擬似配列を生成する。※各要素について、先頭・末尾の空白トークン群は除去される。
\def\bxjs@set@array@from@clist#1#2{\@tempcnta\z@ \@for\bxjs@tmpa:=\@empty#2\do{%\bxjs@trim\bxjs@tmpa \bxjs@cslet{#1/\the\@tempcnta}\bxjs@tmpa \advance\@tempcnta\@one\relax \bxjs@cslet{#1/\the\@tempcnta}\relax}}
\let\c@bxjs@tempcnta\@tempcnta
\def\bxjs@gset@tempcnta{\setcounter{bxjs@tempcnta}}

\bxjs@advance@qc \bxjs@advance@qc\CS{⟨値⟩}: 擬似整数レジスタに値を加算する。
\def\bxjs@advance@qc#1#2{\begingroup \@tempcnta=#1\relax \advance\@tempcnta by#2\relax \global\chardef\bxjs@g@tmpa\@tempcnta \endgroup \let#1\bxjs@g@tmpa}

\begin{Verbatim}
\bxjs@eTeX \bxjs@advance@count ε-Tex\end{Verbatim} 拡張が有効なら通常の整数レジスタ、無効なら擬似整数レジスタを用いる。
\begin{Verbatim}
\def\bxjs@advance@count#1#2{\if\jsWitheTeX\let\bxjs@new@count\newcount\else\def\bxjs@new@count#1{\chardef#1\z@}\let\bxjs@advance@count\bxjs@advance@qc\fi}

\begin{Verbatim}
\jsSetQHLength \jsSetQHLength\CS{⟨長さ式⟩}: \setlengthの変種で、通常のcalcの長さ式の代わりに、「Q/H/trueQ/trueH/zw/zhの単位付きの実数」が記述できる（この場合は式は使えない）。
\def\jsSetQHLength#1#2{\begingroup \bxjs@parse@qh{#2} \ifx\bxjs@tmpb\relax \setlength\@tempdima{#2} \else \def\bxjs@new@count#1{\chardef#1\z@}\def\bxjs@advance@count#1#2{\iftrue\advance#1by#2\relax\else \def\bxjs@new@count#1{\chardef#1\z@}\fi}
\end{Verbatim}
bxjs\parse\qh が Q/H/trueQ/trueH/zw/zh で終わる場合、単位用の寸法値マクロ bxjs\unit@XXX が定義済なら、bxjs@tmpb に #1 に等しい寸法の表現を返し、そうでないならエラーを出す。それ以外では、bxjs@tmpb は \relax になる。
※ (u)p\LaTeX の場合はこれらの和文単位はエンジンでサポートされる。しかし和文フォントの設定が完了するまでは zw/zh の値は正しくない。

今後の段階では Q/H だけが使用可能。

\ifbxjs@after@preamble 〔スイッチ〕文書本体が開始しているか。
\newif\ifbxjs@after@preamble
\bxjs@begin@document@hook BXJS クラス用の文書本体開始時フック。
\AtBeginDocument{\bxjs@begin@document@hook}
\bxjs@post@option@hook \ProcessOptions 直後に実行されるフック。
両文ドライバ読込直前に実行されるフック。

BXJS クラス用の \AtEndPreamble フック。
※ \AtEndPreamble が利用できない場合は無効になる。

一時的な手続き用の制御綴。

\jsInhibitGlue \inhibitglue が定義されていればそれを実行し、未定義ならば何もしない。

2 オプション

これらのクラスは \documentclass{jsarticle} あるいは \documentclass[オプション]{jsarticle} のように呼び出します。
まず、オプションに関連するいくつかのコマンドやスイッチ（論理変数）を定義します。

段組のときに真になる論理変数です。

これを真にすると表題、概要を独立したページに出力します。

\if@restonecol\chapter, \part を右ページ起こしにするかどうかです。横組の書籍では真が標準で、要するに片起こし、奇数ページ起こしになります。

\if@openright\chapter, \part を右ページ起こしにするかどうかです。横組の書籍では真が標準で、要するに片起こし、奇数ページ起こしになります。

\if@openleft [2017-02-24]\chapter, \part を左ページ起こしにするかどうかです。

\if@mainmatter 真なら本文、偽なら前付け・後付けです。偽なら \chapter で章番号が出ません。
BXJS では report 系でも定義されることに注意。

226 \%<book|report>\newif\if@mainmatter \@mainmattertrue 
\if@enablejfam 和文フォントを数式フォントとして登録するかどうかを示すスイッチです。

JS クラスと異なり、初期値は偽とする。

227 \newif\if@enablejfam \@enablejfamfalse 

以下で各オプションを宣言します。

■用紙サイズ　JIS や ISO の A0 判は面積 1 m²、縦横比 1 : \sqrt{2} の長方形の辺の長さを mm 単位に切り捨てたものです。これを基準として順に半截しては mm 単位に切り捨てたものが A1, A2, …です。

B 判は JIS と ISO で定義が異なります。JIS では B0 判の面積が 1.5 m² ですが、ISO では B1 判の辺の長さが A0 判と A1 判の辺の長さの幾何平均です。したがって ISO の B0 判は 1000 mm × 1414 mm です。これに、\LaTeXX\varepsilon の b5paper は 250 mm × 176 mm ですが、pL\LaTeXX\varepsilon の b5paper は 257 mm × 182 mm になっています。ここでは pL\LaTeXX\varepsilon に従いました。

デフォルトは a4paper です。

b5var (B5 変形, 182mm×230mm), a4var (A4 変形, 210mm×283mm) を追加しました。

BXJS クラスではページレイアウト設定に geometry パッケージを用いる。用紙サイズ設定は geometry に渡すオプションの指定と扱われる。228 \@onlypreamble\bxjs@setpaper 
229 \def\bxjs@setpaper#1{\def\bxjs@param@paper{#1}} 
230 \newif\ifbxjs@iso@bsize 
231 \DeclareOption{iso-bsize}{\bxjs@iso@bsizetrue} 
232 \@onlypreamble\bxjs@setpaper@bsize 
233 \def\bxjs@setpaper@bsize#1{\edef\bxjs@param@paper{b#1\ifbxjs@iso@bsize paper\else j\fi}} 
234 \DeclareOption{a3paper}{\bxjs@setpaper{a3paper}} 
235 \DeclareOption{a4paper}{\bxjs@setpaper{a4paper}} 
236 \DeclareOption{a5paper}{\bxjs@setpaper{a5paper}} 
237 \DeclareOption{a6paper}{\bxjs@setpaper{a6paper}} 
238 \DeclareOption{b4paper}{\bxjs@setpaper{b4paper}} 
239 \DeclareOption{b5paper}{\bxjs@setpaper{b5paper}@bsize{4}} 
240 \DeclareOption{b6paper}{\bxjs@setpaper{b5paper}@bsize{5}} 
241 \DeclareOption{b7paper}{\bxjs@setpaper{b5paper}@bsize{6}} 
242 \DeclareOption{a4j}{\bxjs@setpaper{a4paper}} 
243 \DeclareOption{a5j}{\bxjs@setpaper{a5paper}} 
244 \DeclareOption{b4j}{\bxjs@setpaper{b4j}} 
245 \DeclareOption{b5j}{\bxjs@setpaper{b5j}}}
\DeclareOption{a4var}{\bxjs@setpaper{\{210truemm\}{283truemm}}}  \\
\DeclareOption{b5var}{\bxjs@setpaper{\{182truemm\}{230truemm}}}  \\
\DeclareOption{letterpaper}{\bxjs@setpaper{\textit{\{letterpaper\}}}}  \\
\DeclareOption{legalpaper}{\bxjs@setpaper{\textit{\{legalpaper\}}}}  \\
\DeclareOption{executivepaper}{\bxjs@setpaper{\textit{\{executivepaper\}}}}  \\
\DeclareOption{screen}{\bxjs@setpaper{\textit{\{screen\}}}}

geometry の用紙サイズのオプション名を全てサポートする。

\@for\bxjs@tmpa:={a0,a1,a2,c0,c1,c2,c3,c4,c5,c6,ansia,ansib,ansic,ansid,ansie}\do{\edef\bxjs@next{\noexpand\DeclareOption{\bxjs@tmpa\ paper}{\noexpand\bxjs@setpaper{\bxjs@tmpa\ paper}}}\bxjs@next}

\DeclareOption{screen}{\bxjs@setpaper{\textit{\{screen\}}}}

ただし b?paper は iso-bsize の指定に従い ISO と JIS の適切な方の B 列を選択する。

\@for\bxjs@tmpa:={0,1,2,3}\do{\edef\bxjs@next{\noexpand\DeclareOption{b\bxjs@tmpa\ paper}{\noexpand\bxjs@setpaper@bsize{\bxjs@tmpa}}}\bxjs@next}

Pandoc で用紙サイズを指定した場合は出力 \LaTeX \texttt{ソース}において「後ろに paper を付けた名前のオプション」が指定される。これに対処するため、後ろに paper をつけた形を用意する。さらに、「Pandoc で用紙サイズを custom すると実質的に何も設定しない」ようにするため custompaper というオプションを用意する。

\DeclareOption{a4varpaper}{\bxjs@setpaper{\{210truemm\}{283truemm}}}  \\
\DeclareOption{b5varpaper}{\bxjs@setpaper{\{182truemm\}{230truemm}}}  \\
\DeclareOption{screenpaper}{\bxjs@setpaper{\textit{\{screen\}}}}  \\
\DeclareOption{custompaper}{\}

\begin{description}
\item[横置き] 用紙の縦と横の長さを入れ換えます。
\end{description}

\newif\if@landscape  \\
\@landscapefalse  \\
\DeclareOption{landscape}{\@landscape=true}

\begin{description}
\item[slide] オプション slide を新設しました。
\end{description}

[2016-10-08] slide オプションは article 以外では使い物にならなかったので、簡単のため article のみで使えるオプションとしました。

\newif\if@slide  \\
\@slidetrue  \\
%<!slide>\@slidefalse  \\
%<slide>\@slidetrue

BXJS ではスライド用のクラス bxjsslide を用意しているので、本来はこのスイッチは不要なはずである。しかし、JS クラスの一部のコードをそのまま使うために保持している。※この \if@slide という制御綴は、ユニクでないにも関わらず、衝突した場合に正常動作が保たれない、という問題を抱えている。

@ptsize の定義が変わったのでご迷惑をおかけしましたが，標準的なドキュメントクラスと同様にポイント数から 10 を引いたものに直しました。

2003-03-22 14Q オプションを追加しました。
2003-04-18 12Q オプションを追加しました。
2016-07-08 \mag を使わずに各種寸法をスケールさせるためのオプション nomag を新設しました。usemag オプションの指定で従来通りの動作となります。デフォルトは usemag です。
2016-07-24 オプティカルサイズを調整するために NFSS ヘパッチを当てるオプション nomag* を新設しました。

\@ptsize は 10pt，11pt，12pt が指定された時のみ JS クラスと同じ値とし，それ以外は jsUnusualPtSize \(= -20\) にする。

\newcommand{\@ptsize}{0}
\def\bxjs@param@basefontsize{10pt}
\def\jsUnusualPtSize{-20}
\def\bxjs@setbasefontsize{\bxjs@setbasefontsize\@tempdima}{#1}{%}
\def\bxjs@setjbasefontsize\@tempdima{#1}{%}
\setkeys{bxjs}{jbase=#1}

\ifjsc@mag は「mag を使うか」を表すスイッチ。
\ifjsc@mag@xreal は「NFSS にパッチを当てるか」を表すスイッチ。
\newif\ifjsc@mag
\newif\ifjsc@mag@xreal
\let\jsc@magscale@undefined
\DeclareOption{8pt}{\bxjs@setbasefontsize\@tempdima{8pt}}
\DeclareOption{9pt}{\bxjs@setbasefontsize\@tempdima{9pt}}
JSクラス互換のmagstyle設定オプション。
\DeclareOption{usemag}{\let\bxjs@magstyle\bxjs@magstyle@@usemag}
\DeclareOption{nomag}{\let\bxjs@magstyle\bxjs@magstyle@@nomag}
\DeclareOption{nomag*}{\let\bxjs@magstyle\bxjs@magstyle@@xreal}

■トンボオプション  トンボ（crop marks）を出力します。実際の処理は\pLaTeX\ ε本体で行いますが（\plcore.dtx参照）。オプションtombowで日付きのトンボ, オプションtomboで日付なしのトンボを出力します。これらはアスキ版のままです。カウンタ\hour, \minuteは\pLaTeX\ ε本体で宣言されています。

取りあえず、\pLaTeX\系の場合に限り、JSクラスのトンボ関連のコードをそのまま活かしておく。正常に動作する保証はない。

\if j\jsEngine
\hour\time \divide\hour by 60\relax
\@tempcnta\hour \multiply\@tempcnta 60\relax
\minute\time \advance\minute-\@tempcnta
\DeclareOption{tombow}{%\tombowtrue \tombowdatetrue
\setlength{\@tombowwidth}{.1\p@}%
\@bannertoken{\jobname\space(\number\year-\two@digits\month-\two@digits\day
\space\two@digits\hour:\two@digits\minute)%
\maketombowbox}
\DeclareOption{tombo}{%\tombowtrue \tombowdatefalse
\setlength{\@tombowwidth}{.1\p@}%
\maketombowbox}
面付け オプション mentuke で幅ゼロのトンボを出力します。面付けに便利です。これもアスキー版のままでします。

\if j\\jsEngine
\DeclareOption{mentuke}{%
  \tombowtrue \tombowdatefalse
  \setlength{\@tombowwidth}{\z@}%
  \maketombowbox%
\fi

両面、片面オプション twoside で奇数ページ・偶数ページのレイアウトが変わります。
[2003-04-29] vartwoside でどちらのページも傍注が右側になります。

\DeclareOption{oneside}{\@twosidefalse \@mparswitchfalse}
\DeclareOption{twoside}{\@twosidetrue \@mparswitchtrue}
\DeclareOption{vartwoside}{\@twosidetrue \@mparswitchfalse}

二段組 twocolumn で二段組になります。

\DeclareOption{onecolumn}{\@twocolumnfalse}
\DeclareOption{twocolumn}{\@twocolumntrue}

表題ページ titlepage で表題・概要を独立したページに出力します。

\DeclareOption{titlepage}{\@titlepagetrue}
\DeclareOption{notitlepage}{\@titlepagefalse}

右左起こし 書籍では章は通常は奇数ページ起こしになりますが、横組ではこれを openright と表すことにしてあります。openany で偶数ページからでも始まるようになります。
[2017-02-24] openright は横組では奇数ページ起こし、縦組では偶数ページ起こしを表します。ややこしいですが、これは \LaTeX の標準クラスが西欧の横組事情しか考慮せずに、奇数ページ起こしと右起こしを一緒にしてしまったせいです。縦組での奇数ページ起こしと横組での偶数ページ起こしを表現したいので、jsclasses では新たに openleft と追加しました。

%<book|report>\DeclareOption{openright}{\@openrighttrue\@openleftfalse}
%<book|report>\DeclareOption{openleft}{\@openlefttrue\@openrightfalse}
%<book|report>\DeclareOption{openany}{\@openrightfalse\@openleftfalse}

eqnarray 環境と数式の位置 森本さんのご教示にしたがって前に移動しました。

eqnarray (env.) \LaTeX の eqnarray 環境では & でできるアキが大きすぎるようですので、少し小さくします。また、中央の要素も \displaystyle にします。
[2022-09-13] \LaTeX2ε 2021-11-15 (ltmath.dtx 2021/10/14 v1.2j) で\@currentcounter が追加されましたので、追随します。

\def\eqnarray{%
  \stepcounter{equation}%
}
leqno で数式番号が左側になります。fleqn で数式が本文左端から一定距離のところに出力されます。森本さんにしたがって訂正しました。

\begin{verbatim}
\def\@currentlabel{\p@equation\theequation}
\def\@currentcounter{equation}
\global@eqnswtrue
\m@th
\global\@eqcnt\z@
\tabskip\@centering
\let\\@eqncr
$$\everycr{}\halign to\displaywidth\bgroup
\tabskip\z@skip{##}$\@eqnsel
&\global\@eqcnt\@ne \hfil$\displaystyle{{}##{}}$\hfil &
&\global\@eqcnt\tw@
$\displaystyle{##}$\hfil\tabskip\@centering
&\global\@eqcnt\thr@@ \hb@xt@\z@\bgroup\hss##\egroup
\tabskip\z@skip
\cr}
\end{verbatim}

\textit{leqno} で数式番号が左側になります。\textit{fleqn} で数式が本文左端から一定距離のところに出力されます。森本さんにしたがって訂正しました。
文献リスト  文献リストを open 形式（著者名や書名の後に改行が入る）で出力します。これは使われ的事情ではないのでコメントアウトしてあります。

数式フォントとして和文フォントを登録しないオプション  数式中では 16 通りのフォントしか使えません。AMSFonts や mathptmx パッケージを使って数式フォントをたくさん使うと “Too many math alphabets…” というエラーが起こってしまいます。disablejfaq オプションを付ければ、明朝・ゴシックを数式用フォントとして登録するのをやめますので、数式用フォントが二つ節約できます。いずれにしても \textmc や \mbox や amsmath パッケージの \text を使うには数式中で和文フォントを使って使いますので、この新ドキュメントクラスでは標準で和文フォントを数式用に登録しないことにしていたのですが、従来のドキュメントクラスの仕様に合わせることにしました。

\bxjs@enablejfaq [暗黙文字トークン] enablejfaq オプションの状態：

enablejfaq オプションの処理。

JS クラスとの互換のため disablejfaq オプションを定義する。

ドラフト  draft で overfull box の起きた行末に 5pt の罫線を引きます。
[2016-07-13] \ifdraft を定義するのをやめました。

\ifsjDraft [スイッチ] draft オプションが指定されているか。
※ JS クラスの \ifdraft が廃止されるので、BXJS クラスでも \ifdraft を 2.0 版で廃止した。
■和文フォントメトリックの選択
このクラスファイルでは、和文 TFM として東京書籍印刷の小林肇さんの作られた JIS フォントメトリック（jis, jisg）を標準で使うことにしますが、従来の min10, goth10 を使いたいときは mingoth というオプションを指定します。また、winjis オプションで winjis メトリック（OTF パッケージと同じ psitau さん作；ソースに書かれた Windows の機種依存文字が dvips, dvipdfmx などで出力出来るようになる）が使えます。

[2018-02-04] winjis オプションはコッソリ削除しました。代替として、同等ものをパッケージ化 (winjis.sty) して、GitHub にはコッソリ置いておきます。

BXJS クラスではここは和文ドライバの管轄。

■papersize スペシャルの利用
dvips や dviout で用紙設定を自動化するにはオプション papersize を与えます。

BXJS クラスでは geometry パッケージがこの処理を行う。

\ifbxjs@papersize [スイッチ] papersize スペシャルを出力するか。既定で有効であるが、nopapersize オプションで無効にする。
※ JS クラスでは \ifpapersize という制御綴だが、これは採用しない。

■英語化
オプション english を新設しました。

※\if@english は非ユニークで衝突耐性がない。

■jsbook を jsreport もどとに　オプション report を新設しました。
[2017-02-13] 従来は「jsreport 相当」を jsbook の report オプションで提供していましたが、新しく jsreport クラスも作りました。どちらでもお好きな方を使ってください。
BXJS では当初から bxjsreport クラスが用意されている。

■ jslogo パッケージの読み込み
LaTeX 関連のロゴを再定義する jslogo パッケージを読み込まないオプション nojslogo を新設しました。jslogo オプションの指定で従来どおりの動作となります。デフォルトは jslogo で、すなわちパッケージを読み込みます。

BXJS クラスでは、nojslogo を既定とする。

410 \newif\if@jslogo \@jslogofalse
411 \DeclareOption{jslogo}{\@jslogotrue}
412 \DeclareOption{nojslogo}{\@jslogofalse}

■ 複合設定オプション

\texttt{\bxjs@invscale} を書く場所を決める。（JS クラスと同じにはできなそう。）

\texttt{\bxjs@invscale} は \LaTeX における「長さのスケール」の逆関数を求めるもの。例えば \texttt{\bxjs@invscale\dimX{1.3}} は \texttt{\dimX=1.3\dimX} の逆の演算を行う。

※局所化の \texttt{\begingroup \egroup} について、以前は \texttt{\group \egroup} を使っていたが、これだと数式モード中では空のサブ数式を生み出してしまうため修正した。
※元の長さが 128pt 以上の場合でも動作するように修正した。

413 \mathchardef\bxjs@isc@ll=128
414 \mathchardef\bxjs@isc@sl=259
415 \def\bxjs@isc@sl@h{65539 }{ }
416 \def\bxjs@invscale#1#2{\begingroup
417 \@tempdima=#1\relax \@tempdimb#2\p@\relax
418 \ifdim\@tempdima<\bxjs@isc@ll\p@
419 \@tempcnta\@tempdima \multiply\@tempcnta\@cclvi
420 \divide\@tempcnta\@tempdimb \multiply\@tempcnta\@cclvi
421 \else
422 \@tempcnta\@tempdima \divide\@tempcnta\@tempdimb
423 \multiply\@tempcnta\p@ \let\bxjs@isc@sl\bxjs@isc@sl@h
424 \fi
425 \@tempcntb\p@ \divide\@tempcntb\@tempdimb
426 \advance\@tempcntb\@tempdimb \advance\@tempcnta\tw@
427 \@tempdimb\@tempcnta\one
428 \advance\@tempcnta\@tempcntb \advance\@tempcnta\@tempcntb
429 \advance\@tempcnta\bxjs@isc@sl \tempdimc\@tempcnta\one
430 \@whiledim\@tempdimb<\tempdimc\do{\%
431 \@tempcntb\@tempdimb \advance\@tempcntb\@tempdimc
432 \advance\@tempcntb\one \divide\@tempcntb\tw@
433 \ifdim\@tempcntb\p@
434 \advance\@tempcntb\m\one \tempdimc=\@tempcntb\one
435 \else \@tempdimb=\@tempcntb\one \fi\%
436 \xdef\bxjs@gtmpa{\the\@tempdimb}\%
437 \endgroup \#1=\bxjs@gtmpa\relax}
複合設定オプションとは、「エンジンやドライバや和文ドライバの設定を含む、複数の設定を一度に行うオプション」のことである。ある特定の設定を短く書く必要性が高いと判断される場合に用意される。

pandoc オプションは、Pandoc で \LaTeX \textup{用の既定テンプレート}を用いて他形式から \LaTeX （および PDF）形式に変換する用途に最適化した設定を与える。

\begin{verbatim}
\DeclareOption{pandoc}{
  \bxjs@apply@pandoc@opt
}\@onlypreamble\bxjs@apply@pandoc@opt
\def\bxjs@apply@pandoc@opt{\
  和文ドライバを pandoc に、エンジン指定を autodetect-engine に変更する。
  ※実際の和文ドライバ・エンジン設定より優先される。
  \g@addto@macro\bxjs@post@option@hook{\
    \bxjs@oldfontcommandstrue
    \setkeys{bxjs}{ja=pandoc}\
    \let\bxjs@engine@given=*}
  ドライバオプションを dvi=dvipdfmx 相当に変更する。
  ※これは実際のドライバ設定で上書きできる（オプション宣言順に注意）。
  \ifx\bxjs@driver@opt\@undefined\
    \def\bxjs@driver@opt{dvipdfmx}\
    \bxjs@dvi@opttrue
  \fi
  \global\let\bxjs@apply@pandoc@opt\relax}
\end{verbatim}

pandoc+ オプションは、pandoc と同じ設定をした上で、さらに和文パラメタの先頭に _plus を追加する。

\begin{verbatim}
\DeclareOption{pandoc+}{
  \g@addto@macro\bxjs@post@option@hook{\
    \edef\jsJaParam{\bxjs@catopt{_plus}\jsJaParam}}\
  \ExecuteOptions{pandoc}}
\end{verbatim}

■エンジン・ドライバオプション ☃

\begin{verbatim}
\bxjs@engine@given 〔暗黙文字トークン〕オプションで明示されたエンジンの種別。
\bxjs@engine@opt 明示されたエンジンのオプション名。
\end{verbatim}

エンジン明示指定のオプションの処理。
※ 0.9pre 版の暫定仕様と異なり、エンジン名は \texttt{\LaTeX} に限定する。xetex や pdftex は一般的な \LaTeX の慣習に従って「ドライバの指定」とみなすべきだ。

\begin{verbatim}
\DeclareOption{autodetect-engine}{
  \let\bxjs@engine@given=}
\end{verbatim}
暗黙文字トークン オプションで明示されたドライバの種別。

\begin{verbatim}
\let\bxjs@engine@given=n
\end{verbatim}

※ class-nodvidriver は BXJS クラスの仕様上は nodvidriver と完全に等価であるが、「グローバルオプションに何があるか」の点で異なる。
dvipdfmx-if-dvi は 2.0 版より非推奨となった。

■ その他の BXJS 独自オプション ☃

非推奨のオプションについて警告を出す。

\ifbxjs@bigcode 〔スイッチ〕 up\TeX で有効化する ToUnicode CMap として「UTF8-UCS2」の代わりに「UTF8-UTF16」を使うか。BMP 外の文字に対応できる「UTF8-UTF16」の方が望ましいのであるが、このファイルが利用可能かの確実な判定が困難であるため、既定を真とした上で、オプションで指定することとする。
※ 2.0 版より、既定値を常に真とする。

\DeclareOption{nobigcode}{%}
\DeclareOption{bigcode}{%}

nobigcode ／ bigcode オプションの定義。
allowoldfontcommands を既定で有効にするか。

\ifbxjs@oldfontcommands _allowoldfontcommands オプションの定義。
※ oldfontcommands オプションの名前は memoir クラスに倣った。ちなみに KOMA-Script では enabledeprecatedfontcommands であるがこれはちょっとアレなので避けた。

\DeclareOption{nooldfontcommands}{% 
  \bxjs@oldfontcommandsfalse}
\DeclareOption{oldfontcommands}{% 
  \bxjs@oldfontcommandstrue}

■ 無効および廃止されたオプション ☳

\bxjs@register@badopt マクロを登録する。文書本体開始時に、当該オプションが「未使用のグローバルオプション」になっている場合に badopt マクロが実行される。

\ifbxjs@brace@safe \@onlypreamble\bxjs@register@badopt \expandafter\@onlypreamble\csname bxjs@badopt/#1\endcsname \@namedef{bxjs@badopt/#1} \g@addto@macro\bxjs@begin@document@hook{% \@for\bxjs@tmpa:=\@unusedoptionlist\do{\@nameuse{bxjs@badopt/\bxjs@tmpa}}} \fi \bxjs@invalid@opt 無効オプションを宣言する。そのオプションが指定された場合、それがグローバルオプションとして他のパッケージによって使用されていなければ、文書本体開始時にエラーを出す。※古いカーネルでは未使用検査ができないため、その場で警告を出す。

\ifj\jsEngine\else 26
keyval 型のオプション その他のオプションは keyval の機構を用いて処理する。

\DeclareOption*{%
  \bxjs@invalid@opt{tombo}{% Option 'tombo' can be used only on (u)pLaTeX%
  \bxjs@invalid@opt{tombow}{% Option 'tombow' can be used only on (u)pLaTeX%
  \bxjs@invalid@opt{mentuke}{% Option 'mentuke' can be used only on (u)pLaTeX%
  fi

■keyval 型のオプション ☃ その他のオプションは keyval の機構を用いて処理する。

\DeclareOption*{%
  \bxjs@check@ja@prefix \ifx\bxjs@next\relax
  \def\bxjs@next{\bxjs@cls@setkeys{bxjs}}%
  \expandafter\bxjs@next\expandafter{\CurrentOption}%
  \else
  オプションが ja:XXX という形式である場合は japaram=XXX に振り替える。
  \edef\bxjs@next{%
  \noexpand\setkeys{bxjs}{japaram={\bxjs@next}}%
  }\bxjs@next
  fi}

\bxjs@check@ja@prefix オプション文字列が ja: で始まるかを検査し、そうである場合は後続の文字列を \bxjs@next に代入する。

\def\bxjs@check@ja@prefix{%
  \let\bxjs@next\relax
  \expandafter\bxjs@check@ja@prefix@a\CurrentOption\nil ja:\nil\nil\nil}
\def\bxjs@check@ja@prefix@a#1ja:#2\nil\nil#3\nil{
  \ifx\nil#1\nil \def\bxjs@next{#2}\fi}

\bxjs@safe@setkeys 未知のキーに対してエラー無しで無視する \setkeys。※ネスト不可。

\def\bxjs@safe@setkeys#1#2{%
  \let\bxjs@save@KV@errx\KV@errx \let\KV@errx\@gobble
  \setkeys{#1}{#2}%
  \let\KV@errx\bxjs@save@KV@errx}

\bxjs@cls@setkeys 未知のキーに対して (エラー無しで) \OptionNotUsed を行う \setkeys。\DeclareOption* 中で用いる。

\def\bxjs@cls@setkeys#1#2{%
  \let\bxjs@save@KV@errx\KV@errx \let\KV@errx\@gobble
  \setkeys{#1}{#2}%
  \let\KV@errx\bxjs@save@KV@errx}
\if\bxjs@brace@safe\else
  \let\bxjs@cls@setkeys\bxjs@safe@setkeys\fi
\def\bxjs@declaration@enum@option {濠}{(オプション名)\{新値\}}
\def\bxjs@declaration@enum@option* {濠}{(オプション名)\{新値\}}

27
“オプション名”=(値) のオプション指定に対して、\[bxjs\@\{enum \text{名}\}\] を \[bxjs\@\{enum \text{名}\}@\{値\}\] に等置する（後者の制御綴が未定義の場合はエラー）、という動作を規定する。

\onlypreamble\bxjs@declare@enum@option
\def\bxjs@declare@enum@option#1#2#3{% 
 bxjs@csletcs\{bxjs@#2\}\{bxjs@\#2##1\} \def\bxjs@\{#1\}{\@ifundefined{bxjs@\#2\##1}{\bxjs@error@keyval{#1}{##1}}{\bxjs@csletcs\{bxjs@\#2\}\{bxjs@\#2\##1\}}} \@onlypreamble
\bxjs@declare@bool@option\bxjs@declare@bool@option{\{オプション名\}}{\{スイッチ名\}}{\{初期値\}}
“オプション名”=(真偽値) のオプション指定に対して、\if\bxjs@\{スイッチ名\}\を設定する、という動作を規定する。
\onlypreamble\bxjs@declare@bool@option
\def\bxjs@declare@bool@option#1#2#3{% 
 \csname newif\expandafter\endcsname\csname ifbxjs@#2\endcsname \@nameuse{bxjs@#2##1} \define@key{bxjs}{#1}[true]{\@ifundefined{bxjs@#2##1}{\bxjs@error@keyval{#1}{##1}}{\@nameuse{bxjs@#2##1}}} \@onlypreamble
\bxjs@set@keyval\bxjs@set@keyval{\{key\}}{\{value\}}{\{error\}}
\bxjs@kv@\{key\}@\{value\} が定義済ならそれを実行し、未定義ならエラーを出す。
\onlypreamble
\def\bxjs@set@keyval#1#2#3{% 
 bxjs@csletcs\{bxjs@next\}\{bxjs@kv@#1@#2\} \ifx bxjs@next\relax \bxjs@error@keyval{#1}{#2}\else bxjs@next \fi} \@onlypreamble
\bxjs@error@keyval\bxjs@error@keyval{\{key\}}{\{value\}}{\{error\}}
\ClassError\bxjs@clsname{Invalid value 'value' for option #1} \@ehc
\jsScale \[実数値マクロ\] 和文スケール値。
\onlypreamble
\def\jsScale{0.924715} \@onlypreamble
\bxjs@base@opt\bxjs@base@opt 明示された base オプションの値。
\onlypreamble \setkeys{bxjs}{base=#1}% \@undefined
base オプションの処理。
\onlypreamble
\def\bxjs@base@opt{#1}{% 
 \edef\bxjs@base@opt{#1}\bxjs@setbasefontsize{#1}} \@undefined
\define@key{bxjs}{fontsize}{\setkeys{bxjs}{base=#1}}
\texttt{\bxjs@jbase@opt} 明示された \texttt{jbase} オプションの値。
630 \%let\bxjs@jbase@opt\@undefined

\texttt{jbase} オプションの処理。
631 \define\key{bxjs}{jbase}\{edef\bxjs@jbase@opt(#1)}
632 \define\key{bxjs}{jafontsize}\{\setkeys{bxjs}{jbase=\#1}}

\texttt{\bxjs@scale@opt} 明示された \texttt{scale} オプションの値。
633 \%let\bxjs@scale@opt\@undefined

\texttt{scale} オプションの処理。
634 \define\key{bxjs}{scale}\{%
635 \edef\bxjs@scale@opt(#1)\%
636 \let\jsScale\bxjs@scale@opt\%
637 \define\key{bxjs}{jafontscale}\{\setkeys{bxjs}{scale=\#1}}

\texttt{noscale} オプションの処理。
\textbf{TODO:3.0} \texttt{noscale} は廃止の予定。
638 \DeclareOption{noscale}{\bxjs@depre@opt@do{noscale}{scale=1}}

\texttt{\bxjs@param@mag} \texttt{mag} オプションの値。
639 \%let\bxjs@param@mag\relax

\texttt{mag} オプションの処理。
640 \define\key{bxjs}{mag}\{edef\bxjs@param@mag(#1)}

\texttt{paper} オプションの処理。
641 \define\key{bxjs}{paper}\{edef\bxjs@param@paper(#1)}

\texttt{\bxjs@jadriver} 和文ドライバの名前。
642 \let\bxjs@jadriver\relax

\texttt{\bxjs@jadriver@opt} 明示された和文ドライバの名前。
643 \%let\bxjs@jadriver@opt\@undefined

\texttt{ja} オプションの処理。
※ \texttt{jadriver} は 0.9 版で用いられた旧称。
\textbf{TODO:3.0} \texttt{jadriver} は廃止の予定。
※単なる \texttt{ja} という指定は無視される（Pandoc 対策）。
644 \define\key{bxjs}{jadriver}\{%
645 \bxjs@depre@opt@do{jadriver}{ja=\#1}\edef\bxjs@jadriver@opt(#1)}
646 \define\key{bxjs}{ja}\{\relax\%
647 \ifx\relax\#1\else\edef\bxjs@jadriver@opt(#1)\fi

\texttt{\jsJaFont} 和文フォント設定の名前。
648 \let\jsJaFont\@empty

\texttt{jafont} オプションの処理。
649 \define\key{bxjs}{jafont}\{edef\jsJaFont(#1)}

29
和文ドライバパラメタの文字列。
\let\jsJaParam@empty

japaram オプションの処理。
\define@key{bxjs}{japaram}{% \edef\jsJaParam{\bxjs@catopt\jsJaParam{#1}}}%

引数をもつ pandoc・pandoc+ オプションは、その引数を和文パラメタの指定と見なす。
\define@key{bxjs}{pandoc}{% \ExecuteOptions{pandoc}% \edef\jsJaParam{\bxjs@catopt\jsJaParam{#1}}}%
\define@key{bxjs}{pandoc+}{% \ExecuteOptions{pandoc+}% \edef\jsJaParam{\bxjs@catopt\jsJaParam{#1}}}%

\bxjs@magstyle magstyle 設定値。（古いイマイチな名前。）
\let\bxjs@magstyle@@mag=m \let\bxjs@magstyle@@real=r \let\bxjs@magstyle@@xreal=x

（新しい素敵な名前。）
※ただし制御綴としては、*付の名前は扱い難いので、\bxjs@magstyle@@xreal 方を優先させる。
\let\bxjs@magstyle@@usemag\bxjs@magstyle@@mag \let\bxjs@magstyle@@nomag\bxjs@magstyle@@real
\bxjs@cslet{bxjs@magstyle@@nomag*}\bxjs@magstyle@@xreal \bxjs@magstyle@@default は既定の値を表す。
\let\bxjs@magstyle@@default\bxjs@magstyle@@xreal
\let\bxjs@magstyle\bxjs@magstyle@@default

magstyle オプションの処理。
\define@key{bxjs}{magstyle}{% \bxjs@csletcs{bxjs@magstyle}{bxjs@magstyle@@#1}% \ifx\bxjs@magstyle\relax \bxjs@error@keyval{magstyle}{#1}% \let\bxjs@magstyle\bxjs@magstyle@@default \fi}%

