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”

v2.9e　[2026/03/29]

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　14

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　61

5.1　ページレイアウト 62

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　78

8　文書のマークアップ　81

8.1　表題 81
8.2　章・節 86
8.3　リスト環境 99
8.4　パラメータの設定 106
8.5　フロート 107
8.6　キャプション 109

9　フォントコマンド　110

10 相互参照 113
10.1 目次の類 113
10.2 参考文献 118
10.3 索引 120
10.4 脚注 121

11 段落の頭へのグルー挿入禁止 124

12 いろいろなロゴ 128

13 amsmath との衝突の回避 128

14 初期設定 129

15 実験的コード 133

16 BXJS 独自の追加処理 134

付録 A 和文ドライバの仕様 139

付録 B 和文ドライバ：minimal 140
B.1 準備 140
B.2 (u)pTEX 用の設定 143
B.3 pdfTEX 用の処理 148
B.4 XeTEX 用の処理 148
B.5 後処理（エンジン共通） 149

付録 C 和文ドライバ：standard 152
C.1 準備 152
C.2 和文ドライバパラメタ 153
C.3 共通処理 (1) 153
C.4 pTEX 用設定 160
C.5 pdfTEX 用設定：CJK + bxcjkjatype 164
C.6 XeTEX 用設定：xeCJK + zxjatype 166
C.7 LuaTEX 用設定：LuaTEX-ja 169
C.8 共通処理 (2) 173

付録 D 和文ドライバ：modern 173
D.1 フォント設定 173
D.2 fixltx2e 読込 174
D.3 和文カテゴリコード 174
D.4 完了 174

付録 E 和文ドライバ：pandoc 174

E.1 準備 . . . 175
E.2 和文ドライバパラメタ . . . 176
E.3 dupload システム . . . 177
E.4 lang 変数 . . . 178
E.5 geometry 変数 . . . 182
E.6 CJKmainfont 変数 . . . 182
E.7 Option clash 対策 . . . 182
E.8 レイアウト上書き禁止 . . . 182
E.9 paragraph のマーク . . . 183
E.10 全角空白文字 . . . 184
E.11 hyperref 対策 . . . 184
E.12 Pandoc 要素に対する和文用の補正 . . . 185
E.13 ifPDFTeX スイッチ . . . 186
E.14 完了 . . . 187

付録 F 補助パッケージ一覧 **187**

付録 G 補助パッケージ：bxjscompat **187**
G.1 準備 . . . 188
G.2 8bit 欧文 TEX . . . 188
G.3 XHTEX . . . 188
G.4 LuaTEX . . . 190
G.5 完了 . . . 191

付録 H 補助パッケージ：bxjscjkcat **191**
H.1 準備 . . . 191
H.2 和文カテゴリコードの設定 . . . 192
H.3 ギリシャ・キリル文字の扱い . . . 193
H.4 初期設定 . . . 200
H.5 完了 . . . 200

付録 I 補助パッケージ：bxjspandoc **200**
I.1 準備 . . . 201
I.2 パッケージオプション . . . 201
I.3 パッケージ読込の阻止 . . . 201
I.4 fixltx2e パッケージ . . . 202
I.5 cmap パッケージ . . . 202
I.6 microtype パッケージ . . . 202
I.7 Unicode 文字変換対策 . . . 203
I.8 PandoLa モジュール . . . 204
I.9 完了 . . . 204

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 2ε 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。

⟨article⟩	`bxjsarticle.cls`	短いレポート（章なし）のクラス
⟨report⟩	`bxjsreport.cls`	長いレポート（章あり）のクラス
⟨book⟩	`bxjsbook.cls`	書籍用のクラス
⟨slide⟩	`bxjsslide.cls`	スライド用のクラス
⟨minimal⟩	`bxjsja-minimal.def`	minimal 和文ドライバ
⟨standard⟩	`bxjsja-standard.def`	standard 和文ドライバ
⟨modern⟩	`bxjsja-modern.def`	modern 和文ドライバ（未公開）
⟨pandoc⟩	`bxjsja-pandoc.def`	pandoc 和文ドライバ
⟨compat⟩	`bxjscompat.sty`	古いやつをどうにかする補助パッケージ
⟨cjkcat⟩	`bxjscjkcat.sty`	modern ドライバ用の補助パッケージ
⟨ancpandoc⟩	`bxjspandoc.sty`	Pandoc 用の補助パッケージ

※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 LATEX3 Project の `classes.dtx` と株式会社アスキーの `jclasses.dtx` に基づいてもと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近は 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 upLATEX 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。

⟨article⟩	jsarticle.cls	論文・レポート用
⟨book⟩	jsbook.cls	書籍用
⟨report⟩	jsreport.cls	レポート用
⟨jspf⟩	jspf.cls	某学会誌用
⟨kiyou⟩	kiyou.cls	某紀要用

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。

```
%<*minijs>
%% if jsclasses loaded, abort loading this package
\ifx\@jsc@uplatextrue\@undefined\else
  \PackageInfo{minijs}{jsclasses does not need minijs, exiting}
  \expandafter\endinput
\fi
%% "fake" jsarticle
\expandafter\def\csname ver@jsarticle.cls\endcsname{}
%</minijs>
```

\bxjs@clsname 文書クラスの名前です。エラーメッセージ表示などで使われます。

```
%<*class>
%% このファイルは日本語文字を含みます.
%<article>\def\bxjs@clsname{bxjsarticle}
%<book>\def\bxjs@clsname{bxjsbook}
%<report>\def\bxjs@clsname{bxjsreport}
%<slide>\def\bxjs@clsname{bxjsslide}
```

\ifjsc@needsp@tch [2016-08-22] 従来 jsclasses では，pLaTeX や LaTeX の不都合な点に対して，クラスファイル内で独自に対策を施していました。しかし，2016 年以降，コミュニティ版 pLaTeX が次第に対策コードをカーネル内に取り込むようになりました。そこで，新しい pLaTeX カーネルと衝突しないように，日付が古い場合だけパッチをあてる場合があります。この処理に使用するフラグを定義します。

```
%</class>
%<*class|minijs>
\newif\ifjsc@needsp@tch
\jsc@needsp@tchfalse
%</class|minijs>
%<*class>
```

■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。

```
\let\jsArticle=a
\let\jsBook=b
\let\jsReport=r
\let\jsSlide=s
%<article>\let\jsDocClass\jsArticle
%<book>\let\jsDocClass\jsBook
%<report>\let\jsDocClass\jsReport
%<slide>\let\jsDocClass\jsSlide
```

\bxjs@test@engine `\bxjs@test@engine\制御綴{⟨コード⟩}`： **\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}
```

\jsEngine 〔暗黙文字トークン〕エンジン（TEX 処理系）の種別： `j` = pTEX 系、`x` = XETEX、`p` = pdfTEX、`l` = LuaTEX、`J` = NTT JTEX、`O` = Omega 系、`n` =以上の何れでもない。
※ pdfTEX と LuaTEX については DVI モードの場合も含む。

```
\let\jsEngine=n
\bxjs@test@engine\kanjiskip{\let\jsEngine=j}
\bxjs@test@engine\jintercharskip{\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。

- XETEX： 0.997 版（2007 年）以上

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。
3.0 版での**クラス本体の**処理系バージョン要件は以下の通りである。