\bxjs@geometry geometry オプションの指定値。
\let\bxjs@geometry@@class=c \let\bxjs@geometry@@user=u
\bxjs@declare@enum@option{geometry}{geometry}{class}
\ifbxjs@fancyhdr [スイッチ] fancyhdr の指定値。fancyhdr パッケージに対する調整を行うか。
682 \bxjs@declare@bool@option{fancyhdr}{fancyhdr}{true}

\ifbxjs@dvi@opt [スイッチ] dvi オプションが指定されたか。
683 \newif\ifbxjs@dvi@opt
DVI モードのドライバとドライバ種別の対応。
684 \let\bxjs@dvidriver@@dvipdfmx=\bxjs@driver@@dvipdfmx
685 \let\bxjs@dvidriver@@dvips=\bxjs@driver@@dvips
686 \let\bxjs@dvidriver@@dviout=\bxjs@driver@@dvimode
687 \let\bxjs@dvidriver@@xdvi=\bxjs@driver@@dvimode
688 \let\bxjs@dvidriver@@nodvidriver=\bxjs@driver@@none
689 \bxjs@cslet{bxjs@dvidriver@@class-nodvidriver}=\bxjs@driver@@none
dvi オプションの処理。
690 \define@key{bxjs}{dvi}{%}
691 \bxjs@csletcs{bxjs@tmpa}{bxjs@dvidriver@@#1}%
692 \ifx\bxjs@tmpa\relax
693 \bxjs@error@keyval{dvi}{#1}%
694 \else
\fi

\ifbxjs@layout@buggyhmargin [スイッチ] bxjsbook の左右マージンがアレか。
※ layout が v1 の場合はアレになる。
695 \newif\ifbxjs@layout@buggyhmargin

\ifbxjs@force@chapterabstract [スイッチ] abstract 環境を chapterabstract にするか。
※ bxjsbook では常に真。bxjsreport では layout が v1 の場合に真になる。
700 \newif\ifbxjs@force@chapterabstract
701 \%<book>\bxjs@force@chapterabstracttrue

layout オプションの処理。
702 \%<book>\bxsks@overwrite@layout@v1\{%
703 \%<book>\bxjs@overwrite@layout@buggyhmargin\true
704 \%<report>\bxjs@force@chapterabstract\true
705 \%
706 \%<book>\bxsks@overwrite@layout@v2\{%
707 \%<book>\bxjs@overwrite@layout@buggyhmargin\false
708 \%<report>\bxjs@force@chapterabstract\false
709 \%
710 \define@key{bxjs}{layout}{%}
711 \bxjs@set@keyval{layout}{#1}{}%}

\bxjs@textwidth@limit textwidth-limit の指定値。
\textwidth の指定値。
\edef\bxjs@textwidth@opt{#1}

\numberoflines の指定値。
\edef\bxjs@numberoflines@opt{#1}

\paragraphmark の指定値。パラグラフのマーク。
\edef\bxjs@paragraphmark{#1}

\wholezwlines の指定値。
\ifbxjs@wholezwlines
  \bxjs@declarebooloption{whole-zw-lines}{whole@zw@lines}{true}
\else
  \true
\fi

\jaspacecmd の指定値。
\ifbxjs@jaspacecmd
  \bxjs@declarebooloption{jaspace-cmd}{jaspace@cmd}{true}
\else
  \true
\fi

\fixatcmd の指定値。
\ifbxjs@fixatcmd
  \bxjs@declarebooloption{fix-at-cmd}{fix@at@cmd}{true}
\else
  \true
\fi

\hyperrefenc の指定値。
\ifbxjs@hyperrefenc
  \bxjs@declarebooloption{hyperref-enc}{hyperref@enc}{true}
\else
  \true
\fi

\everyparhook の指定値。
\ifbxjs@everyparhook
  \bxjs@declareenumoption{everyparhook}{everyparhook}{\if j\jsEngine compat\else modern\fi}
\else
  \bxjs@everyparhook@@none
\fi

\labelsection の指定値。
\ifbxjs@labelsection
  \bxjs@declareenumoption{label@section}{label@section}{compat}
\else
  \bxjs@label@section@@none
\fi

\usezw の指定値。
\ifbxjs@usezw
  \bxjs@declarebooloption{use-zw}{usezw}{true}
\else
  \true
\fi

TODO: 3.0 zw/nozw は廃止の予定。
\ifbxjs@disguise@js 〔スイッチ〕 disguise-js の指定値。

TODO: js/nojs は廃止の予定。

\ifbxjs@precisetext 〔スイッチ〕 precise-text の指定値。

\ifbxjs@simplejasetup 〔スイッチ〕 simple-ja-setup の指定値。

\ifbxjs@plautopatch 〔スイッチ〕 plautopatch の指定値。

■オプションの実行

\LaTeX カーネルの 2021/06/01 より前の版では、クラスやパッケージのオプションのトークン列の中に波括弧が含まれると正常に処理ができない。これに対処する為 \ dönemoveelement の実装に少し手を加えて「第 2 引数が空の場合の処理をショートカットする」ことにより、この場合に波括弧を含む第 1 引数が通るようにする。

※ クラスに \ DeclareOption* があり \ OptionNotUsed を使っていない場合は \@unusedoptions は常に空のままであることを利用している。
デフォルトのオプションを実行します。multicols や url を \RequirePackage するのはやめました。

\ExecuteOptions{a4paper,oneside,onecolumn,notitlepage,final}
%<book>\ExecuteOptions{a4paper,twoside,onecolumn,titlepage,openright,final}
%<report>\ExecuteOptions{a4paper,oneside,onecolumn,titlepage,openany,final}
%<slide>\ExecuteOptions{36pt,a4paper,landscape,oneside,onecolumn,titlepage,final}
\ProcessOptions\relax
\bxjs@post@option@hook

後処理
※ landscape の処理のコードは BXJS では無意味なので除外する。

\if@slide
  \def\maybeblue{\@ifundefined{ver@color.sty}{}{\color{blue}}}
\fi
%<*jsclasses>
\if@landscape
  \setlength\@tempdima {\paperheight}
  \setlength\paperheight{\paperwidth}
  \setlength\paperwidth {\@tempdima}
\fi
%</jsclasses>

※グローバルオプションの整理 2021/06/01 より前の版の \LaTeX X カーネルでは、グローバルオプションのトークン列に { } が含まれていると、後のパッケージで \ProcessOptions* がエラーを起こす。従って、このようなオプションは除外することにする。

**TODO** 2021/06/01 版以降のカーネルについてこの処理を廃止する。（仕様変更に準じる扱いとする。）

\def\bxjs@tmpdo{%
  \def\bxjs@tmpa{\@gobble}%
  \expandafter\bxjs@tmpdo@a@\classoptionslist,\@nil,%
  \let\@classoptionslist\bxjs@tmpa}
\def\bxjs@tmpdo@a#1,{%
  \ifx\@nil#1\relax\else
  \bxjs@tmpdo@b#1\@nil%
  \if@tempswa \edef\bxjs@tmpa{\bxjs@tmpa,#1}\fi
  \expandafter\bxjs@tmpdo@a
  \fi}
\def\bxjs@tmpdo@b#1#{
  \ifx\@nil#1\@nil \@tempswatrue \else \@tempswafalse \fi}
\bxjs@tmpdo

papersize、10pt、noscale の各オプションは他のパッケージと衝突を起こす可能性があるため、グローバルオプションから外す。

**TODO** noscale オプションは廃止予定。
使用エンジンの検査・自動判定 デフォルトで現在使われているエンジンが pLaTeX か upLaTeX かを判定します。ユーザによって platex オプションまたは uplatex オプションが明示的に指定されている場合は、実際に使われているエンジンと一致しているかを検査し、一致しない場合はエラーメッセージを表示します。

2016-11-09 pLaTeX/ upLaTeX を自動判別するオプション autodetect-engine を新設しました。upLaTeX の場合は、グローバルオプションに uplatex を追加することで、自動判定に応じて otf パッケージにも uplatex オプションが渡るようにします。

2023-02-12 autodetect-engine 指定時の挙動を規定化しました。また platex を新設しました。オプション autodetect-engine, platex, uplatex のうち最後に指定されたものが有効になります。

正規化前の和文ドライバの値を \bxjs@jadriver に設定する。

エンジン明示指定のオプションが与えられた場合は、それが実際のエンジンと一致するかを検査する。

エンジン指定が autodetect-engine であり、かつ実際のエンジンが (u)pLaTeX だった場合は、本来のエンジンオプションをグローバルオプションに加える。

(この時点で \bxjs@engine は \bxjs@engine@given と同じ規則で分類したコードをもっている。)
エンジンが pTkX-ng の場合、グローバルオプションに uplatex を追加する。

■ドライバ指定 ☃ ドライバ指定のオプションが与えられた場合は、それがエンジンと整合するかを検査する。

DVI 出力のエンジンである場合の追加処理。
ドライバオプションがない場合は警告を出す。
※ただし ja 非指定の場合はスキップする（0.3 版との互換性のため）。

\if\tempswa
\if\fbxs@driver@opt\undefined
\if \(/\exp^{III}\ \text{else}\)\else\if\fbxs@jadriver@opt\undefined \text{F}\else \text{T}\fi\fi \%\ClassWarningNoLine{\fbxs@clsname}
{A driver option is MISSING!!}MessageBreak
You should properly specify one of the valid\MessageBreak
driver options according to the DVI driver\MessageBreak
that is in use:\MessageBreak
\@spaces dvips, dvipdfmx, dviout, xdvi,\MessageBreak
\@spaces nodvidriver}
\fi
\fi

\iff x=XXX が指定されていた場合は、XXX が指定された時と同じ動作にする。（グローバルオプションに XXX を追加する。）
\iff\fbxs@dvi@opt
\edef\fbxs@next{\%\let\noexpand\fbxs@driver@given\csname bxxs@dvidriver@@\fbxs@driver@opt\endcsname\noexpand\g@addto@macro\noexpand\@classoptionslist{,\fbxs@driver@opt}\%\}
\fbxs@next
\fi
\fi

エンジンが pTkX-ng の場合、グローバルオプションに dvipdfmx を追加する。ただし、エンジンオプションが platex-ng*（*付）の場合、および既に dvipdfmx が指定されてい
る場合を除く。
\iff jsWithpTkXng
\iff\fbxs@driver@given\fbxs@driver@0@dvi\iff\let\noexpand\fbxs@driver@given\fbxs@driver@0@dvipdfmx\else\iff\else\iff\else\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\iff\ifi
標準の和文ドライバの名前の定数。

\def\bxjs@@minimal{minimal}
\def\bxjs@@standard{standard}
\def\bxjs@@pandoc{pandoc}
\def\bxjs@@modern{modern}
\bxjs@jadriver の正規化。値が未指定の場合は minimal に変える。ただしエンジンが \(\text{upTeX}\) である場合は standard に変える。
※ (u)pTeX 以外で ja を省略するのは 2.0 版より非推奨となった。

\ifx\bxjs@jadriver\relax
\ifx j\jsEngine
\let\bxjs@jadriver\bxjs@@standard
\else
\ClassWarningNoLine\bxjs@clsname
\{The option 'ja' is MISSING!!\MessageBreak
\} So 'ja=\text{minimal}' is assumed as fallback, but\MessageBreak
\text{such implicit setting is now DEPRECATED!}\MessageBreak
\You should write 'ja=\text{minimal}' explicitly,\MessageBreak
\text{if it is intended}\}
\let\bxjs@jadriver\bxjs@@minimal
\fi
\fi
\fi

\text{plautopatch が真の場合はここで plautopatch を読み込む。}
※この時点で既に読み込まれているパッケージは、calc、keyval、iftex。
※ Pandoc モードでは plautopatch の既定値を真とする。
\ifx\bxjs@jadriver\bxjs@@pandoc \ifx\bxjs@plautopatch@given\@undefined
\ifjsWitheTeX
\bxjs@plautopatchtrue
\fi\fi\fi\fi
\fi
\else
\requirepackage{plautopatch}[2018/08/22]\%v0.3
\fi

エンジンオプションがない場合はエラーを出す。
※ただし ja 非指定の場合はスクリッブする。
\ifx\bxjs@jadriver@opt\@undefined\else
\ifx j\jsEngine \ifbxs@plautopatch
\\RequirePackage{plautopatch}[2018/08/22]\%v0.3
\fi\fi\fi

新しい\text{LuaTeX}（0.87 版以降）では mag がアレなので、magstyle=usemag が指定されて
いた場合はエラーを出す。（この場合の既定値は nomag* であり、エラーの場合は既定値に
置き換えられる。）
ClassError{The engine does not support 'magstyle=usemag'}%\MessageBreak
LuaTeX v0.87 or later no longer supports the "mag" feature of TeX.\MessageBreak
The default value 'nomag*' is used instead.\MessageBreak
@ehc}
\fi
\fi

base、jbase、scale の値を用いて和文スケール値を解決する。
※\bxjs@param@basefontsize と \jsScale へのオプション値の反映は既に実施されていることに注意。jbase 非指定の場合はこのままでよい。
\if\bxjs@jbase@opt@undefined\else
jbase 指定済で base 未指定の場合は、\jsScale の値を採用して和文基底サイズを決定する。
\jsSetQHLength@tempdima{\bxjs@jbase@opt}\
\bxjs@invscale@tempdima\jsScale
\bxjs@setbasefontsize@tempdima\
\else
jbase と base がともに指定済の場合は、それらの値から和文スケール値を決定する。
\if\bxjs@basefontsize@undefined\else
\ClassWarningNoLine{Redundant 'scale' option is ignored}\
\fi
\jsSetQHLength@tempdima{\bxjs@jbase@opt}\
@tempdimb=\bxjs@param@basefontsize\relax
\edef\jsScale{\strip@pt@tempdimb}\
\bxjs@invscale@tempdima\jsScale
\edef\jsScale{\strip@pt@tempdima}\
\fi
\fi
\Cjascale 和文クラス共通仕様（※ただし ZR 氏提唱）における、和文スケール値の変数。
\let\Cjascale\jsScale

disguise-js=true 指定時は、jsarticle（または jsbook）クラスを読み込むように振舞う。
※「2 つのクラスを読み込んだ状態」は \LoadClass を使用した場合に出現するので、別に異常ではない。
\if\bxjs@disguise@js
%<book|report>\def\bxjs@js@clsname{jsbook}
%<!book&!report>\def\bxjs@js@clsname{jsarticle}
@namedef{ver@\bxjs@js@clsname.cls}2001/01/01 (bxjs)
\fi

color/graphics パッケージが持つ出力用紙サイズ設定の機能は、BXJS クラスでは余計なので無効にしておく。このため、グローバルで noersetpagesize を設定しておく。
oldfontcommands オプション指定時は \allowoldfontcommands 命令を実行する。

\ifbxjs@oldfontcommands
  \AtEndOfClass{\allowoldfontcommands}
\fi

■ papersize スペシャルの出力  dvi ファイルの先頭に dvips の papersize special を書き込むことで、出力用紙サイズを設定します。これは dvipdfmx や最近の dviout にも有効です。どうやら papersize special には true 付の単位は許されず、かつ単位は常に true なものと扱われるようです。そこで、後で出てくる（☆）の部分、「\mag にあわせてスケール」よりも手前で実行しておくことになります。

トンボの付いたときの用紙サイズは無意味ですが、いわゆる「ノビ」サイズという縦横 1 インチずつ長い用紙に出力することを考え、1 インチずつ加えました。ところが LaTeX2ε はトンボ出力幅を両側に 1 インチとっていますので、dvips 使用時に

```
-0.5in -0.5in
```
というオプションを与えて両側 0.5 インチのトンボにするといいでしょう。

2003-05-17  トンボをプレビューに使うことを考えて 1 インチを 2 インチにしました。
2016-07-11  memo 歴史のマニュアルによると、トンボを含めた用紙の寸法は \stockwidth, \stockheight と呼ぶようですので、これを使うことにしました。
2017-01-11  テンポオプションが指定されているとき「だけ」\stockwidth, \stockheight を定義するようになりました。
2020-04-16  EEPX2ε 2020-10-01 でカーネルの \shipout コードが拡張され
\AtBeginDvi の実行タイミングが変化したので、この時点で発行する \special の中身を展開しておくようにしました。こうしないと、用紙サイズ設定を間違っててしまいます (Issue #72)。
2022-09-12  次期 EEPX2ε カーネルに\stockwidth, \stockheight が追加されるよう

BXJS では出力用紙サイズ記録は geometry パッケージが行う。
また、JS クラスと異なり、\stockwidth, \stockheight は常に定義される。
■基準となる行送り

基準となる行送りをポイント単位で表したものです。

拡大率の設定

■拡大率の設定

サイズの変更は \TeX のプリミティブ \texttt{mag} を使って行います。9 ポイントについては行送りも若干縮めました。サイズについては全面的に見直しました。

・ geometry 側でオプション \texttt{truedimen} を指定してください。
・ geometry 側でオプション \texttt{mag} は使えない。

設定すべき \texttt{mag} 値を (基底サイズ)/(10pt) × 1000 と算出。BXJS クラスでは、\texttt{mag} を直接指定したい場合は、geometry 側ではなくクラスのオプションで行うものとする。
993 \def\bxjs@tmpa#1#2#3#4\@nil{\@tempdima=#2#3.#4\p@}
994 \expandafter\bxjs@tmpa\the\@tempcnta\@nil
995 \edef\jsc@magscale{\strip@pt\@tempdima}
996 \let\jsBaseFontSize\bxjs@param@basefontsize

[2016-07-08] \jsc@mpt および \jsc@mmm に、それぞれ 1pt および 1mm を拡大させた値を格納します。以降のレイアウト指定ではこちらを使います。

※ 2.9 版において \p@? 表記を廃止。
997 \newdimen\jsc@mpt
998 \newdimen\jsc@mmm
999 \ifjsc@mag
1000 \jsc@mpt=1\p@
1001 \jsc@mmm=1mm
1002 \else
1003 \jsc@mpt=\jsc@magscale\p@
1004 \jsc@mmm=\jsc@magscale mm
1005 \fi

ここで pTEX の zw に相当する単位として用いる長さ変数 \jsZw を作成する。約束により、これは \jsScale \times (指定フォントサイズ) に等しい。

use-zw が真の時は zw を \jsZw と同義にする。
1006 \newdimen\jsZw
1007 \jsZw=10\jsc@mpt \jsZw=\jsScale\jsZw
1008 \ifbxjs@usezw
1009 \providecommand*{\zw}{\jsZw}
1010 \fi

\zwspace 全角幅の水平空き。
1011 \def\zwspace{\hskip\jsZw\relax}

そして、magstyle が nomag* の場合は、NFSS にパッチを当てる。
1012 \ifjsc@mag@xreal
1013 \RequirePackage{type1cm}
1014 \let\jsc@get@external@font\get@external@font
1015 \def\get@external@font{\jsc@preadjust@extract@font\get@external@font}
1016 \def\jsc@fstrunc#1{\textwidth}
1017 \ifbxjs@TUenc
1018 \expandafter\let\csname TU/lmr/m/n/10\endcsname\relax
1019 \else
1020 \expandafter\let\csname OT1/cmr/m/n/10\endcsname\relax
1021 \let\csname OMX/cmex/m/n/10\endcsname\relax
1022 \expandafter\let\csname OMX/cmex/m/n/10\endcsname\relax
1023 \def\get@external@font{%
1024 \jsc@get@external@font\get@external@font}
1025 \def\jsc@fstrunc#1{%

42
\edef\jsc@tmpa{\strip@pt#1}\%  
\expandafter\jsc@fstrunc@a\jsc@tmpa.****\@nil\%  
\def\jsc@fstrunc@a#1.#2#3#4#5#6\@nil{\%  
\if#5*\else\%  
\edef\jsc@tmpa{#1\%  
\ifnum#2#3>\z@ .#2\ifnum#3>\z@ #3\fi\fi\}%  
\fi\}%  
\def\jsc@preadjust@extract@font{\%  
\let\jsc@req@size\f@size\%  
\dimen@\f@size\p@ \jsc@invscale\dimen@\jsc@magscale\%  
\advance\dimen@.005pt\relax \jsc@fstrunc\dimen@\%  
\let\jsc@ref@size\jsc@tmpa\%  
\let\f@size\jsc@ref@size\%  
\def\execute@size@function#1{\%  
\let\jsc@cref@size\f@size\%  
\let\f@size\jsc@req@size\%  
\csname s@fct@#1\endcsname\}%  
\let\jsc@DeclareErrorFont\DeclareErrorFont\%  
\def\ DeclareErrorFont#1#2#3#4#5{\%  
\@tempdimc#5\p@ \@tempdimc\jsc@magscale\@tempdimc\%  
\edef\jsc@tmpa{{#1}{#2}{#3}{#4}{\strip@pt\@tempdimc}}\%  
\expandafter\jsc@DeclareErrorFont\jsc@tmpa\}%  
\def\gen@sfcnt{\%  
\edef\mandatory@arg{\mandatory@arg\jsc@cref@size}\%  
\empty@sfcnt\}%  
\def\genb@sfcnt{\%  
\edef\mandatory@arg{%  
\mandatory@arg\expandafter\genb@x\jsc@cref@size..\@@}\%  
\empty@sfcnt\}%  
\ifbxjs@TUenc\else\%  
\DeclareErrorFont{OT1}{cmr}{m}{n}{10}\%  
\fi\%  
\fi\%  

[2016-11-16] latex.ltx (ltspace.dtx) で定義されている smallskip の単位 pt を \jsc@mpt に置き換えた smallskip を定義します。これは \maketitle で用いられます。medskip と bigskip は必要ないのでコメントアウトしています。

\def\jsc@smallskip{\vspace\jsc@smallskipamount}\%  
%\def\jsc@medskip{\vspace\jsc@medskipamount}\%  
%\def\jsc@bigskip{\vspace\jsc@bigskipamount}\%  
\newlength{\smallskipamount}\%  
\smallskipamount=3\m@pt plus 1\m@pt minus 1\m@pt\%  
%\newlength{\medskipamount}\%  
%\medskipamount =6\m@pt plus 2\m@pt minus 2\m@pt\%  
%\newlength{\bigskipamount}\%  
%\bigskipamount =10\m@pt plus 4\m@pt minus 4\m@pt\%  

%\def\smallskip{\vspace{3\m@pt plus 1\m@pt minus 1\m@pt}}\%  
%\def\medskip{\vspace{6\m@pt plus 2\m@pt minus 2\m@pt}}\%  
%\def\bigskip{\vspace{10\m@pt plus 4\m@pt minus 4\m@pt}}\%  

%\def\smallskip{\vspace{3\m@pt plus 1\m@pt minus 1\m@pt}}\%  
%\def\medskip{\vspace{6\m@pt plus 2\m@pt minus 2\m@pt}}\%  
%\def\bigskip{\vspace{10\m@pt plus 4\m@pt minus 4\m@pt}}\%  

\newskip\smallskipamount\%  
\smallskipamount=3\m@pt plus 1\m@pt minus 1\m@pt\%  
\newskip\medskipamount\%  
\medskipamount =6\m@pt plus 2\m@pt minus 2\m@pt\%  
\newskip\bigskipamount\%  
\bigskipamount =10\m@pt plus 4\m@pt minus 4\m@pt\%  

1067 \% jsc@bigskipamoun =12\jsc@mpt plus 4\jsc@mpt minus 4\jsc@mpt 
\paperwidth, \paperheight を \mag にあわせてスケールしておきます（☆）。
[2016-07-11] 新しく追加した \stockwidth, \stockheight も \mag にあわせてスケールします。
[2017-01-11] トンボオプションが指定されているとき「だけ」\stockwidth, \stockheight が定義されています。

■ pagesize スペシャルの出力 [2003-05-17] dvipdfm(x) の pagesize スペシャルを出力します。
[2004-08-08] 今の dvipdfmx は dvips 用スペシャルを理解するようなので外しました。

\%\ifpapersize
\setlength{\@tempdima}{\paperwidth}
\setlength{\@tempdimb}{\paperheight}
\iftombow
\advance \@tempdima 2truein
\advance \@tempdimb 2truein
\fi
\AtBeginDvi{\special{pdf: pagesize width \the\@tempdima space height \the\@tempdimb}}
\fi

3 和文フォントの変更

和文フォントの設定は和文ドライバの管轄。
ここでは、jsclasse.dtx との差分を抑制するために、オリジナルのコードを無効化した
状態で挿入しておく。

\%</class>
\%</jsclasse>
\%</class>

JIS の 1 ポイントは 0.3514mm（約 1/72.28 インチ）、PostScript の 1 ポイントは 1/72 インチですが、\TeX では 1/72.27 インチを 1pt（ポイント）、1/72 インチを 1bp（ビッグポイン
ト）と表します。QuarkXPress などの DTP ソフトは標準で 1/72 インチを 1 ポイント
としますが、以下ではすべて 1/72.27 インチを 1pt としています。1 インチは定義により
25.4mm です。
さらにややこしいことに、\pTeX（アスキーが日本語化した \TeX）の公称 10 ポイントの和
文フォント（min10 など）は、実寸（標準の字送り量）が 9.62216pt です。これは 3.3818mm,
写研の写植機の単位では 13.527 級、PostScript の単位では 9.5862 ポイントになります。
jis フォントなどもこの値を踏襲しています。
この公称 10 ポイントのフォントを、ここでは 13 級に縮小して使うことにします。そのた
ためには、13/13.527 = 0.961 倍すればいいことになります（min10 や jis の場合）。9.62216
ポイントの和文フォントをさらに0.961倍したことにより，約9.25ポイント，DTPで使う単位（1/72インチ）では9.21ポイントということになり，公称10ポイントといっても実は9ポイント強になります。

[2018-02-04] 上記のとおりの「クラスファイルが意図する和文スケール値（1 zw ÷要求サイズ）を表示する実数値マクロ \Cjascale を定義します。このマクロが定義されている場合，OTF パッケージ（2018/02/01以降のバージョン）はこれに従います。jsarticle，jsbook，jsreport では，9.62216pt*0.961/10pt = 0.924690 です。
某学会誌では、和文フォントを PostScript の 9 ポイントにするために、$9/(9.62216 \times 72/72.27) = 0.93885$ 倍します。

[2018-02-04] 和文スケール値 \Cjascale は 9.62216pt × 0.93885/10 pt = 0.903375 です。

和文でイタリック体、斜体、サンセリフ体、タイプライタ体の代わりにゴシック体を使うことにします。

[2003-03-16] イタリック体、斜体について、和文でゴシックを当てていましたが、数学の定理環境などで多量のイタリック体を使うことがあり、ゴシックにすると黒黒となってしまうという弊害がありました。amsthm を使わない場合は定理の本文が明朝になるように \newtheorem 環境を手直しをしてのいていましたが、\TeX が数学で多用されることを考えると、イタリック体に明朝体を当てたほうがいいように思えてきましたので、イタリック体・斜体に対応する和文を明朝体に変えることにしました。
も和文対応にしました。

\begin{verbatim}
[2004-11-03] \rmfamily も和文対応にしました。
1154 % \DeclareFontShape{\jsc@JYn}{mc}{bx}{n}{<->ssub*gt/m/n}{% in \jsc@JYmmc
1155 % \DeclareFontShape{\jsc@JYn}{gt}{bx}{n}{<->ssub*gt/m/n}{% in \jsc@JYngt
1156 \DeclareFontShape{\jsc@JYn}{mc}{m}{it}{<->ssub*mc/m/n}{%
1157 \DeclareFontShape{\jsc@JYn}{mc}{m}{sl}{<->ssub*mc/m/n}{%
1158 \DeclareFontShape{\jsc@JYn}{mc}{m}{sc}{<->ssub*mc/m/n}{%
1159 \DeclareFontShape{\jsc@JYn}{gt}{m}{it}{<->ssub*gt/m/n}{%
1160 \DeclareFontShape{\jsc@JYn}{gt}{m}{sl}{<->ssub*gt/m/n}{%
1161 \DeclareFontShape{\jsc@JYn}{mc}{bx}{it}{<->ssub*gt/m/n}{%
1162 \DeclareFontShape{\jsc@JYn}{mc}{bx}{sl}{<->ssub*gt/m/n}{%
1163 % \DeclareFontShape{\jsc@JTn}{mc}{bx}{n}{<->ssub*gt/m/n}{} % in \jsc@JTnmc
1164 % \DeclareFontShape{\jsc@JTn}{gt}{bx}{n}{<->ssub*gt/m/n}{} % in \jsc@JTngt
1165 \DeclareFontShape{\jsc@JTn}{mc}{m}{it}{<->ssub*mc/m/n}{%
1166 \DeclareFontShape{\jsc@JTn}{mc}{m}{sl}{<->ssub*mc/m/n}{%
1167 \DeclareFontShape{\jsc@JTn}{mc}{m}{sc}{<->ssub*mc/m/n}{%
1168 \DeclareFontShape{\jsc@JTn}{gt}{m}{it}{<->ssub*gt/m/n}{%
1169 \DeclareFontShape{\jsc@JTn}{gt}{m}{sl}{<->ssub*gt/m/n}{%
1170 \DeclareFontShape{\jsc@JTn}{mc}{bx}{it}{<->ssub*gt/m/n}{%
1171 \DeclareFontShape{\jsc@JTn}{mc}{bx}{sl}{<->ssub*gt/m/n}{%
1172 %\fi %restart

[2002-02-02] \LaTeX \varepsilon 2020-02-02 で NFSS が拡張され、それに伴いオリジナルの
\rmfamily などの定義が変化しました。\DeclareRobustCommand で直接定義すると、
これを上書きして NFSS の拡張部分を壊してしまうので、新たに提供されたフックに
コードを挿入します。従来のコードも \LaTeX \varepsilon 2019-10-01 以前のために残してありますが、
mweights パッケージ対策も施しました (forum:2763)。

[2020-10-04] \LaTeX \varepsilon 2020-10-01 では \AddToHook を利用します。

\end{verbatim}

次のコマンドはイタリック補正なども含めて定義されていますが、和文ではイタリック補正はあまり役に立たず、欧文・和文間のグルーが入らないという副作用もありますので、単純な定義に直します。

[2016-08-26] 和欧文間の \xkanjiskip が入らない問題は、plfonts.dtx v1.3i (2000/07/13) の時点で修正されていました。逆に、amsmath パッケージを読み込んだ場合に、数式内の添字で文字サイズが変化するようになるはずのところが、変わらなくなっていましたので、修正しました。

[2017-09-03] Yue ZHANG さん作の fixjfm パッケージが \documentclass より前に \RequirePackage{fixjfm} として読み込まれていた場合には、その定義を優先するため、このクラスファイルでは再定義しません。

[2017-09-19] 2010 年の pTeX の修正で、イタリック補正と和欧文間の \xkanjiskip の衝突が起きなくなっていますから、もうここにあるような単純化は必要ありません。ただし、このクラスファイルが古い TeX 環境で利用される可能性も捨てきれないので、とりあえず残しておきます。

\textmc 次のコマンドはイタリック補正なども含めて定義されていますが、和文ではイタリック補正はあまり役に立たず、欧文・和文間のグルーが入らないという副作用もありますので、単純な定義に直します。

[2016-08-26] 和欧文間の \xkanjiskip が入らない問題は、plfonts.dtx v1.3i (2000/07/13) の時点で修正されていました。逆に、amsmath パッケージを読み込んだ場合に、数式内の添字で文字サイズが変化するようになるはずのところが、変わらなくなっていましたので、修正しました。

[2017-09-03] Yue ZHANG さん作の fixjfm パッケージが \documentclass より前に \RequirePackage{fixjfm} として読み込まれていた場合には、その定義を優先するため、このクラスファイルでは再定義しません。

[2017-09-19] 2010 年の pTeX の修正で、イタリック補正と和欧文間の \xkanjiskip の衝突が起きなくなっていますから、もうここにあるような単純化は必要ありません。ただし、このクラスファイルが古い TeX 環境で利用される可能性も捨てきれないので、とりあえず残しておきます。

\textgt 新クラスでも disablejfam オプションを与えなければ数式内で日本語が使えるようにし
ました。
さらに 2005/12/01 版の LaTeX に対応した pLaTeX に対応しました（Thanks: ymt さん）。
1223 \def\reDeclareMathAlphabet{\mathbb{#1}}{\mathbb{#2}}{\mathbb{#3}}{\string @\expandafter\@gobbletwo\string #4}{}
1224 \ifx\expandafter\string #5\expandafter\string #6\expandafter\@gobble\expandafter\string #7\expandafter\@gobbletwo\string #8\expandafter\@gobble\expandafter\string #9\expandafter\@gobble\expandafter\string #10\expandafter\@gobbletwo\string #11\expandafter\@gobble\expandafter\string #12\expandafter\@gobble\expandafter\string #13\expandafter\@gobbletwo\string #14}
1225 \begin{group}
1226 \let\protect\noexpand\relax
1227 \edef\tempaa{\relax}
1228 \edef\tempbb{\relax}
1229 \edef\tempcc{\tempaa\tempbb}
1230 \expandafter\endgroup\tempcc
1231 \edef\reDeclareMathAlphabet{\mathbb{#1}}{\mathbb{#2}}{\mathbb{#3}}{\string @\expandafter\@gobbletwo\string #4}{}
1232 \def\DualLang@mathalph@bet#1#2{\relax\ifmmode\ifx\math@bgroup\bgroup% 2e normal style (\textstyle{...})
1233 \bgroup\let\DualLang@Mfontsw\DLMfontsw@standard
1234 \else
1235 \ifx\math@bgroup\relax% 2e two letter style (\text{...})
1236 \let\DualLang@Mfontsw\DLMfontsw@oldstyle
1237 \else
1238 \ifx\math@bgroup\empty% 2.09 oldlfont style (\text ...)}\end{latex}\end{document}
これは \textsterling 命令で実際に呼び出される文字です。従来からの OT1 エンコーディングでは \$ のイタリック体が \textsterling なので cmti が使われていましたが、1994 年春からは cmu (upright italic, 直立イタリック体) に変わりました。しかし cmu はその性格からして実験的なものであり、\textsterling 以外で使われることは思えないので、ここでは cmti に戻してしまいます。

[2003-08-20] Computer Modern フォントを使う機会も減り、T1 エンコーディングが一般的になってきました。この定義はもうあまり意味がないので消します。

禁則パラメータも若干修正します。
アスキーの kinsoku.dtx では次の三つが 5000 に設定されています。これを 10000 に再設定します。

\prebreakpenalty\jis"2147=10000 \% 5000 '  
\postbreakpenalty\jis"2148=10000 \% 5000 "  
\prebreakpenalty\jis"2149=10000 \% 5000 ''

「TeX」「〒515」の記号と数字の間に四分アキが入らないようにします。

\inhibitxspcode`!=1  
\inhibitxspcode`!=2

以前の版では、たとえば「ベース名.拡張子」のように和文文字で書いたとき、ビリオドの後に四分アキが入らないようにするため
のようにしていました。ただ、「Foo Inc. は……」のように書いたときにもスペースが入らなくなるので、ちょっとまずい修正だったかもしれません。元に戻しました。
とりあえず「ベース名, \mbox{拡張子}」と書いてください。
「C や C++ では……」と書くと、C++ の直後に四分アキが入らないのでバランスが悪くなります。四分アキが入るようにしました。% の両側も同じです。

これ以外に T1 エンコーディングで 80 〜 ff の文字もすべて欧文文字ですので、両側の和文文字との間にスペースが入らなければなりません。
欧文といえば、LaTeX の `\def\@{\spacefactor\@m}` という定義 (\@m は 1000) では I watch TV.\@. と書くと V とピリオドのペアカーニングが効かなくなります。そこで、次のような定義に直し、I watch TV.\@. と書くことにします。
BXJS クラスでの変更点：