- TEX： 3.0 版 [1990/03] 以上
- pTEX： 2.0 版 [1995/03] 以上
- upTEX： 0.10 版 [2007/07] 以上
- pdfTEX： 1.40 版 [2007/01] 以上
- LuaTEX： 0.60 版 [2010/04] 以上
- XETEX： 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敢えて検査しないことにする。
※クラス本体での要件は敢えて緩くしている。標準和文ドライバ（minimal も含む）についてまた別に要件を定めるので、実質的にはそち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。

```
\@tempswatrue
```

```
\if x\jsEngine
  \ifdim\the\XeTeXversion\XeTeXrevision\p@<0.997\p@
    \@tempswafalse \fi
\fi
```

非サポートのバージョン場合は強制終了させる。

```
\if@tempswa \expandafter\@gobble
\else
  \ClassError\bxjs@clsname
   {The engine in use is all too old}
   {It's a fatal error. I'll quit right now.}
  \expandafter\@firstofone
\fi{\endinput\@@end}
```

万が一「2.09 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。

```
\if@compatibility
  \ClassError\bxjs@clsname
   {Something went chaotic!\MessageBreak
    (How come '\string\documentstyle' is there?)\MessageBreak
    I cannot go a single step further...}
   {If the chant of '\string\documentstyle' was just a blunder of yours,\MessageBreak
    then there'll still be hope....}
  \expandafter\@firstofone
\else \expandafter\@gobble
\fi{\typeout{Farewell!}\endinput\@@end}
```

`\bxjs@if@format@at@least` `\bxjs@if@format@at@least{⟨日付⟩}{⟨真⟩}{⟨偽⟩}`： LaTeX カーネルの版が指定の日付以降であるか。

```
\def\bxjs@if@format@at@least{\@ifl@t@r\fmtversion}
```

`\bxjs@if@package@at@least` `\bxjs@if@package@at@least{⟨名前⟩}{⟨日付⟩}{⟨真⟩}{⟨偽⟩}`： その名前のパッケージの指定の日付以降の版が読み込まれているか。そもそも読み込まれていない場合は偽になる。
※ 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。

```
\bxjs@if@format@at@least{2017/04/15}{%
  \let\bxjs@if@package@at@least\@ifpackagelater
}{%else
  \def\bxjs@if@package@at@least#1#2{%
    \@ifpackageloaded{#1}{\@ifpackagelater{#1}{#2}}{\@secondoftwo}}}
```

`\ifjsWithupTeX` 〔スイッチ〕エンジンが「内部漢字コードが Unicode の upTeX」であるか。
※つまり、`\jsEngine` = `j` である場合、このスイッチが真なら upLaTeX、偽なら pLaTeX である。2023 年 6 月に pLaTeX の TeX 処理系が「ε-pTeX」から「内部漢字コードが非 Unicode の ε-upTeX」に変わったが、これによる影響はない。

```
\newif\ifjsWithupTeX
\ifx\ucs\@undefined\else \ifnum\ucs"3000="3000
  \jsWithupTeXtrue
```

```
\fi\fi
\let\if@jsc@uplatex\ifjsWithupTeX
```

`\ifjsWithpTeXng`〔スイッチ〕エンジンが pTEX-ng であるか。

```
\newif\ifjsWithpTeXng
\bxjs@test@engine\ngbanner{\jsWithpTeXngtrue}
```

`\ifjsWitheTeX`〔スイッチ〕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。

```
\newif\ifjsWitheTeX
\bxjs@test@engine\eTeXversion{\jsWitheTeXtrue}
```

`\ifjsInPdfMode`〔スイッチ〕pdfTEX・LuaTEX が PDF モードで動作しているか。

```
\newif\ifjsInPdfMode
\@nameuse{jsInPdfMode\ifnum0%
  \ifx\pdfoutput\@undefined\else\the\pdfoutput\fi
  \ifx\outputmode\@undefined\else\the\outputmode\fi
  >0 true\else false\fi}
```

`\ifbxjs@explIII`〔スイッチ〕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。

```
\newif\ifbxjs@explIII
\bxjs@if@format@at@least{2020/02/02}{\bxjs@explIIItrue}{}
```

`\ifbxjs@brace@safe`〔スイッチ〕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。

※`\@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。

```
\newif\ifbxjs@brace@safe
\bxjs@if@format@at@least{2021/06/01}{\bxjs@brace@safetrue}{}
```

`\ifbxjs@TUenc`〔スイッチ〕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。

```
\newif\ifbxjs@TUenc
\def\bxjs@tmpa{TU}\edef\bxjs@tmpb{\f@encoding}
\ifx\bxjs@tmpa\bxjs@tmpb
  \bxjs@TUenctrue
\fi
```

`\ifbxjs@old@hook@system`〔スイッチ〕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}
```

\bxjs@nclong *無しの \newcommand で定義した引数無しマクロが \long 付になるならば \long、\long 無になるなら \relax と等価になる。
※カーネルの 2025/11/01 版において当該のマクロ定義が \long 無になるように仕様変更された。

```
\newcommand\bxjs@nclong{}\long\def\bxjs@tmpa{}
\ifx\bxjs@nclong\bxjs@tmpa \let\bxjs@nclong\long
\else \let\bxjs@nclong\relax
\fi
```

■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 LaTeX 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\g@addto@macro\csname\bxjs@clsname.cls-h@@k\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}%
```

```
}{}
```

■BXJS クラス特有の設定

LuaTeX の場合、本クラス用の Lua モジュールを用意する。

```
\ifx l\jsEngine
  \directlua{ bxjs = {} }
\fi
```

`\bxjs@protected` ε-TeX 拡張が有効な場合にのみ `\protected` の効果をもつ。

```
\ifjsWitheTeX \let\bxjs@protected\protected
\else \let\bxjs@protected\@empty
\fi
```

`\bxjs@robust@def` 無引数の頑強な命令を定義する。

```
\ifjsWitheTeX
  \def\bxjs@robust@def{\protected\def}
\else
  \def\bxjs@robust@def{\DeclareRobustCommand*}
\fi
```

`\bxjs@CGHN` 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。

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

`\bxjs@cond` `\bxjs@cond\ifXXX……\fi{⟨真⟩}{⟨偽⟩}`

TeX の if-文（`\ifXXX……⟨真⟩\else⟨偽⟩\fi`）を末尾呼出形式に変換するためのマクロ。

```
\@gobbletwo\if\if \def\bxjs@cond#1\fi{%
  #1\expandafter\@firstoftwo
  \else\expandafter\@secondoftwo
  \fi}
```

TODO:2.9 `\bxjs@expanded` を定義する。

`\bxjs@cslet` `\bxjs@cslet{⟨名前 1⟩}`**\制御綴**：

```
\def\bxjs@cslet#1{%
  \expandafter\let\csname#1\endcsname}
```

`\bxjs@csletcs` `\bxjs@csletcs{⟨名前 1⟩}{⟨名前 2⟩}`：

```
\def\bxjs@csletcs#1#2{%
  \expandafter\let\csname#1\expandafter\endcsname\csname#2\endcsname}
```

\bxjs@catopt `\bxjs@catopt{⟨文字列 1⟩}{⟨文字列 2⟩}`： 2 つの文字列を , で繋いだ文字列。ただし少なくとも一方が空の場合は , を入れない。完全展開可能。

```
\def\bxjs@catopt#1#2{%
  #1\if\relax#1\relax\else\if\relax#2\relax\else,\fi\fi#2}
```

\bxjs@ifplus `\@ifstar` の + 版。

```
\def\bxjs@ifplus#1{\@ifnextchar+{\@firstoftwo{#1}}}
```

\bxjs@trim `\bxjs@trim\CS` で、`\CS` の内容のトークン列を先頭と末尾の空白トークン群を除去したものに置き換える。

```
\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: \@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: \@nil\@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\@ne}
  \bxjs@cslet{#1/\the\@tempcnta}\relax}
```

\bxjs@gset@tempcnta calc の整数式を用いて `\@tempcnta` の値を設定する。

```
\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}
```

\bxjs@new@count ε-TeX 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。

\bxjs@advance@count

```
\ifjsWitheTeX
  \let\bxjs@new@count\newcount
  \def\bxjs@advance@count#1#2{\advance#1by#2\relax}
\else
  \def\bxjs@new@count#1{\chardef#1\z@}
  \let\bxjs@advance@count\bxjs@advance@qc
\fi
```

\bxjs@add@class@option グローバルオプションを追加する。

※ 2021/06/01 版以降の LaTeX カーネルでは従来の \@classoptionslist に加えて「無加工のクラスオプションリスト」を格納するための \@raw@classoptionslist が用意されているので、その場合はそのリストにも追加する。

```
\def\bxjs@add@class@option#1{%
  \edef\bxjs@args{{#1}}%
  \expandafter\bxjs@add@class@option@a\bxjs@args}
\def\bxjs@add@class@option@a#1{%
  \g@addto@macro\@classoptionslist{,#1}%
  \ifx\@raw@classoptionslist\@undefined\else
    \g@addto@macro\@raw@classoptionslist{,#1}%
  \fi}
```

\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}%
      \xdef\bxjs@g@tmpa{\the\@tempdima}%
    \else \global\let\bxjs@g@tmpa\bxjs@tmpb
    \fi
  \endgroup
  #1=\bxjs@g@tmpa\relax}
```

\bxjs@parse@qh #1 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)pLaTeX 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。

```
\if j\jsEngine \def\bxjs@parse@qh@units{zw,zh}
\else \def\bxjs@parse@qh@units{trueQ,trueH,Q,H,zw,zh}
\fi
\def\bxjs@parse@qh#1{%
  \let\bxjs@tmpb\relax
  \@for\bxjs@tmpa:=\bxjs@parse@qh@units\do{%
    \ifx\bxjs@tmpb\relax
      \edef\bxjs@next{{\bxjs@tmpa}{#1}}%
      \expandafter\bxjs@parse@qh@a\csname bxjs@unit@\bxjs@tmpa\expandafter
          \endcsname\bxjs@next
    \fi}}
\def\bxjs@parse@qh@a#1#2#3{%
  \def\bxjs@next##1#2\@nil##2\@nnil{\bxjs@parse@qh@b{##1}{##2}#1}%
  \bxjs@next#3\@nil#2\@nil\@nnil}
\def\bxjs@parse@qh@b#1#2#3{%
```

```
  \ifx\@nnil#2\@nnil\else
    \ifx#3\relax
      \ClassError\bxjs@clsname
       {You cannot use '\bxjs@tmpa' here}{\@ehc}%
      \def\bxjs@tmpb{0pt}%
    \else
      \@tempdimb#3\relax \@tempdimb#1\@tempdimb
      \edef\bxjs@tmpb{\the\@tempdimb}%
    \fi
  \fi}
```

今の段階では Q/H だけが使用可能。

```
  \def\bxjs@unit@Q{0.25mm}\let\bxjs@unit@H\bxjs@unit@Q
```

`\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。

```
\@onlypreamble\bxjs@begin@document@hook
\def\bxjs@begin@document@hook{\bxjs@after@preambletrue}
\AtBeginDocument{\bxjs@begin@document@hook}
```

`\bxjs@post@option@hook` `\ProcessOptions` 直後に実行されるフック。

```
\@onlypreamble\bxjs@post@option@hook
\let\bxjs@post@option@hook\@empty
```

`\bxjs@pre@jadriver@hook` 和文ドライバ読込直前に実行されるフック。

```
\@onlypreamble\bxjs@pre@jadriver@hook
\let\bxjs@pre@jadriver@hook\@empty
```

`\bxjs@endpreamble@hook` 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。

```
\@onlypreamble\bxjs@endpreamble@hook
\let\bxjs@endpreamble@hook\@empty
\AtEndOfClass{%
  \ifx\AtEndPreamble\@undefined\else
    \AtEndPreamble{\bxjs@endpreamble@hook}%
  \fi}
```

一時的な手続き用の制御綴。

```
\@onlypreamble\bxjs@tmpdo
\@onlypreamble\bxjs@tmpdo@a
\@onlypreamble\bxjs@tmpdo@b
\@onlypreamble\bxjs@tmpdo@c
\@onlypreamble\bxjs@tmpdo@d
```

`\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。

```
\bxjs@robust@def\jsInhibitGlue{%
  \ifx\inhibitglue\@undefined\else \inhibitglue \fi}
```

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。

\if@restonecol 段組のときに真になる論理変数です。

```
234 \newif\if@restonecol
```

\if@titlepage 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。

```
235 \newif\if@titlepage
```

\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。

```
236 %<book|report>\newif\if@openright
```

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

```
237 %<book|report>\newif\if@openleft
```

\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。

```
238 %<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。

```
239 \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 $1\,\mathrm{m}^2$，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 $1.5\,\mathrm{m}^2$ 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 $1000\,\mathrm{mm} \times 1414\,\mathrm{mm}$ です。このため，LaTeX 2_ε の b5paper は $250\,\mathrm{mm} \times 176\,\mathrm{mm}$ ですが，pLaTeX 2_ε の b5paper は $257\,\mathrm{mm} \times 182\,\mathrm{mm}$ になっています。ここでは pLaTeX 2_ε にならって JIS 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。

```
\@onlypreamble\bxjs@setpaper
\def\bxjs@setpaper#1{\def\bxjs@param@paper{#1}}
\newif\ifbxjs@iso@bsize
\DeclareOption{iso-bsize}{\bxjs@iso@bsizetrue}
\@onlypreamble\bxjs@setpaper@bsize
\def\bxjs@setpaper@bsize#1{\edef\bxjs@param@paper{%
  b#1\ifbxjs@iso@bsize paper\else j\fi}}
\DeclareOption{a3paper}{\bxjs@setpaper{a3paper}}
\DeclareOption{a4paper}{\bxjs@setpaper{a4paper}}
\DeclareOption{a5paper}{\bxjs@setpaper{a5paper}}
\DeclareOption{a6paper}{\bxjs@setpaper{a6paper}}
\DeclareOption{b4paper}{\bxjs@setpaper@bsize{4}}
\DeclareOption{b5paper}{\bxjs@setpaper@bsize{5}}
\DeclareOption{b6paper}{\bxjs@setpaper@bsize{6}}
\DeclareOption{a4j}{\bxjs@setpaper{a4paper}}
\DeclareOption{a5j}{\bxjs@setpaper{a5paper}}
\DeclareOption{b4j}{\bxjs@setpaper{b4j}}
\DeclareOption{b5j}{\bxjs@setpaper{b5j}}
\DeclareOption{a4var}{\bxjs@setpaper{{210truemm}{283truemm}}}
\DeclareOption{b5var}{\bxjs@setpaper{{182truemm}{230truemm}}}
\DeclareOption{letterpaper}{\bxjs@setpaper{letterpaper}}
\DeclareOption{legalpaper}{\bxjs@setpaper{legalpaper}}
\DeclareOption{executivepaper}{\bxjs@setpaper{executivepaper}}
```

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{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 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 と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{screen}}
```

```
\DeclareOption{custompaper}{}
```