- fix-at-cmd オプションが偽の場合は再定義しない。
- 固定の 3000 でなく実際のピリオドの sfcode 値を使う。
- 「防御的な \@」での不具合を防ぐため、大文字直後の \@ は標準と同等の動作にする。

4 フォントサイズ

フォントサイズを変える命令（\normalsize, \small など）の実際の挙動の設定は、三つの引数をとる命令 \setfontsize を使って、たとえば

\setfontsize{\normalsize}{10}{16}

のようにして行います。これは

\normalsize は 10 ポイントのフォントを使い、行送りは 16 ポイントである

という意味です。ただし、処理を速くするため、以下では 10 と同義の \LaTeX の内部命令 \@xpt を使っています。この \@xpt の類は次のものがあり、\LaTeX 本体で定義されています。

\@vpt 5 \@vipt 6 \@viipt 7
\@viipt 8 \@vipt 9 \@xpt 10
\@xipt 10.95 \@xiipt 12 \@xiiipt 14.4

ここでは \setfontsize の定義を少々変更して、段落の字下げ \parindent, 和文文字間のスペース \kanjiskip, 和文・欧文間のスペース \xkanjiskip を変更しています。

\kanjiskip は \LaTeX で Opt plus .4pt minus .5pt に設定していますが、これはそもそも文字サイズの変更に応じて変わるべきものです。それに、プラスになったりマイナスになったりするのには、追い出しと追い込みの混在が生じ、統一性を欠きます。なるべく追い出しになるようにプラスの値だけにしたいところですが、ごくわずかなマイナスは許すことにしました。
\kankanjiskip については、四分つまり全角の 1/4 を標準として、追い出すために三分あるいは二分まで延長するのが一般的ですが、ここでは Times や Palatino のスペースがほぼ四分であることに着目して、これに一致させています。これなら書くときにスペースを空けても空けなくても同じ出力になります。
\parindent については、0（以下）でなければ全角幅（lzw）に直します。
[2008-02-18] english オプションで \parindent を 1em にしました。

\fontsize 命令（\large 等でなく）でフォントサイズ変更した場合にもフックが実行されるように、@setFontsize ではなく set@fontsize に対してバッチを当てるように変更。

\bxjs@patch@set@fontsize \set@fontsize にバッチを当てる。
※\set@fonts で書き換えるパッケージへの対策のため、クラス読み込中に複数回実行する。前回の実行直後から \set@fontsize が更新されている場合にのみ実際にバッチを当てる。

TODO\[3.0\] 新しい \LaTeX X カーネルで selectfont フックを利用する。

\bxjs@patch@set@fontsiz

\jsFontSizeChanged フォントサイズ変更時に呼ばれるフック。\jsZw を再設定している。その後でユーザ定義用のフック \jsResetDimen を実行する。
ユーザ定義用のフック。
\providecommand*{\jsResetDimen}{}

クラスファイルの内部では、拡大率も考慮した\jsc@setfontsize を\@setfontsize の変わりに用いることにします。

\@setfontsize\def\jsc@setfontsize#1#2#3{%\@setfontsize#1{#2\jsc@mpt}{#3\jsc@mpt}}

\ifjs@mag\let\jsc@setfontsize\@setfontsize\else
\def\jsc@setfontsize#1#2#3{%\edef\bxjs@sfs@next{%\unexpanded{\@setfontsize#1}{\the\dimexpr#2\jsc@mpt\relax}{\the\dimexpr#3\jsc@mpt\relax}}\bxjs@sfs@next}
\fi\fi

これらのグルーをもってしても行分割ができない場合は、\emergencystretch に訴えます。

これはフォントサイズ非依存なので\Cwd で書くのが適当だが、\Cwd はまだ定義されていない。

\emergencystretch 3\jsZw

欧文用に行間を狭くする論理変数と、それを真・偽にするためのコマンドです。

\narrowbaselines [2003-06-30] 数式に入るとここで\narrowbaselines を実行しているので\n\abovedisplayskip 等が初期化されてしまうという shintok さんのご指摘に対して、しっかり愛好家さんが次の修正を教えてくださいました。
\n2008-02-18 english オプションで最初の段落のインデントをしないようにしました。

TODO: Hasumi さん [qa:54539] のご指摘は考慮中です。

別行立て数式に入るときに\narrowbaselines が呼ばれるが、このコードでは「数式中で\normalsize などのサイズ命令（\@currsiz 体）が呼ばれた」となり警告が出る。JS クラスでは、\@setfontsize 中の\@nomath 実行を消して「そもそもサイズ命令で警告が出ない」ようにしている。警告が常に出ないので望ましくないので、BXJS クラスの実装では、\narrowbaselines の時だけ警告が出ないようにする。

\newif{\ifnarrowbaselines}
\ife@english

56
microtype パッケージを読み込んだ場合、\normalsize 等のフォントサイズ変更命令の定義の中に if 文が使われていると、不可解なエラーが発生する。これは microtype が邪悪なトリックを使用しているせいなのだが、一応こちら側で対策をとることにする。

\bxjs@if@narrowbaselines スイッチ narrowbaselines を \TeX 式条件文にしたもの。

\normalsize 標準のフォントサイズと行送りを選ぶコマンドです。

本文 10 ポイントのときの行送りは、欧文の標準クラスファイルでは 12 ポイント、アジアの和文クラスファイルでは 15 ポイントになっていますが、ここでは 16 ポイントにしました。ただし narrowbaselines で欧文用の 12 ポイントになります。

公称 10 ポイントの和文フォントが約 9.25 ポイント（アジアのもののが 0.961 倍）であることもあり、行送りがかなりゆったりとしたと思います。実際、16/9.25 ≈ 1.73 であり、和文の推奨値の一つ「二分四分」（1.75）に近づきました。

microtype 対策のため if 文を避ける。後の \small、\footnotesize も同様。
数式の上のアキ（\abovedisplayskip）、短い数式の上のアキ（\abovedisplayshortskip）、数式の下のアキ（\belowdisplayshortskip）の設定です。

[2003-02-16] ちょっと変えました。

[2009-08-26] \TeX{} Q & A 52569 から始まる議論について読んでいましたが、結局、微調整してみることにしました。

\abovedisplayskip 11\jscp@pt \@plus3\jscp@pt \@minus4\jscp@pt
\abovedisplayshortskip \z@ \@plus3\jscp@pt
\belowdisplayskip 9\jscp@pt \@plus\jscp@pt \@minus4\jscp@pt
\belowdisplayshortskip \belowdisplayskip

最後に、リスト環境のトップレベルのパラメータ \@listI を、\@listi にコピーしておきます。\@listI の設定は後で出てきます。

\let\@listi\@listI}

ここで実際に標準フォントサイズで初期化します。

%</class><
%

%<class|minijs>
%% initialize
%
%<class>

\Cht 基準となる長さの設定をします。\pTeX{} 2 でカーネル（plfonts.dtx）で宣言されているパラメータに実際の値を設定します。たとえば \Cwd は \normalfont の全角幅（1zw）です。

\Cwd [2017-08-31] 基準とする文字を「全角空白」（EUC コード 0xA1A1）から「漢」（JIS コード 0x3441）へ変更しました。

\Chs

\Cwd 等の変数は \pTeX{} 系以外では未定義なのでここで定義する。

\ifx\Chs\@undefined \newdimen\Chs \fi
\ifx\Cdp\@undefined \newdimen\Cdp \fi
\ifx\Cwd\@undefined \newdimen\Cwd \fi
\ifx\Cvs\@undefined \newdimen\Cvs \fi
\ifx\Chs\@undefined \newdimen\Chs \fi

規約上、現在の \jsZw の値が \Cwd である。BXJS では \Chs と \Cdp は単純に \Cwd の 88\% と 12\% の値とする。

\setlength\Chs{0.88\jsZw}
\setlength\Cdp{0.12\jsZw}
\setlength\Cwd{1\jsZw}
\setlength\Cvs{\baselineskip}
\setlength\Chs{1\jsZw}

\small \small も \normalsize と同様に設定します。行送りは、\normalsize が 16 ポイントなら、割合からすれば 16 \times 0.9 = 14.4 ポイントになりますが、\small の使い方が考えて、
ここでは和文 13 ポイント、欧文 11 ポイントとします。また、\texttt{topsep} と \texttt{parsep} は、元はそれぞれ 4 ± 2, 2 ± 1 ポイントでしたが、ここではゼロ（\texttt{z@}）にしました。

\begin{verbatim}
\newcommand{\small}{\texttt{\@ixpt{11}}

%<kiyou> \texttt{\@ixpt{8.8888}}%}

\def\@listi{
\itemsep \parsep
\itemsep \parsep}
\end{verbatim}

\texttt{\@viiipt{9.5}} を追加。

\begin{verbatim}
\footnotesize
\footnotesize
\scriptsize
\tiny
\large
\Large
\Huge
\HUGE
\end{verbatim}

\texttt{\@viiipt{11}} を追加。
別行立て数式の中では \narrowbaselines にします。和文の行送りのまでは、行列や場合分けの行送り、連分数の高さなどが不釣合いに大きくなるためです。

本文中の数式の中では \narrowbaselines にしていません。本文中ではなるべく行送りが変わるように大きいものを使わず、行列は amsmath の smallmatrix 環境を使うのがいいでしょう。

しかし、このおかげで別行数式の上下のスペースが少し違ってしまいしました。とりあえず amsmath の equation 関係は okumacro のほうで逃げていますが、もっとかめ逃げ道があればお教えください。

見出し用のフォントは \bfseries 固定ではなく、\headfont という命令で定めることにします。これは太ゴシックが使えるときは \sffamily \bfseries でいいと思いますが、通常の中ゴシックでは単に \sffamily だけのほうがよさそうです。『pLaTeX 美文書作成入門』（1997 年）では \sffamily \fontseries{sbc} として新ゴ M と合わせましたが、\fontseries{sbc} はちょっと幅が狭いように感じました。

5 レイアウト

■二段組

\columnsep \columnsep は二段組のときの左右の段間の幅です。元は 10pt でしたが、2zw にしました。

\columnseprule このスペースの中央に \columnseprule の幅の罫線が引かれます。

■段落

\lineskip 上下の行の文字が \lineskiplimit より接近したら、\lineskip より近づかないようにし、\normallineskip をします。元は 0pt でしたが 1pt に変更しました。normal... の付いた方は保存用です。

\lineskip
\normallineskip
\normallineskiplimit
ベースラインストレッチ 実際の行送りが \baselineskip の何倍かを表すマクロです。たとえば

\renewcommand{\baselinestretch}{2}

とすると、行送りが通常の 2 倍になります。ただし、これを設定すると、たとえ \baselineskip が伸縮するように設定しても、行送りの伸縮ができなくなります。行送りの伸縮はしないのが一般的です。

\renewcommand{\baselinestretch}{1}

\parskip \parskip は段落間の追加スペースです。元は 0pt plus 1pt になっていましたが、ここではゼロにしました。\parindent は段落の先頭の字下げ幅です。

\setlength{\parskip}{0pt}
\if@slide
\setlength{\parindent}{0pt}
\else
\setlength{\parindent}{1\Cwd}
\fi

\@lowpenalty \nopagebreak, \nolinebreak は引数に応じて次のペナルティ値のうちどれかを選ぶようになっています。ここはオリジナル通りです。

\@lowpenalty 51
\@medpenalty 151
\@highpenalty 301

\interlinepenalty 段落中の改ページのペナルティです。デフォルトは 0 です。
\brokenpenalty ページの最後の行がハイフンで終わる際のペナルティです。デフォルトは 100 です。

5.1 ページレイアウト

BXJS ではページレイアウトの処理は geometry パッケージが担当している。

■準備

\begin{document} フックのコード内で、geometry パッケージが挿入するコードの直前で実行されるフック。
\end{document}
現状ではここで \texttt{mag} を設定している。
\texttt{topskip} も指定する。
\begin{verbatim}
\ifjsc@mag
\texttt{mag=\bxjs@param\@mag}
\fi
\end{verbatim}
\begin{verbatim}
\setlength{\topskip}{10\textsc{mpt}}
\end{verbatim}
\begin{verbatim}
\textsc{jsSetQHLength} のための和文単位の定義。
\end{verbatim}
\begin{verbatim}
\def\bxjs@unit@trueQ{0.25true\text{mm}}\let\bxjs@unit@trueH\bxjs@unit@trueQ
\def\bxjs@unit@zw{\textsc{Zw}}\let\bxjs@unit@zh\bxjs@unit@zw
\end{verbatim}
\begin{verbatim}
\bxjs@param\@paper が長さ指定の場合、\texttt{geometry} の形式 (\texttt{papersize=\{W,H\}}) に変換
する。\{W\}-\{H\} の形式について。
\end{verbatim}
\begin{verbatim}
\@tempswafalse
\def\bxjs@tmpdo\@ifnextchar\bgroup\bxjs@tmpdo@a\remove@to@nnil
\def\bxjs@tmpdo@a#1\@ifnextchar\bgroup\bxjs@tmpdo@b\remove@to@nnil
\def\bxjs@tmpdo@b#1\@ifnextchar\@nnil\bxjs@tmpdo@c\remove@to@nnil
\def\bxjs@tmpdo@c\@nnil{\@tempswatrue
\edef\bxjs@param\@paper{papersize={\bxjs@tmpa}}}
\expandafter\bxjs@tmpdo\bxjs@param\@paper\@nnil
\end{verbatim}
\begin{verbatim}
\texttt{W,H} の形式について。
\end{verbatim}
\begin{verbatim}
\if@tempswafalse
\def\bxjs@tmpa\@nil,\@nil
\def\bxjs@tmpdo#1,#2,#3\@nnil{\def\bxjs@tmpb{#3}\ifx\bxjs@tmpa\bxjs@tmpb
\@tempswatrue\edef\bxjs@param\@paper{papersize={#1,#2}}\fi}
\expandafter\bxjs@tmpdo\bxjs@param\@paper,\@nil,\@nil\@nnil
\fi
\end{verbatim}
\begin{verbatim}
\texttt{W*H} の形式について。
\end{verbatim}
\begin{verbatim}
\if@tempswafalse
\def\bxjs@tmpa\@nil*,\@nil
\def\bxjs@tmpdo#1*#2*#3\@nnil{\def\bxjs@tmpb{#3}\ifx\bxjs@tmpa\bxjs@tmpb
\@tempswatrue\edef\bxjs@param\@paper{papersize={#1,#2}}\fi}
\expandafter\bxjs@tmpdo\bxjs@param\@paper*,\@nil*,\@nil\@nnil
\fi
\end{verbatim}
\begin{verbatim}
\texttt{\textsc{bxjs}@layout@paper} \texttt{geometry} の用紙設定のオプション。
\end{verbatim}
\begin{verbatim}
\edef\bxjs@layout@paper{\ifjsc@mag truedimen,\fi}
\end{verbatim}
\begin{verbatim}
\if@landscape landscape,\fi
\bxjs@layout\texttt{geometry} のページレイアウトのオプション列。文書クラス毎に異なる。
\end{verbatim}
\begin{verbatim}
\%<*article|report>
\def\bxjs@layout@base{%
ヘッド・フッタ領域の横幅。

textwidth オプションの設定を反映する。

\fullwidth [寸法レジスタ] ヘッダ・フッタ領域の横幅。
1682 \newdimen\fullwidth

\textwidth\limit における、\textwidth 上限の値。

\jsTextWidthLimit における、\textwidth 上限の値。

1683 \%</book>

\bxjs@preproc@layout  における、\textwidth の前処理。

geometry は \topskip が標準の行高 (\ht\strutbox) より小さくならないようにする
自動調整を行うが、これをどうするかは未検討。今のところ、単純に回避（無効化）して
いる。

1699 \def\bxjs@preproc@layout{%
1700  \edef\bxjs@save@ht@strutbox{\the\ht\strutbox}
1701  \ht\strutbox=10\jsc@mpt}

\bxjs@postproc@layout  における、\textwidth の前処理。

geometry のドライバを再設定する。

\ht\strutbox の値を元に戻す。

\textwidth の値を補正する。
bxjsbook の場合は、geometry が設定した \textwidth は \fullwidth として扱い、その値から実際の \textwidth を導出する。

\fi
\fullwidth=\textwidth

\textwidth 関連の調整。
\@tempdima=\bxjs@textwidth\@limit\relax
\ifbxjs@whole@zw@lines
  \advance\@tempdima.005pt\relax
  \divide\@tempdima\Cwd \multiply\@tempdima\Cwd
\fi
\ifdim\textwidth>\@tempdima
  \textwidth=\@tempdima
  \addtolength\evensidemargin{\fullwidth-\textwidth}
\fi
\%</book>

\textheight 関連の調整。
\@tempdimb=\textheight
\advance\textheight-\topskip
\advance\textheight.005pt\relax
\divide\textheight\baselineskip \multiply\textheight\baselineskip
\advance\textheight\topskip
\advance\@tempdimb-\textheight
\advance\topmargin0.5\@tempdimb
\headheight 関連の調整。
\@tempdima=\topskip
\advance\headheight\@tempdima
\advance\topmargin-\@tempdima
marginpar 関連の調整。
\setlength\marginparsep{\columnsep}
\setlength\marginparpush{\baselineskip}
\setlength\marginparwidth{\paperwidth-\oddsidemargin-1truein% -\textwidth-10\jsc@mmm-\marginparsep}
\ifbxjs@whole@zw@lines
  \divide\marginparwidth\Cwd \multiply\marginparwidth\Cwd
\fi
連動する変数。
\maxdepth=.5\topskip
\stockwidth=\paperwidth
\stockheight=\paperheight
}\jsGeometryOptions

geometry パッケージに渡すオプションのリスト。
※ geometry=user 指定時にユーザが利用することを想定している。
\edef\jsGeometryOptions{%
  \bxjs@layout@paper,\bxjs@layout}
}}
\section*{geometry パッケージを読み込む}

geometry パッケージの begin-document フックの処理に割り込む。
※ \LaTeX \のフックシステムがある場合はムニュマニューナ。
\begin{verbatim}
def\bxjs@geometry@name{geometry}
\if\bxjs@old@hook@system
\let\bxjs@apply@bd@pre@geometry@hook\AtBeginDocument
\else
\def\bxjs@apply@bd@pre@geometry@hook{%
  \AddToHook{begindocument}{\bxjs@geometry@name}}
\fi
\end{verbatim}

\texttt{geometry=class} の場合に、実際に geometry パッケージを読み込む。
\begin{verbatim}
\if\fx\bxjs@geometry\bxjs@geometry@@class
  \PassOptionsToPackage{driver=none}{geometry}
\fi
\end{verbatim}

ここで geometry を読み込む。
※ geometry の begin-document フックにおいて、\LaTeX\ の旧版互換を有効にする。
\begin{verbatim}
\bxjs@apply@bd@pre@geometry@hook{\noexpand\RequirePackage[\bxjs@layout\paper,\bxjs@layout]{geometry}}
\end{verbatim}

\texttt{\bxjs@geometry@driver} が用いるドライバの名前。
※この値は一度決めた後は変わってしまわないので、\texttt{\bxjs@postproc@layout} において書き戻す処理を入れている。
\begin{verbatim}
\let\bxjs@geometry@driver\Gm@driver
\end{verbatim}
BXJS の 2.0 版より、geometry の 4.x 版のサポートは廃止された。

エンジンが platex-ng の時は geometry のドライバを pdftex にする。

geometry パッケージを読み込まない ☃
geometry=user の場合の処理。
　この場合はユーザが何らかの方法（例えば geometry を読み込む）でページレイアウトを設定する必要がある。もし、本体開始時に \textwidth がカーネル設定の値（.5\maxdimen）のままになっている場合はエラーを出す。

*\jsUseMinimalPageLayout は動作テスト用。
はとりあえず無効にしておく。

\let\bxjs@geometry@driver\relax
\def\setpagelayout{%\bxjs@ifplus{\bxjs@pagelayout@a}{%else\@ifstar{\bxjs@pagelayout@a}{\bxjs@pagelayout@a}}}
\def\bxjs@pagelayout@a#1{%ClassError\bxjs@clsname{Command \string\setpagelayout is not supported,\MessageBreak because 'geometry' value is not 'class'}\@eha}
\fi\fi

■縦方向のスペース

ここでは、jsclasse.dtx との差分を抑制するために、オリジナルのコードを無効化した状態で挿入しておく。

\\headheight \topskip は本文領域上端と本文 1 行目のベースラインとの距離です。あまりぎりぎりの値にすると、本文中に \(\int\) のような高い文字が入ったときに 1 行目のベースラインが他のページより下がってしまいます。ここでは本文の公称フォントサイズ（10pt）にします。

[2003-06-26] \headheight はヘッダの高さで、元は 12pt でしたが、新ドキュメントクラスでは \topskip と等しくしていました。ところが、fancyhdr パッケージで \headheight が小さいとおかしいことになるようですので、2 倍に増やしました。代わりに、版面の上下揃えの計算では \headheight ではなく \topskip を使うことにしました。

[2016-08-17] 圏点やルビが一行目に来た場合に下がるのを防ぐため、\topskip を 10pt から 1.38zw に増やしました。\headheight は従来と同じ 20pt とします。

\setlength{\topskip}{1.38zw}%% from 10\jsc@mpt (2016-08-17)
\if@slide
\setlength{\headheight}{0\jsc@mpt}
\else
\setlength{\headheight}{20\jsc@mpt}%% from 2\topskip (2016-08-17); from \topskip (2003-06-26)
\fi

\footskip \footskip は本文領域下端とフッタ下端との距離です。標準クラスファイルでは、book で 0.35in（約 8.89mm）、book 以外で 30pt（約 10.54mm）となっていましたが、ここでは A4 状のときちょうど 1cm となるように、\paperheight の 0.03367 倍（最小 \baselineskip）としました。書籍については、フッタは使わないことにし、ゼロにしました。
はヘッダ下端と本文領域上端との距離です。元は book で 18pt（約 6.33mm）、それ以外で 25pt（約 8.79mm）になっていた。ここでは article は \footskip - \topskip としました。

[2016-10-08] article の slide のとき、および book の非 report と kiyou のときに \headsep を減らしてこわっていたのを修正しました（2016-08-17 での修正漏れ）。

\headsep \headsep はヘッダ下端と本文領域上端との距離です。元は book で 18pt（約 6.33mm）、それ以外で 25pt（約 8.79mm）になっていた。ここでは article は \footskip - \topskip としました。

[2016-10-08] article の slide のとき、および book の非 report と kiyou のときに \headsep を減らしてこわっていたのを修正しました（2016-08-17 での修正漏れ）。
maxdepth は本文最下行の最大の深さで、plain TeX や LaTeX 2.09 では 4pt に固定でした。LaTeX2e では maxdepth + topskip を本文フォントサイズの 1.5 倍にしたいのですが、topskip は本文フォントサイズ（ここでは 10pt）に等しいので、結局 maxdepth は topskip の半分の値（具体的には 5pt）にします。

\newdimen\fullwidth
この \fullwidth は article では紙幅 \paperwidth の 0.76 倍を超えない全角幅の整数倍（二段組では全角幅の偶数倍）にします。0.76 倍という数値は A4 縦置きの場合に紙幅から約 2 インチを引いた値になるように選びました。book では紙幅から 36 ミリを引いた値にしました。

\textwidth 書籍以外では本文領域の幅 \textwidth は \fullwidth と等しくします。article では A4 縦置きで 49 文字となります。某学会誌スタイルでは 50zw（25 文字×2段）+段間 8mm とします。
本文部分の高さ \textheight は、紙の高さ \paperheight の 0.83 倍から、ヘッダの高さ、ヘッダと本文の距離、本文とフッタ下端の距離、\topskip を引き、それを \baselineskip の倍数に切り捨て、最後に \topskip を加えます。念のため 0.1 ポイント余分に加えており、0.83 倍という数値は、A4 録置きの場合に紙の高さから上下マージン各約 1 インチを引いた値になるように選びました。

某学会誌スタイルでは 44 行にします。
[2003-06-26] \headheight を \topskip に直しました。以前はこの二つは値が同じであったので、変化はないはずです。
[2016-08-26] \topskip を 10pt から 1.38zw に増やしましたので、その分 \textheight
を増やします（2016-08-17での修正漏れ）。
[2016-10-08] article の slide のときに headheight はゼロなので、さらに修正しました（2016-08-17での修正漏れ）。

\begin{verbatim}
%<article|book|report>
\if@slide
  \setlength{\textheight}{0.95\paperheight}
\else
  \setlength{\textheight}{0.83\paperheight}
\fi
\addtolength{\textheight}{-10\jcs@mpt}%% from -\topskip (2016-10-08); from -\headheight (2003-06-26)
\addtolength{\textheight}{-\headsep}
\addtolength{\textheight}{-\footskip}
\divide\textheight{\baselineskip}
\addtolength{\textheight}{\topskip}
\multiply\textheight{\baselineskip}
\end{verbatim}

\flushbottom[2016-07-18] \textwidth に念のため 0.1 ポイント余裕を持たせているのと同様に、\flushbottom にも余裕を持たせます。元の \LaTeX2e での完全な \flushbottom の定義は

\begin{verbatim}
def\flushbottom{\let\@textbottom\relax \let\@texttop\relax}
\end{verbatim}

ですが、次のようにします。

\begin{verbatim}
def\flushbottom{}\def\@textbottom{\vskip\z@ \@plus.1\jcs@mpt}\let\@texttop\relax
\end{verbatim}

\flushbottom \textwidth に念のため 0.1 ポイント余裕を持たせているのと同様に、\flushbottom にも余裕を持たせます。元の \LaTeX2e での完全な \flushbottom の定義は

\begin{verbatim}
def\flushbottom{\let\@textbottom\relax \let\@texttop\relax}
\end{verbatim}

ですが、次のようにします。

\begin{verbatim}
def\flushbottom{}\def\@textbottom{\vskip\z@ \@plus.1\jcs@mpt}\let\@texttop\relax
\end{verbatim}

\flushbottom \textwidth に念のため 0.1 ポイント余裕を持たせているのと同様に、\flushbottom にも余裕を持たせます。元の \LaTeX2e での完全な \flushbottom の定義は

\begin{verbatim}
def\flushbottom{\let\@textbottom\relax \let\@texttop\relax}
\end{verbatim}

ですが、次のようにします。
\addtolength{\oddsidemargin}{-1in}
\else
\addtolength{\oddsidemargin}{-\inv@mag in}
\fi
\setlength{\evensidemargin}{\oddsidemargin}
\if@mparswitch
\addtolength{\evensidemargin}{\fullwidth}
\addtolength{\evensidemargin}{-\textwidth}
\fi
\addtolength{\evensidemargin}{-\marginparsep}
\setlength{\marginparwidth}{\paperwidth}
\addtolength{\marginparwidth}{-\oddsidemargin}
\addtolength{\marginparwidth}{-\inv@mag in}
\addtolength{\marginparwidth}{-\textwidth}
\addtolength{\marginparwidth}{-10\jsc@mmm}
\divide{\marginparwidth}{\@tempdima}
\multiply{\marginparwidth}{\@tempdima}
\topmargin 上マージン（紙の上端とヘッダ上端の距離）から1インチ引いた値です。
\headheight 上マージン を \topskip に直しました。以前はこの二つは値が同じであったので、変化はないはずです。
\topskip を10ptから1.38zwに直しましたが、\topmargin は従来の値から変わらないように調節しました。…つもりでしたが、\textheight を増やし忘れていたので変わってしまいました（2016-08-26 修正済み）。
\setlength{\topmargin}{\paperheight}
\addtolength{\topmargin}{-\textheight}
\if@slide
\addtolength{\topmargin}{-\headheight}
\else
\addtolength{\topmargin}{-\textheight}
\fi
%<kiyou>\setlength{\topmargin}{81truebp}
\if@tombow
\addtolength{\topmargin}{-1in}
\else
\addtolength{\topmargin}{-\inv@mag in}
\fi
%</jsclasses>
■脚注
ここからのコードは以下の点を除いて JS クラスのものを踏襲する。

- zw の代わりに \jsZw を用いる。
- article/report/book/slide の切り分けの処理が異なる。

各脚注の頂に入る支柱（strut）の高さです。脚注間に余分のアキが入らないように、\footnotesize の支柱の高さ（行送りの 0.7 倍）に等しくします。

ここは元々は
\{\footnotesize\global\setlength\footnotesep{\baselineskip}\}
としていたが、そもそも {\global\setlength}\~ は calc 使用時には有意義な動作をしない。\global\footnotesep だと所望の値が得られるが、同時に \footnotesize のフォントを固定させてしまうという副作用をもつ。なので、実際の設定値を直接使うことにする。

\footnotesep=11\jcs@mpt \footnotesep=0.7\footnotesep
\footins \skip\footins は本文の最終行と最初の脚注との間の距離です。標準の 10 ポイントクラスでは 9 plus 4 minus 2 ポイントになっていますが、和文の行送りを考えてもうちょっと大きくします。
\setlength{\skip\footins}{16\jcs@mpt \@plus 5\jcs@mpt \@minus 2\jcs@mpt}

■フロート関連
フロート (図, 表) 関連のパラメータは \LaTeX2e 本体で定義されていますが、ここで設定変更します。本文ページ（本文とフロートが共存するページ）とフロートだけのページで設定が異なります。ちなみに、カウンタは内部では \c@ を名前に冠したマクロになっています。
\c@topnumber \topnumber カウンタは本文ページ上部のフロートの最大数です。
\setcounter{topnumber}{9}
\topfraction 本文ページ上部のフロートが占有できる最大の割合です。フロートが入りやすいように、元の値 0.7 を 0.8 [2003-08-23: 0.85] に変えてあります。
\renewcommand{\topfraction}{.85}
\c@bottomnumber \bottomnumber カウンタは本文ページ下部のフロートの最大数です。
\setcounter{bottomnumber}{9}
\bottomfraction 本文ページ下部のフロートが占有できる最大の割合です。元は 0.3 でした。
\renewcommand{\bottomfraction}{.8}
カウンタは本文ページに入りうるフロートの最大数です。

2007 \setcounter{totalnumber}{20}

本文ページに最低限書かなければならない本文の割合です。フロートが入りやすいように元の 0.2 を 0.1 に変えました。

2008 \renewcommand{\textfraction}{.1}

フロートだけのページでのフロートの最小割合です。これも 0.5 を 0.8 に変えてあります。

2009 \renewcommand{\floatpagefraction}{.8}

二段組のとき本文ページ上部に出力できる段抜きフロートの最大数です。ちょっと増やしました。

2010 \setcounter{dbltopnumber}{9}

二段組のとき本文ページ上部に出力できる段抜きフロートが占めうる最大の割合です。0.7を0.8に変えてあります。

2011 \renewcommand{\dbltopfraction}{.8}

二段組のときフロートだけのページに入れるべき段抜きフロートの最小割合です。0.5を0.8に変えてあります。

2012 \renewcommand{\dblfloatpagefraction}{.8}

フロートだけのページに入るグルーです。\@fptop はページ上部、\@fpbot はページ下部、\@fpsep はフロート間に入ります。
6 改ページ（日本語 \TeX 開発コミュニティ版のみ）

\pltx@cleartorightpage
\pltx@cleartoleftpage
\pltx@cleartooddpage
\pltx@cleartoevenpage

2017-02-24 コミュニティ版 pLaTeX の標準クラス 2017/02/15 に合わせて、同じ命令を追加しました。

1. \pltx@cleartorightpage：右ページになるまでページを繰る命令
2. \pltx@cleartoleftpage：左ページになるまでページを繰る命令
3. \pltx@cleartooddpage：奇数ページになるまでページを繰る命令
4. \pltx@cleartoevenpage：偶数ページになるまでページを繰る命令

となっています。

\def\pltx@cleartorightpage{\clearpage\if@twoside
\ifodd\c@page
\hbox{}\thispagestyle{empty}\newpage
\if@twocolumn\hbox{}\newpage\fi
\else
\ifydir
\hbox{}\thispagestyle{empty}\newpage
\if@twocolumn\hbox{}\newpage\fi
\fi
\else
\ifydir
\hbox{}\thispagestyle{empty}\newpage
\if@twocolumn\hbox{}\newpage\fi
\fi\fi}

\def\pltx@cleartoleftpage{\clearpage\if@twoside
\ifodd\c@page\else
\hbox{}\thispagestyle{empty}\newpage
\if@twocolumn\hbox{}\newpage\fi
\fi\fi}

\def\pltx@cleartooddpage{\clearpage\if@twoside
\ifodd\c@page\else
\hbox{}\thispagestyle{empty}\newpage
\if@twocolumn\hbox{}\newpage\fi
\fi\fi}

\def\pltx@cleartoevenpage{\clearpage\if@twoside
\ifodd\c@page\else
\hbox{}\thispagestyle{empty}\newpage
\if@twocolumn\hbox{}\newpage\fi
\fi\fi}
BXJS クラスでは \iftdirm 等が使えないので、横組を仮定した定義を用いる。
\let\pltx@cleartorightpage\pltx@cleartooddpage
\let\pltx@cleartoleftpage\pltx@cleartoevenpage
\vsize の値がアレな場合は本体開始まで \clearpage を無効にする。
\ifdim\vsize=\z@\begingroup\toks0\expandafter{\clearpage}\xdef\clearpage{\noexpand\ifbxjs@after@preamble\the\toks0\noexpand\fi}\endgroup\fi
\cleardoublepage

\cleardoublepage[2017-02-24] コミュニティ版 pLTpX の標準クラス 2017/02/15 に合わせて, report と book クラスの場合に\cleardoublepage を再定義します。
\%<book|report>
\if@openleft\let\cleardoublepage\pltx@cleartoleftpage\else\if@openright\let\cleardoublepage\pltx@cleartorightpage\fi\fi
\%</book|report>

7 ページスタイル

ページスタイルとして, \LaTeX{}2ε（欧文版）の標準クラスでは empty, plain, headings, myheadings があります。このうち empty, plain タイプは \LaTeX{}2ε 本体で定義されています。

アスキーのクラスファイルでは headnombre, footnombre, bothstyle, jpl@in が追加されていますが、ここでは欧文標準のものだけにしました。

ページスタイルは \ps@... の形のマクロで定義されています。
\@evenhead \@oddhead, \@evenfoot, \@oddfoot は偶数・奇数ページの柱（ヘッダ, フッタ）を出力する命令です。これらは \fullwidth 幅の \hbox 中で呼び出されます。
\@evenfoot \@oddfoot \ps@... の中で定義しておきます。

柱の内容は, \chapter が呼び出す \chaptermark{何々}, \section が呼び出す \sectionmark{何々} で設定します。柱を扱う命令には次のものがあります。

\markboth{左}{右} 両方の柱を設定します。
\markright{右} 右の柱を設定します。
\leftmark 左の柱を出力します。
\rightmark 右の柱を出力します。

柱を設定する命令は、右の柱が左の柱の下位にある場合は十分まともに動作します。たと
えば左マークを `\chapter`, 右マークを `\section` で変更する場合がこれにあたります。しかし、同一ページに複数の `\markboth` があると、おかしな結果になることがあります。

`\tableofcontents` のような命令で使われる `\markboth` は、`\ps@...` コマンド中で `\markboth` か `@gobbletwo` (何もしない) に `\let` されます。

\ps@empty  `empty` ページスタイルの定義です。`HTeX` 本体で定義されているものをコメントアウトした形で載せておきます。
2073  % `\def\ps@empty{`
2074  % `\let\@mkboth\@gobbletwo`
2075  % `\let\@oddhead\@empty`
2076  % `\let\@oddfoot\@empty`
2077  % `\let\@evenhead\@empty`
2078  % `\let\@evenfoot\@empty`}