■横置き　用紙の縦と横の長さを入れ換えます。

```
\newif\if@landscape
\@landscapefalse
\DeclareOption{landscape}{\@landscapetrue}
```

■slide　オプション slide を新設しました。

[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
```

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、ユニー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。

```
%<!slide>\@slidefalse
%<slide>\@slidetrue
```

■サイズオプション　10pt，11pt，12pt のほかに，8pt，9pt，14pt，17pt，21pt，25pt，30pt，36pt，43pt を追加しました。これは等比数列になるように選んだものです（従来の 20pt も残しました）。\@ptsize の定義が変だ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 へ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}
```

\bxjs@setbasefontsize 基底フォントサイズを実際に変更する。

```
\def\bxjs@setbasefontsize#1{%
```

Q 単位の長さ指定をサポートするため \jsSetQHLength を使う。

※クラスオプションのトークン列の中に展開可能なトークンがある場合、LaTeX はクラスファイルの読込の前にそれを展開しようとする。このため、この位置で \jQ をサポートすることは原理的に不可能である。

```
  \jsSetQHLength\@tempdima{#1}%
  \edef\bxjs@param@basefontsize{\the\@tempdima}%
  \ifdim\@tempdima=10pt          \renewcommand\@ptsize{0}%
  \else\ifdim\@tempdima=10.95pt \renewcommand\@ptsize{1}%
  \else\ifdim\@tempdima=12pt     \renewcommand\@ptsize{2}%
  \else \bxjs@nclong\edef\@ptsize{\jsUnusualPtSize}\fi\fi\fi}
```

TODO: 恐らく 14pt と base=14.4pt 等の関係も全く等価であるべき。

```
\def\bxjs@setjbasefontsize#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{8pt}}
\DeclareOption{9pt}{\bxjs@setbasefontsize{9pt}}
\DeclareOption{10pt}{\bxjs@setbasefontsize{10pt}}
\DeclareOption{11pt}{\bxjs@setbasefontsize{10.95pt}}
\DeclareOption{12pt}{\bxjs@setbasefontsize{12pt}}
\DeclareOption{14pt}{\bxjs@setbasefontsize{14.4pt}}
\DeclareOption{17pt}{\bxjs@setbasefontsize{17.28pt}}
\DeclareOption{20pt}{\bxjs@setbasefontsize{20pt}}
\DeclareOption{21pt}{\bxjs@setbasefontsize{20.74pt}}
\DeclareOption{25pt}{\bxjs@setbasefontsize{24.88pt}}
\DeclareOption{30pt}{\bxjs@setbasefontsize{29.86pt}}
\DeclareOption{36pt}{\bxjs@setbasefontsize{35.83pt}}
\DeclareOption{43pt}{\bxjs@setbasefontsize{43pt}}
\DeclareOption{12Q}{\bxjs@setjbasefontsize{3mm}}
\DeclareOption{14Q}{\bxjs@setjbasefontsize{3.5mm}}
\DeclareOption{10ptj}{\bxjs@setjbasefontsize{10pt}}
\DeclareOption{10.5ptj}{\bxjs@setjbasefontsize{10.5pt}}
\DeclareOption{11ptj}{\bxjs@setjbasefontsize{11pt}}
\DeclareOption{12ptj}{\bxjs@setjbasefontsize{12pt}}
```

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 2_ε 本体で行います（plcore.dtx 参照）。オプション tombow で日付付きのトンボ，オプション

tombo で日付なしのトンボを出力します。これらはアスキー版のままです。カウンタ \hour, \minute は pLaTeX 2ε 本体で宣言されています。

取りあえず、pTeX 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}
\fi
```

■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も表現したいので，jsclasses では新たに openleft も追加しました。

```
%<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] LaTeX 2_ε 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。

[2025-05-26] LaTeX 2_ε 2025-06-01 に追従し，最初の$$を\dollardollar@begin に変更しました. \providecommand で\dollardollar@begin を定義しているので, 古い LaTeX 2_ε カーネルでも問題ありません.

```
\providecommand\dollardollar@begin{$$}
\def\eqnarray{%
   \stepcounter{equation}%
   \def\@currentlabel{\p@equation\theequation}%
   \def\@currentcounter{equation}%
   \global\@eqnswtrue
   \m@th
   \global\@eqcnt\z@
   \tabskip\@centering
   \let\\\@eqncr
   \dollardollar@begin\everycr{}\halign to\displaywidth\bgroup
       \hskip\@centering$\displaystyle\tabskip\z@skip{##}$\@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}
```

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。

[2022-09-13] LaTeX 2_ε 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。

```
\DeclareOption{leqno}{\input{leqno.clo}}
```

```
\DeclareOption{fleqn}{\input{fleqn.clo}%
% fleqn 用の eqnarray 環境の再定義
  \def\eqnarray{%
    \stepcounter{equation}%
    \def\@currentlabel{\p@equation\theequation}%
    \def\@currentcounter{equation}%
    \global\@eqnswtrue\m@th
    \global\@eqcnt\z@
    \tabskip\mathindent
    \let\\=\@eqncr
    \setlength\abovedisplayskip{\topsep}%
    \ifvmode
      \addtolength\abovedisplayskip{\partopsep}%
    \fi
    \addtolength\abovedisplayskip{\parskip}%
    \setlength\belowdisplayskip{\abovedisplayskip}%
    \setlength\belowdisplayshortskip{\abovedisplayskip}%
    \setlength\abovedisplayshortskip{\abovedisplayskip}%
    \dollardollar@begin\everycr{}\halign to\linewidth% $$
    \bgroup
      \hskip\@centering$\displaystyle\tabskip\z@skip{##}$\@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
    }}
```

■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。これは使われることはないのでコメントアウトしてあります。

```
% \DeclareOption{openbib}{%
%   \AtEndOfPackage{%
%    \renewcommand\@openbib@code{%
%       \advance\leftmargin\bibindent
%       \itemindent -\bibindent
%       \listparindent \itemindent
%       \parsep \z@}%
%    \renewcommand\newblock{\par}}}
```

■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。disablejfam 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 を使えば数式中で和文フォントが使えますので，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@enablejfam〔暗黙文字トークン〕enablejfam オプションの状態：

```
%\let\bxjs@enablejfam\@undefined
```

enablejfam オプションの処理。

```
\def\bxjs@kv@enablejfam@true{\let\bxjs@enablejfam=t}
\def\bxjs@kv@enablejfam@false{\let\bxjs@enablejfam=f}
\def\bxjs@kv@enablejfam@default{\let\bxjs@enablejfam\@undefined}
\define@key{bxjs}{enablejfam}[true]{%
  \bxjs@set@keyval{enablejfam}{#1}{}}
```

JS クラスとの互換のため disablejfam オプションを定義する。

```
\DeclareOption{disablejfam}{\let\bxjs@enablejfam=f}
```

※実際に何らかの設定を行うのは和文ドライバである。和文ドライバとエンジンの組合せにより、enablejfam が default である場合に「数式和文ファミリ」が有効と無効の選択は異なるし、またそもそも有効と無効の一方しか選択できない場合もある。

■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。

\ifjsDraft〔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 が廃止されたので、BXJS クラスでも \ifdraft を 2.0 版で廃止した。

```
\newif\ifjsDraft
\DeclareOption{draft}{\jsDrafttrue \overfullrule=5pt }
\DeclareOption{final}{\jsDraftfalse \overfullrule=0pt }
```

■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 などを使いたいときは 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，同等な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できる。

※ JS クラスでは \ifpapersize という制御綴だが、これは採用しない。

```
\newif\ifbxjs@papersize
\bxjs@papersizetrue
\DeclareOption{nopapersize}{\bxjs@papersizefalse}
\DeclareOption{papersize}{\bxjs@papersizetrue}
```

■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。

```
\newif\if@english
\@englishfalse
\DeclareOption{english}{\@englishtrue}
```

■jsbook を jsreport もどきに　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。

```
\newif\if@jslogo \@jslogofalse
\DeclareOption{jslogo}{\@jslogotrue}
\DeclareOption{nojslogo}{\@jslogofalse}
```

■複合設定オプション

TODO:3.x　\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。）

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

※局所化の \begingroup〜\endgroup について、以前は \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 128 pt 以上の場合でも動作するように修正した。

```
\mathchardef\bxjs@isc@ll=128
\mathchardef\bxjs@isc@sl=259
\def\bxjs@isc@sl@h{65539 }
\def\bxjs@invscale#1#2{%
  \begingroup \@tempdima=#1\relax \@tempdimb#2\p@\relax
    \ifdim\@tempdima<\bxjs@isc@ll\p@
      \@tempcnta\@tempdima \multiply\@tempcnta\@cclvi
      \divide\@tempcnta\@tempdimb \multiply\@tempcnta\@cclvi
    \else
      \@tempcnta\@tempdima \divide\@tempcnta\@tempdimb
      \multiply\@tempcnta\p@ \let\bxjs@isc@sl\bxjs@isc@sl@h
    \fi
    \@tempcntb\p@ \divide\@tempcntb\@tempdimb
    \advance\@tempcnta-\@tempcntb \advance\@tempcnta-\tw@
    \@tempdimb\@tempcnta\@ne
    \advance\@tempcnta\@tempcntb \advance\@tempcnta\@tempcntb
    \advance\@tempcnta\bxjs@isc@sl \@tempdimc\@tempcnta\@ne
    \@whiledim\@tempdimb<\@tempdimc\do{%
      \@tempcntb\@tempdimb \advance\@tempcntb\@tempdimc
      \advance\@tempcntb\@ne \divide\@tempcntb\tw@
      \ifdim #2\@tempcntb>\@tempdima
        \advance\@tempcntb\m@ne \@tempdimc=\@tempcntb\@ne
      \else \@tempdimb=\@tempcntb\@ne \fi}%
    \xdef\bxjs@gtmpa{\the\@tempdimb}%
  \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 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。

```
\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}
```

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。

```
\DeclareOption{pandoc+}{%
  \g@addto@macro\bxjs@post@option@hook{%
    \edef\jsJaParam{\bxjs@catopt{_plus}\jsJaParam}}%
  \ExecuteOptions{pandoc}}
```

■エンジン・ドライバオプション ⛄

\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。

```
%\let\bxjs@engine@given\@undefined
```

\bxjs@engine@opt 明示されたエンジンのオプション名。

```
%\let\bxjs@engine@opt\@undefined
```

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 ...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」とみなすべきだから。

```
\DeclareOption{autodetect-engine}{%
  \let\bxjs@engine@given=*}
\DeclareOption{latex}{%
  \def\bxjs@engine@opt{latex}%
  \let\bxjs@engine@given=n}
\DeclareOption{platex}{%
  \def\bxjs@engine@opt{platex}%
  \let\bxjs@engine@given=j}
\DeclareOption{uplatex}{%
  \def\bxjs@engine@opt{uplatex}%
  \let\bxjs@engine@given=u}
\DeclareOption{xelatex}{%
  \def\bxjs@engine@opt{xelatex}%
  \let\bxjs@engine@given=x}
\DeclareOption{pdflatex}{%
  \def\bxjs@engine@opt{pdflatex}%
  \let\bxjs@engine@given=p}
\DeclareOption{lualatex}{%
  \def\bxjs@engine@opt{lualatex}%
  \let\bxjs@engine@given=l}
\DeclareOption{platex-ng}{%
  \def\bxjs@engine@opt{platex-ng}%
  \let\bxjs@engine@given=g}
\DeclareOption{platex-ng*}{%
```

```
  \def\bxjs@engine@opt{platex-ng*}%
  \let\bxjs@platexng@nodrv=t%
  \let\bxjs@engine@given=g}
```

`\bxjs@driver@given` 〔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。

```
%\let\bxjs@driver@given\@undefined
\let\bxjs@driver@@dvimode=0
\let\bxjs@driver@@dvipdfmx=1
\let\bxjs@driver@@pdfmode=2
\let\bxjs@driver@@xetex=3
\let\bxjs@driver@@dvips=4
\let\bxjs@driver@@none=5
```

`\bxjs@driver@opt` 明示された「ドライバ指定」のオプション名。

```
%\let\bxjs@driver@opt\@undefined
```

※ `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。

```
\DeclareOption{dvips}{%
  \def\bxjs@driver@opt{dvips}%
  \let\bxjs@driver@given\bxjs@driver@@dvips}
\DeclareOption{dviout}{%
  \def\bxjs@driver@opt{dviout}%
  \let\bxjs@driver@given\bxjs@driver@@dvimode}
\DeclareOption{xdvi}{%
  \def\bxjs@driver@opt{xdvi}%
  \let\bxjs@driver@given\bxjs@driver@@dvimode}
\DeclareOption{dvipdfmx}{%
  \def\bxjs@driver@opt{dvipdfmx}%
  \let\bxjs@driver@given\bxjs@driver@@dvipdfmx}
\DeclareOption{nodvidriver}{%
  \def\bxjs@driver@opt{nodvidriver}%
  \let\bxjs@driver@given\bxjs@driver@@none}
\DeclareOption{class-nodvidriver}{%
  \def\bxjs@driver@opt{class-nodvidriver}%
  \let\bxjs@driver@given\bxjs@driver@@none}
\DeclareOption{pdftex}{%
  \def\bxjs@driver@opt{pdftex}%
  \let\bxjs@driver@given\bxjs@driver@@pdfmode}
\DeclareOption{luatex}{%
  \def\bxjs@driver@opt{luatex}%
  \let\bxjs@driver@given\bxjs@driver@@pdfmode}
\DeclareOption{xetex}{%
  \def\bxjs@driver@opt{xetex}%
  \let\bxjs@driver@given\bxjs@driver@@xetex}
```

`dvipdfmx-if-dvi` は 2.0 版より非推奨となった。