\ps@plainhead `plainhead` はシンプルなヘッダだけのページスタイルです。
\ps@plainfoot `plainfoot` はシンプルなフッタだけのページスタイルです。
\ps@plain `plain` は `book` では `plainhead`, それ以外では `plainfoot` になります。
2079  `\def\ps@plainfoot{`
2080  `% `\let\@mkboth\@gobbletwo`
2081  `% `\let\@oddhead\@empty`
2082  `% `\def\@oddfoot{\normalfont\hfil\thepage\hfil}`
2083  `% `\let\@evenhead\@empty`
2084  `% `\let\@evenfoot\@oddfoot`
2085  `\ps@plainhead{`
2086  `% `\let\@mkboth\@gobbletwo`
2087  `% `\let\@oddhead\@empty`
2088  `% `\let\@evenhead\@empty`
2089  `% `\def\@evenhead{`
2090  `% `\if@mparswitch \hss \fi`
2091  `% `\hbox to \fullwidth{\textbf{\thepage}\hfil\leftmark}`
2092  `% `\if@mparswitch\else \hss \fi}`
2093  `% `\def\@oddhead{`
2094  `% `\hbox to \fullwidth{\hfil\textbf{\thepage}\hfil}`
2095  `% `\hbox to \fullwidth{\hfil\textbf{\thepage}\hfil}`
2096  `% `\def\ps@plain{`
2097  `% `\let\@mkboth\@gobbletwo`
2098  `% `\let\@oddfoot\@empty`
2099  `% `\let\@evenfoot\@empty`
2100  `% `\def\@evenfoot{`
2101  `% `\if@mparswitch \hss \fi`
2102  `% `\hbox to \fullwidth{\textbf{\thepage}\hfil\leftmark}`
2103  `% `\if@mparswitch\else \hss \fi`}

\ps@headings `headings` スタイルはヘッダに見出しとページ番号を出力します。ここではヘッダにアンダーラインを引くようにしてみました。

まず `article` の場合です。
2107  `% `<article|slide>
2108  `% `\if@twoside
2109  `% `\def\ps@headings{`
2110  `% `\let\@oddfoot\@empty`
2111  `% `\let\@evenfoot\@empty`
2112  `% `\def\@evenhead{`
2113  `% `\if@mparswitch \hss \fi`
2114  `% `\underline{\textbf{\thepage}\hfil\leftmark}`
2115  `% `\if@mparswitch\else \hss \fi`}

78

\bxjs@maybe@autoxspacing \autoxspacing が定義済ならばそれを実行する。
※\autoxspacing は未定義の可能性があるため代わりに用いる。
最後は学会誌の場合です。

8 文書のマークアップ

8.1 表題

\title これらは \LaTeX 本体で次のように定義されています。ここではコメントアウトした形で示します。

\author
副題を設定する。

\jsSubtitle

※プレアンブルにおいて \newcommand*{\subtitle}{...} が行われることへの対策として、\subtitle の定義を \title の実行まで遅延させることにする。もしどうしても主題より前に副題を設定したい場合は、\jsSubtitle 命令を直接用いればよい。

**TODO: 3.0** \subtitle の遅延処理は Pandoc モードに移す。

本体を \jsSubtitle として定義する。

\newcommand*{\jsSubtitle}[1]{\gdef\bxjs@subtitle{#1}}
\let\bxjs@subtitle\@undefined
\title にフックを入れる。
\renewcommand*{\title}[1]{\bxjs@decl@subtitle\gdef\@title{#1}}
\g@addto@macro\bxjs@begin@document@hook{\bxjs@decl@subtitle}
\def\bxjs@decl@subtitle{\global\let\bxjs@decl@subtitle\relax}
\ifx\subtitle\@undefined\global\let\subtitle\jsSubtitle\fi
\global\let\jsSubtitle\relax

\bxjs@annihilate@subtitle \subtitle 命令を無効化する。

※独自の \subtitle が使われている場合は無効化しない。
\def\bxjs@annihilate@subtitle{%
\ifx\subtitle\jsSubtitle \global\let\subtitle\relax \fi
\global\let\jsSubtitle\relax
}

某学会誌スタイルで使う英語のタイトル, 英語の著者名, キーワード, メールアドレスです。

\etitle 2192 %<**jspf>
\eaauthor 2193 \newcommand*{\etitle}[1]{\gdef\@etitle{#1}}
\eaauthor 2194 \newcommand*{\eaauthor}[1]{\gdef\@eaauthor{#1}}
\keywords 2195 \newcommand*{\keywords}[1]{\gdef\@keywords{#1}}
\email 2196 \newcommand*{\email}[1]{\gdef\@email{#1}}
\ AUTHORSEMAIL 2197 \newcommand*{\AUTHORSEMAIL}[1]{\gdef\@email{author's e-mail: #1}}
\keywords 2198 \%

従来の標準クラスでは、文書全体のページスタイルを empty にしても表題のあるページだけ plain になってしまうことがありました。これは \maketitle の定義中に \thispagestyle {plain} が入っているためです。この問題を解決するために、「全体のページスタイルが empty でないならこのページのスタイルを plain にする」という次の命令を作ることにしました。
\plainifnotempty 2199 %
\ifx \@oddhead \@empty

81
\maketitle 表題を出力します。著者名を出力する部分は、欧文の標準クラスファイルでは \large、和文のものでは \Large になっていましたが、ここでは \large にしました。

[2016-11-16] 新設された nomag および nomag* オプションの場合をデフォルト（usemag 相当）に合わせるため、\smallskip を\jcs@smallskip に置き換えた。
\smallskip のまでは nomag(*) の場合にスケールしなくなり、レイアウトが変わってしまいます。

\maketitle

独立した表題ページを作らない場合の表題の出力形式です。

\maketitle

\begin{center}

\let\footnote\thanks

{\LARGE \@title \par}
\ifx\bxjs@subtitle\@undefined\else
\vskip3\jsc@mpt
{\normalsize \bxjs@subtitle\par}
\fi
\vskip 1.5em

{\large
\lineskip .5em
\begin{tabular}{c}
\@author

\end{tabular}\par}
\vskip 1em

{\large \@date}
\end{center}
\par\vskip 1.5em
\%<article|slide> \ifvoid\@abstractbox\else\centerline{\box\@abstractbox}\vskip1.5em\fi
\} 
\fi
\%</article|book|report|slide>
\%<jspf>
\newcommand{\maketitle}{\par
\begingroup
\renewcommand{\thefootnote}{\@fnsymbol{\c@footnote}}%
\def\@makenumbermark{\rlap{\@textsuperscript{\normalfont\@thefnmark}}}%
\longdef{\makefntext[#1]{\advance\leftskip 3\jsZw
\parindent 1}\jsZw
\noindent
\llap{\@textsuperscript{\normalfont\@thefnmark}}\vskip0.3\jsZw
\#1}}%
\twocolumn[\maketitle]
\plainifnotempty
\@thanks
\endgroup
\setcounter{footnote}{0}
は柱に使う
8.2 章・節

label-section オプション対応のための処理。

\bxjs@label@sect 節付 #1 の番号を出力する。節付 XXX に対して、\labelXXX が定義済ならそれが出力書式を表す。未定義ならばカウンタの出力書式 \theXXX が使われる。

\def\bxjs@label@sect#1{%
  \@ifundefined{label#1}{\@nameuse{the#1}}{\@nameuse{label#1}}}

\secapp 節番号の接頭辞。
\secpos 節番号の接尾辞。

\ifnum\bxjs@label@section=\bxjs@label@section@@compat\else
  \def\@secapp{\presectionname}
  \def\@secpos{\postsectionname}
\fi

\labelsection 節番号の出力書式。
\ifnum\bxjs@label@section=\bxjs@label@section@@modern
  \def\labelsection{\@secapp\thesection\@secpos}
\fi

■構成要素 \@startsection マクロは6個の必須引数と、オプションとして*と1個のオプション引数と1個の必須引数をとります。

\@startsection(名){レベル}{字下げ}{前アキ}{後アキ}{スタイル}
  *[別見出し](見出し)

それぞれの引数の意味は次の通りです。

名　ユーザレベルコマンドの名前です（例: section）。
レベル　見出しの深さを示す数値です（chapter=1, section=2, ...）。この数値がsecnumdepth 以下のとき見出し番号を出力します。
字下げ　見出しの字下げ量です。
前アキ　この値の絶対値が見出し上側の空きです。負の場合は、見出し直後の段落をインデントしません。
後アキ　正の場合は、見出しの下の空きです。負の場合は、絶対値が見出しの右の空きです（見出しと同じ行から本文を始めます）。
スタイル　見出しの文字スタイルの設定です。
*　この * 印がないと、見出し番号を付け、見出し番号のカウンタに1を加算します。
別見出し　目次や索引に出力する見出しです。
見出し
見出しです。
見出しの命令は通常 \@startsection とその最初の 6 個の引数として定義されます。
次は \@startsection の定義です。情報処理学会論文誌スタイルファイル（ipsjcommon.sty）を参考にさせていただきましたが、完全に行送りが \baselineskip の整数倍にならなくてもいいから前の行と重ならないようにしました。
\begin{verbatim}
def\@startsection#1#2#3#4#5#6{%
  \if@noskipsec \leavevmode \fi
  \par
  \@tempskipa #4\relax
  \if@english \@afterindentfalse \else \@afterindenttrue \fi
  \ifdim \@tempskipa <\z@ \@tempskipa -\@tempskipa \@afterindentfalse \fi
  \if@nobreak \everypar\everyparhook{}\else \addpenalty\@secpenalty \addvspace\@tempskipa \fi
  \ifdim \@tempskipa >\z@ \if@slide\else \null \vspace*{-\baselineskip} \fi
  \vskip\@tempskipa \fi
  \noindent
  \@ifstar{\@ssect{#3}{#4}{#5}{#6}}{\@dblarg{\@sect{#1}{#2}{#3}{#4}{#5}{#6}}}
}\@sect
\@xsect は、前のアキがちょうどゼロの場合にもうまくいくように、多少変え
てあります。\everyparhook の挿入は \everyparhook=compat の時のみ行
う。
\end{verbatim}
\bxjs@if@ceph \bxjs@if@ceph{\langle コード \rangle}: everyparhook=compat である場合にのみ \langle コード \rangle を実行
する。
\begin{verbatim}
\ifnum\bxjs@everyparhook=\bxjs@everyparhook@compat \let\bxjs@if@ceph@firstofone
\end{verbatim}
2432 \else \let\bxjs@if@ceph\@gobble
2433 \fi

2434 \def\@sect#1#2#3#4#5#6[#7]{%
2435 \ifnum #2>\c@secnumdepth
2436 \let\@svsec\@empty
2437 \else
2438 \refstepcounter{#1}%
2439 \protected@edef\@svsec{\@seccntformat{#1}\relax}%
2440 \fi
2441 \% 見出し後の空きを \@tempskipa にセット
2442 \@tempskipa #5\relax
2443 \% 条件判断の順序を入れ替えました
2444 \ifdim \@tempskipa<\z@ 2445 \def\@svsechd{%
2446 #6{\hskip #3\relax
2447 \@svsec #8}%
2448 \@tempskipa #8\relax
2449 \csname #1mark\endcsname{#7}%
2450 \addcontentsline{toc}{#1}{% 88
2451 \ifnum #2>\c@secnumdepth \else
2452 \protect\numberline{\bxjs@label@sect{#1}}%
2453 \fi
2454 \fi
2455 \begin{paragraph}
2456 \interlinepenalty \ifnum #2>\c@secnumdepth \else
2457 \addcontentsline{toc}{#1}{% 88
2458 \interlinepenalty \ifnum #2>\c@secnumdepth \else
2459 \csname #1mark\endcsname{#7}%
2460 \addcontentsline{toc}{#1}{% 88
2461 \endparagraph
2462 \csname #1mark\endcsname{#7}%
2463 \addcontentsline{toc}{#1}{% 88
2464 \ifnum #2>\c@secnumdepth \else
2465 \protect\numberline{\bxjs@label@sect{#1}}%
2466 \fi
2467 \fi
2468 \fi
2469 \@sect{#5}}%

二つ挿入した \everyparhook のうち後者が \paragraph 類の後で 2 回実行され、それ以降は前者が実行されます。

[2016-07-28] slide オプションと twocolumn オプションを同時に指定した場合の罫線の位置を微調整しました。
■柱関係の命令

\chaptermark \...mark の形の命令を初期化します（第 7 節参照）。\chaptermark 以外は \LaTeX 本体で定義済みです。

\sectionmark
\subsectionmark
\subsubsectionmark
\paragraphmark
\subparagraphmark

89
カウンタの定義
\c@secnumdepth secnumdepth は第何レベルの見出しまで番号を付けるかを決めるカウンタです。
\c@chapter 見出し番号のカウンタです。\newcounter の第 1 引数が新たに作るカウンタです。これは
\c@section 第 2 引数が増加するたびに 0 に戻されます。第 2 引数は定義済みのカウンタです。
\c@subsection
\c@subsubsection
\c@paragraph
\c@subparagraph

\thepart カウンタの値を出力する命令 \the 何々 を定義します。
\thechapter カウンタを出力するコマンドには次のものがあります。
\thesection \arabic{COUNTER} 1, 2, 3, ...
\thesubsection \roman{COUNTER} i, ii, iii, ...
\thesubsubsection \Roman{COUNTER} I, II, III, ...
\theparagraph \alph{COUNTER} a, b, c, ...
\theparagraph \Alph{COUNTER} A, B, C, ...
\kansuji{COUNTER} 一, 二, 三, ...

以下ではスペース節約のため @ の付いた内部表現を多用しています。
\@chapapp の初期値は \texttt{prechaptername}（第）です。
\@chappos の初期値は \texttt{postchaptername}（章）です。
\appendix は \@chapapp を \texttt{appendixname} に、\@chappos を空に再定義します。

■前付、本文、後付 本のうち章番号があるのが「本文」、それ以外が「前付」、「後付」です。

\frontmatter ページ番号をローマ数字にし、章番号を付けないようにします。
[2017-03-05] \frontmatter と \texttt{mainmatter} の 2 つの命令は、改丁または改ページした後で \texttt{pagening...} でノンブルを 1 にリセットします。長い間 \frontmatter は \texttt{openany} のときに単なる改ページとしていたが、これではノンブルをリセットする際に偶奇逆転が起こる場合がありました。\texttt{openany} かどうかに依らず奇数ページまで繰るように修正することで、問題を解消しました。実は、\LaTeX の標準クラスでは 1998 年に修正されていた問題です（コミュニティ版 \texttt{pL\LaTeX} の標準クラス 2017/03/05 も参照）。

\mainmatter ページ番号を算用数字にし、章番号を付けるようにします。

\backmatter 章番号を付けないようにします。ページ番号の付け方は変わりません。
新しい部を始めます。

\seccdef を使って見出しを定義しています。このマクロは二つの引数をとります。

\seccdef{星なし}{星あり}

星なし * のない形の定義です。
星あり * のある形の定義です。

\seccdef は次のようにして使います。

\def\chapter {... \seccdef \CMDA \CMDB }
\def\CMDA [#1]{...} % \chapter[...]{...} の定義
\def\CMDB #1{...} % \chapter*[...]{...} の定義

まず book と report のクラス以外です。

2572 %<>!book&!report>
2573 \newcommand\part{%
2574 \if@noskipsec \leavespace \fi
2575 \par
2576 \addvspace{4ex} %
2577 \if@english \afterindentfalse \else \afterindenttrue \fi
2578 \seccdef\part@spart}
2579 %</book&!report>

book および report クラスの場合は、少し複雑です。

2580 %<>book/report>
2581 \newcommand\part{%
2582 \if@openleft
2583 \cleardoublepage
2584 \else\if@openright
2585 \cleardoublepage
2586 \else
2587 \clearpage
2588 \fi\fi
2589 \thispagestyle{empty}% 欧文用標準スタイルでは plain
2590 \if@twocolumn
2591 \onecolumn
2592 \@restonecoltrue
2593 \else
2594 \@restonecolfalse
2595 \fi
2596 \null\vfill
2597 \seccdef\part@spart}
2598 %</book&report>

\part 部の見出しを出力します。\bfseries を \headfont に変えました。

book および report クラス以外では secnumdepth が −1 より大きいとき部番号を付け

92
ます。

book および report クラスでは secnumdepth が −2 より大きいとき部番号を付けます。

\@spart 番号を付けない部です。
\@endpart と \@part の最後で実行されるマクロです。両面印刷のときは白ページを追加します。二段組のときには、二段組に戻します。

[2016-12-13] openany のときには白ページが追加されるのは変なので、その場合は追加しないようにしました。このバグは \LaTeX{} では classes.dtx v1.4b (2000/05/19) で修正されています。

\chapter 章の最初のページスタイルは、全体が empty でなければ plain にします。また、\@topnum を 0 にして、章見出しの上に図や表が来ないようにします。
\@chapter 章見出しを出力します。secnumdepth が 0 以上かつ \@mainmatter が真のとき章番号を出力します。

\@chapter[#1]{#2}
\def\@chapter[#1]{#2}{% 
\ifnum \c@secnumdepth >\m@ne \if@mainmatter 
\refstepcounter{chapter} %
\typeout{\@chapapp\thechapter\@chappos} %
\addcontentsline{toc}{chapter}{#1} %
\else
\addcontentsline{toc}{chapter}{#1} %
\fi %
\chaptermark{#1} %
\addtocontents{lof}{\protect\addvspace{10\@vs}} %
\addtocontents{lot}{\protect\addvspace{10\@vs}} %
\if@twocolumn
\@topnewpage[\@makechapterhead{#2}] %
\else
\@makechapterhead{#2} %
\@afterheading
\fi %
\@makechapterhead 実際に章見出しを組み立てます。\bfseries を \headfont に変えました。
\def\@makechapterhead#1{%
\vspace*{2\@vs} %欧文は 50pt
\parindent \z@ \raggedright \normalfont
\ifnum \c@secnumdepth >\m@ne %
\huge \headfont \@chapapp\thechapter\@chappos
\par
\nobreak
\vskip \@vs % 欧文は 20pt
\fi
\fi
\interlinepenalty\@M
\Huge \headfont #1
\vskip 3\@vs % 欧文は 40pt
\@schapter \chapter*{...} コマンドの本体です。\chaptermark を補いました。
\def\@schapter#1{%
番号なしの章見出しです。

下位レベルの見出し

同上です。
後アキ

対処しました（forum:1982）。

従来は \paragraph の最初に出るマークを「■」に固定していましたが、このマークを変更可能にするため \jsParagraphMark というマクロに切り出しました。これで、たとえば

\renewcommand{\jsParagraphMark}{★}

とすれば「★」に変更できますし、マークを空にすることも容易です。なお、某学会クラスでは従来どおりマークは付けません。

※ BXJS クラスでは、1.1 版 [2016-02-14] から \jsParagraphMark をサポートしている。段落のマーク（■）が必ず和文フォントで出力されるようにする。
8.3 リスト環境

第 $k$ レベルのリストの初期化をするのが $\@list k$ です ($k = i, ii, iii, iv$)。$\@list k$ は $\leftmargin$ を $\leftmargin k$ に設定します。

\leftmargini 二段組であるかないかに応じてそれぞれ 2em, 2.5em でしたのが、ここでは全角幅の 2 倍にしました。

[2002-05-11] 3zw に変更しました。
[2005-03-19] 二段組は 2zw に戻しました。

\leftmarginii i, iii, iv は $\labelsep$ とそれぞれ ‘(m)’, ‘vi’, ‘M’ の幅との和より大きくすることになっています。ここでは全角幅の整数倍に丸めました。

\leftmarginiii
\labelsep \labelsep はラベルと本文の間の距離です。\labelwidth はラベルの幅です。これは二分
\labelwidth に変えました。
\setlength \labelsep {0.5\jsZw} % .5em
\setlength \labelwidth{\leftmargini}
\addtolength\labelwidth{-\labelsep}
\partopsep リスト環境の前に空行がある場合、\parskip と \topsep に \partopsep を加えた値だけ
縦方向の空白ができます。0 に改変しました。
\setlength\partopsep{\z@} % {2\p@ \@plus 1\p@ \@minus 1\p@}
\@beginparpenalty リストや段落環境の前後、リスト項目間に挿入されるペナルティです。
\@endparpenalty
\@itempenalty
\@listi \@listi は \leftmargin, \parsep, \topsep, \itemsep などのトップレベルの定義を
\@listi します。この定義は、フォントサイズコマンドによって変更されます（たとえば \small の
中では小さい値に設定されます）。このため、\normalsize がすべてのパラメータを戻せる
ように、\@listi で \@listi のコピーを保存します。元の値はかなり複雑ですが、ここでは
簡素化していきました。特に最初と最後に行送りの半分の空きが入るようにしてありました。
アスキの標準スタイルではトップレベルの itemize, enumerate 環境でだけ最初と
最後に行送りの半分の空きが入るようになっていました。
[2004-09-27] \topsep のグルー +0.2 -0.1 \baselineskip を思い切って外しました。
\def\@listi\{\leftmargin\leftmargini
\parsep \z@\topsep \z@\parsep \z@ \itemsep \z@\relax}
\let\@listi\@listi

念のためパラメータを初期化します（実際には不要のようにです）。
\@listi
\@listii 第 2〜6 レベルのリスト環境のパラメータの設定です。
\@listiii \@listiii \@listiv \@listiv \@listv \@listv \@listvi
\@listvi \@listvi
\def\@listiii{\leftmargin\leftmargini
\labelwidth\leftmargini \advance\labelwidth-\labelsep
\topsep \z@\parsep \z@}
\def\@listiii{\leftmargin\leftmarginii
\labelwidth\leftmarginii \advance\labelwidth-\labelsep
\topsep \z@\parsep \z@}
\def\@listii{\leftmargin\leftmarginiii
\labelwidth\leftmarginiii \advance\labelwidth-\labelsep
\topsep \z@\parsep \z@}
\def\@listi{\leftmargin\leftmarginiv
\labelwidth\leftmarginiv \advance\labelwidth-\labelsep
\topsep \z@\parsep \z@}
enumerate 環境  enumerate 環境はカウンタ enumi, enumii, enumiii, enumiv を使います。enumi は第 n レベルの番号です。

\theenumi 出力する番号の書式を設定します。これらは LATEX 本体（1tlists.dtx 参照）で定義済みで、ここでは表彰を変えています。\@arabic, \@alph, \@roman, \@Alph はそれぞれ算用数字、小文字アルファベット、小文字ローマ数字、大文字アルファベットで番号を出力する命令です。

\renewcommand{\theenumi}{\@arabic\c@enumi}
\renewcommand{\theenumii}{\@alph\c@enumii}
\renewcommand{\theenumiii}{\@roman\c@enumiii}
\renewcommand{\theenumiv}{\@Alph\c@enumiv}

\labelenumi enumerate 環境の番号を出力する命令です。第 2 レベル以外は最後に英文のピリオドが付きますが、これは好みに応じて取り払ってください。第 2 レベルの番号のかっこは和文用に換え、その両側に入る余分なグルーを \inhibitglue で取り除いています。

\newcommand*{\jsInJaParen}{[1]{\mbox{\jsInhibitGlue}(#1)\jsInhibitGlue}}
\newcommand{\labelenumi}{\theenumi.}
\newcommand{\labelenumii}{\jsInJaParen{\theenumii}}
\newcommand{\labelenumiii}{\theenumiii.}
\newcommand{\labelenumiv}{\theenumiv.}

\p@enumii \p@enumiii は \ref コマンドで enumerate 環境の第 n レベルの項目が参照されるときの書式です。これも第 2 レベルは和文用かっこのありでした。

\p@enumiv \renewcommand{\p@enumii}{\theenumii}
\renewcommand{\p@enumiii}{\theenumii\jsInhibitGlue(\theenumii )}
\renewcommand{\p@enumiv}{\p@enumiii\theenumiii}

itemize 環境  itemize 環境の第 n レベルのラベルを作るコマンドです。
\labelitemi
\labelitemii
\labelitemiii
\labelitemiv
■description 環境

description (env.) 本来の description 環境では、項目名が短いと、説明部分の頭がそれに引きずられて左に出てしまいます。これを解決した新しい description の実装です。

\begin{environment}{description}{%
\list{}{%
\labelwidth=\leftmargin
\labelsep=1\textwidth
\advance \labelwidth by -\labelsep
\let \makelabel=\descriptionlabel}\endlist}

\descriptionlabel description 環境のラベルを出力するコマンドです。好みに応じて #1 の前に適当な空き（たとえば \hspace{1\textwidth}）を入れるのもいいと思います。

\newcommand*{\descriptionlabel}[1]{\normalfont\headfont #1\hfil}

■概要

abstract (env.) 概要（要旨、梗概）を出力する環境です。book クラスでは各章の初めにちょっとしたことを書くのに使います。titlepage オプション付きの article クラスでは、独立したページに出力されます。abstract 環境は元は quotation 環境で作られていたが、quotation 環境の右マージンをゼロにしたので、list 環境で作り直しました。

JSPF スタイルでは実際の出力は \maketitle で行われます。

bxjsreport クラスの abstract 環境は:

• \texttt{layout=v1} の場合は jsbook + report の動作を継承する。つまり jsbook と同じになる。
• \texttt{layout=v2} の場合は新設の jsreport の動作を継承する。つまり jsarticle (+ titlapage) と同じになる。

chapterabstract (env.) jsbook の abstract 環境（「各章の初めにちょっとしたことを書く」ためのもの）を chapterabstract と呼ぶことにする。

\begin{environment}{chapterabstract}{%
\begin{list}{%\itemindent=1\textwidth\rightmargin=0pt\leftmargin=5\textwidth}{%\item\endlist}

\vspace{\baselineskip}\end{environment}

101
“普通の” abstract 環境の定義。

BXJS クラスでは、概要の最初の段落に段落下げが入るようにする。

102
■キーワード

keywords (env.) キーワードを準備する環境です。実際の出力は \maketitle で行われます。

■verse 環境

verse (env.) 詩のための verse 環境です。
■ quotation 環境

quotation (env.) 段落の頭の字下げ量を 1.5em から \parindent に変えました。また、右マージンを 0 にしました。

2969 \item \relax \{\endlist

2970 \newenvironment{quotation}{%  
2971 \list{}{\%  
2972 \listparindent\parindent  
2973 \itemindent\listparindent  
2974 \rightmargin \z@\%  
2975 \item \relax \{\endlist

■ quote 環境

quote (env.) quote 環境は、段落がインデントされないことを除き、quotation 環境と同じです。

2976 \newenvironment{quote}{}%  
2977 \{\list{}{\rightmargin\z@}\item \relax \{\endlist

■ 定理など lthm.dtx 参照。たとえば次のように定義します。

\newtheorem{definition}{定義}%  
\newtheorem{axiom}{公理}%  
\newtheorem{theorem}{定理}%

[2001-04-26] 定理の中はイタリック体になりましたが、それでは和文がゴシック体になってしまいますので、\itshape を削除しました。

[2009-08-23] \bfseries を \headfont に直し、\labelsep を 1zw にし、括弧を全角にしました。

2978 \def\@begintheorem#1#2{\trivlist\labelsep=1\jsZw  
2979 \item[\hskip \labelsep]{\headfont #1\ #2}}%  
2980 \def\@opargbegintheorem#1#2#3{\trivlist\labelsep=1\jsZw  
2981 \item[\hskip \labelsep]{\headfont #1\ #2 (\#3)}}

titlepage (env.) タイトルを独立のページに出力するのに使われます。

[2017-02-24] コミュニティ版 pLaTeX の標準クラス 2017/02/15 に合わせて、book クラスでタイトルを必ず奇数ページに送るようにしました。といっても、横組クラスしかありませんでしたので、従来の挙動は何も変わっていません。また、book 以外の場合のページ番号のリセットもコミュニティ版 pLaTeX の標準クラス 2017/02/15 に合わせましたので、こちらも片面印刷あるいは独立のタイトルページを作らないクラスばかりでしたので、従来の挙動は何も変わらずに済みました。

2982 \newenvironment{titlepage}{%  
2983 %<book> \pltx@cleartooddpage % 2017-02-24  
2984 \if@twocolumn  
2985 \restonecoltrue\onecolumn  
2986 \else
付録

\appendix 本文と付録を分離するコマンドです。

8.4 パラメータの設定

array と tabular 環境

\arraycolsep array 環境の列間には \arraycolsep の 2 倍の幅の空きが入ります。
\setlength{\arraycolsep}{5\jasc@mpt}

\tabcolsep tabular 環境の列間には \tabcolsep の 2 倍の幅の空きが入ります。
\setlength{\tabcolsep}{6\jasc@mpt}
\arrayrulewidth array, \texttt{tabular} 環境内の罫線の幅です。
3022 \setlength{\arrayrulewidth}{.4\p@}
\doublerulesep array, \texttt{tabular} 環境での二重罫線間のアキです。
3023 \setlength{\doublerulesep}{2\p@}

■ \texttt{tabbing} 環境
\texttt{tabbingsep} \' コマンドで入るアキです。
3024 \setlength{\tabbingsep}{\labelsep}

■ \texttt{minipage} 環境
\texttt{@mpfootins} \texttt{minipage} 環境の脚注の \texttt{\skip@mpfootins} は通常のページの \texttt{\skip\footins} と同じ働きをします。
3025 \texttt{\skip@mpfootins} = \texttt{\skip\footins}

■ \texttt{framebox} 環境
\texttt{\fboxsep} \texttt{\fbox}, \texttt{\framebox} で内側のテキストと枠との間の空きです。
3026 \setlength{\fboxsep}{3\jots}
3027 \setlength{\fboxrule}{.4\p@}

■ \texttt{equation} と \texttt{eqnarray} 環境
\texttt{\theequation} 数式番号を出力するコマンドです。
3028 \%<!book&!report>\renewcommand{\theequation}{\@arabic\c@equation}
3029 \%<*book|report>
3030 \%\addtoreset{equation}{chapter}
3031 \renewcommand{\theequation}
3032 \%\ifnum \c@chapter>\z@ \thechapter.\fi \@arabic\c@equation
3033 \%</book|report>
\texttt{\jot} \texttt{eqnarray} の行間に余分に入るアキです。デフォルトの値をコメントアウトして示しておきます。
3034 \% \setlength{\jot}{3pt}
\texttt{\@eqnnum} 数式番号の形式です。デフォルトの値をコメントアウトして示しておきます。
\texttt{\theequation} \texttt{\@InhibitGlue}（\texttt{\theequation}）\texttt{\@InhibitGlue} のように和文かっこを使うことも可能です。
3035 \% \def{\@eqnnum}{{\theequation}}
\texttt{amsmath} パッケージを使う場合は \texttt{\tagform@} を次のように修正します。
3036 \% \def{\tagform@}{\maketag@@@{\ignorespaces#1\unskip\@@italiccorr}}
8.5 フロート

タイプ TYPE のフロートオブジェクトを扱うには、次のマクロを定義します。

\fps@TYPE フロートを置く位置（float placement specifier）です。
\ftype@TYPE フロートの番号です。2 の累乗（1, 2, 4, ...) でなければなりません。
\ext@TYPE フロートの目次を出力するファイルの拡張子です。
\fnum@TYPE キャプション用の番号を生成するマクロです。
\@makecaption\langle num\rangle\langle text\rangle キャプションを出力するマクロです。\langle num\rangle は \fnum@... の生成する番号、\langle text\rangle はキャプションのテキストです。テキストは適当な幅の \parbox に入ります。

figure 環境

\c@figure 図番号のカウンタです。
\thefigure 図番号を出力するコマンドです。
\fps@figure figure のパラメータです。\figurename の直後に - が入っていましたが、ここでは外し \ftype@figure ました。
\ext@figure \def\fps@figure{tbp}
\fnum@figure \def\ftype@figure{1}
\def\fnum@figure{\figurename\nobreak\thefigure}

figure (env.) * 形式は段抜きのフロートです。
figure* (env.) \newenvironment{figure}{\@float{figure}}{\end@float} \newenvironment{figure*}{\@dblfloat{figure}}{\end@dblfloat}

table 環境

\c@table 表番号カウンタと表番号を出力するコマンドです。アスキー版では \thetable. が \thetable \thetable{}・になっていますが、ここではオリジナルのままにしています。
table のパラメータです。	ablename の直後に - が入っていましたが、ここでは外しました。

\begin{table}[htb]
\centering
\begin{tabular}{|c|c|c|}
\hline
A & B & C \\
\hline
1 & 2 & 3 \\
\hline
\end{tabular}
\caption{Caption for table}
\end{table}

8.6 キャプション

\@makecaption コマンドにより呼び出され、実際にキャプションを出力するコマンドです。第 1 引数はフロートの番号、第 2 引数はテキストです。

\abovecaptionskip それぞれキャプションの前後に挿入されるスペースです。\belowcaptionskip が 0 になっていたので、キャプションを表の上につけた場合にキャプションと表がくっついてしまうのを直しました。

実際のキャプションを出力します。オリジナルと異なり、文字サイズを \textsize にし、キャプションの幅を 2cm 狭くしました。

[2003-11-05] ロジックを少し変えてみました。

[2018-12-11] 遅くなりましたが、\texttt{listings} パッケージを使うときに title を指定すると “1zw” が出力されてしまう問題 (forum:1543, Issue #71) 対処しました。
フォントコマンド

ここでは \LaTeX 2.09 で使われていたコマンドを定義します。これらはテキストモードと数式モードのどちらでも動作します。これらは互換性のためのもので、できるだけ \text... と \math... を使ってください。

[2016-07-15] KOMA-Script 中の \scr@DeclareOldFontCommand に倣い、これらの命令を使うときには警告を発することにしました。
警告を最初の一回だけ発することにしました。また、例外的に警告を出さないようにするスイッチも付けます。

\if@jsc@warnoldfontcmd
\if@jsc@warnoldfontcmdexception
BXJSクラスでは不使用。
\if@jsc@warnoldfontcmdexception
は\allow/disallowoldfontcommandsの状態を表す。
\newif\if@jsc@warnoldfontcmd
\@jsc@warnoldfontcmdtrue
\newif\if@jsc@warnoldfontcmdexception
\@jsc@warnoldfontcmdexceptionfalse

\jscDeclareOldFontCommand
\newcommand*{\jscDeclareOldFontCommand}[3]{%
\g@addto@macro\bxjs@oldfontcmd@list{\do#1}%
\DeclareOldFontCommand[#1]{%\bxjs@oldfontcmd(#1)#2}%
\bxjs@oldfontcmd(#1)#3%}
\DeclareRobustCommand*{\jsc@warnoldfontcmd}{%
\ClassInfo\bxjs@clsname
{Old font command '{\string#1}' is used!!\MessageBreak
The first occurrence is}%
}
\allowoldfontcommands“二文字フォント命令”の使用を許可する（警告しない）。
\disallowoldfontcommands“二文字フォント命令”の使用に対して警告を出す。
\DeclareRobustCommand*{\allowoldfontcommands}{%\@jsc@warnoldfontcmdexceptiontrue}
\DeclareRobustCommand*{\disallowoldfontcommands}{%\@jsc@warnoldfontcmdexceptionfalse}
\let\bxjs@oldfontcmd@list\@empty
\def\bxjs@oldfontcmd#1{%
\expandafter\bxjs@oldfontcmd@a\csname bxjs@ofc/\string#1\endcsname#1}
\def\bxjs@oldfontcmd@a#1#2{%
\if@jsc@warnoldfontcmdexceptionelse
\global\@jsc@warnoldfontcmdexceptionfalse
\ifx#1\relax
\global\let#1=t%
\@jsc@warnoldfontcmdfalse
\fi
\fi
\fi
110
単純に \AtEndDocument のフックの中で \bxjs@warnoldfontcmd@final を実行した場合、最終ページのヘッダ・フッタの中にある二文字フォント命令はそれより後に実行されるため捕捉できない。これに対処するため、\end{document} 中に実行される \clearpage の処理の直後に \bxjs....final が呼ばれるようにする。
※新しい \LaTeX \ではフックシステムの機能を利用する。

\iffx\bxjs@old@hook@system
\AtEndDocument{\g@addto@macro\clearpage{\bxjs@warnoldfontcmd@final}}
\else
\AddToHook{enddocument/afterlastpage}{\bxjs@warnoldfontcmd@final}
\fi