```
\DeclareOption{dvipdfmx-if-dvi}{\bxjs@depre@opt@do{dvipdfmx-if-dvi}{dvi=dvipdfmx}}
```

■その他の BXJS 独自オプション TODO:3.x 互換用オプションを分離する。

\bxjs@depre@opt 非推奨のオプションについて警告を出す。

\bxjs@depre@opt@do
```
\@onlypreamble\bxjs@depre@opt
\def\bxjs@depre@opt#1#2{%
  \ClassWarningNoLine\bxjs@clsname
   {The old option '#1' is DEPRECATED\MessageBreak
    and may be abolished in future!\MessageBreak
    You should instead write:\MessageBreak
    \space\space #2}}
\@onlypreamble\bxjs@depre@opt@do
\def\bxjs@depre@opt@do#1#2{%
  \bxjs@depre@opt{#1}{#2}%
  \setkeys{bxjs}{#2}}
```

\ifbxjs@bigcode 〔スイッチ〕upTEX 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。

```
\newif\ifbxjs@bigcode \bxjs@bigcodetrue
```

nobigcode ／ bigcode オプションの定義。

```
\DeclareOption{nobigcode}{%
  \bxjs@bigcodefalse}
\DeclareOption{bigcode}{%
  \bxjs@bigcodetrue}
```

\ifbxjs@oldfontcommands 〔スイッチ〕\allowoldfontcommands を既定で有効にするか。

```
\newif\ifbxjs@oldfontcommands
```

nooldfontcommands、oldfontcommands 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 であるがこれはチョットアレなので避けた。

```
\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 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
  \def\bxjs@register@badopt#1{%
    \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。

```
\@onlypreamble\bxjs@invalid@opt
\ifbxjs@brace@safe
  \def\bxjs@invalid@opt#1#2{%
    \bxjs@register@badopt{#1}{\ClassError\bxjs@clsname{#2}\@ehc}}
\else
  \def\bxjs@invalid@opt#1#2{%
    \DeclareOption{#1}{\ClassWarningNoLine\bxjs@clsname{#2}}}
\fi
```

JS クラスにはあるが BXJS クラスにはないオプションを「無効オプション」として宣言する。

※ `ltjsclasses` クラスでも警告を出している。

```
\bxjs@invalid@opt{winjis}{%
  This class does not support 'winjis' option}
\bxjs@invalid@opt{mingoth}{%
  This class does not support 'mingoth' option}
\bxjs@invalid@opt{jis}{%
  This class does not support 'jis' option}
\if j\jsEngine\else
\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\@nnil}
\def\bxjs@check@ja@prefix@a#1ja:#2\@nil#3\@nnil{%
  \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
  \def\KV@errx##1{\OptionNotUsed}%
  \setkeys{#1}{#2}%
  \let\KV@errx\bxjs@save@KV@errx}
\ifbxjs@brace@safe\else
  \let\bxjs@cls@setkeys\bxjs@safe@setkeys
\fi
```

\bxjs@declare@enum@option \bxjs@declare@enum@option{⟨オプション名⟩}{⟨enum 名⟩}{⟨初期値⟩}
“⟨オプション名⟩=⟨値⟩” のオプション指定に対して、\[bxjs@⟨enum 名⟩] を \[bxjs@⟨enum 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@@#3}%
  \define@key{bxjs}{#1}{%
    \@ifundefined{bxjs@#2@@##1}{%
      \bxjs@error@keyval{#1}{##1}%
    }{\bxjs@csletcs{bxjs@#2}{bxjs@#2@@##1}}}}
```

\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#3}%
  \define@key{bxjs}{#1}[true]{%
    \@ifundefined{bxjs@#2##1}{%
      \bxjs@error@keyval{#1}{##1}%
```

```
    }{\@nameuse{bxjs@#2##1}}}}
```

`\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。

```
\def\bxjs@set@keyval#1#2#3{%
  \bxjs@csletcs{bxjs@next}{bxjs@kv@#1@#2}%
  \ifx\bxjs@next\relax
    \bxjs@error@keyval{#1}{#2}%
    #3%
  \else \bxjs@next
  \fi}
\@onlypreamble\bxjs@error@keyval
\def\bxjs@error@keyval#1#2{%
  \ClassError\bxjs@clsname
   {Invalid value '#2' for option #1}\@ehc}
```

`\jsScale` 〔実数値マクロ〕和文スケール値。

```
\def\jsScale{0.924715}
```

`\bxjs@base@opt` 明示された base オプションの値。

```
%\let\bxjs@base@opt\@undefined
```

base オプションの処理。

```
\define@key{bxjs}{base}{%
  \edef\bxjs@base@opt{#1}%
  \bxjs@setbasefontsize{#1}}
\define@key{bxjs}{fontsize}{\setkeys{bxjs}{base=#1}}
```

`\bxjs@jbase@opt` 明示された jbase オプションの値。

```
%\let\bxjs@jbase@opt\@undefined
```

jbase オプションの処理。

```
\define@key{bxjs}{jbase}{\edef\bxjs@jbase@opt{#1}}
\define@key{bxjs}{jafontsize}{\setkeys{bxjs}{jbase=#1}}
```

`\bxjs@scale@opt` 明示された scale オプションの値。

```
%\let\bxjs@scale@opt\@undefined
```

scale オプションの処理。

```
\define@key{bxjs}{scale}{%
  \edef\bxjs@scale@opt{#1}%
  \let\jsScale\bxjs@scale@opt}
\define@key{bxjs}{jafontscale}{\setkeys{bxjs}{scale=#1}}
```

noscale オプションの処理。

TODO:3.0 noscale は廃止の予定。

```
\DeclareOption{noscale}{\bxjs@depre@opt@do{noscale}{scale=1}}
```

`\bxjs@param@mag` mag オプションの値。

```
\let\bxjs@param@mag\relax
```

mag オプションの処理。

```
\define@key{bxjs}{mag}{\edef\bxjs@param@mag{#1}}
```

paper オプションの処理。

```
\define@key{bxjs}{paper}{\edef\bxjs@param@paper{#1}}
```

`\bxjs@jadriver` 和文ドライバの名前。

```
\let\bxjs@jadriver\relax
```

`\bxjs@jadriver@opt` 明示された和文ドライバの名前。

```
%\let\bxjs@jadriver@opt\@undefined
```

ja オプションの処理。

※ jadriver は 0.9 版で用いられた旧称。

TODO:3.0 jadriver は廃止の予定。

※単なる ja という指定は無視される（Pandoc 対策）。

```
\define@key{bxjs}{jadriver}{%
  \bxjs@depre@opt{jadriver}{ja=#1}\edef\bxjs@jadriver@opt{#1}}
\define@key{bxjs}{ja}[\relax]{%
  \ifx\relax#1\else\edef\bxjs@jadriver@opt{#1}\fi}
```