\bsc ボールドシリーズにします。通常のミディアムシリーズに戻すコマンドは \mdseries です。
\icsc\jsscDeclareOldFontCommand{\mc}{\normalfont\mcfamily}{\mathmc}
\icsc\jsscDeclareOldFontCommand{\gt}{\normalfont\gtfamily}{\mathgt}
\icsc\jsscDeclareOldFontCommand{\rm}{\normalfont\rmfamily}{\mathrm}
\icsc\jsscDeclareOldFontCommand{\sf}{\normalfont\sffamily}{\mathsf}
\icsc\jsscDeclareOldFontCommand{\tt}{\normalfont\ttfamily}{\mathit}
\icsc\jsscDeclareOldFontCommand{\bf}{\normalfont\bfseries}{\mathbf}
\icsc\jsscDeclareOldFontCommand{\it}{\normalfont\itshape}{\mathit}

111
10 相互参照

10.1 目次の類

\section コマンドは .toc ファイルに次のような行を出力します。

\contentsline {section}{タイトル}{ページ}

たとえば \section に見出し番号が付く場合、上の「タイトル」は

\numberline {番号}{見出し}

となります。この「番号」は \thesection コマンドで生成された見出し番号です。

figure 環境の \caption コマンドは .lof ファイルに次のような行を出力します。

\contentsline {figure}{
umberline {番号}{キャプション}}{ページ}

この「番号」は \thefigure コマンドで生成された図番号です。

table 環境も同様です。

\contentsline {...}{\l@... というコマンドを実行するので、あらかじめ \l@chapter, \l@section, \l@figure などを定義しておかなければなりません。これらのもとは \l@dottedtocline コマンドを使って定義します。これは

\l@dottedtocline{レベル}{インデント}{幅}{タイトル}{ページ}

という書式です。

レベル この値が \tocdepth 以下のときだけ出力されます。\chapter はレベル 0, \section はレベル 1, 等々です。

インデント 左側の字下げ量です。

幅 「タイトル」に \numberline コマンドが含まれる場合、節番号が入る箱の幅です。

\@pnumwidth ページ番号の入る箱の幅です。

\@tocrmarg 右マージンです。\@tocrmarg ≥ \@pnumwidth とします。

\@dotsep 点の間隔です（単位 mu）。

\c@tocdepth 目次ページに出力する見出しレベルです。元は article で 3, その他で 2 でしたのが、ここでは一つずつ減らしています。
■目次

\tableofcontents

目次を生成します。

\newcommand\prechaptername

[2013-12-30] \prechaptername

などから見積もった目次のラベルの長さです。(by ts)

\newdimen\jstocwidth

\newcommand{\tableofcontents}{%

%<*book|report>

\settowidth\jstocwidth{\headfont\prechaptername%postchaptername}%

%<book\report>

\settowidth@tempdima{\headfont\appendixname}%

\ifdim\jstocwidth<\@tempdima \setlength\jstocwidth{\@tempdima} \fi

\ifdim\jstocwidth<2\jstw 

\divide\jstocwidth by 2 \advance\jstocwidth 1\jstw\fi

\if@twocolumn

\@restonecoltrue\onecolumn

\else

\@restonecolfalse

\chapter*{\contentsname}%

%</book\report>

%<book\report> \if@restonecol\twocolumn\fi

}\section*{\contentsname}%

%</book\report>

%<book\report>

%}</book\report>

%<!book&!report>

\settowidth\jstocwidth{\headfont\presectionname\postsectionname}%

%<!book\report>

%<!book\report>

\settowidth\jstocwidth{\headfont\appendixname}%

\ifdim\jstocwidth<\@tempdima \setlength\jstocwidth{\@tempdima} \fi

\ifdim\jstocwidth<2\jstw 

\divide\jstocwidth by 2 \advance\jstocwidth 1\jstw\fi

\section*{\contentsname}%

\@mkboth{\contentsname}{\contentsname}%

%</!book\report>

%</!book\report>

%</!book\report>

\@starttoc{toc}%

%<book\report> \if@restonecol\twocolumn\fi

%}</book\report>

\@part

部の目次です。

\newcommand*{\l@part}[2]{%}

\ifnum \c@tocdepth >-2 \relax

%<!book\report> \addpenalty@secpenalty

%<book\report> \addpenalty{-%@highpenalty}\%

\addvspace(2.25em @plus\jstw)\%

\begingroup

\parindent \z\%

% \pnumwidth should be \tocrmarg

\rightskip \pnumwidth

\rightskip \tocrmarg
章の目次です。

[2013-12-30] \@lnumwidth を \jsectoclwidth から決めるようにしてみました。（by ts）
インデントと幅はそれぞれ 1.5em, 2.3em でしたが、1zw, 3.683zw に変えました。

[2013-12-30] 上のインデントは jsc@tocl@width から決めるようにしました。(by ts)

さらに下位レベルの目次項目の体裁です。あまり使ったことがありませんので、要修正かも

[2013-12-30] ここも jsc@tocl@width から決めるようにしてみました。(by ts)
\numberline 欧文版 LaTeX では numberline(...) は幅 \@tempdima の箱に左詰めで出力する命令ですが、アスキー版では \@tempdima の代わりに \@lnumwidth という変数で幅を決めるように再定義しています。後続文字が全角か半角かでスペースが変わらないように \hspace を入れておきました。
\newdimen\@lnumwidth
\def\numberline#1{\hbox{$#1\hfil}\hspace{0pt}}

\@dottedtocline\LaTeX\本体（ltsect.dtx 参照）での定義と同じですが、\@tempdima を \@lnumwidth に変えています。
\renewcommand{\jsTocLine}{\leaders{\hbox{\hss.\hss}}\hfill}

とします。
\def\jsTocLine{\leaders\hbox{\hss}\hfill}

■図目次と表目次
\listoffigures 図目次を出力します。
\newcommand{\listoffigures}{%
図目次の項目を出力します。
\newcommand*{\l@figure}{\@dottedtocline{1}{1.5em}{3.68em}}

表目次を出力します。
\newcommand{\listoftables}{\if@twocolumn\@restonecoltrue\onecolumn
\else\@restonecolfalse\fi
\chapter*{\listtablename}
\@mkboth{\listtablename}{\listtablename}\}

表目次は図目次と同じです。
\let\l@table\l@figure

10.2 参考文献

オープンスタイルの参考文献で使うインデント幅です。元は 1.5em でした。
\newdimen\bibindent
\setlength{\bibindent}{2em}
\newenvironment{thebibliography}{\@jsc@warnoldfontcmdexceptiontrue
\global\let\presectionname\relax
\global\let\postsectionname\relax
\begin{thebibliography}{1}{1}{\bibindent}
参考文献リストを出力します。
[2016-07-16] \LaTeX 2.09 で使われていたフォントコマンドの警告を、文献スタイル（.bst）
ではよく \bf がいまだに用いられることが多いため、thebibliography 環境内では例外的
に出さないようにしました。
はデフォルトでは小さなスペースを生成します。\newblock \newcommand{\newblock}{\hskip .11em\@plus.33em\@minus.07em}

\bibitem のラベルを作ります。ltbibl.dtx で定義されていますが、コンマとかっこを和文フォントにするには次のようにします。とりあえずコメントアウトしておきましたので、必要に応じて生かしてください。

\cite 文献の番号を出力する部分は ltbibl.dtx で定義されていますが、コンマとかっこを和文フォントにするには次のようにします。とりあえずコメントアウトしておきましたので、必要に応じて生かしてください。かっこ前後に入るグルーを \inhibitGlue で取っていきますので、オリジナル同様、Knuth\cite{knu} と同じように半角空白で囲んでください。

\begin{thebibliography}{118}
\end{thebibliography}
引用番号を上ツキの1)のようなスタイルにするには次のようにします。\citeの先頭に\unskipを付けて先行のスペース（〜も）を帳消しにしています。

10.3 索引

theindex（env.）2〜3段組の索引を作成します。最後が偶数ページのときにマージンがずれる現象を直しました（Thanks:藤村さん）。
索引項目の字下げ幅です。\@idxitem は \item の項目の字下げ幅です。
\subitem \newcommand{\@idxitem}{\par\hangindent 4\jsZw} % 元 40pt
\subsubitem \newcommand{\@idxitem}{\par\hangindent 6\jsZw} % 元 30pt

索引で先頭文字ごとのブロックの間に入るスペースです。
\newcommand{\indexspace}{\par \vskip 10\jsc@mpt \@plus5\jsc@mpt \@minus3\jsc@mpt \relax}

索引の \see, \seealso コマンドで出力されるものです。デフォルトはそれぞれ see, see also という英語ですが、ここではとりあえず両方とも「→」に変えました。⇒（$\Rightarrow$）などでもいいでしょう。
\newcommand{\seename}{\if@english see\else \rightarrow \fi}
\newcommand{\alsoname}{\if@english see also\else \rightarrow \fi}

10.4 脚注

和文の句読点・閉じかっこ類の直後で用いた際に余分なアキが入るのを防ぐため、\inhibitglue を入れることにします。pLIPeX の日付が 2016/09/03 より新しい場合は、このパッチが必要なのでああんせません。

パッチの必要性は「\pltx@foot@penalty が未定義か」で行う。\inhibitglue の代わりに \jsInhibitGlue を使う。

\newcommand{\pltx@foot@penalty}{\@undefined}
\def{\footnote}{\jsInhibitGlue\footnote@ve}
\def{\footnotemark}{\jsInhibitGlue\footnotemarks@ve}
\fi

脚注番号を付ける命令です。ここでは脚注番号の前に記号 * を付けています。「注 1」の形式に
するには `\textasteriskcentered` を注 `\kern0.1em` にしてください。`@xfootnotenext` と合わせて、もし脚注番号が空なら記号も出力しないようにしてあります。

[2002-04-09] インプリメントの仕方を変えたため消しました。


[2013-05-14] pcore.ltx に倣った形に書き直しました（Thanks: 北川さん）。


pTeX 依存のコードなので、minimal 和文ドライバ実装に移動。

\textbackslash thefootnote 脚注番号に * 印が付くようにしました。ただし、番号がゼロのときは * 印も脚注番号も付きません。

[2003-08-15] `\textasteriskcentered` ではフォントによって下がりすぎるので変更しました。

[2016-10-08] TODO: 脚注番号が `newtxtex` や `newpхтex` の使用時におかしなかっ

3474 \def\thefootnote\lbrace\ifnum\c@footnote>\z@ \leavevmode\lower.5ex\hbox{*}\@arabic\c@footnote\fi\rbrace

3475 \% \def\thefootnote\lbrace\ifnum\c@footnote>\z@ \leavevmode\lower.5ex\hbox{*}\@arabic\c@footnote\fi\rbrace

\textbackslash footnoterule 本文と脚注の間の簿線です。

3476 \renewcommand\{\footnoterule\}{% \kern-2.6\jsc@\@pt \kern-.4\p@ \hrule width .4\columnwidth \kern 2.6\jsc@\@pt\}

\textbackslash c@footnote 脚注番号は章ごとにリセットされます。

3480 \%<book\report\@$addtoreset{footnote}\{chapter\}$

\textbackslash @footnotetext 脚注で `\verb` が使えるように改変してあります。Jeremy Gibbons, _T_{\TeX} and TUG NEWS_, Vol. 2, No. 4 (1993), p. 9

3477 \% `kern-2.\jsc@\@pt \kern-.4\p@`
v1.2e で parhook 対応として \parhook が追加されていますが、実は同時に \color@endgroup
も \endgraf するように変更されていますので、不要だと思います。というわけで追加しま
せん。
\long\def\@footnotetext{%
  \insert\footins\bgroup
  \normalfont\footnotesize
  \interlinepenalty\interfootnotelinepenalty
  \splittopskip\footnotesep
  \splitmaxdepth \dp\strutbox \floatingpenalty \@MM
  \hsize\columnwidth \@parboxrestore
  \def\@currentcounter{footnote}\
  \protected@edef\@currentlabel{%
    \csname p@footnote\endcsname\@thefnmark}
  \color@begingroup
  \@makefntext{\rule\z@\footnotesep\ignorespaces}
  \futurelet\jsc@next\jsc@fo@t
  \def\jsc@fo@t\{\ifcat\bgroup\noexpand\jsc@next \let\jsc@next\jsc@f@@t
  \else \let\jsc@next\jsc@f@t\fi \jsc@next
  \def\jsc@f@@t\{\bgroup\aftergroup\jsc@@foot\let\jsc@next\}
  \def\jsc@f@t#1{#1\jsc@@foot
  \def\jsc@@foot\{\@finalstrut\strutbox\color@endgroup\egroup
  \ifx\pltx@foot@penalty\@undefined\else
    \ifhmode\null\fi
    \ifnum\pltx@foot@penalty=\z@\else
      \penalty\pltx@foot@penalty
      \pltx@foot@penalty\z@\fi
    \fi\fi\}
\@makefntext 実際に脚注を出力する命令です。\@makefntext は脚注の番号を出力する命令です。ここでは脚注が左端から一定距離に来るようにしてあります。
\newcommand\@makefntext[1]{%\rule\z@\footnotesep\ignorespaces}\
  \futurelet\jsc@next\jsc@fo@t
  \def\jsc@fo@t\{\ifcat\bgroup\noexpand\jsc@next \let\jsc@next\jsc@f@@t
  \else \let\jsc@next\jsc@f@t\fi \jsc@next
  \def\jsc@f@@t\{\bgroup\aftergroup\jsc@@foot\let\jsc@next\}
  \def\jsc@f@t#1{#1\jsc@@foot
  \def\jsc@@foot\{\@finalstrut\strutbox\color@endgroup\egroup
  \ifx\pltx@foot@penalty\@undefined\else
    \ifhmode\null\fi
    \ifnum\pltx@foot@penalty=\z@\else
      \penalty\pltx@foot@penalty
      \pltx@foot@penalty\z@\fi
    \fi\fi}
\@xfootnotenext 最初の \footnotetext{...} は番号が付きません。著者の所属などを脚注の欄に書くときに便利です。
  すでに \footnote を使った後なら \footnotetext[0]{...} とすれば番号を付けない
脚注になります。ただし、この場合は脚注番号がリセットされてしまうので、工夫が必要
です。
[2002-04-09] インプリメントの仕方を変えたため消しました。
ここまでのコードは JS クラスを踏襲する。

11 段落の頭へのグルー挿入禁止

段落頭のかぎかっこなどを見かけた字半下げから全角1字下げに直します。

\jsInhibitGlueAtParTop「段落頭の括弧の空き補正」の処理を \jsInhibitGlueAtParTop という命令にして、これ
を再定義可能にした。
\let\jsInhibitGlueAtParTop\@empty
\everyparhook 全ての段落の冒頭で実行されるフック。これの初期値を先述の \jsInhibitGlueAtParTop
とする。
\def\everyparhook{\jsInhibitGlueAtParTop}
\ifnum\bxjs@everyparhook=\bxjs@everyparhook@@compat
\g@addto@macro\bxjs@begin@document@hook{\everypar{\everyparhook}}
\fi

[2016-07-18] \inhibitglue の発行対象を \inhibitxspcode が 2 に設定されているものすべてに拡大しました。
[2016-12-01] すぐ上の変更で @tempa を使っていたのがよくなかったので、プレフィックスを付けて \jsc@tempa にしました (forum:2085)。
[2017-02-13] \jsc@tempa は実はテンポラリではなく「この処理専用のユニーク制御綴」である必要があります。間違って別の箇所で使う危険性が高いので, 専用の命令 \jsc@ig@temp に置き換えました (Issue #54)。

次の \@inhibitglue は JS クラスでの \jsInhibitGlueAtParTop の実装である。エンジンが (u)platex の場合はこれを採用する。
ここからしばらく「（本物の）\everypar に追加した \everyparhook を保持する」ためのパッチ処理が続く。これは、\everyparhook=compat の場合にのみ実行する。

これだけではいけないようですね。あちこちに \everypar を初期化するコマンドが隠されていた。

まず、環境の直後の段落です。

\[2016-11-19\] ltlists.dtx 2015/05/10 v1.0t の変更に追随して \clubpenalty のリセットを追加しました。

\[2017-08-31\] minipage 環境にも対策します。
\item 命令の直後です。
二つ挿入した `everyparhook` のうち後者が `section` 類の直後に 2 回、前者が 3 回目以降に実行されます。

「`everyparhook` 用のパッチ処理」はここまで。

※ `luatexja` を読み込んだ場合に `ltjcore.sty` によって上書きされるのを防ぐため遅延させる。
12 いろいろなロゴ

LaTeX 関連のロゴを作り直します。
[2016-07-14] ロゴの定義は jslogo パッケージに移転しました。後方互換のため、jsclasses ではデフォルトでこれを読み込みます。nojslogo オプションが指定されている場合は読み込みません。

BXJS クラスでも jslogo オプション指定の場合に jslogo パッケージを読み込むようにした。ただし JS クラスと異なり、既定では読み込みない。
※\小、\上小の制御綴は定義しない。

13 amsmath との衝突の回避

amsmath パッケージでは行列中で \@ifnextchar を再定義していますが、これが WiFiX の ProvidesFile で悪さをする例が FTeX で報告されています。これを避けるための tDB さんのフィックスを挿入しております。副作用がありましたらお知らせください。

この現象については私の TeX 揭示板 4273〜、16058〜で議論がありました。なお、AMS 関係のパッケージを読み込む際に psamsfonts オプションを与えても回避できません（Thanks: しっぽ愛好家さん）。
[2016-11-19] 本家の ltclass.dtx 2004/01/28 v1.1g で修正されているのでコメントアウトしました。
14 初期設定

■いろいろな語

\newcommand{\prepartname}{\if@english Part~\else 第\fi}
\newcommand{\postpartname}{\if@english\else 部\fi}
\newcommand{\prechaptername}{\if@english Chapter~\else 第\fi}
\newcommand{\postchaptername}{\if@english\else 章\fi}
\newcommand{\presectionname}{}
\newcommand{\postsectionname}{}
\newcommand{\contentsname}{\if@english Contents\else 目次\fi}
\newcommand{\listfigurename}{\if@english List of Figures\else 図目次\fi}
\newcommand{\listtablename}{\if@english List of Tables\else 表目次\fi}
\newcommand{\refname}{\if@english References\else 参考文献\fi}
\newcommand{\bibname}{\if@english Bibliography\else 参考文献\fi}
\newcommand{\indexname}{\if@english Index\else 索引\fi}
\newcommand{\figurename}{\if@english Fig.~\else 図\fi}
\newcommand{\tablename}{\if@english Table~\else 表\fi}
\newcommand{\appendixname}{\if@english Appendix~\else 付録\fi}
\newcommand{\abstractname}{\if@english Abstract\else 概要\fi}

■今日の日付

\LaTeX\ で処理した日付を出力します。和暦にするには \texttt{和暦} と書いてください。
環境変数 SOURCE_DATE_EPOCH / FORCE_SOURCE_DATE が設定されている場合は“今日”が過去・未来の日付になる可能性がある。BXJS クラスでは、和暦の扱いは bxwareki パッケージに任せる。
※ 2.0 版より、完全に bxwareki に任せる。

西暦 8 ビット欧文 TpX ではそもそも非 ASCII の制御綴は使えないのであるが、JS クラスのユーザ命令である \西暦/\和暦 だけは擬似的に使えるようにする。欧文 TpX では

- \西暦 = \``e8``a5``bf``e6``9a``a6
- \和暦 = \``e5``92``8c``e6``9a``a6

と扱われるため、``e8 と ``e5 を「固定の引数付のマクロ」として定義すればよい。もちろん、同じバイットで始まる他の名前（例えば \西暦 true）とは共存できないので、この 2 つのユーザ命令以外の非 ASCII の制御綴は使わないようにする。

TpX エンジンの種類により処理を分ける。

```latex
\ifjsSeireki [スイッチ] \西暦 スイッチ（\if 西暦）の代わりに用いる。
\newif\ifjsSeireki \jsSeirekitrue
\bxjs@decl@Seireki@cmds
\ifjsSeireki \jsSeirekitrue
\newif\if\西暦 \西暦 true
\bxjs@decl@Seireki@cmds
\def\西暦{\西暦 true}
\def\和暦{\西暦 false\bxjs@wareki@used}
\def\bxjs@if@use@seireki{\bxjs@cond\ifjsSeireki\fi}
\def\bxjs@iai{\noexpand~}
\Seireki \西暦/\和暦 の代わりになる ASCII 名の命令も（念のため）用意しておく。
\Wareki \newif\if\西暦 \西暦 true
\newif\if\ Wareki \西暦 false\bxjs@wareki@used
\def\bxjs@if@use@seireki{\bxjs@cond\ifjsSeireki{\fi}
\def\bxjs@iai{\noexpand~}
\Wareki \西暦 false\bxjs@wareki@used
```

8 ビット欧文 TpX の場合。

```latex
8 ビット欧文 TpX ではない場合。ここでは JS クラスと合わせるため \西暦 スイッチを使う。
\newif\if 西暦 \西暦 true
\newif\if\Wareki \西暦 false\bxjs@wareki@used
```
西暦における年の表記の「年」以前の部分（元号+年数）。
※heiseiの代替となる機能（だから常に和暦を扱う）。

heisei数を表す整数レジスタで、元号が「平成」である場合にのみ定義される。
※JSクラスと互換の機能。

ただしbxwarekiが使えない場合は西暦表示にフォールバックする。

today英語、西暦、和暦で場合分けをする。
※diffの都合のためまたjsclassesのコードを挿入する。
texjporg版の日本語用Babel定義ファイル（japanese.1df）が読み込まれた場合に影響を受けないようにする。

\g@addto@macro\bxjs@begin@document@hook{%  
  \ifx\bbl@jpn@maybekansuji\@undefined\else  
  \bxjs@decl@Seireki@cmds  
  \g@addto@macro\datejapanese{%  
    \let\today\bxjs@today}%  
  \fi}%

■ハイフネーション例外  TeX のハイフネーションルールの補足です（ペンディング：english）

\hyphenation{ado-be post-script ghost-script phe-nom-e-no-log-i-cal man-u-scripts}

■ページ設定  ページ設定の初期化です。

%<slide>\pagestyle{empty}%
%<article/report>\pagestyle{plain}%
%<book>\pagestyle{headings}%
pagenumbering{arabic}
\if@twocolumn\twocolumn\sloppy\flushbottom\else\onecolumn\raggedbottom\fi
%<*slide>\renewcommand\familydefault{\sfdefault}\raggedright%</slide>

15 実験的コード

この節はJSクラスの話で、BXJSクラスには当てはまらない。

[2016-11-29]コミュニティ版pLTExで新設されたテスト用パッケージ（exppl2eパッケージ）が文書クラスより先に読み込まれていた場合は、jsclassesもテスト版として動作します。この処置はjsarticle、jsbook、jsreportにのみ行い、jspfとkiyouは除外しておきます。exppl2eパッケージが読み込まれていない場合は通常版として動作しますので、ここで終了します。
以上です。
16 ＢＸＪＳ 独自の追加処理

■ "command" の補填 fontspec で提供される "command" 命令と strongenv 環境を全てのエンジンで使えるようにする。
※この実装は特にエンジンや和文処理パッケージに依存しないはずであるが、現状では standard 和文ドライバでの提供となっていて、そこで有効化のオプションが定義されていない。ここで "ja～" の名前で定義することにする。

\textbf{強調用の宣言型命令。}
\begin{verbatim}
\xdef\bxjs@robust@def\jqs\textbf{StrongText}{\bxjs@strong@text}\
\end{verbatim}

fontspec と互換の \textbf{strongfontdeclare} 命令も提供する。既定の設定は \textbf{bfseries} (太字)である。
※\textbf{strongfontdeclare} は試験的機能とする。
\begin{verbatim}
\xdef\bxjs@const@unit{#1}{#2}{#3}{#4}\
\end{verbatim}

共通命令の実装

\begin{verbatim}
\xdef\bxjs@const@unit\xdef}\bxjs@const@unit{#1}{#2}{#3}{#4}\
\end{verbatim}

\textbf{\TeX} 拡張が使えない場合は、何らかの寸法パラメタに値を保持する必要があるが、レジスタは貴重なので代わりに "ダミーの TFM を定義してその \textbf{fontdimen} を使う" というテク
ニックを用いる（アレ）。

\begin{verbatim}
else
  \let\bxjs@Ct\fontdimen \font\bxjs@Ut=cmtex9 at 0.98245pt
  \bxjs@Ct8\bxjs@Ut=8sp \bxjs@Ct16\bxjs@Ut=\z@
  \def\bxjs@const@unit#1#2{%
    \expandafter\bxjs@const@unit@a\csname bxjs@#2@\endcsname#1}
  \def\bxjs@const@unit@a#1#2#3{%
    \chardef#1\bxjs@Ct8\bxjs@Ut 
    \bxjs@advance@qc#1\@ne \bxjs@Ct8\bxjs@Ut#1sp
    \bxjs@Ct#1\bxjs@Ut=#3\relax \def#2{\bxjs@Ct#1\bxjs@Ut}}
  fi

\jQ \jQ と \jH はともに 0.25mm に等しい。
\jH \bxjs@const@unit\jQ{jQ}{0.25mm}
\let\jH\jQ
\trueQ \trueQ と \trueH はともに 0.25true mm に等しい。
\trueH \ifjsc\mag
  \@tempdimb=\jsBaseFontSize\relax
  \edef\bxjs@tmpa{\strip@pt\@tempdimb}\
  \@tempdima=2.5mm
  \bxjs@invscale\@tempdima\bxjs@tmpa
  \bxjs@const@unit\trueQ{trueQ}{\@tempdima}
  \@tempdima=10pt
  \bxjs@invscale\@tempdima\bxjs@tmpa
  \bxjs@const@unit\truept{truept}{\@tempdima}
  \else \let\trueQ\jQ \let\bxjs@truept\p@
  \fi
\let\trueH\trueQ
\end{verbatim}
\ascQ \ascQ は \trueQ を和文スケール値で割った値。例えば、\fontsize{12\ascQ}{16\trueH}
\ascpt とすると、和文が 12Q になる。

同様に、\ascpt は \truept を和文スケールで割った値。
\ascpt \bxjs@const@unit\ascQ{ascQ}{\@tempdima}
\ascpt \bxjs@const@unit\ascpt{ascpt}{\@tempdima}
\jfontsize \jfontsize{\langle \text{フォントサイズ} \rangle \langle \text{行送り} \rangle}： 和文フォント規準で、すなわち、1zw が \langle \text{フォントサイズ} \rangle に等しかなるようにフォントサイズを指定する。この命令の引数では、Q/H の単位が使用できる。
\DeclareRobustCommand*\jsJaFontSize[2]{%
  \begingroup
  \bxjs@jafontsize@a{#1}\
  \@tempdimb=\jsInverseScale\@tempdima
  \bxjs@jafontsize@a{#2}\
  \xdef\bxjs@g@tmpa{\noexpand\fontsize{\the\@tempdimb}{\the\@tempdima}}%
  \endgroup\bxjs@g@tmpa
\end{verbatim}

\end{verbatim}

134
続いて、和文間空白・和欧文間空白関連の命令を実装する。この実装はエンジンや和文処理パッケージに依存するが、ここでは共通の基盤となる部分を実装する。

\def\bxjs@kanjiskip{0pt}
\jsSetKanjiSkip
和文間空白の量を設定する。
※\setkanjiskipの実体。
\newcommand\jsGetKanjiSkip{\bxjs@kanjiskip}
\jsApplyKanjiSkip{⟨グルー値⟩}：和文間空白を実際に設定するためのエンジン依存のコード。
\def\bxjs@reset@kanjiskip{\if\bxjs@kanjiskip@enabled
\setlength{\@tempskipa}{\bxjs@kanjiskip}\else \@tempskipa\z@\fi}
\if\bxjs@kanjiskip@enabled 和文間空白の挿入が有効か。
※エンジン側の機能で制御する場合は、このスイッチは常に真にしておく。
\newif\ifbxjs@kanjiskip@enabled \bxjs@kanjiskip@enabledtrue
\jsEnableKanjiSkip 和文間空白の挿入を有効／無効にする。
\jsDisableKanjiSkip ※pTeX系のエンジンやluatexjaのパッケージを使用する場合はそれ自体がもつ制御機能を利用するため、これらの命令は使わない。
\bxjs@reset@kanjiskip 現在の和文間空白の設定を実際に反映させる。
和文間空白について同様のものを用意する。
\begin{verbatim}
\def\bxjs@xkanjiskip{0pt}
\DeclareRobustCommand*{\jsSetXKanjiSkip}[1]{%
  \bxjs@let@lenexpr\bxjs@xkanjiskip{#1}%
  \bxjs@reset@xkanjiskip}
\newcommand*{\jsGetXKanjiSkip}{\bxjs@xkanjiskip}
\newif\ifbxjs@xkanjiskip@enabled \bxjs@xkanjiskip@enabledtrue
\bxjs@robust@def{\jsEnableXKanjiSkip}{{%}
  \bxjs@xkanjiskip@enabledtrue
  \bxjs@reset@xkanjiskip}
\bxjs@robust@def{\jsDisableXKanjiSkip}{{%}
  \bxjs@xkanjiskip@enabledfalse
  \bxjs@reset@xkanjiskip}
\bxjs@robust@def{\bxjs@reset@xkanjiskip}{{%
  \ifbxjs@xkanjiskip@enabled
    \setlength{\@tempskipa}{\bxjs@xkanjiskip}%
  \else \@tempskipa\z@ \fi
  \jsApplyXKanjiSkip\@tempskipa}}
\let\jsApplyXKanjiSkip\@gobble
\end{verbatim}

和文・和欧文間空白の初期値。
\begin{verbatim}
\AtEndOfPackage{%
  \jsSetKanjiSkip{0pt plus .1\jsZw minus .01\jsZw}%
  \if\jsDocClass\jsSlide \jsSetXKanjiSkip{.1em}%%%%
  \else \jsSetXKanjiSkip{.25em plus .15em minus .06em}%%%%
  \fi
}
\end{verbatim}

■和文空白命令
\begin{verbatim}
\def\jaenspace{.5\jsZw}
\def\jathinspace{\bxjs@xkanjiskip}
\end{verbatim}

全角空白文字１つからなる名前の制御綴。\texttt{"zwspace} と等価になる。
\begin{verbatim}
\def\{"zwspace}
\end{verbatim}

非数式中では \texttt{\jathinspace} と等価になるように再定義する。
※数式中では従来通り（\:と等価）。
\begin{verbatim}
\bjxprotected\def\bjxchoice@jathinspace{\
\relax\ifmmode \mskip\medmuskip\
\else \jathinspace\ignorespaces\fi}
\jsAtEndOfClass{\
\ifjsWitheTeX \let\>\bjxchoice@jathinspace\
\else \def\>\protect\bjxchoice@jathinspace\%\
\fi}
\jaspace
\jlreq
クラスと互換の命令。
\begin{verbatim}
\DeclareRobustCommand*{\jaspace}[1]{\
\expandafter\ifx\csname bxjs@jaspace@@#1\endcsname\relax\ClassError\bxjs@clsname{Unknown jaspace: #1}{\@eha}\
\else \csname bxjs@jaspace@@#1\endcsname\fi}
\def\bxjs@jaspace@@zenkaku{\hskip 1\jsZw\relax}
\def\bxjs@jaspace@@nibu{\hskip .5\jsZw\relax}
\def\bxjs@jaspace@@shibu{\hskip .25\jsZw\relax}
\end{verbatim}
終わり。
\end{verbatim}
\jaspace jlreq クラスと互換の命令。
\begin{verbatim}
\DeclareRobustCommand*{\jaspace}[1]{\
\expandafter\ifx\csname bxjs@jaspace@@#1\endcsname\relax\ClassError\bxjs@clsname{Unknown jaspace: #1}{\@eha}\
\else \csname bxjs@jaspace@@#1\endcsname\fi}
\def\bxjs@jaspace@@zenkaku{\hskip 1\jsZw\relax}
\def\bxjs@jaspace@@nibu{\hskip .5\jsZw\relax}
\def\bxjs@jaspace@@shibu{\hskip .25\jsZw\relax}
\end{verbatim}
終わり。

■和文ドライバ読込
フックを実行する。
\begin{verbatim}
\bxjs@pre@jadriver@hook
和文ドライバのファイルを読み込む。
\input{bxjsja-\bxjs@jadriver.def}
おしまい。
\end{verbatim}

137
付録 A 和文ドライバの仕様

次の命令が BXJS クラス本体と和文ドライバの連携のために用意されている。このうち、★印を付けたものは “書込” が許されるものである。

- \jsDocClass ［文字トークンの let］文書クラスの種類を示し、次のいずれかと一致する（\if で判定可能）。
  \jsArticle bxjsarticle クラス
  \jsBook bxjsbook クラス
  \jsReport bxjsreport クラス
  \jsSlide bxjsslide クラス
- \jsEngine ［文字トークンの let］使用されているエンジンの種別。（\if で判定可能）。
  p pdfTeX（DVI モードも含む）
  l LuaTeX（〃）
  x XƎTeX
  j pTeX または upTeX
  n 以上の何れでもない
- \jsWithupTeX [スイッチ] 使用されているエンジンが upTeX であるか。
- \jsWithTeX [スイッチ] 使用されているエンジンが ε-TeX 拡張であるか。
- \jsInPdfMode [スイッチ] 使用されているエンジンが（pdfTeX・LuaTeX の）PDF モードであるか。
- \jsUnusualPtSize [整数定数を表す文字列のマクロ] 基底フォントサイズが 10pt、11pt、12pt のいずれでもない場合の @ptsize の値。（@ptsize 自体があまり有用でないと思われる。）
- \jsScale [実数を表す文字列のマクロ] 和文フォントサイズの要求サイズに対するスケール。クラスオプション scale で指定される。（既定値は 0.924715。）
- \jsJaFont [マクロ] 和文フォント設定を表す文字列。クラスオプション jafont で指定された値。
- \jsJaParam [マクロ] 和文モジュールに渡すパラメタを表す文字列。この値が何を表すかは決まってなくて、各々の和文モジュールが独自に解釈する。クラスオプション japaram で指定された値。
- \jsInhibitGlue [マクロ] inhibitglue という命令が定義されていればそれを実行し、そうでなければ何もしない。JS クラスで inhibitglue を用いている箇所は全て \jsInhibitGlue に置き換えられている。従って、inhibitglue は未定義でも動作するが、その実装がある場合は BXJS クラスはそれを活用する。
- \jsInhibitGlueAtParTop [マクロ] ★ 段落先頭におけるカギ括弧の位置調整を行うマクロ。全ての段落先頭で呼び出される。
- \jsZw [内部寸法値] 「現在の全角幅」を表す変数。JS クラスで zw 単位で設定されている長さパラメタはこの変数を単位として設定されている。この変数の値は実際に
用いられる「和文フォント」のメトリックに基づくのではなく、機械的に \_jsScale \times (フォントサイズ) であると定められている（フォントサイズ変更の度に再設定される。従って、「和文コンポーネント」はこの設定が辻褄が合うように和文フォントサイズを調整する必要がある。ほとんどの場合、和文フォントを NFSS で規定する際に \_jsScale の値をスケール値として与えれば上手くいく。

- \_jsFontSizeChanged [マクロ] フォントサイズが変更された時に必ず呼び出される（呼び出すべき）マクロ。
- \_jsResetDimen [マクロ] 上記 \_jsFontSizeChanged 中で呼び出される、ユーザ（和文モジュール）用のフック。フォントサイズに依存するパラメタをここで設定することができる。既定の定義は空。

以下で標準で用意されている和文ドライバの実装を示す。

### 付録 B 和文ドライバ：minimal

ja オプションの指定が無い場合に適用されるドライバ。また、standard ドライバはまずこのドライバファイルを読み込んでいます。

このドライバでは、各エンジンについての必要最低限の処理だけを行っている。日本語処理のためのパッケージ（xeCJK ルアTEX-ja 等）を自分で読み込んで適切な設定を行うという使用状況を想定している。

ただし、(u)pTEX エンジンについては例外で、和文処理機構の選択の余地がないため、このドライバにおいて、「JS クラスと同等の指定」を完成させるためのコードを記述する。

**TODO:** minimal のコード中に何を置べきかについて検討する。現状では、本来は「minimal にすら依存しない」はずのものが minimal 中に置かれている。

**TODO:** 3.0 とりあえず、新しい補助ファイルを導入する。文書クラスや和文ドライバの種別に関わらず必ず読み込まれるもの。

#### B.1 準備

このファイルは日本語文字を含みます

このファイルは日本語文字を含みます

■環境検査 minimal 和文ドライバの処理系バージョン要件はクラス本体と同じとする。

ただし「公式にはサポート外」のエンジンが使われている場合は強制終了させる。

※ NTT jTEX と Omega 系。

```latex
\let\bxjs@tmpa\relax
\ifx J\jsEngine \def\bxjs@tmpa{NTT-jTeX}\fi
\ifx O\jsEngine \def\bxjs@tmpa{Omega}\fi
\ifx\bxjs@tmpa\relax \expandafter\@gobble
\else
\ClassError\bxjs@clsname
\{The engine in use (\bxjs@tmpa) is not supported\}
```

139
It's a fatal error. I'll quit right now.

■補助マクロ

\DeclareJaTextFontCommand 和文書体のための、「余計なこと」をしない \DeclareTextFontCommand。

\DeclareJaMathFontCommand 和文数式フォントが無効な場合に、それをエミュレートするもの。

\bxjs@if@sf@default \familydefault の定義が “sfdefault” である場合に引数のコードを実行する。

\jsInverseScale \jsScale の逆数。

\jsLetHeadChar \CS{（トークン列）}： トークン列の先頭の文字を抽出し、\CS をその文字トークン（に展開されるマクロ）として定義する。
\def\jsLetHeadChar#1#2{\%}
\begingroup
\escapechar=`\ %
\let\bxjs@tmpa={brace-match-hack}
\bxjs@let@hchar@exp#2}\%
\endgroup
\let#1\bxjs@g@tmpa}
\def\bxjs@let@hchar@exp{\%}
\futurelet\@let@token\bxjs@let@hchar@exp@a}
\def\bxjs@let@hchar@exp@a{\%}
\bxjs@cond\ifcat\noexpand\@let@token\bgroup\fi{波括弧}
\bxjs@let@hchar@out\let\relax
\bxjs@cond\ifcat\noexpand\@let@token@sptoken\fi{空白}
\bxjs@let@hchar@out\let\space%
\bxjs@cond\if\noexpand\@let@token@backslashchar\fi{バックスラッシュ}
\bxjs@let@hchar@out\let@backslashchar
\{(\bxjs@let@hchar@exp@b)})}
\def\bxjs@let@hchar@exp@b#1{\%
\expandafter\bxjs@let@hchar@exp@c\string#1?\@nil#1}
\def\bxjs@let@hchar@exp@c#1#2\@nil{\%
%\message{<#1#2>}%
\bxjs@cond\if#1@backslashchar\fi{制御綴}
\bxjs@cond\expandafter\ifx\noexpand\@let@token\@let@token\fi{\%
\bxjs@let@hchar@out\let\relax
}%\else
\expandafter\bxjs@let@hchar@exp
%\else
\bxjs@let@hchar@chr#1%
}\}
\def\bxjs@let@hchar@chr#1{%}
\def\bxjs@let@hchar@out@def{(#1))}
\def\bxjs@let@hchar@out@#1{\%
\global@#1\bxjs@@g@tmpa@#2\relax
\toks@bgroup}@skip to right brace
UTF-8 のバイト列を扱うコード。
\chardef\bxjs@let@hchar@csta=128
\chardef\bxjs@let@hchar@cstb=192
\chardef\bxjs@let@hchar@cstc=224
\chardef\bxjs@let@hchar@cstd=240
\chardef\bxjs@let@hchar@cste=248
\let\bxjs@let@hchar@chr@ue@a\bxjs@let@hchar@chr
\def\bxjs@let@hchar@chr@ue#1{%}
\@tempcnta=`\#1\relax
%\message{the@\@tempcnta}%
\bxjs@cond\ifnum\@tempcnta<\bxjs@let@hchar@csta\fi{%}
\bxjs@let@hchar@chr@ue@a@#1%
\{(\bxjs@cond\ifnum\@tempcnta<\bxjs@let@hchar@cstb\fi{%
B.2 (u)pTeX 用の設定

基本的に、JS クラスのコードの中で、「和文コンポーネントの管轄」として BXJS クラスで除外されている部分に相当するが、若干の変更が加えられている。

■補助マクロ \jsLetHeadChar を UTF-8 バイト列と和文文字トークンに対応させる。

\edef\jsc@JYn{\ifjsWithupTeX JY2\else JY1\fi}
\edef\jsc@JTn{\ifjsWithupTeX JT2\else JT1\fi}
\edef\jsc@pfx{\ifjsWithupTeX u\fi}
\def\bxjs@declarefontshape {\ifjsWithupTeX \else \fi}

■エンジン依存の定義 最初にエンジン (pTeX か upTeX か) に依存する定義を行う。
\ifjsWithupTeX は BXJS において定義されているスイッチで、エンジンが upTeX であるかを表す。
\jsc@JYn および \jsc@JTn は標準の和文横書きおよび縦書き用エンコーディングを表す。

\edef\jsc@JYn{\ifjsWithupTeX JY2\else JY1\fi}
\edef\jsc@JTn{\ifjsWithupTeX JT2\else JT1\fi}
\edef\jsc@pfx{\ifjsWithupTeX u\fi}
\def\bxjs@declarefontshape {\ifjsWithupTeX \else \fi}
めるため一旦マクロにしておく。xjs@sizereference は全角幅を測定する時に参照するフォント。

まず upTeX の場合の定義を示す。JS クラスの uplatex オプション指定時の定義と同じである。

\begin{verbatim}
\onlypreamble\bxjs@declarefontshape
\ifjsWithupTeX
\DeclareFontShape{JY2}{mc}{m}{n}{<->s*\bxjs@scale\upjpnrm-h}{}%
\DeclareFontShape{JY2}{gt}{m}{n}{<->s*\bxjs@scale\upjpngt-h}{}%
\end{verbatim}

\begin{verbatim}
\ifjsWithpTeXng \def\bxjs@declarefontshape{\DeclareFontShape{JY1}{mc}{m}{n}{<->s*\bxjs@scale\jis}{}%
\DeclareFontShape{JY1}{gt}{m}{n}{<->s*\bxjs@scale\jisg}{}%
\end{verbatim}

\begin{verbatim}
\def\bxjs@sizereference{upjisr-h}
\end{verbatim}

\begin{verbatim}
\else
\def\bxjs@sizereference{jis}
\end{verbatim}

\begin{verbatim}
\end{verbatim}

■和文フォントスケールの補正

実際、pTeX の標準的な和文フォント（JFM のこと、例えば jis）では、指定された jsScale（この値を \textit{s} とする）をそのまま使って定義すると期待通りの大きさにならない。これらの JFM では 1zw の大きさが指定されたサイズではなく既にスケール（この値を \textit{f} とする；jis では 0.962216 倍）が掛けられた値になっているからである。そのため、ここでは \textit{s/f} を求めてその値をマクロ \bxjs@scale に保存する。

\begin{verbatim}
\begingroup
\def\bxjs@tmpa:\bxjs@sizereference\space at 10pt
\end{verbatim}
幅が丁度10ptなら補正は不要

\ifdim\wd\z@=10pt
global\let\bxjs@scale\jsScale\else
(10*s)/(10*f)として計算、\bxjs@invscaleはBXJSで定義
 endorse\bxjs@tmpa{\strip@pt\wd\z@}
@endgroup
% typeout{\string\bxjs@scale : \bxjs@scale}

■和文フォント関連定義 \bxjs@scaleが決まったので先に保存した標準和文フォント宣言を実行する。
\bxjs@declarefontshape

フォント代替の明示的定義。
\DeclareFontShape{\jsc@JYn}{mc}{m}{it}{<->ssub*mc/m/n}{}
\DeclareFontShape{\jsc@JYn}{mc}{m}{sl}{<->ssub*mc/m/n}{}
\DeclareFontShape{\jsc@JYn}{mc}{m}{sc}{<->ssub*mc/m/n}{}
\DeclareFontShape{\jsc@JYn}{gt}{m}{it}{<->ssub*gt/m/n}{}
\DeclareFontShape{\jsc@JYn}{gt}{m}{sl}{<->ssub*gt/m/n}{}
\DeclareFontShape{\jsc@JYn}{gt}{bx}{it}{<->ssub*gt/m/n}{}
\DeclareFontShape{\jsc@JYn}{gt}{bx}{sl}{<->ssub*gt/m/n}{}
\DeclareFontShape{\jsc@JTn}{mc}{m}{it}{<->ssub*mc/m/n}{}
\DeclareFontShape{\jsc@JTn}{mc}{m}{sl}{<->ssub*mc/m/n}{}
\DeclareFontShape{\jsc@JTn}{mc}{b}{it}{<->ssub*mc/bx/n}{}
\DeclareFontShape{\jsc@JTn}{mc}{b}{sl}{<->ssub*mc/bx/n}{}
\DeclareFontShape{\jsc@JTn}{gt}{m}{it}{<->ssub*gt/m/n}{}
\DeclareFontShape{\jsc@JTn}{gt}{m}{sl}{<->ssub*gt/m/n}{}
\DeclareFontShape{\jsc@JTn}{gt}{bx}{it}{<->ssub*gt/bx/n}{}
\DeclareFontShape{\jsc@JTn}{gt}{bx}{sl}{<->ssub*gt/bx/n}{}
\DeclareFontShape{\jsc@JTn}{mc}{b}{n}{<->ssub*mc/bx/n}{}
\DeclareFontShape{\jsc@JTn}{mc}{b}{it}{<->ssub*mc/bx/n}{}
\DeclareFontShape{\jsc@JTn}{mc}{b}{sl}{<->ssub*mc/bx/n}{}
\DeclareFontShape{\jsc@JTn}{gt}{b}{n}{<->ssub*gt/bx/n}{}
\DeclareFontShape{\jsc@JTn}{gt}{b}{it}{<->ssub*gt/bx/n}{}
\DeclareFontShape{\jsc@JTn}{gt}{b}{sl}{<->ssub*gt/bx/n}{}
144
欧文総称フォント命令で和文フォントが連動するように修正する。その他の和文フォント関係の定義を行う。
※ 2020/02/02 の NFSS の改修に対する jsclasses の対策を取り入れた。
\begin{verbatim}
\ifnum\fmtversion{2020/10/01}
  {\jssc@needsp\tchfalse}{\jssc@needsp\tchtrue}
\if\jssc@needsp\tch
  % --- for 2020-02-02 or older BEGIN
  \ifx\@rmfamilyhook\@undefined % old
    \DeclareRobustCommand\rmfamily
    {\not@math@alphabet\rmfamily\mathrm
     \romanfamily\rmdefault\kanjifamily\mcdefault\selectfont}
  \else % 2020-02-02
    \g@addto@macro\@rmfamilyhook{\prepare@family@series@update@kanji{mc}\mcdefault}
  \fi
\else % --- for 2020-02-02 or older END & for 2020-10-01 BEGIN
  \ifx\mweights@init\@undefined\else % mweights.sty is loaded
    \selectfont is executed twice but I don’t care about speed...
  \fi
\fi
\else % 2020-10-01
  \ifx\DeclareFixJFMCJKTextFontCommand\@undefined
    \DeclareJaTextFontCommand{\textmc}{\mcfamily}
    \DeclareJaTextFontCommand{\textgt}{\gtfamily}
  \fi
\fi
\fi\else
\fi\fi
\fi
\end{verbatim}
念のため。

これ以降では、bxjs@parse@qh の処理は pTeX 系では不要になるので無効化する（つまり jsSetQHLength は setlength と等価になる）。

\begin{verbatim}
def\bxjs@parse@qh#1{\let\bxjs@tmpb\relax}
\let\bxjs@parse@qh@a\@undefined
\let\bxjs@parse@qh@b\@undefined
\end{verbatim}

■パラメタの設定

\begin{verbatim}
\renewcommand\kanjifamilydefault{\gtdefault}
\selectfont
\end{verbatim}

\begin{verbatim}
\renewcommand\kanjifamilydefault{\gtdefault}
\selectfont
\end{verbatim}

"80〜"FF の範囲の \spcode を 3 に変更。

\begin{verbatim}
\begingroup
\catcode`\!=0
bxjs@ptex@dir 現在の組方向： t=縦、 y=横、 ?=その他。
\end{verbatim}

エスケープ文字の変更はここまで。

■minijs パッケージのブロックやっておく。

\@namedef{ver@minijs.sty}{}

B.3 pdfT\TeX 用の処理

\else\if p\jobname T\else n\jobname T\else F\fi\fi T

\jsLetHeadChar を UTF-8 バイト列に対応させる。

ムニャムニャ。

\@onlypreamble\bxjs@cjk@loaded
\let\bxjs@cjk@loaded\relax
\g@addto@macro\bxjs@begin@document@hook{%
\@ifpackageloaded{CJK}{\bxjs@cjk@loaded}{%}
}

B.4 X\TeX 用の処理

\else\if x\jobname \else F\fi\fi F

\bxjs@let@hchar@chr について、「BMP 外の文字の文字トークンに対して \string を適用するとサロゲートペアに分解される」という問題に対する応急措置を施す。

\def\bxjs@let@hchar@chr#1{%
@tempcnta`#1\relax \divide\@tempcnta"800\relax
}
オプションの実際の処理内容。

\bxjs@do@precisetext  オプションの実際の処理内容。
\bxjs@do@simplejasetup  オプションの実際の処理内容。

TODO: 3.0 バージョン要件を見直して暫定措置を解除する。

\bxjs@do@simplejasetup  オプションの実際の処理内容。
\bxjs@do@precisetext  オプションの実際の処理内容。

\jsSimpleJaSetup  日本語出力用の超簡易的な設定。

B.5 後処理（エンジン共通）
段落頭でのグルー挿入禁止 本体開始時において \everyparhook を検査して、“結局何もしない” ことになっている場合は、副作用を完全に無くすために \everyparhook を空にする。

\everyparhook=modern の場合の、\everyparhook の有効化の実装。
※本体開始時ではなく最初から有効化していることに注意。

fancyhdr 対策 fancyhdr オプションの値が true であり、かつ fancyhdr が使用された場合に以下の対策を行う。
- デフォルトの書式設定に含まれる “二文字フォント命令” を除去する。
- bxjsbook においてヘッダ・フッタの横幅を \fullwidth に変える。

fancyhdr の初期設定に関する改変の処理。fancyhdr 読込完了と \pagestyle{fancy} 実行の間で実行されるべき。

ヘッダ・フッタの要素の書式について、それが既定のままであれば、“二文字フォント命令”を除去したものに置き換える。
※和文なので \sl は無い方がよいはず。
が（定義済で）\textwidth よりも大きい場合、ヘッダ・フッタの横幅を \fullwidth に合わせる。

\fullwidth が（定義済で）\textwidth よりも大きい場合、ヘッダ・フッタの横幅を \fullwidth に合わせる。

\fullwidth が（定義済で）\textwidth よりも大きい場合、ヘッダ・フッタの横幅を \fullwidth に合わせる。
付録 C 和文ドライバ：standard ☃

標準のドライバ。

- \rmfamily/\sffamily/\ttfamily での和文ファミリ連動
- \mcfamily/\gtfamily
- \textmc/\textgt
- \setkanjiskip/\getkanjiskip
- \setxkanjiskip/\getxkanjiskip
- \autospacing/\noautospacing
- \autoxspacing/\noautoxspacing

C.1 準備

\%<\*standard>
\%
このファイルは日本語文字を含みます
まず minimal ドライバを読み込む。
\input{bxjsja-minimal.def}

simplejasetup は standard では無効になる。
\bxjs@simplejasetupfalse

■環境検査

TODO: 3.0 以下で 3.0 版でのバージョン要件の予定について述べておく。
standard 和文ドライバの処理系バージョン要件 (minimal からの差分) は以下の通りである。

- upTEX： 0.29 版 [2010/01] 以上
- LuaTeX： 0.85 版 [2015/11] 以上
- XƎTEX： 0.9999 版 [2013/03] 以上

加えて、以下の要件を定める。

- pTeX 系以外のエンジンでは \TeXX 拡張を必須とする。
  ※ bxcjkjatype パッケージが \TeXX 拡張を要求するため。
- LuaTeX の DVI モードはサポートしない。
  ※ LuaTeX-ja パッケージがサポートしていないため。

■パッケージ読込 利用可能な場合は etoolbox パッケージを読み込む。
※ 1.3 版は「etoolbox パッケージ」としての最古の版であるらしい。\AtEndPreamble はこの版で既に利用可能である。
\ifjsWithTeX
\IfFileExists{etoolbox.sty}{%
\RequirePackage{etoolbox}[2007/10/08]\% v1.3
}\fi

C.2 和文ドライバパラメタ

japaram の値を key-value リストとして解釈する。keyval のファミリは bxjsStd とする。

\ifbxjs@jp@jismmiv 2004JIS 字形を優先させるか。
\newif\ifbxjs@jp@jismmiv
\jis2004 オプションの処理。
\newif\ifbxjs@jp@jismmivfalse
\define@key{bxjsStd}{jis2004}[true]{%}
\bxjs@set@keyval{jis2004}{#1}{}
\ifbxjs@jp@units
和文用単位 (zw, zh, (true)Q, (true)H) を使えるようにするか。
\newif\ifbxjs@jp@units
\units オプションの処理。
\newif\ifbxjs@jp@unitsfalse
\define@key{bxjsStd}{units}[true]{%}
\bxjs@set@keyval{units}{#1}{}
\bxjs@jp@font フォントパッケージの追加オプション。
\let\bxjs@jp@font@empty
\font オプションの処理。
\edef\bxjs@jp@font{\bxjs@catopt\bxjs@jp@font{#1}}\}
\ifbxjs@jp@strong@cmd
strong 命令を補填するか。
\newif\ifbxjs@jp@strong@cmd\bxjs@jp@strong@cmdtrue
\strong-cmd オプションの処理。
\newif\ifbxjs@jp@strong@cmdfalse\bxjs@jp@strong@cmdfalse
\define@key{bxjs}{strong-cmd}[true]{%}
\bxjs@set@keyval{strongcmd}{#1}{}
実際の japaram の値を適用する。
\def\bxjs@next#1\{\bxjs@safe@setkeys{bxjsStd}{#1}{}
\expandafter\bxjs@next\expandafter{\jsJaParam}

C.3 共通処理 (1)
jis2004 パラメタ  jis2004 パラメタが有効の場合は、グローバルオプションに jis2004 を追加する。
※ otf や luatexja-preset 等のパッケージがこのオプションを利用する。

4378 \let\jafontsize\jsJaFontSize

■和文用単位のサポート エンジンが (u)pTeX の場合は units を無効にする。
4386 \if \jEngine
4387 \bxjs\jp@unitsfalse
4388 \fi

units パラメタが有効の場合は、bxcalc パッケージの \useTeXunits 命令を実行して和文用単位を有効化する。
4389 \ifbxjs\jp@units
4390 \IfFileExists{bxcalc.sty}{}
4391 \RequirePackage{bxcalc}\[2018/01/28]\v1.0a
4392 \ifx\useTeXunits\@undefined
4393 \PackageWarningNoLine\bxjs@clsname
4394 \{Cannot support pTeX units (zw etc.), since\MessageBreak
4395 the package 'bxcalc' is too old}\%
4396 \bxjs\jp@unitsfalse
4397 \else \useTeXunits
4398 \fi
4399 \} else
4400 \PackageWarningNoLine\bxjs@clsname
4401 \{Cannot support pTeX units (zw etc.), since\MessageBreak
4402 the package 'bxcalc' is unavailable}\%
4403 \bxjs\jp@unitsfalse
4404 \}
4405 \fi

bxcalc で和文用単位をサポートした場合は、\bxjs@parse@qh の処理は不要になるので無効化する。
4406 \ifbxjs\jp@units
4407 \def\bxjs@parse@qh#1{\let\bxjs@tmpb\relax}
4408 \let\bxjs@parse@qh#1\@undefined
4409 \let\bxjs@parse@qh#2\@undefined
4410 \fi

\bxjs@let@lenexpr \bxjs@let@lenexpr\CS{(長さ式): 長さ式に bxcalc の展開を適用した結果のトークン列を \CS に代入する。
\ifbxjs@jp@units
\def\bxjs@let@lenexpr#1#2{%\edef#1{#2}\
\expandafter\CUXParseExpr\expandafter#1\expandafter{#1}}
\else
\def\bxjs@let@lenexpr{\edef}
\fi

■\strong 命令の補填

\strong 現在未定義（fontspec が未読込）である場合は、クラス本体で定義した \jsStrongText strongenv（env.）を利用して定義する。

\ifbxjs@jp@strong@cmd\jsAtEndOfClass{%\ifx\strong\@undefined\ifx\strongenv\@undefined
\newcommand*{\strongenv}{\jsStrongText}\
\DeclareTextFontCommand{\strong}{\jsStrongText}\
\newcommand*{\strongfontdeclare}{\jsStrongDeclare}\fi\fi
\fi

■和文フォント指定の扱い standard 和文ドライバでは \jsJaFont の値を和文フォントの“プリセット”の指定として用いる。プリセットの値は、\TeX Live の kanji-config-updmap コマンドで使う“ファミリ”と同じにすることを想定する。特別な値として、auto は kanji-config-updmap で現在指定されているファミリを表す。

\bxjs@adjust@jafont \jsJaFont に入っている和文フォント設定の値を“調整”して、その結果を \bxjs@tmpa に返す。#1 が f の場合は“非埋込 (noEmbed)”の設定が禁止される。この禁止の場合も含め、何か異常がある場合は \bxjs@tmpa は空になる。
\onlypreamble\bxjs@adjust@jafont
\def\bxjs@adjust@jafont#1{%\ifx\jsJaFont\bxjs@@auto
\bxjs@get@kanjiEmbed
\ifx\bxjs@jaEmbed\relax
\let\bxjs@tmpa\@empty
\else
\let\bxjs@tmpa\bxjs@jaEmbed
\ifx\bxjs@jaVariant\bxjs@@hziv \bxjs@apply@mmiv
\fi
\else
\let\bxjs@tmpa\bxjs@jaEmbed
\ifx\bxjs@jaEmbed\relax
\let\bxjs@tmpa\@empty
\else
\let\bxjs@tmpa\bxjs@jaFont
\fi
\fi
\fi
\else
\let\bxjs@tmpa\@empty
\fi
\fi
\fi

\ClassWarningNoLine\bxjs@clsname
{Option 'jafont=noEmbed' is ignored, because it is\MessageBreak
not available on the current situation}%
現在の\texttt{updmap}の\texttt{jaEmbed}・\texttt{jaVariant}パラメータの値。\texttt{bxjs@get@kanjiEmbed}により実際に設定値が取得されてここに設定される。

※古い版の\texttt{updmap}では\texttt{kanjiEmbed}・\texttt{kanjiVariant}であった。
\let\bxjs@jaEmbed\bxjs@g@tmpa
\let\bxjs@jaVariant\bxjs@g@tmpb
\bxjs@resolve@jafont@paren

パラメータ値内の () を解決する。\bxjs@resolve@jafont@paren\CS で、\CS の内容中の (...) を \bxjs@jafont@paren{...} に置き換える。

\def\bxjs@resolve@jafont@paren#1{\def\bxjs@tmpb{\let#1}\expandafter\bxjs@resolve@jafont@paren@a#1\@nil\@nil\@nnil#1}
\def\bxjs@resolve@jafont@paren@a#1(#2)#3\@nil#4\@nnil#5{\ifx\relax#4\relax \bxjs@tmpb#5\else \edef\bxjs@tmpa{#1\bxjs@jafont@paren{#2}#3}\bxjs@tmpb\bxjs@tmpa\fi}

■和文として出力 「欧文扱い」となっている文字を和文として出力するための機能。
\jachar \jachar{( 文字 )}： 和文文字として出力する。
\newcommand*\jachar[1]{\begingroup\jsLetHeadChar\bxjs@tmpa{#1}\ifx\bxjs@tmpa\relax\ClassWarningNoLine\bxjs@clsname{Illegal argument given to \string\jachar}\else\expandafter\bxjs@jachar\expandafter{\bxjs@tmpa}\fi\endgroup}
\jsJaChar を \jachar と等価にする。

下請けの \bxjs@jachar の実装はエンジンにより異なる。

■hyperref 対策 出力ページサイズに制限する処理は geometry パッケージが行うので、hyperref 側の処理は無効にしておく。
\PassOptionsToPackage{setpagesize=false}{hyperref}
\bxjs@fix@hyperref@unicode hyperref の unicode オプションの値を固定する。
```latex
\@namedef{KV@Hyp@bxjs/hook}##1{%
KV@Hyp@unicode{##1}%
\defKV@Hyp@unicode####1{%
\expandafter\ifx\csname if##1\expandafter\endcsname
\csname if####1\endcsname\else
\ClassWarningNoLine\bxjs@clsname
(Blcoked hyperref option 'unicode=####1')%
\fi
}
}
\jsCheckHyperrefUnicode
hyperrefのunicodeオプションの値を検証するための本体開始時のフック。
※pxjahyper-uni.defはこのフックを\relaxに上書きすることで検証を無効化している。
\@onlypreamble\jsCheckHyperrefUnicode
\let\jsCheckHyperrefUnicode\@empty
\g@addto@macro\bxjs@begin@document@hook{\jsCheckHyperrefUnicode}
\bxjs@check@hyperref@unicode
hyperrefのunicodeオプションの値を本体開始時に検証する。
\@onlypreamble\bxjs@check@hyperref@unicode
\def\bxjs@check@hyperref@unicode#1{%
\g@addto@macro\jsCheckHyperrefUnicode{%
\@tempswafalse
\begingroup
\expandafter\ifx\csname ifHy@unicode\endcsname\relax
\aftergroup\@tempswatrue \fi
\expandafter\ifx\csname ifHy@unicode\expandafter\endcsname
\csname if#1\endcsname
\aftergroup\@tempswatrue \fi
\endgroup
\if@tempswa
\ClassError\bxjs@clsname
{The value of hyperref 'unicode' key is not suitable}\MessageBreak
for the present engine (must be #1)}%
{\@ehc}%
\fi}
\bxjs@urgent@special
DVIのなるべく早い位置にspecialを出力する。
\@onlypreamble\bxjs@urgent@special
\@onlypreamble\bxjs@urgent@special@a
\LatexXカーネルの新フック管理が導入済かを調べる。未導入の古い版である場合。
\if\bxjs@old@hook@system
\def\bxjs@urgent@special#1{%
\AtBeginDvi{\special{#1}}%
\g@addto@macro\bxjs@begin@document@hook{\jsCheckHyperrefUnicode}
\ifpackageloaded{atbegshi}{%
\begingroup
\expandafter\ifx\csname ifHy@unicode\endcsname\relax
\aftergroup\@tempswatrue \fi
\expandafter\ifx\csname ifHy@unicode\expandafter\endcsname
\csname if#1\endcsname
\aftergroup\@tempswatrue \fi
\endgroup
\if@tempswa
\ClassError\bxjs@clsname
{The value of hyperref 'unicode' key is not suitable}\MessageBreak
for the present engine (must be #1)}%
{\@ehc}%
\fi}
```

157
導入済の場合。
※自己が先行する必要がある対象のパッケージを適宜追加する。
※pxjahyper パッケージの処理と合わせる。
\def\bxjs@urgent@special#1{\AddToHook{shipout/firstpage}[pxjahyper/enc]{\special{#1}}}\def\bxjs@urgent@special@a{\DeclareHookRule{shipout/firstpage}{pxjahyper/enc}{<}{hyperref}}\global\let\bxjs@urgent@special@a\relax\fi
\newcommand*{\setkanjiskip}{\jsSetKanjiSkip}\newcommand*{\getkanjiskip}{\jsGetKanjiSkip}\def\jsApplyKanjiSkip#1{\kanjiskip=#1\relax}\newcommand*{\setxkanjiskip}{\jsSetXKanjiSkip}\newcommand*{\getxkanjiskip}{\jsGetXKanjiSkip}\def\jsApplyXKanjiSkip#1{\xkanjiskip=#1\relax}\def\bxjs@jachar#1{\bxjs@jachar@a#1....\@nil}\def\bxjs@jachar@a#1#2#3#4#5\@nil{\ifx.#2#1\relax\else\ifx.#3\relax\@tempcnta`#1 \multiply\@tempcnta64 \advance\@tempcnta`#2 \advance\@tempcnta`#3 \advance\@tempcnta`#4 \advance\@tempcnta`#5\fi\fi\fi\fi\fi\else\ifx.#4\relax\fi\fi\fi\fi\fi

C.4 p\TeX{} 用設定

\if j\jsEngine

\newcommand*{\setkanjiskip}{\jsSetKanjiSkip}\newcommand*{\getkanjiskip}{\jsGetKanjiSkip}\def\jsApplyKanjiSkip#1{\kanjiskip=#1\relax}\newcommand*{\setxkanjiskip}{\jsSetXKanjiSkip}\newcommand*{\getxkanjiskip}{\jsGetXKanjiSkip}\def\jsApplyXKanjiSkip#1{\xkanjiskip=#1\relax}\def\bxjs@jachar#1{\bxjs@jachar@a#1....\@nil}\def\bxjs@jachar@a#1#2#3#4#5\@nil{\ifx.#2#1\relax\else\ifx.#3\relax\@tempcnta`#1 \multiply\@tempcnta64 \advance\@tempcnta`#2 \advance\@tempcnta`#3 \advance\@tempcnta`#4 \advance\@tempcnta`#5\fi\fi\fi\fi\fi

\jaJaChar がサブマクロ。
\def\bxjs@jachar#1{\bxjs@jachar@a#1....\@nil}\def\bxjs@jachar@a#1#2#3#4#5\@nil{\ifx.#2#1\relax\else\ifx.#3\relax\@tempcnta`#1 \multiply\@tempcnta64 \advance\@tempcnta`#2 \advance\@tempcnta`#3 \advance\@tempcnta`#4 \advance\@tempcnta`#5\fi\fi\fi\fi\fi

\ifx.#2\relax\fi\ifx.#3\relax\@tempcnta`#1 \multiply\@tempcnta64 \advance\@tempcnta`#2 \advance\@tempcnta`#3 \advance\@tempcnta`#4 \advance\@tempcnta`#5\fi\fi\fi\fi\fi

\ifx.#2\relax\fi\ifx.#3\relax\@tempcnta`#1 \multiply\@tempcnta64 \advance\@tempcnta`#2 \advance\@tempcnta`#3 \advance\@tempcnta`#4 \advance\@tempcnta`#5\fi\fi\fi\fi\fi
符号値が@tempcntaの和文文字を出力する処理。
\begin{Verbatim}
\ifjsWithupTeX
\def\bxjs@jachar@b{\kchar@\@tempcnta}
\else
\def\bxjs@jachar@b{\ifx\bxUInt\@undefined\else\bxUInt{\@tempcnta}\fi}
\fi
\fi
\fi

和欧文間空白の命令\jathinspaceの実装。
\begin{Verbatim}
\ifbxjs@jaspace@cmd
\def\jathinspace{\hskip\xkanjiskip}
\fi
\end{Verbatim}

■jis2004パラメータ pxchfonとpxbabelでは2004JISを指定するオプションの名がprefer2004jisである。

\begin{Verbatim}
\ifbxjs@jp@jismiv
\PassOptionsToPackage{prefer2004jis}{pxchfon}
\PassOptionsToPackage{prefer2004jis}{pxbabel}
\fi
\end{Verbatim}

■和文フォント指定の扱いpTeXは既定でkanji-config-updmapの設定に従うため、\jsJaFontがautoの場合は何もする必要がない。無指定でもautoでない場合は、\jsJaFontをオプションにしてpxchfonパッケージを読み込む。ここで、和文ドライバパラメータfontが指定されている場合は、その値をpxchfonのオプションに追加する。

\begin{Verbatim}
\let\bxjs@jafont@paren@firstofone
\let\bxjs@tmpa\jsJaFont
\ifx\bxjs@tmpa\bxjs@@auto
\let\bxjs@tmpa\@empty
\else\ifx\bxjs@tmpa\bxjs@@noEmbed
\def\bxjs@tmpa{noembed}
\fi\fi
\bxjs@resolve@jafont@paren\bxjs@tmpa
\edef\bxjs@tmpa{\bxjs@catopt\bxjs@tmpa\bxjs@font}
\ifx\bxjs@tmpa\@empty\else
\begin{Verbatim}
\end{Verbatim}
\fi
\end{Verbatim}
■otf パッケージ対策 インストールされている otf パッケージが scale オプションに対応している場合は scale=(\jsScale の値) を事前に otf に渡す。
※ scale 対応は 1.7b6 版 [2013/11/17] から。
※ otf.sty の中に「\RequirePackage{keyval}」の行が存在するかにより判定している。
（もっといい方法はないのか……。）

■hyperref 対策 hyperref の unicode オプションに対する調整を行う。
※ pxjahyper パッケージの「unicode 対応」サポートの履歴：
  • 0.7 版 [2021-02-13]：upTeX 上に限り unicode 対応。
  • 0.9c 版 [2021-06-06]：pxjahyper-uni.def ファイルを追加。
  • 1.0 版 [2022-04-01]：pLaTeX 上の unicode 対応を試験的サポート。
  • 1.3 版 [2023-03-01]：pLaTeX 上の unicode 対応を正式サポート。
unicode オプションが偽であることを検証する。ただし、pxjahyper パッケージまたは pxjahyper-uni.def が読み込まれて（前提条件を満たしていて）「unicode 対応」が行われた場合は検証は無効化される。

\bxjs@check@hyperref@unicode{false}

\bxjs@plautopatch@new は「pxjahyper の自動読込に対応した版の plautopatch が読み込まれているか」のフラグ。

\bxjs@if@package@at@least{plautopatch}{2020/05/25}{% v0.9g

「unicode を有効にするか」を判定する。まず必要条件として「pxjahyper-uni.def が存在すること」「\bxjs@plautopatch@new が真、または、ファイルフックが利用可能である」と検査する。

※ pxjahyper-uni.def をもつ pxjahyper の版であれば、up\LaTeX 上の unicode には対応していることに注意。

\let\bxjs@avail@hy@unicode=f
\if \ifx t\bxjs@plautopatch@new T%\else\ifbxjs@old@hook@system F\else T\fi\fi T%
\IfFileExists{pxjahyper-uni.def}{\let\bxjs@avail@hy@unicode=t}{}
\fi
\if t\bxjs@avail@hy@unicode
\ifjsWithupTeX
必要条件が満たされていて、かつ up\LaTeX である場合の処理。もしファイルフックが利用可能ならば、hyperref が読み込まれた場合にその直後に pxjahyper-uni.def が読まれるようにする。

※そうでないなら、前提条件より pxjahyper が読み込まれるはずなので何もしなくてよい。
\let\bxjs@old@hook@system\else
\AddToHook{\bxjs@CGHN{package/hyperref/after}}{%
\input{pxjahyper-uni.def}
\fi
\else
必要条件が満たされていて、かつ p\LaTeX である場合の処理。pxjahyper が「p\LaTeX 上の unicode 対応をもつほど新しい版（1.3 版以降）」であるかを判定する方法はない。しかし、新しい \LaTeX システムで unicode を無効にするのは避けたいので、\LaTeX カーネルが 2023/06/01 版以降である場合に pxjahyper を十分に新しいと推定することにする。すなわち「pxjahyper が読み込まれるはず」かつ「\LaTeX カーネルが新しい」かを判定する。
\let\bxjs@avail@hy@unicode=f
\ifx t\bxjs@plautopatch@new
\bxjs@if@format@at@least{2023/06/01}{%v0.9g
\let\bxjs@avail@hy@unicode=t}{}
\fi
\fi
\fi
この時点で「unicode を有効にできるか」の判定結果がフラグ \bxjs@avail@hy@unicode
に入っている。unicode を有効にできない場合は unicode の既定値を偽に設定する。