`\jsJaFont` 和文フォント設定の名前。

```
\let\jsJaFont\@empty
```

jafont オプションの処理。

```
\define@key{bxjs}{jafont}{\edef\jsJaFont{#1}}
```

`\jsJaParam` 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 の方を優先させる。

```
\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@@usemag
\ifx l\jsEngine \ifnum\luatexversion>86
  \let\bxjs@magstyle@@default\bxjs@magstyle@@xreal
\fi\fi
\ifjsWithpTeXng
  \let\bxjs@magstyle@@default\bxjs@magstyle@@xreal
\fi
\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。

```
\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。

```
\newif\ifbxjs@dvi@opt
```

DVI モードのドライバとドライバ種別との対応。

```
\let\bxjs@dvidriver@@dvipdfmx=\bxjs@driver@@dvipdfmx
\let\bxjs@dvidriver@@dvips=\bxjs@driver@@dvips
\let\bxjs@dvidriver@@dviout=\bxjs@driver@@dvimode
\let\bxjs@dvidriver@@xdvi=\bxjs@driver@@dvimode
\let\bxjs@dvidriver@@nodvidriver=\bxjs@driver@@none
\bxjs@cslet{bxjs@dvidriver@@class-nodvidriver}\bxjs@driver@@none
```

dvi オプションの処理。

```
\define@key{bxjs}{dvi}{%
  \bxjs@csletcs{bxjs@tmpa}{bxjs@dvidriver@@#1}%
  \ifx\bxjs@tmpa\relax
    \bxjs@error@keyval{dvi}{#1}%
  \else
```

\bxjs@driver@given を未定義にしていることに注意。

```
    \def\bxjs@driver@opt{#1}%
    \let\bxjs@driver@given\@undefined
    \bxjs@dvi@opttrue
  \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。

```
\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。

```
\newif\ifbxjs@force@chapterabstract
%<book>\bxjs@force@chapterabstracttrue
```

layout オプションの処理。

```
\@namedef{bxjs@kv@layout@v1}{%
%<book>\bxjs@layout@buggyhmargintrue
%<report>\bxjs@force@chapterabstracttrue
}
\@namedef{bxjs@kv@layout@v2}{%
%<book>\bxjs@layout@buggyhmarginfalse
%<report>\bxjs@force@chapterabstractfalse
}
\define@key{bxjs}{layout}{%
  \bxjs@set@keyval{layout}{#1}{}}
```

\bxjs@textwidth@limit textwidth-limit の指定値。

```
%\let\bxjs@textwidth@limit@opt\@undefined
\define@key{bxjs}{textwidth-limit}{%
  \bxjs@depre@opt{textwidth-limit}{textwidth=#1zw}%
  \edef\bxjs@textwidth@limit@opt{#1}}
```

\bxjs@textwidth@opt textwidth の指定値。

```
%\let\bxjs@textwidth@opt\@undefined
\define@key{bxjs}{textwidth}{\edef\bxjs@textwidth@opt{#1}}
\define@key{bxjs}{line_length}{\setkeys{bxjs}{textwidth=#1}}
```

\bxjs@number@of@lines@opt number-of-lines の指定値。

```
%\let\bxjs@number@of@lines@opt\@undefined
\define@key{bxjs}{number-of-lines}{\edef\bxjs@number@of@lines@opt{#1}}
\define@key{bxjs}{number_of_lines}{\setkeys{bxjs}{number-of-lines=#1}}
```

\bxjs@paragraph@mark paragraph-mark の指定値。パラグラフのマーク。

```
%\let\bxjs@paragraph@mark\@undefined
\define@key{bxjs}{paragraph-mark}{%
  \edef\bxjs@paragraph@mark{#1}}
```

\ifbxjs@whole@zw@lines〔スイッチ〕whole-zw-lines の指定値。

```
\bxjs@declare@bool@option{whole-zw-lines}{whole@zw@lines}{true}
```

\ifbxjs@jaspace@cmd〔スイッチ〕jaspace-cmd の指定値。

```
\bxjs@declare@bool@option{jaspace-cmd}{jaspace@cmd}{true}
\define@key{bxjs}{xkanjiskip-cmd}[true]{\setkeys{bxjs}{jaspace-cmd=#1}}
```

\ifbxjs@fix@at@cmd〔スイッチ〕fix-at-cmd の指定値。

```
\bxjs@declare@bool@option{fix-at-cmd}{fix@at@cmd}{true}
```

\ifbxjs@hyperref@enc〔スイッチ〕hyperref-enc の指定値。

```
\bxjs@declare@bool@option{hyperref-enc}{hyperref@enc}{true}
```

\bxjs@everyparhook everyparhook の指定値。

```
\chardef\bxjs@everyparhook@@none=0
\chardef\bxjs@everyparhook@@compat=1
\chardef\bxjs@everyparhook@@modern=2
\bxjs@declare@enum@option{everyparhook}{everyparhook}{%
  \if j\jsEngine compat\else modern\fi}
```

\bxjs@label@section label-section の指定値。

```
\chardef\bxjs@label@section@@none=0
\chardef\bxjs@label@section@@compat=1
\chardef\bxjs@label@section@@modern=2
\bxjs@declare@enum@option{label-section}{label@section}{compat}
```

\ifbxjs@usezw〔スイッチ〕use-zw の指定値。

TODO:3.0 zw/nozw は廃止の予定。

```
\bxjs@declare@bool@option{use-zw}{usezw}{true}
\DeclareOption{nozw}{\bxjs@depre@opt@do{nozw}{use-zw=false}}
\DeclareOption{zw}{\bxjs@depre@opt@do{zw}{use-zw=true}}
```

\ifbxjs@disguise@js〔スイッチ〕disguise-js の指定値。

TODO:3.0 js/nojs は廃止の予定。

```
\bxjs@declare@bool@option{disguise-js}{disguise@js}{true}
\DeclareOption{nojs}{\bxjs@depre@opt@do{nojs}{disguise-js=false}}
\DeclareOption{js}{\bxjs@depre@opt@do{js}{disguise-js=true}}
```

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

```
\bxjs@declare@bool@option{precise-text}{precisetext}{false}
\DeclareOption{noprecisetext}{\bxjs@depre@opt@do{noprecisetext}{precise-
  text=false}}
\DeclareOption{precisetext}{\bxjs@depre@opt@do{precisetext}{precise-
  text=true}}
```

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

```
\bxjs@declare@bool@option{simple-ja-setup}{simplejasetup}{true}
\DeclareOption{nosimplejasetup}{\bxjs@depre@opt@do{nosimplejasetup}{simple-
  ja-setup=false}}
```

```
\DeclareOption{simplejasetup}{\bxjs@depre@opt@do{simplejasetup}{simple-ja-
  setup=true}}
```

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

```
\bxjs@declare@bool@option{plautopatch}{plautopatch}{false}
\g@addto@macro\bxjs@plautopatchtrue{\let\bxjs@plautopatch@given\@undefined}
\g@addto@macro\bxjs@plautopatchfalse{\def\bxjs@plautopatch@given{false}}
```

■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 \@removeelement 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。

```
\ifbxjs@brace@safe\else
\let\bxjs@org@removeelement\@removeelement
\def\@removeelement#1#2#3{%
  \def\reserved@a{#2}%
  \ifx\reserved@a\@empty \let#3\@empty
  \else \bxjs@org@removeelement{#1}{#2}{#3}%
  \fi}
\fi
```