\if f\bxjs@avail@hy@unicode
\PassOptionsToPackage{unicode=false}{hyperref}
\fi

\fi
tounicode special 命令を出力する。
\if \ifx\bxjs@driver@given\bxjs@driver@@dvipdfmx T%
\else\ifjsWithpTeXng T\else F\fi\fi T%
\IfFileExists{pxjahyper-enc.sty}{\@tempswatrue}{\@tempswafalse}
\if@tempswa
\RequirePackage{pxjahyper-enc}[2020/10/05]%v0.6
\ifbxjs@bigcode\else \suppressbigcode \fi
\else
\ifnum\jis"2121="A1A1 %euc
\bxjs@urgent@special{pdf:tounicode EUC-UCS2}
\else\ifnum\jis"2121="8140 %sjis
\bxjs@urgent@special{pdf:tounicode 90ms-RKSJ-UCS2}
\else\ifnum\jis"2121="3000 %uptex
\ifbxjs@bigcode
\bxjs@urgent@special{pdf:tounicode UTF8-UTF16}
\else
\PassOptionsToPackage{bigcode}{pxjahyper}
\fi\fi\fi\fi
\else
\let\bxToUnicodeSpecialDone=t
\fi
\fi
\fi
\let\bxToUnicodeSpecialDone=t

■和文数式ファミリ 和文数式ファミリは既定で有効とする。すなわち enablejfam=false
以外の場合は @enablejfam を真にする。
\ifx f\bxjs@enablejfam\else
\@enablejfamtrue
\fi

実際な和文用の数式ファミリの設定を行う。
\if@enablejfam
\DeclareSymbolFont{mincho}{\jsc@JYn}{mc}{m}{n}
\DeclareSymbolFontAlphabet{\mathmc}{mincho}
\SetSymbolFont{mincho}{bold}{\jsc@JYn}{gt}{m}{n}
\jfam\symmincho
\DeclareMathAlphabet{\mathgt}{\jsc@JYn}{gt}{m}{n}
\g@addto@macro\bxjs@begin@document@hook{%
\reDeclareMathAlphabet\undefined
\reDeclareMathAlphabet を適用した数式英字フォント命令は \LaTeX の通常の命令とは定

162
義文の形が異なる。このため bm パッケージを読み込んで \( \text{\texttt{bm\{\texttt{\newcommand{\mathbf}{\@mathbf}{\@mathgt}\texttt{\newcommand{\mathsf}{\@mathsf}{\@mathgt}}}\}}} \) を実行するとエラーが発生する。これを回避するための暫定対応として、bm が読み込まれた場合は数式英字フォントの和文連動を無効にする。

**TODO**: ユーザが数式英字フォントの和文連動を制御できるようにする。

```latex
\@ifpackageloaded{bm}{}{%else
\reDeclareMathAlphabet{\mathrm}{\@mathmc}{\@mathgt}{
\reDeclareMathAlphabet{\mathbf}{\@mathbf}{\@mathgt}{
\reDeclareMathAlphabet{\mathsf}{\@mathsf}{\@mathgt}{
}%
\fi}
```

C.5 pdftex 用設定：CJK + bxcjkjatype

■ bxcjkjatype パッケージの読込 jstex が指定されている場合は、その値を bxcjkjatype のオプション（プリセット指定）に渡す。ここで値が auto である場合は \( \texttt{bxjs\{get\{kanjiEmbed\}} \) を実行する。スケール値（\( \texttt{jsScale} \) の反映は bxcjkjatype の側で行われる。

※ Pandoc モードでは autotilde を指定しない。

```latex
\begin{footnotesize}
\begin{center}
\textbf{Autotilde が制限される環境}
\end{center}
\end{footnotesize}
```

■ hyperref 対策 bxcjkjatype 使用時は unicode にするべき。

```latex
\if\bxjs@hyperref@enc
\PassOptionsToPackage{unicode}{hyperref}
\fi
\begin{hypersetup}
\PassOptionsToPackage{unicode}{hyperref}
```

※ パッケージオプションでの指定に対応するのは、「アクティブな高位バイトトークンがその場で展開されてしまう」ため困難である。

```latex
\ifx\bxjs@cjkl\undefined
\begingroup
```

163
が和欧文間空白である場合は PDF 文字列中で空白文字でなく空に展開させる。

- が和欧文間空白である場合は PDF 文字列中で空白文字でなく空に展開させる。

■ 共通命令の実装

\newcommand*{\setkanjiskip}{\jsSetKanjiSkip}
\newcommand*{\getkanjiskip}{\jsGetKanjiSkip}
\newcommand*{\autospacing}{\jsEnableKanjiSkip}
\newcommand*{\noautospacing}{\jsDisableKanjiSkip}
\protected\def\bxjs@cjkglue{\hskip\jsKanjiSkip}
\newcommand*{\setxkanjiskip}{\jsSetXKanjiSkip}
\newcommand*{\getxkanjiskip}{\jsGetXKanjiSkip}
\newcommand*{\autoxspacing}{\jsEnableXKanjiSkip}
\newcommand*{\noautoxspacing}{\jsDisableXKanjiSkip}
\protected\def\bxjs@cjkecglue{\hskip\jsXKanjiSkip}
\def\bxjs@cjktilde{\bxjs@cjkglue}\ifx\bxjs@cjktilde\@empty\else\fi
\def\bxjs@cjktildecmd{~}
\def\bxjs@org@LetUnexpandableSpace{\HyPsd@LetUnexpandableSpace}
\let\bxjs@org@LetUnexpandableSpace\bxjs@LetUnexpandableSpace
\let\bxjs@cjktilde\bxjs@cjktildecmd
\protected\def\bxjs@cjkremovespace{\bxjs@cjktildecmd}
\def\bxjs@cjkremovespacecmd{~}
\def\bxjs@org@SetKanjiSkip{\jsSetKanjiSkip}
\def\bxjs@org@GetKanjiSkip{\jsGetKanjiSkip}
\def\bxjs@org@EnableKanjiSkip{\jsEnableKanjiSkip}
\def\bxjs@org@DisableKanjiSkip{\jsDisableKanjiSkip}
\def\bxjs@org@SetXKanjiSkip{\jsSetXKanjiSkip}
\def\bxjs@org@GetXKanjiSkip{\jsGetXKanjiSkip}
\def\bxjs@org@EnableXKanjiSkip{\jsEnableXKanjiSkip}
\def\bxjs@org@DisableXKanjiSkip{\jsDisableXKanjiSkip}
\def\bxjs@setjachar#1{\jachar#1}
\jachar のサブマクロの実装。
和欧文間空白の命令 \jathinspace の実装。

\ifxjs@jaspace@cmd
\protected\def\jathinspace{\CJKecglue}
\fi

■和文数式ファミリ  CJK パッケージは（恐らく）数式文字として CJK 文字をサポートしていない。従って @enablejfm は常に偽になる。
\ifx t\bxjs@enablejfm
\ClassWarningNoLine\bxjs@clsname{You cannot use 'enablejfm=true', since the\MessageBreak
CJK package does not support Japanese math}
\fi

C.6 X\TeX 用設定: xecjk + zxjatype

■zxjatype パッケージの読み込み スケール値（\jsScale）の反映は zxjatype の側で行われる。
\RequirePackage{zxjatype}
\PassOptionsToPackage{no-math}{fontspec}%!
\PassOptionsToPackage{xetex}{graphicx}%!
\PassOptionsToPackage{xetex}{graphics}%!
\ifx\zxJaFamilyName\@undefined
\ClassError\bxjs@clsname{xeCJK or zxjatype is too old}@ehc
\fi

■和文フォント定義 \jsJaFont が指定された場合は、その値をオプションとして zxjafont を読み込む。非指定の場合は原ノ味フォントを使用する。
※ 2.0 版より既定を IPAex から原ノ味に変更。
\bxjs@adjust@jafont{f}
\let\bxjs@jafont@paren\@gobble
\bxjs@resolve@jafont\bxjs@tmpa
\edef\bxjs@tmpa{\bxjs@catopt\bxjs@tmpa\bxjs@jp@font}
\ifx\bxjs@tmpa\@empty
\setCJKmainfont[BoldFont=HaranoAjiGothic-Medium.otf]{HaranoAjiMincho-Regular.otf}
\else
\edef\bxjs@tmpa{\noexpand\RequirePackage[\bxjs@tmpa]{zxjafont}[2013/01/28]% v0.2a}
\bxjs@next
\fi

■hyperref 対策 unicode オプションの指定に関する話。
XelTeXの場合は、xdvipdfmxがUTF-8→UTF-16の変換を行う機能を持っているため、本来はspecial命令の文字列の文字コード変換は不要である。ところが、hyperrefでの方針としては、XelTeXの場合にもパッケージ側で文字コード変換を行う方が望ましいと考えている。実際、unicodeを無効にしていると警告が出て強制的に有効化される。一方で、過去（r35125まで）のxdvipdfmxでは、文字列をUTF-16に変換した状態で与えるのは不正と見なしていて警告が発生する。

これを踏まえて、ここでは、「XelTeXのバージョンが0.99992以上の場合にunicodeを既定で有効にする」ことにする。

※\TeXの小数の精度は十進で4桁までしか保証されないので、\strcmpを利用して文字列で比較している。（整数部が多桁になっても大丈夫。）しかし実は、\strcmpプリミティブが追加されたのは0.9994版（2009年6月）かららしい。

TODO:3.0バージョン要件を見直して暫定措置を解除する。

\begin{verbatim}
\ifx\strcmp\@undefined\else %未定義なら条件を満たさない
\ifnum\strcmp{\the\XeTeXversion\XeTeXrevision}{0.99992} \m@ne
\ifbxjs@hyperref@enc
\PassOptionsToPackage{unicode}{hyperref}
\fi
\fi
\fi
\end{verbatim}

■段落頭でのグルー挿入禁止 どうやら、zxjatypeの\inhibitglueの実装が極めて粗雑なため、1.0版での実装では全く期待通りの動作をしていないし、そもそも（少なくとも現状の）xeCJKでは、段落頭での\inhibitglueは実行しないほうがJSクラスの出力に近いものが得られるらしい。

従って、\jsInhibitGlueAtParTopは結局何もしないことになる。

強制改行直後のグルー禁止処理、のような怪しげな何か。

\begin{verbatim}
\AtEndOfClass{%
\def\getline #1{%
\ifvmode \@nolnerr
\else
\unskip \reserved@e {\reserved@f#1}\nobreak \hfil \break \null
\nobreak \hskip-1sp\hskip1sp\relax
\ignorespaces
\fi}
\end{verbatim}

■共通命令の実装

\begin{verbatim}
\newcommand*{\jsKanjiSkip}
\newcommand*{\jsXKanjiSkip}
\ifx\CJKecglue\@undefined
\def\CJKtilde{\CJK@global\def~{\CJKecglue\ignorespaces}}
\fi
\end{verbatim}
\newcommand*{\autospacing}{\jsEnableKanjiSkip}
\newcommand*{\noautospacing}{\jsDisableKanjiSkip}
\protected\def\bxjs@CJKglue{\hskip\jsKanjiSkip}
\def\jsApplyKanjiSkip#1{\jsKanjiSkip=#1\relax\xeCJKsetup{CJKglue={\bxjs@CJKglue}}}
\newcommand*{\setxkanjiskip}{\jsSetXKanjiSkip}
\newcommand*{\getxkanjiskip}{\jsGetXKanjiSkip}
\newcommand*{\autoxspacing}{\jsEnableXKanjiSkip}
\newcommand*{\noautoxspacing}{\jsDisableXKanjiSkip}
\protected\def\bxjs@CJKecglue{\hskip\jsXKanjiSkip}
\def\jsApplyXKanjiSkip#1{\jsXKanjiSkip=#1\relax\xeCJKsetup{CJKecglue={\bxjs@CJKecglue}}}
\mcfamily、\gtfamily は本来は zxjatype の方で定義すべきであろうが、現状は暫定的にここで定義する。
\ifx\mcfamily\@undefined\protect\def\mcfamily{\CJKfamily{\CJKrmdefault}}\protected\def\gtfamily{\CJKfamily{\CJKsfdefault}}\fi
\jachar のサブマクロの実装。
\def\bxjs@jachar#1{\xeCJKDeclareCharClass{CJK}{`#1}\relax#1}
\jathinspace の実装。
\ifbxjs@jaspace@cmd\protect\def\jathinspace{\CJKecglue}\fi
■和文数式ファミリ 和文数式ファミリは既定で無効とする。すなわち enablejfam=true の場合にのみ @enablejfam を真にする。
\ife\bxjs\@enablejfam\@enablejfamtrue\fi
実際には和文用の数式ファミリの設定を行う。
※FIXME: 要検討。
\if@enablejfam\xeCJKsetup{CJKmath=true}\fi
C.7 LuaTeX 用設定: LuaTeX-ja
\else\if1\jsEngine
■LuaTeX-ja パッケージの読込 luatexja とともに luatexja-fontspec パッケージを読み込む。
luatexja は自前の \zw（これは実際の現在和文フォントに基づく値を返す）を定義するので、\zw の定義を消しておく。なお、レイアウト定義の「全角幅」は「規定」に基づく \jsZw であることに注意が必要。
※ 1.0b 版から「graphics パッケージに pdftex オプションを渡す」処理を行っていたが、1.4 版で廃止された。

\let\zw\@undefined
\RequirePackage{luatexja}
\edef\bxjs@next{% 
\noexpand\RequirePackage\jsScale\{}luatexja-fontspec\{2015/08/26}\%
} \bxjs@next

\set@fontsize へのパッチ適用を再度行う。

フォント代替の明示的定義。
\DeclareFontShape{JY3}{mc}{m}{it}{<->ssub*mc/m/n}{}
\DeclareFontShape{JY3}{mc}{m}{sl}{<->ssub*mc/m/n}{}
\DeclareFontShape{JY3}{mc}{m}{sc}{<->ssub*mc/m/n}{}
\DeclareFontShape{JY3}{gt}{m}{it}{<->ssub*gt/m/n}{}
\DeclareFontShape{JY3}{gt}{m}{sl}{<->ssub*gt/m/n}{}
\DeclareFontShape{JY3}{mc}{bx}{it}{<->ssub*gt/m/n}{}
\DeclareFontShape{JY3}{mc}{bx}{sl}{<->ssub*gt/m/n}{}
\DeclareFontShape{JY3}{gt}{bx}{it}{<->ssub*gt/m/n}{}
\DeclareFontShape{JY3}{gt}{bx}{sl}{<->ssub*gt/m/n}{}
\DeclareFontShape{JY3}{mc}{b}{n}{<->ssub*mc/bx/n}{}
\DeclareFontShape{JY3}{mc}{b}{it}{<->ssub*mc/bx/n}{}
\DeclareFontShape{JY3}{mc}{b}{sl}{<->ssub*mc/bx/n}{}
\DeclareFontShape{JY3}{gt}{b}{n}{<->ssub*gt/bx/n}{}
\DeclareFontShape{JY3}{gt}{b}{it}{<->ssub*gt/bx/n}{}
\DeclareFontShape{JY3}{gt}{b}{sl}{<->ssub*gt/bx/n}{}
\DeclareFontShape{JT3}{mc}{m}{it}{<->ssub*mc/m/n}{}
\DeclareFontShape{JT3}{mc}{m}{sl}{<->ssub*mc/m/n}{}
\DeclareFontShape{JT3}{mc}{m}{sc}{<->ssub*mc/m/n}{}
\DeclareFontShape{JT3}{gt}{m}{it}{<->ssub*gt/m/n}{}
\DeclareFontShape{JT3}{gt}{m}{sl}{<->ssub*gt/m/n}{}
\DeclareFontShape{JT3}{mc}{bx}{it}{<->ssub*gt/m/n}{}
\DeclareFontShape{JT3}{mc}{bx}{sl}{<->ssub*gt/m/n}{}
\DeclareFontShape{JT3}{gt}{bx}{it}{<->ssub*gt/m/n}{}
\DeclareFontShape{JT3}{gt}{bx}{sl}{<->ssub*gt/m/n}{}
\DeclareFontShape{JT3}{mc}{b}{n}{<->ssub*mc/bx/n}{}
\DeclareFontShape{JT3}{mc}{b}{it}{<->ssub*mc/bx/n}{}
\DeclareFontShape{JT3}{mc}{b}{sl}{<->ssub*mc/bx/n}{}
\DeclareFontShape{JT3}{gt}{b}{n}{<->ssub*gt/bx/n}{}
\DeclareFontShape{JT3}{gt}{b}{it}{<->ssub*gt/bx/n}{}
\DeclareFontShape{JT3}{gt}{b}{sl}{<->ssub*gt/bx/n}{}

■和文フォント定義  \jsJaFont が指定された場合は、その値をオプションとして luatexja-preset を読み込む。非指定の場合は原ノ味フォントを指定する（luatexja-
preset は読み込まない。
※ 2.0 版より既定を IPAex から原ノ味に変更。

\let\bxjs@jafont@paren\@gobble
\def\bxjs@tmpa{noembed}
\fi
\edef\bxjs@tmpa{\bxjs@catopt\bxjs@tmpa\bxjs@jp@font}
\ifx\bxjs@tmpa\@empty
\defaultjfontfeatures{Kerning=Off}
\setmainjfont[BoldFont=HaranoAjiGothic-Medium.otf,JFM=ujis]{HaranoAjiMincho-Regular.otf}
\setsansjfont[BoldFont=HaranoAjiGothic-Medium.otf,JFM=ujis]{HaranoAjiGothic-Medium.otf}
\else
\edef\bxjs@next{%
\noexpand\RequirePackage[\bxjs@tmpa]{luatexja-preset}%
\bxjs@next
\fi
欧文総称フォント命令で和文フォントが連動するように修正する。その他の和文フォント関係の定義を行う。
\DeclareRobustCommand\rmfamily{\not@math@alphabet\rmfamily\mathrm\romanfamily\rmdefault\kanjifamily\mcdefault\selectfont}
\DeclareRobustCommand\sffamily{\not@math@alphabet\sffamily\mathsf\romanfamily\sfdefault\kanjifamily\gtdefault\selectfont}
\DeclareRobustCommand\ttfamily{\not@math@alphabet\ttfamily\mathtt\romanfamily\ttdefault\kanjifamily\gtdefault\selectfont}
\long\def\jttdefault{\gtdefault}
\unless\ifx\@ltj@match@familytrue\@undefined
\reDeclareMathAlphabet\mathrm{\mathmc}\mathrm{\mathmc}%
\reDeclareMathAlphabet\mathbf{\mathgt}\mathbf{\mathgt}%
\reDeclareMathAlphabet\mathsf{\mathgt}\mathsf{\mathgt}%
\renewcommand\kanjifamilydefault{\gtdefault}
\bxjs@if@sf@default{%
\renewcommand\kanjifamilydefault{\gtdefault}}

■和文パラメータの設定
% 次の３つは既定値の通り
\ltjsetparameter{prebreakpenalty={"",10000}}
\setparameter{postbreakpenalty={`",10000}}
\setparameter{prebreakpenalty={`",10000}}
\setparameter{jaxspmode={`",1}}
\setparameter{jaxspmode={`",2}}
\setparameter{alxspmode={`",3}}
\setparameter{alxspmode={`",3}}

■ 段落頭でのグルー挿入禁止 基本的に現状の ltj* クラスの処理に合わせる。
※ jsInhibitGlueAtParTop は使わない。

\ltjfakeparbegin 現在の LuaTeX-ja で定義されているマクロで、段落中で段落冒頭用の処理を発動する。未定義である場合には、同等のものを用意する。
\ifx\ltjfakeparbegin\@undefined
\protected\def\ltjfakeparbegin{%
\ifhmode
\relax\directlua{luatexja.jfmglue.create_beginpar_node()}}
\fi
\fi

ltj* クラスの定義と同等になるようにパッチを当てる。
\unless\ifnum\bxjs@everyparhook=\bxjs@everyparhook@@none
\begingroup
\let\%\@percentchar \def\@#1{\[\[\detokenize{#1}\]\]}
\@gobble\if\def\bxjs@tmpa{\@\{\everypar\{\everyparhook\fi}}
\ifnum\bxjs@everyparhook=\bxjs@everyparhook@@compat
\@gobble\if\def\bxjs@tmpa{\@\{\everypar{\everyparhook\fi}}
\directlua{
local function patchcmd(cs, code, from, to)
tex.sprint(code:gsub(from:gsub(\%W, \%\%0), \%0..to):gsub("macro:\", \%gdef..cs, 1):gsub("->", "{", 1)..")
end
patchcmd\[\[\meaning\@xsect\]", \[\[\meaning\@xsect\]", \@\{\hskip\-\@tempskipa\}, \@\ltjfakeparbegin\]
patchcmd\[\[\meaning\@item\]", \[\[\meaning\@item\]", \@\bxjs@tmapp, \@\ltjfakeparbegin\]}
\endgroup
\fi

■ hyperref 対策 unicode にするべき。
※ 1.6c 版より、固定ではなく既定設定+検証に切り替えた。
\ifbxjs@hyperref@enc
\PassOptionsToPackage{unicode}{hyperref}
\bxjs@check@hyperref@unicode{true}
\fi

■ 共通命令の実装
\newcommand*{\setkanjiskip}{\jsSetKanjiSkip}
4979 \newcommand*{\getkanjiskip}{\jsGetKanjiSkip}
4980 \newcommand*{\setxkanjiskip}{\jsSetXKanjiSkip}
4981 \newcommand*{\getxkanjiskip}{\jsGetXKanjiSkip}
4982 \protected\def\autospacing{%
4983 \ltjsetparameter{autospacing=true}}
4984 \protected\def\noautospacing{%
4985 \ltjsetparameter{autospacing=false}}
4986 \protected\def\autoxspacing{%
4987 \ltjsetparameter{autoxspacing=true}}
4988 \protected\def\noautoxspacing{%
4989 \ltjsetparameter{autoxspacing=false}}
4990 \def\jsApplyKanjiSkip#1{%
4991 \ltjsetparameter{kanjiskip={#1}}}
4992 \def\jsApplyXKanjiSkip#1{%
4993 \ltjsetparameter{xkanjiskip={#1}}}

\jachar のサブマクロの実装。
4994 \def\bxjs@jachar#1{%
4995 \ltjjachar`#1\relax}
4996 \jathinspace の実装。
4997 \ifx\bxjs@jaspace@cmd
4998 \protected\def\jathinspace{%
4999 \hskip\ltjgetparameter{xkanjiskip}\relax}
5000 \fi
5001

■和文数式ファミリ LuaTeX-ja では和文数式ファミリは常に有効で、既にこの時点で必要
5002 な設定は済んでいる。従って @enablejfam は常に真になる。
5003 \ifx f\bxjs@enablejfam
5004 \ClassWarningNoLine\bxjs@clsname
5005 \MessageBreak
5006 \{You cannot use 'enablejfam=false', since the\MessageBreak
5007 LuaTeX-ja always provides Japanese math families\}
5008 \fi
5009

C.8 共通処理 (2)
5010 \fi\fi\fi\fi

■共通命令の実装
\textmc minimal ドライバ実装中で定義した \DeclareJaTextFontCommand 利用する。
\textgt 5006 \ifx\DeclareFixJFMCJKTextFontCommand\@undefined
5007 \DeclareJaTextFontCommand{\textmc}{\mcfamily}
5008 \DeclareJaTextFontCommand{\textgt}{\gtfamily}
5009 \fi
5010
\mathmc この時点で未定義である場合に限り、\DeclareJaMathFontCommand 利用したフォール
\mathgt パックの定義を行う。
5011 \ifx\mathmc\@undefined
以上で終わり。

付録 D  和文ドライバ：modern ☃

モダンな設定。
standard ドライバの設定を引き継ぐ。

D.1 フォント設定

T1 エンコーディングに変更する。
※以下のコードは \usepackage{T1}{fontenc} と同等。
\input{bxjsja-standard.def}

\fontencoding{T1}\selectfont

基本フォントを Latin Modern フォントファミリに変更する。
※以下は \usepackage{noamth}{lmodern} と同じ。ユーザは後で lmodern を好きなオプションを付けて読み込むことができる。
\input{t1enc.def}\
\fontencoding{T1}\selectfont

大型演算子用の数式フォントの設定。
※ amssfonts パッケージと同等にする。
\DeclareFontShape{OMX}{cmex}{m}{n}{%\cmex@opt{10}}\expandafter\let\csname OMX/cmex/m/n/10\endcsname\relax
amssmath 読込時に上書きされるのを防ぐ。
\def\cmex@opt{10}

D.2 fixltx2e 読込

※ fixltx2e 廃止前の \LaTeX \ カーネルの場合。
D.3 和文カテゴリコード

和文カテゴリコード設定のための補助パッケージを読みこむ。

D.4 完了

おしまい。

付録 E 和文ドライバ：pandoc

「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。

E.1 準備

xeCJK で space が有効になるのを阻止する。
※ bxjsja-standard.def の中で xeCJK が読み込まれるためこの位置に置いている。

standard ドライバの設定を引き継ぐ。

■環境検査

TODO: 3.0 以下で 3.0 版までのバージョン要件の予定について述べておく。
pandoc 和文ドライバの処理系バージョン要件は standard と同じとする。加えて、以下の要件を定める。

• pTeX 系も含めて全てのエンジン種別で \-pTeX 拡張を要求する。
• 特に etoolbox の 2.0 版以上を要求する。
※もちろん他にも追加の依存パッケージがある。

■パッケージ読込
bxjspandoc パッケージを読み込む。

\RequirePackage{bxjspandoc}
\-pTeX ではない場合に警告を出す。
This engine does not support e-TeX extension!
Some feature might not work properly
\fi

\ifbxjs@bxghost@available

その他の依存パッケージを読み込む。
\RequiresPackage{iftex}[2013/04/04]% v0.2
\fi

E.2 和文ドライバパラメタ

keyval のファミリは bxjsPan とする。
\ifbxjs@jp@fix@strong

\let\bxjs@kv@fixstrong@true\bxjs@jp@fix@strongtrue
\let\bxjs@kv@fixstrong@false\bxjs@jp@fix@strongfalse
\define@key{bxjsPan}{fix-strong}{true}{%\bxjs@set@keyval{fixstrong}{#1}{}}

\ifbxjs@jp@fix@code

\let\bxjs@kv@fixcode@true\bxjs@jp@fix@codetrue
\let\bxjs@kv@fixcode@false\bxjs@jp@fix@codefalse
\define@key{bxjsPan}{fix-code}{true}{%\bxjs@set@keyval{fixcode}{#1}{}}

174
重要要素に適用される書体変更の種類。

\begin{verbatim}
#strongオプションの処理。
\def\bxjs@kv@strong@bold\{\chardef\bxjs@jp@strong=0\}
\def\bxjs@kv@strong@sans\{\chardef\bxjs@jp@strong=1\}
\def\bxjs@kv@strong@boldsans\{\chardef\bxjs@jp@strong=2\}
\define@key{bxjsPan}{strong}{%}
\ifbxjs@jp@or@indent
\ifbxjs@jp@or@secnumdepth
\ifbxjs@jp@or@block@heading
プレアンプルでのレイアウト上書きを許可するか。既定値は真。
\fi
\newif\ifbxjs@jp@or@indent \bxjs@jp@or@indenttrue
\newif\ifbxjs@jp@or@secnumdepth \bxjs@jp@or@secnumdepthtrue
\newif\ifbxjs@jp@or@block@heading \bxjs@jp@or@block@headingtrue
\fi
\fi
\fi
\define@key{bxjsPan}{_plus}\[]\}{%}
\bxjs@jp@or@indentfalse
\bxjs@jp@or@secnumdepthfalse
\bxjs@jp@or@block@headingfalse}
\fi
\let\bxjs@kv@orindent@true\bxjs@jp@or@indenttrue
\let\bxjs@kv@orindent@false\bxjs@jp@or@indentfalse
\define@key{bxjsPan}{or-indent}\[true\]{%}
\let\bxjs@kv@orsecnumdepth@true\bxjs@jp@or@secnumdepthtrue
\let\bxjs@kv@orsecnumdepth@false\bxjs@jp@or@secnumdepthfalse
\define@key{bxjsPan}{or-secnumdepth}\[true\]{%}
\let\bxjs@kv@orblockheading@true\bxjs@jp@or@block@headingtrue
\let\bxjs@kv@orblockheading@false\bxjs@jp@or@block@headingfalse
\define@key{bxjsPan}{or-block-heading}\[true\]{%}
\fi
\let\bxjs@next\bxjs@safe@setkeys{bxjsPan}{#1}\}
\expandafter\bxjs@next\expandafter{\jsJaParam}
\end{verbatim}

クラスでpandoc+が指定された場合、内部和文パラメータ_plusが和文ドライバに渡される。この場合、レイアウト上書きを禁止する。
※_plusは必ずパラメータ列の先頭にあるので、個別のおパラメータ設定の方が常に優先される。
\define@key{bxjsPan}{_plus}\[]\}{%}
\bxjs@jp@or@indentfalse
\bxjs@jp@or@secnumdepthfalse
\bxjs@jp@or@block@headingfalse}
レイアウト上書き許可オプション(or-indent・or-secnumdepth・or-block-heading)の処理。
\let\bxjs@kv@orindent@true\bxjs@jp@or@indenttrue
\let\bxjs@kv@orindent@false\bxjs@jp@or@indentfalse
\define@key{bxjsPan}{or-indent}true{%}
\let\bxjs@kv@orsecnumdepth@true\bxjs@jp@or@secnumdepthtrue
\let\bxjs@kv@orsecnumdepth@false\bxjs@jp@or@secnumdepthfalse
\define@key{bxjsPan}{or-secnumdepth}true{%}
\let\bxjs@kv@orblockheading@true\bxjs@jp@or@block@headingtrue
\let\bxjs@kv@orblockheading@false\bxjs@jp@or@block@headingfalse
\define@key{bxjsPan}{or-block-heading}true{%}
\fi
実際のjaparamの値を適用する。
\def\bxjs@next\bxjs@safe@setkeys{bxjsPan}#1\}
\expandafter\bxjs@nnext\expandafter{\jsJaParam}

E.3 duploadシステム

TODO: 新しいカーネルで利用可能な機構での代替を検討する。カーネルへのパッチは排除したいので。
パッケージが重複して読み込まれたときに“option clash”的検査をスキップする。この時に何らかのコードを実行させることができる。
指定の名前の特定のファイルの読込が \filewithoptions で指示されて、しかもそのファイルが読込済である場合に、オプション重複検査をスキップして、代わりに〈定義本体〉のコードを実行する。このコード中で #1 は渡されたオプション列のテキストに置換される。

5103 \onlypreamble\bxjs@set@dupload@proc
5104 \def\bxjs@set@dupload@proc#1{%
5105 \expandafter\bxjs@set@dupload@proc@a\csname bxjs@dlp/#1\endcsname
5106 \onlypreamble\bxjs@set@dupload@proc@a
5107 \def\bxjs@set@dupload@proc@#1{%
5108 \onlypreamble#1\def#1##1
5109 \bxjs@cslet\bxjs@dlp/#1\@undefined
5110 \@if@ptions
5111 \@if@ptions 重启定義。
5112 \let\bxjs@org@if@ptions\@if@ptions
5113 \onlypreamble\bxjs@org@reset@ptions
5114 \let\bxjs@org@reset@ptions\relax
5115 \def\@if@ptions#1#2#3{%
5116 \let\bxjs@next\@secondoftwo
5117 \def\bxjs@tmpa{#1}\def\bxjs@tmpb{\@currext}%
5118 \ifx\bxjs@tmpa\bxjs@tmpb
5119 \expandafter\if\csname bxjs@dlp/#2.#1\endcsname\relax
5120 \let\bxjs@next@\@firstoftwo \fi
5121 \fi
5122 \bxjs@next\bxjs@do@dupload@proc\bxjs@org@if@ptions{#1}{#2}{#3}
5123 \g@addto@macro\bxjs@begin@document@hook{%
5124 \let\@if@ptions\bxjs@org@if@ptions
5125 \onlypreamble\bxjs@do@dupload@proc
5126 \def\bxjs@do@dupload@proc#1#2#3{%
5127 \if\bxjs@org@reset@ptions\relax
5128 \let\bxjs@org@reset@ptions\@reset@ptions
5129 \fi
5130 \bxjs@csletcs{\bxjs@next}{\bxjs@dlp/#2.#1}%
5131 \def\@reset@ptions{%
5132 \let\@reset@ptions\bxjs@org@reset@ptions
5133 \@reset@ptions
5134 \bxjs@next{#3}%%
5135 \@firstoftwo}

E.4 lang 変数

lang=ja という言語指定が行われると、2.12 版より前の Pandoc はこれに対応していなかったため不完全な Babel や Polyglossia の設定を出力してしまっていた。現在では lang=ja 指定について正しく InPpX 側の言語名 japanese に変換されるようになっているが、それでも日本語指定の場合は相変わらず調整処理が必要である。

176
 bxjs@polyglossia@options Polyglossia のオプション列のテキスト。“実際にには読み込まれていない” 場合は \relax になる。

\let\bxjs@polyglossia@options\relax

エンジンが Xe\TeX か Lua\TeX の場合が対象になる。
※この場合 etoolbox が使用可能になっている。

パッケージの読込を検知するため読込済のマークを付けて dupload の処理を仕込む。

\pandocSkipLoadPackage{polyglossia}
\bxjs@set@dupload@proc{polyglossia.sty}{% %\bxjs@unset@dupload@proc{polyglossia.sty} %\ClassWarning\bxjs@clsname{Package polyglossia is requested} %\def\bxjs@polyglossia@options{#1} %

polyglossia の読込が指示された場合、直後に \setmainlanguage が実行されることを想定して、フック用の \setmainlanguage を定義する。
※最初に \setmainlanguage 以外が実行された場合はエラーになる。

\newcommand*{\setmainlanguage}[2]{%もし、\setmainlanguage の引数が空または japanese だった場合はメインが日本語である（lang=ja 指定）と見なす。
\ifboolexpr{test{\ifblank{##2}}or test{\ifstrequal{##2}{japanese}}}{{%\ClassWarning\bxjs@clsname{Main language is 'japanese', thus fallback\MessageBreak definitions will be employed} %\csundef{ver@polyglossia.sty}%}{%else %\ClassWarning\bxjs@clsname{Main language is '#2',\MessageBreak thus polyglossia will be loaded}% %\csundef{ver@polyglossia.sty}%}

それ以外は、改めて polyglossia を読み込んで、本来の処理を実行する。
\}

\ClassWarning\bxjs@clsname{Main language is '#2',\MessageBreak thus polyglossia will be loaded} %\csundef{ver@polyglossia.sty} %
プレアンブルで polyglossia の読込が指示されなかった場合、Polyglossia と連携するパッケージの誤動作を防ぐため、\AtEndPreamble において読込済マークを外す。

\def\bxjs@next{%
  \noexpand\RequirePackage[\bxjs@polyglossia@options]{polyglossia}[\]
  }\bxjs@next

\setmainlanguage[##1]{##2}{}

プレアンブルで polyglossia の読込が指示されなかった場合、Polyglossia と連携するパッケージの誤動作を防ぐため、\AtEndPreamble において読込済マークを外す。

\g@addto@macro\bxjs@endpreamble@hook{%
  \ifx\bxjs@polyglossia@options\relax
    \csundef{ver@polyglossia.sty}%
  \fi}

\bxjs@pandoc@polyglossia@ja

Pandoc 側で lang=ja が指定されていた場合の処理。この場合は Polyglossia の処理を無効化するためにダミーの定義を行う。すなわち、サブ言語 xxx の各々について、xxx 環境と \textxxx 命令を（特に何も加工しないものとして）定義する。この目的のため、\setotherlanguage{s} をダミーを定義する命令として定義する。

\begin{verbatim}
\@onlypreamble\bxjs@pandoc@polyglossia@ja
\def\bxjs@pandoc@polyglossia@ja{%
  \renewcommand*{\setmainlanguage}[2][]{}%
  \newcommand*{\setotherlanguage}[2][]{}%
  \iftoken{##2}{\relax}{%else
    \cslet{##2}\@empty \cslet{end##2}\@empty
    \cslet{text##2}\@firstofone}}%
  \newcommand*{\setotherlanguages}[2][]{}%
  \@for\bxjs@tmpa:=##2\do{%\setotherlanguage{\bxjs@tmpa}}%
\end{verbatim}

Polyglossia の読込済マークは外れるようにしておく。

\begin{verbatim}
\let\bxjs@polyglossia@options\relax}
\fi
\end{verbatim}

■Babel について 現在の Pandoc では、テンプレートで用いられる多言語パッケージとしてエンジンの種別によらずに Babel が使われる。
※ XƎTeX では 2.15 版で、LuaTeX は 2.6 版で Polyglossia から Babel に変更されている。

\begin{verbatim}
\bxjs@babel@options
Babel のオプション列のテキスト。“実際には読み込まれていない” 場合は \relax になる。
\let\bxjs@babel@options\relax
\end{verbatim}

パッケージの読込を検知するため読込済のマークを付けて dupload の処理を仕込む。

\begin{verbatim}
\pandocSkipLoadPackage{babel}
\bxjs@set@dupload{proc=babel.sty}{%}
\bxjs@unset@dupload{proc=babel.sty}%
\ClassWarning{\bxjs@clsname}{Package babel is requested}%
\@tempswafalse \let\bxjs@babel@options\@empty
\end{verbatim}

バッケージオプションに言語名が空の main= がある場合は、main=japanese に置き換える。

\begin{verbatim}
\@tempswafalse \let\bxjs@babel@options\empty
\end{verbatim}

178
プレアンプルで babel の読込が指示されなかった場合、読込済マークを外す。

3.0 版より前の japanese.ldf はサポート対象エンジンが限られていた。ここでは、エンジンの種類を問わず、「japanese.ldf が古い場合は読込を回避してダミー定義で代替する」という対策を入れる。実は japanese.ldf で行う定義は bxorigcapt の機能等により実質的に全て無効化されている。最新の環境においては「japanese 指定の Babel + bxorigcapt パッケージ」の状態にしておきたい。

filehook の機能を用いて japanese.ldf の読込にフックを仕込む。

バージョンを判定する部分。
※filehook にも細工を入れている理由は、初期の japanese.ldf には ProvidesLanguage が記述されていなかったため。
ダミーの言語定義。

ダミーの言語定義。

\ifx t\bxjs@japanese@ldf@skipped
\ClassWarningNoLine\bxjs@clname
\{Loading japanese.ldf is skipped\}%
\fi

\ifundef\l@japanese{\chardef\l@japanese\z@}{%}
\let\datejapanese\@empty\let\captionsjapanese\@empty
\let\extrasjapanese\@empty\let\noextrasjapanese\@empty
\main@language{japanese}%
\fi

\g@addto@macro\bxjs@begin@document@hook{%
\let\bxjs@begin@japanese@ldf@hook\relax
\let\bxjs@end@japanese@ldf@hook\relax}
\fi

lang 対策はこれで終わり。

E.5 geometry 変数

geometry を “再度読み込んだ” 場合に、そのパラメータで \setpagelayout* が呼ばれるようにする。
\bxjs@set@dupload@proc{geometry.sty}{%
\setpagelayout*{#1}}%

E.6 CJKmainfont 変数

LuaTeX (+ LuaTeX-ja) の場合に CJKmainfont 変数が指定された場合は \setmainjfont の指定にまわす。
\if l\jsEngine
\pandocSkipLoadPackage{xeCJK}
\providecommand*{\setCJKmainfont}{\setmainjfont}
\fi

E.7 Option clash 対策

xeCJK パッケージについて。
※ xeCJK はクラス内で既に読み込まれているので、space は（意図通りに）無効になる。
※ v2.8～v2.9.2 の間。
\if x\jsEngine
\expandafter\g@addto@macro\csname opt@xeCJK.sty\endcsname{%
 ,space)
\fi

E.8 レイアウト上書き禁止

レイアウト上書き禁止の実装は etoolbox の機能を使う。
■indentについて
indent 変数を指定しない場合に「段落表現形式をインデント方式に変更する」動作を抑止する。


parskip がある場合はそれを読み込むようとするため、parskip の読込をブロックする。

parskip がない場合はパラメタを変更しようとするため、該当のパラメタを復帰させる。

■secnumdepthについて
secnumdepth の値を決めるのは numbersections 変数（-N/--number-sections オプションに連動する）や secnumdepth 変数であるが、何れにしても secnumdepth の値は書き換えられる。そのため、secnumdepth を復帰させる。

■block-headingについて
paragraph、subparagraph を別行見出しに変える処理を抑止する。
※2.7.1版以前では別行見出し変更が既定で有効であった。
E.9 paragraph のマーク

BXJS クラスでは \paragraph の見出しの前に \jsParagraphMark で指定したマークが付加され、既定ではこれは“■”である。しかし、この規定は \paragraph が本来のレイアウトを保っている、すなわち「行内見出しである」「節番号が付かない」ことが前提になっていると考えられる。Pandoc はこの規定を変更することがある（特に既定で \paragraph を別行見出しに再定義する）ため、変更された場合は \jsParagraphMark の既定値を空にする。

Pandoc がプレアンプルで行う再定義の結果を調べるため、begin-document フックを利用する。

まず、マーク変更が必要かを調べる。\oldparagraph という制御綴が定義済の場合、Pandoc が \paragraph の様式を変更したということなので、マーク変更が必要である。

パラグラフ番号付きの場合は、マーク変更が必要である。

「マーク変更が必要」である場合、\jsParagraphMark が既定値のままであれば空に変更する。

E.10 全角空白文字

LaTeX でない入力では、全角空きを入れるために全角空白文字（U+3000）が使われる可能性があるので、全角空白文字を和文文字でなく空きとして扱うようにしておく。
※ (u)LaTeX では対応できないので対象外。

\pandocZWSpace 全角空白文字の入力で実行されるコード。

\def\pandocZWSpace{\zwspace}

全角空白文字の入力で \pandocZWSpace が実行されるようにする。
E.11 hyperref 対策

hyperref の unicode オプションの固定を行う。

TODO: unicode オプションの固定処理は可能なら廃止したい。hyperref の開発状況を鑑みる限り、固定処理は危険なので。

E.12 Pandoc 要素に対する和文用の補正

■重要要素 重要 (Strong) 要素に対する \LaTeX 出力は \textbf となるが、代わりに \textbf を書き換える（うわぁ）。

■インラインコード要素 インラインコード (Code) 要素に対する \LaTeX 出力は \texttt となる。\texttt の両端に欧文ゴーストが入るようにする。さらに \verb の外側にも欧文ゴーストが入るようにする。
いる。

E.13 ifPDFTeX スイッチ

Pandoc モードでは Pandoc の既定テンプレートを（無理やり）(u)pTeX に対応させることを目的にしている。

旧版のテンプレートでは ifxetex と ifluatex パッケージを読み込んだ上で「XeTeX でも LuaTeX でもないものは pdfTeX」という前提の動作をしていた。よって、(u)pTeX に対応させる際には「pdfTeX 用の処理が実行される」ことを前提にすればよかった。

ところが、Pandoc の 2.12 版では ifxetex パッケージが導入されて「pdfTeX の処理を直接
\ifPDFTeX で行う」ように改修された。このため、(u)p\TeX での実行でどのコードが実行されるかを予期することが困難になってしまった。
これに対処するため、「文書ファイルのプレアンプル実行中に限って \ifPDFTeX が（実際とは異なり）真になるようにする」という細工を施すことで、従来通り「pdf\TeX 用の処理が実行される」前提が維持されるようにする。

プレアンプル開始時の \ifPDFTeX は真で、終了時に偽装を無効化する。\filehook のフックで「パッケージ読込中は偽装を解除する」ことを実現している。

E.14 完了
おしまい。

和文ドライバ実装はここまで。
付録 F 補助パッケージ一覧

BXJS クラスの機能を実現するために用意されたものだが、他のクラスの文書で読み込んで利用することもできる。

- **bxjscompat** : 古いやつをどうにかするナニカ。
- **bxjsjckcat** : modern ドライバ用の和文カテゴリを適用する。
- **bxjspandoc** : Pandoc 用のナニカ。

付録 G 補助パッケージ：bxjscompat

古いやつをどうにかするためのムニャムニャ。
※すなわち BXJS クラスにおいては「新しいシステムにおいては bxjscompat がなくても正常に動作する」状態を保つべき。

G.1 準備

\begin{verbatim}
\def\bxac@pkgname{bxjscompat}
\end{verbatim}

エンジンの種別。

\begin{verbatim}
\let\bxac@engine=n
\edef\bxac@do\string#1{\edef\bxac@tmpa{#1}\edef\bxac@tmpb{\meaning#1}\ifx\bxac@tmpa\bxac@tmpb#2\fi}
\bxac@do\kanjiskip{\let\bxac@engine=j}
\bxac@do\XeTeXversion{\let\bxac@engine=x}
\bxac@do\luatexversion{\let\bxac@engine=l}
\end{verbatim}

もし BXJS クラスの読込中にこのパッケージが読み込まれているならば、BXJS のクラスの終わりまで実行を遅延する。

\begin{verbatim}
\ifx\jsAtEndOfClass\@undefined
\let\bxac@delayed@if@bxjs\@firstofone
\else \let\bxac@delayed@if@bxjs\jsAtEndOfClass\fi
\end{verbatim}

\begin{verbatim}
\newif\ifbxac@in@old@behavior
\let\ImposeOldLuaTeXBehavior\relax
\let\RevokeOldLuaTeXBehavior\relax
\end{verbatim}

\begin{verbatim}
\IfImposedOldLuaTeXBehavior \IfImposedOldLuaTeXBehavior は 0.85 版以降の LuaTeX を一時的に pdLaTeX と互換である\RevokeOldLuaTeXBehavior ように見せかける。\RevokeOldLuaTeXBehavior で元に戻すことができる。
\end{verbatim}

※エンジンが LuaTeX 以外の場合は何もしない。
G.2 8bit 欧文 TeX

5388 \ifx n\bxac@engine
    和文を含むマクロ定義を通用させるため、高位バイトをアクティブ化しておく。
5389 \@tempcnta="80 \loop \ifnum\@tempcnta<"100
5390 \catcode\@tempcnta\active
5391 \advance\@tempcnta\@ne
5392 \repeat
    以上。
5393 \fi

G.3 XƎTeX

5394 \ifx x\bxac@engine

■文字クラスの設定  XƎTeX の文字クラス（\XeTeXcharclass）の Unicode 規定に基づく設定は、初期の版ではフォーマットに組み込まれていたが、2016/02/01 以降の IXTeX カーネルでは「必要に応じて後から設定用のファイルを読み込む」方式に変更された。ここでは「設定されている状態」を担保する。
※ちなみに、XƎTeX に「文字間トークン挿入」の機能が導入されたのは 0.997 版（2007 年頃）からのようだ。
ただし xeCJK が読込済ならば（そちらが適切に設定しているはずなので）何もしない。
5395 \ifx\XeTeXcharclass\@undefined\else
5396 \bxac@delayed@if@bxjs{%
5397 \@ifpackageloaded{xeCJK}{%else
設定が未実行の状態ならば、設定用のファイルを読む。
5398 \ifx\xe@alloc@intercharclass\@undefined\else
5399 \ifnum\xe@alloc@intercharclass=\z@
5400 \PackageInfo\bxac@pkgname
5401 {Setting up interchar class for CJK...\@gobble}%
5402 \InputIfFileExists{load-unicode-xetex-classes.tex}{}%
5403 \xe@alloc@intercharclass=3
5404 }{%else
5405 \PackageWarning\bxac@pkgname
5406 {Cannot find file 'load-unicode-xetex-classes.tex'\%}
5407 \@gobble}%
5408 }%
5409 \fi\fi

フォーマット組込だった時代の設定は不完全なところがあるので補正する。
5410 \ifnum\XeTeXcharclass="3041=\z@
5411 \PackageInfo\bxac@pkgname
5412 {Adjusting interchar class for CJK...\@gobble}%
5413 \@for\bxac@tmpb:={{%
5414 3041,3043,3045,3047,3049,3063,3083,3085,3087,308E,%
5415 3095,3096,30A1,30A3,30A5,30A7,30A9,30C3,30E3,30E5,%

187
漢字および完成形ハングルのカテゴリコードが確実に11になっているようにする。

\chardef\bxac@tmpb=11
\def\bxac@do#1#2{%
  \@tempcnta=#1\relax
  \unless\ifnum\catcode\@tempcnta=\bxac@tmpb
  \chardef\bxac@tmpa=#2\relax
  \@whilenum{\@tempcnta<\bxac@tmpa}\do{%
    \catcode\@tempcnta\bxac@tmpb \advance\@tempcnta\@ne
  }%
  \fi}
\bxac@do{"4E00}{"9FCD}

以上。

G.4 LuaTeX

0.82〜0.84版のLuaTeXを（0.81版以前と同様に）pdfTeXの拡張であるように見せる処理。
※恐らく必要な場面はなかったと思われるので、外しておく。

\unless\ifnum\luatexversion<80 \ifnum\luatexversion<85
  \chardef\pdftexversion=200
  \def\pdftexrevision{0}
  \let\pdftexbanner\luatexbanner
\fi\fi
\ImposeOldLuaTeXBehavior
\RevokeOldLuaTeXBehavior

\begingroup\expandafter\expandafter\expandafter\endgroup
\expandafter\ifx\csname outputmode\endcsname\relax\else

該当する場合、以下の5つのpdfTeX拡張プリミティブを復帰させることになる。
\def\bxac@ob@list{%
  \do\let\pdfoutput{\outputmode}%
  \do\let\pdfpagewidth{\pagewidth}%
  \do\let\pdfpageheight{\pageheight}%
  \do\protected\edef{\pdfvariable horigin}{\pdfvariable horigin}%
  \do\protected\edef{\pdfvariable vorigin}{\pdfvariable vorigin}%
\def\bxac@ob@do#1#2#3#4\do\begingroup
\expandafter\bxac@ob@do@a\csname bxac@\string#2\endcsname{#1}#2}
漢字および完成形ハングルのカテゴリコードが確実に11になっているようにする。

\directlua{
local function range(cs, ce, cc, ff)
  if ff or not tex.getcatcode(cs) == cc then
    local setcc = tex.setcatcode
    for c = cs, ce do setcc(c, cc) end
  end
end
range(0x3400, 0x4DB5, 11, false)
ifnum\luatexversion>64
  range(0x4DB5, 0x4DBF, 11, true)
  range(0x4E00, 0x9FCC, 11, false)
  range(0x9FCD, 0x9FFF, 11, true)
  range(0xAC00, 0xD7A3, 11, false)
  range(0x20000, 0x2A6D6, 11, false)
  range(0x2A6D7, 0x2A6FF, 11, true)
  range(0x2A800, 0x2B734, 11, false)
  range(0x2B735, 0x2B73F, 11, true)
  range(0x2B740, 0x2B81D, 11, false)
  range(0x2B81E, 0x2B81F, 11, true)
  range(0x2B820, 0x2CEA1, 11, false)
  range(0x2CEA2, 0x2FFFD, 11, true)
\fi
}

以上。

\G.5 完了

おしまい。
付録 H 補助パッケージ：bxjscjkcat

modern ドライバ用の和文カテゴリを適用する。

H.1 準備

\def\bxjx@pkgname{bxjscjkcat}
\newcount\bxjx@cnta
\onlypreamble\bxjx@tmpdo
\onlypreamble\bxjx@tmpdo@a
\onlypreamble\bxjx@tmpdo@b

\bxjx@engine エンジンの種別。
\let\bxjx@engine=n
\def\bxjx@tmpdo#1#2{\edef\bxjx@tmpa{\string#1}\edef\bxjx@tmpb{\meaning#1}\ifx\bxjx@tmpa\bxjx@tmpb #2\fi}
\bxjx@tmpdo\kanjiskip{\let\bxjx@engine=j}
\bxjx@tmpdo\enablecjktoken{\let\bxjx@engine=u}
\bxjx@tmpdo\XeTeXversion{\let\bxjx@engine=x}
\bxjx@tmpdo\pdftexversion{\let\bxjx@engine=p}
\bxjx@tmpdo\luatexversion{\let\bxjx@engine=l}

それぞれのエンジンで、前提となる日本語処理パッケージが実際に読み込まれているかを検査する。
\def\bxjx@tmpdo#1#2{\if#1\bxjx@engine\@ifpackageloaded{#2}{}{%else\PackageError{bxjx@pkgname}{Package '#2' must be loaded}{Package loading is aborted.\MessageBreak@ehc}\endinput}\fi}
\bxjx@tmpdo{p}{bxcjkjatype}
\bxjx@tmpdo{x}{xeCJK}
\bxjx@tmpdo{l}{luatexja}

古い LATEXの場合、\TextOrMathはfixltx2eパッケージで提供される。
\ifx\TextOrMath\@undefined
\RequirePackage{fixltx2e}
\fi
H.2 和文カテゴリコードの設定

upLaTeX の場合、和文カテゴリコードの設定を LuaTeX-ja と（ほぼ）等価なものに変更する。
※ LuaTeX-ja との相違点：A830, A960, 1B000。

H.3 ギリシャ・キリル文字の扱い

「特定 CJK 曖昧文字」について、和文・欧文扱いを制御できるようにする。ここで「特定 CJK 曖昧文字」とは以下の該当する文字の集合を指す：

- Unicode と JIS X 0213 に共通して含まれるギリシャ文字・キリル文字。
• Latin-1 の上位部分と JIS X 0208 に共通して含まれる文字（LuaTeX-ja の定める "範囲 8"）。

\bxjx@grkcyr@list「特定 CJK 暖昧文字」に関する情報をもつ \do-リスト。各項目の形式は以下の通り：
\do{Unicode 符号値}{\text{対象 fontenc}}{\text{テキスト LICR}}{\text{数式 LICR}}
※数式で使わない文字は \text{数式 LICR} を空にする。
5556 \onlypreamble\bxjx@grkcyr@list
5557 \def\bxjx@grkcyr@list{%
5558 \do{0391}{LGR}{\textAlpha}{A}% % GR. C. L. ALPHA
5559 \do{0392}{LGR}{\textBeta}{B}% % GR. C. L. BETA
5560 \do{0393}{LGR}{\textGamma}{\Gamma}% % GR. C. L. GAMMA
5561 \do{0394}{LGR}{\textDelta}{\Delta}% % GR. C. L. DELTA
5562 \do{0395}{LGR}{\textEpsilon}{E}% % GR. C. L. EPSILON
5563 \do{0396}{LGR}{\textZeta}{\Zeta}% % GR. C. L. ZETA
5564 \do{0397}{LGR}{\textEta}{\Eta}% % GR. C. L. ETA
5565 \do{0398}{LGR}{\textTheta}{\Theta}% % GR. C. L. THETA
5566 \do{0399}{LGR}{\textIota}{\Iota}% % GR. C. L. IOTA
5567 \do{039A}{LGR}{\textKappa}{\Kappa}% % GR. C. L. KAPPA
5568 \do{039B}{LGR}{\textLambda}{\Lambda}% % GR. C. L. LAMDA
5569 \do{039C}{LGR}{\textMu}{\Mu}% % GR. C. L. MU
5570 \do{039D}{LGR}{\textNu}{\Nu}% % GR. C. L. NU
5571 \do{039E}{LGR}{\textXi}{\Xi}% % GR. C. L. XI
5572 \do{039F}{LGR}{\textOmicron}{O}% % GR. C. L. OMICRON
5573 \do{03A0}{LGR}{\textPi}{\Pi}% % GR. C. L. PI
5574 \do{03A1}{LGR}{\textRho}{\Rho}% % GR. C. L. RHO
5575 \do{03A2}{LGR}{\textSigma}{\Sigma}% % GR. C. L. SIGMA
5576 \do{03A3}{LGR}{\textTau}{\Tau}% % GR. C. L. TAU
5577 \do{03A4}{LGR}{\textUpsilon}{\Upsilon}% % GR. C. L. UPSILON
5578 \do{03A5}{LGR}{\textPhi}{\Phi}% % GR. C. L. PHI
5579 \do{03A6}{LGR}{\textChi}{\Chi}% % GR. C. L. CHI
5580 \do{03A7}{LGR}{\textPsi}{\Psi}% % GR. C. L. PSI
5581 \do{03A8}{LGR}{\textOmega}{\Omega}% % GR. C. L. OMEGA
5582 \do{03B1}{LGR}{\textAlpha}{\alpha}% % GR. S. L. ALPHA
5583 \do{03B2}{LGR}{\textBeta}{\beta}% % GR. S. L. BETA
5584 \do{03B3}{LGR}{\textGamma}{\gamma}% % GR. S. L. GAMMA
5585 \do{03B4}{LGR}{\textDelta}{\delta}% % GR. S. L. DELTA
5586 \do{03B5}{LGR}{\textEpsilon}{\epsilon}% % GR. S. L. EPSILON
5587 \do{03B6}{LGR}{\textZeta}{\zeta}% % GR. S. L. ZETA
5588 \do{03B7}{LGR}{\textTheta}{\theta}% % GR. S. L. ETA
5589 \do{03B8}{LGR}{\textIota}{\iota}% % GR. S. L. IOTA
5590 \do{03B9}{LGR}{\textKappa}{\kappa}% % GR. S. L. KAPPA
5591 \do{03BA}{LGR}{\textLambda}{\lambda}% % GR. S. L. LAMDA
5592 \do{03BB}{LGR}{\textMu}{\mu}% % GR. S. L. MU
5593 \do{03BC}{LGR}{\textNu}{\nu}% % GR. S. L. NU
5594 \do{03BD}{LGR}{\textXi}{\xi}% % GR. S. L. XI
5595 \do{03BE}{LGR}{\textOmicron}{o}% % GR. S. L. OMICRON
5597 \do{03CO}{LGR}{\textPi}{\pi}% % GR. S. L. PI
面倒なので補っておく。

\providecommand*{\mathdegree}{{}^\circ}

\ifbxjx@gcc@cjk

〔スイッチ〕「特定 CJK 曖昧文字」を和文扱いにするか。

\newif\ifbxjx@gcc@cjk

\greekasCJK 〔公開命令〕「特定 CJK 曖昧文字」を和文扱いにする。

\newcommand*\greekasCJK{\bxjx@gcc@cjktrue}

\nogreekasCJK 〔公開命令〕「特定 CJK 曖昧文字」を欧文扱いにする。

\newcommand*\nogreekasCJK{\bxjx@gcc@cjkfalse}

\mathdegree 面倒なので補っておく。

\providecommand*{\mathdegree}{{}^\circ}

\ifbxjx@gcc@cjk

〔スイッチ〕「特定 CJK 曖昧文字」を和文扱いにするか。

\newif\ifbxjx@gcc@cjk

\greekasCJK 〔公開命令〕「特定 CJK 曖昧文字」を和文扱いにする。

\newcommand*\greekasCJK{\bxjx@gcc@cjktrue}

\nogreekasCJK 〔公開命令〕「特定 CJK 曖昧文字」を欧文扱いにする。

\newcommand*\nogreekasCJK{\bxjx@gcc@cjkfalse}
ラテン文字で代用される数式ギリシャ文字の出力を行う。(基準文字) (mathchardef の制御綴) の数式クラスと数式ファミリを引き継いで、(出力文字) (ASCII 文字トークン) の文字コードの数式文字を出力する。例えば、\Pi の意味が \mathchar"7005 である場合、\bxjx@fake@grk\{\Pi\} は \mathchar"7042 を実行する。

※フォントパッケージ使用時の再定義を考慮して、(基準文字) が mathchardef であるかを検査し、そうでない場合はフォールバックとして単に ⟨出力文字⟩ を実行する。

\def\bxjx@tmpdo#1\relax{%
  \def\bxjx@fake@grk##1##2{%
    \expandafter\bxjx@fake@grk@a\meaning##2#1\@nil{##1}{##2}%
  }
  \def\bxjx@fake@grk@a##1#1##2\@nil##3##4{%
    \ifx\##1\%
      \bxjx@cnta##4\divide\bxjx@cnta\@cclvi
      \multiply\bxjx@cnta\@cclvi \advance\bxjx@cnta`##3\relax
      \mathchar\bxjx@cnta
    \else ##3\fi}
  \expandafter\bxjx@tmpdo\string\mathchar\relax
}

■pdfTeX・upTeX の場合

\ifnum0\if p\bxjx@engine1\fi\if u\bxjx@engine1\fi>0

  • ⟨bxjx@KC/(符号値)⟩： その文字が「特定曖昧 CJK 文字」に該当する場合に定義済
  になる。

  まず inputenc を読み込んで入力エンコーディングを utf8 に変更する。
  ※「既定 UTF-8 化」後の \LaTeX においても、必ず「inputenc が明示的に読み込まれた」
  状態になる。

  \def\bxjx@tmpa{utf8}
  \ifx\bxjx@tmpa\inputencodingdoingname
    \PackageWarningNoLine\bxjx@pkgname{Input encoding changed to utf8}\
    \inputencoding{utf8}
  \fi

  upTeX の場合に、「特定曖昧 CJK 文字」を含むブロックの和文カテゴリコードを変更
  する。

  \if u\bxjx@engine
  \kcatcode"0370=15
  \kcatcode"0400=15
  \kcatcode"0500=15
  \fi

各文字について \DeclareUnicodeCharacter を実行する。

\def\bxjx@tmpdo#1{%
  \tempcnta="#1\relax
  \bxjx@fake@grk\{\bxjx@fake@grk\{出力文字\}\}⟨基準文字⟩： ラテン文字で代用される数式ギリシャ文字の出力を行う。(基準文字) (mathchardef の制御綴) の数式クラスと数式ファミリを引き継いで、(出力文字) (ASCII 文字トークン) の文字コードの数式文字を出力する。例えば、\Pi の意味が \mathchar"7005 である場合、\bxjx@fake@grk\{\Pi\} は \mathchar"7042 を実行する。

※フォントパッケージ使用時の再定義を考慮して、(基準文字) が mathchardef であるかを検査し、そうでない場合はフォールバックとして単に ⟨出力文字⟩ を実行する。

\def\bxjx@tmpdo#1\relax{%
  \def\bxjx@fake@grk##1##2{%
    \expandafter\bxjx@fake@grk@a\meaning##2#1\@nil{##1}{##2}%
  }
  \def\bxjx@fake@grk@a##1#1##2\@nil##3##4{%
    \ifx\##1\%
      \bxjx@cnta##4\divide\bxjx@cnta\@cclvi
      \multiply\bxjx@cnta\@cclvi \advance\bxjx@cnta`##3\relax
      \mathchar\bxjx@cnta
    \else ##3\fi}
  \expandafter\bxjx@tmpdo\string\mathchar\relax
}

■pdfTeX・upTeX の場合

\ifnum0\if p\bxjx@engine1\fi\if u\bxjx@engine1\fi>0

  • ⟨bxjx@KC/（符号値）⟩： その文字が「特定曖昧 CJK 文字」に該当する場合に定義済
  になる。

  まず inputenc を読み込んで入力エンコーディングを utf8 に変更する。
  ※「既定 UTF-8 化」後の \LaTeX においても、必ず「inputenc が明示的に読み込まれた」
  状態になる。

  \def\bxjx@tmpa{utf8}
  \ifx\bxjx@tmpa\inputencodingdoingname
    \PackageWarningNoLine\bxjx@pkgname{Input encoding changed to utf8}\
    \inputencoding{utf8}
  \fi

  upTeX の場合に、「特定曖昧 CJK 文字」を含むブロックの和文カテゴリコードを変更
  する。

  \if u\bxjx@engine
  \kcatcode"0370=15
  \kcatcode"0400=15
  \kcatcode"0500=15
  \fi

各文字について \DeclareUnicodeCharacter を実行する。

\def\bxjx@tmpdo#1{%
  \tempcnta="#1\relax
  \bxjx@fake@grk\{\bxjx@fake@grk\{出力文字\}\}
引数が数式 LICR である場合は \textbackslash bxjx\textbackslash fake\textbackslash grk で出力する。大文字なら \Pi、小文字なら \pi を基準文字にする。

それ以外は数式 LICR をそのまま実行する。

以上の処理を「特定 CJK 曖昧文字」の各々に適用する。

\textbackslash bxjx\textbackslash DeclareUnicodeCharacter を \textbackslash bxjx\textbackslash DeclareUnicodeCharacter を改変して、「特定 CJK 曖昧文字」の場合に再定義を抑止したもの。
5738 \let\bxjx@org@DeclareUnicodeCharacter\DeclareUnicodeCharacter
5739 \onlypreamble\bxjx@DeclareUnicodeCharacter
5740 \def\bxjx@DeclareUnicodeCharacter#1#2{%
5741 \count@=#1\relax
5742 \expandafter\ifx\csname bxjx@KC/\the\count@\endsname\relax
5743 \bxjx@org@DeclareUnicodeCharacter{#1}{#2}%
5744 \else
5745 \wlog{ \space\space skipped defining Unicode char U+#1}%
5746 \fi}
5747 \bxjx@ja@or@not\bxjx@ja@or@not{⟨和文用定義⟩}{⟨対象 fontenc⟩}{⟨LICR⟩}： \[no\]greekasCJK の状態
5748 に応じて和文または欧文で文字を出力する。
5749 \def\bxjx@ja@or@not#1#2#3{%
5750 \ifbxjx@gcc@cjk #1%
5751 \else\expandafter\ifx\csname T@#2\endcsname\relax #1%
5752 \else \UseTextSymbol{#2}{#3}%
5753 \fi\fi}
5754 \DeclareFontEncoding@
5755 \DeclareFontEncoding@ にパッチを当てて、\DeclareFontEncoding の実行中だけ変改
5756 後の \DeclareUnicodeCharacter が使われるようにする。
5757 \begingroup
5758 \toks@\expandafter{\DeclareFontEncoding@{#1}{#2}{#3}}
5759 \edef\next{\def\noexpand\DeclareFontEncoding@##1##2##3{%
5760 \noexpand\bxjx@swap@DUC@cmd
5761 \the\toks@
5762 \noexpand\bxjx@swap@DUC@cmd}}
5763 \endgroup\next
5764 \def\bxjx@swap@DUC@cmd{%
5765 \let\bxjx@tmpa\DeclareUnicodeCharacter
5766 \let\DeclareUnicodeCharacter\bxjx@tmpa
5767 \let\bxjx@tmpa\relax
5768 以上。
5769 ■XƎTEX・LuaTeX の場合
5770 \else\ifnum0\if x\bxjx@engine1\fi\if l\bxjx@engine1\fi>0
5771 各文字について、数式中の動作を定義する。
5772 \def\bxjx@tmpdo#1{%
5773 \bxjx@cnta="#1\relax
5774 \begingroup
5775 \lccode`~={\bxjx@cnta
5776 \lowercase{\endgroup
5777 197
数式 L1C1R が空なら何もしない。空でない場合、upLTEX の場合と同じ方法で "数式中の動作" を決定し、当該の文字を math active にしてその動作を設定する。

「Unicode な数式」の設定が行われているかを（簡易的に）検査して、そうでない場合にのみ、以上の処理を「特定 CJK 曖昧文字」の各々に適用する。

LuaTEX の場合は、LuaTEX-ja の jacharrange の設定を変更する。
※ "範囲 2" がギリシャ・キリル文字、"範囲 8" が Latin-1 の記号。
以上。

\section*{H.4 初期設定}

「特定 CJK 曖昧文字」を欧文扱いにする。

\section*{H.5 完了}

おしまい。

\section*{付録Ⅰ 補助パッケージ：bxjspandoc ☃️}

Pandoc の \LaTeX 用標準テンプレートをより幸せに使うための設定。BXJS クラスの
pandoc ドライバのコードの中の、“汎用的” に使える部分を切り出したもの。つまり現在の
pandoc ドライバはこのパッケージを読み込む。
※テンプレートの \TeX コードより前に読み込む必要があるため、専ら文書クラス内での読み
込みに限られる。

\subsection*{I.1 準備}

\begin{verbatim}
\def\bxjsp@pkgname{bxjspandoc}
\def\bxjsp@engine{n}
\let\bxjsp@engine=\empty
\AtBeginDocument{\bxjsp@begin@document@hook}
\if\bxjsp@babel@used 〔スイッチ〕 Babel が読み込まれたか。
\fi
\end{verbatim}
I.2 パッケージオプション

english オプションが指定されている場合、\ldots の調整を抑止する。
※つまり、「グローバルの english オプション」が指定されている場合も抑止の対象になる。
BXJS クラスの英語モードを想定しているが、それ以外の場合でも、一般的な \LaTeX の習慣として、グローバルの english は「その文書の基底言語が英語である」ことを示す。

I.3 パッケージ読込の阻止

\pandocSkipLoadFile は特定のファイルを（\@filewithoptions の処理に関して）読込済であるとマークする。
\pandocSkipLoadPackage の機能を用いてパッケージの読込を阻止する。

I.4 fixltx2e パッケージ

テンプレートでは fixltx2e パッケージを読み込むが、最近（2015 年版以降）の \LaTeX ではこれで警告が出る。これを抑止する。
\LaTeX カーネルが新しい場合は fixltx2e を読込済にする。
I.5  cmap パッケージ

エンジンが (u)pLATEX のときに cmap パッケージが読み込まれるのを阻止する。（実際は警告が出るだけで無害であるが。）
5844 \ifj\bxjsp@engine
5845 \pandocSkipLoadPackage{cmap}
5846 \fi

I.6 microtype パッケージ

警告が多すぎなので消す。
5847 \ifj\bxjsp@engine \else
5848 \PassOptionsToPackage{verbose=silent}{microtype}
5849 \fi

エンジンが (u)pLATEX のときに microtype パッケージが読み込まれるのを阻止し、さらにテンプレートで使われている命令を通すためにダミーの定義を行う。※昔は standard ドライバでこの処理を行っていたが、元来は Pandoc 用の処理なので、1.5版で pandoc に移動。
5850 \ifj\bxjsp@engine
5851 \pandocSkipLoadPackage{microtype}
5852 \newcommand*\UseMicrotypeSet[2]{1}
5853 \fi

I.7 Unicode 文字変換対策

Pandoc で LATEX 形式に書き出す場合は、元データ中の一部の Unicode 文字を「LATEX の表記」に置き換える。その中には日本語文書で問題になるものが含まれる。

...→\ldots
'→`
"→``
``→''

日本語 LATEX では「LATEX の表記」は欧文扱い、Unicode 文字は和文扱いとして使い分ける習慣があるので、このような置換が行われるのは好ましくない。
これららの置換のうち、後の 4つは Pandoc の --no-tex-ligatures オプションを指定すれば抑止できるが、「…」の置換を抑止する機能はないようである。そこで、「\ldots を「…」に戻す」という処置を行う。
\pandocLdots Pandoc 用の \ldots の実装。非数式である場合は代わりに … を実行する。
※以前は「Pandoc が必ず \ldots の形で書き出す」ことを利用して後続に {} があるかで「元が … であるか」を判断していた。ところが、Pandoc 2.7 版で {} を必ずしも付けなくなったため、1.9f 版で非数式の \ldots を全て … に戻す動作に変更した。
5854 \DeclareRobustCommand{\pandocLdots}{%
5855 \let\bxjsp@do\bxjsp@ja@ellipsis
5856 \ifmmode \let\bxjsp@do\bxjsp@org@ldots
5857 \fi

201
ldots の実装を \pandocLdots に置き換える。

もしここで newcommand\pandocLdots{ldots} という定義である場合は置き換えない。

english オプションが指定されていてかつ Babel が読み込まれていない場合も置き換えない。

ldots の直後の文字が非英字の場合、Pandoc は「…」のように空白を入れずに並べて出力する。 「Pandoc は非英字と見なすが XƎTEX・LuaTEX は英字と見なす（または将来その可能性がある）」 文字で、特に日本語文書に現れるものについて、非英字扱いにしておく。

※ Pandoc は「Unicode 7.0 で GC が Letter」な文字を英字と判定している。

I.8 PandoLa モジュール

インストール済であれば読み込む。

I.8 PandoLa モジュール
1.9 完了

おしまい。

補助パッケージ実装はここまで。