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。

```
%<article>\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 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:3.0 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#{\bxjs@tmpdo@c}
\def\bxjs@tmpdo@c#1\@nil{%
  \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:3.0 noscale オプションは廃止予定。

```
\@expandtwoargs\@removeelement
  {papersize}\@classoptionslist\@classoptionslist
\@expandtwoargs\@removeelement
  {10pt}\@classoptionslist\@classoptionslist
\@expandtwoargs\@removeelement
  {noscale}\@classoptionslist\@classoptionslist
```

■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。

```
\ifx\bxjs@jadriver@opt\@undefined\else
```

```
  \let\bxjs@jadriver\bxjs@jadriver@opt
\fi
```

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。

```
\let\bxjs@tmpb\jsEngine
\ifx j\bxjs@tmpb\ifjsWithpTeXng
  \let\bxjs@tmpb=g
\fi\fi
\ifx j\bxjs@tmpb\ifjsWithupTeX
  \let\bxjs@tmpb=u
\fi\fi
\ifx p\bxjs@tmpb\ifjsInPdfMode\else
  \let\bxjs@tmpb=n
\fi\fi
```

（この時点で `\bxjs@tmpb` は `\bxjs@engine@given` と同じ規則で分類したコードをもっている。）

```
\ifx *\bxjs@engine@given
  \let\bxjs@engine@given\bxjs@tmpb
```

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。

```
  \ifx j\bxjs@engine@given
    \bxjs@add@class@option{platex}
  \else\ifx u\bxjs@engine@given
    \bxjs@add@class@option{uplatex}
  \fi\fi
\fi
\ifx\bxjs@engine@given\@undefined\else
  \ifx\bxjs@engine@given\bxjs@tmpb\else
    \ClassError\bxjs@clsname
     {Option '\bxjs@engine@opt' used on wrong engine}\@ehc
  \fi
\fi
```

エンジンが pTeX-ng の場合、グローバルオプションに `uplatex` を追加する。

```
\ifjsWithpTeXng
  \bxjs@add@class@option{uplatex}
\fi
```

■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。

```
\@tempswatrue
\ifx \bxjs@driver@given\@undefined\else
  \ifjsInPdfMode
    \ifx\bxjs@driver@given\bxjs@driver@@pdfmode\else
      \@tempswafalse
```

```
    \fi
  \else\ifx x\jsEngine
    \ifx\bxjs@driver@given\bxjs@driver@@xetex\else
      \@tempswafalse
    \fi
  \else
    \ifx\bxjs@driver@given\bxjs@driver@@pdfmode
      \@tempswafalse
    \else\ifx\bxjs@driver@given\bxjs@driver@@xetex
      \@tempswafalse
    \fi\fi
    \ifjsWithpTeXng\ifx\bxjs@driver@given\bxjs@driver@@dvipdfmx\else
      \@tempswafalse
    \fi\fi
  \fi\fi
\fi
\if@tempswa\else
  \ClassError\bxjs@clsname
   {Option '\bxjs@driver@opt' used on wrong engine}\@ehc
\fi
```

DVI 出力のエンジンである場合の追加処理。

```
\ifjsInPdfMode \@tempswafalse
\else\ifx x\jsEngine \@tempswafalse
\else\ifjsWithpTeXng \@tempswafalse
\else \@tempswatrue
\fi\fi\fi
\if@tempswa
```

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）。

```
  \ifx\bxjs@driver@opt\@undefined
    \if \ifbxjs@explIII T\else\ifx\bxjs@jadriver@opt\@undefined F\else T\fi\fi T%
      \ClassWarningNoLine\bxjs@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
```

dvi=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。）

```
  \ifbxjs@dvi@opt
    \bxjs@csletcs{bxjs@driver@given}{bxjs@dvidriver@@\bxjs@driver@opt}
    \bxjs@add@class@option{\bxjs@driver@opt}
  \fi
```

```
\fi
```

エンジンが pTEX-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。

```
\ifjsWithpTeXng
  \ifx\bxjs@driver@given\bxjs@driver@@dvipdfmx
    \let\bxjs@platexng@nodrv\@undefined
  \else\ifx t\bxjs@platexng@nodrv\else
    \bxjs@add@class@option{dvipdfmx}
  \fi\fi
\fi
```

ドライバが nodvidriver であった場合の処理。DVI ウェア依存の処理を全て無効化する。

```
\ifx\bxjs@driver@given\bxjs@driver@@none
  \bxjs@papersizefalse
\fi
```

■その他の BXJS 特有の後処理 \documentclass より前に plautopatch パッケージが読み込まれている場合は bxjs@plautopatch を真にする。

```
\@ifpackageloaded{plautopatch}{%
  \bxjs@plautopatchtrue
}{}
```

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 (u)pTEX 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=minimal' is assumed as fallback, but\MessageBreak
      such implicit setting is now DEPRECATED!\MessageBreak
      You should write 'ja=minimal' explicitly,\MessageBreak
      if it is intended}
    \let\bxjs@jadriver\bxjs@@minimal
  \fi
\fi
```

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
\ifx j\jsEngine \ifbxjs@plautopatch
  \RequirePackage{plautopatch}[2018/08/22]%v0.3
\fi\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 非指定の場合はスキップする。

```
\ifx\bxjs@jadriver@opt\@undefined\else
  \ifx\bxjs@engine@given\@undefined
    \ClassError\bxjs@clsname
     {An engine option must be explicitly given}%
     {When you use a Japanese-driver you must specify a correct\MessageBreak
      engine option.\MessageBreak\@ehc}
\fi\fi
```

新しい 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。）

```
\ifx\bxjs@magstyle@@default\bxjs@magstyle@@mag\else
  \ifx\bxjs@magstyle\bxjs@magstyle@@mag
    \let\bxjs@magstyle\bxjs@magstyle@@default
    \ClassError\bxjs@clsname
     {The engine does not support 'magstyle=usemag'}%
     {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。

```
\ifx\bxjs@jbase@opt\@undefined\else
  \ifx\bxjs@base@opt\@undefined
```

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。

```
    \ifx\bxjs@scale@opt\@undefined\else
      \ClassWarningNoLine\bxjs@clsname
       {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）クラスを読込済のように振舞う。※「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。

```
\ifbxjs@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 `nosetpagesize` を設定しておく。

```
\bxjs@add@class@option{nosetpagesize}
```

`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て，1 インチずつ加えました。ところが pLaTeX 2_ε 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

```
-O -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] memoir クラス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] トンボオプションが指定されているとき「だけ」`\stockwidth`，`\stockheight` を定義するようにしました。

[2020-10-04] LaTeX 2$_\varepsilon$ 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しまいます (Issue #72)。

[2022-09-12] 次期 LaTeX 2$_\varepsilon$ カーネルに\stockwidth，\stockheight が追加されるようですので，クラスファイル側では未定義のときのみこれらの長さ変数を定義します。h20y6m さん，ありがとうございます。

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。

```
\ifx\stockwidth\@undefined\newdimen\stockwidth\fi
\ifx\stockheight\@undefined\newdimen\stockheight\fi
\begingroup\expandafter\expandafter\expandafter\endgroup
\expandafter\ifx\csname iftombow\expandafter\endcsname\csname iftrue\endcsname
  \setlength{\stockwidth}{\paperwidth}
  \setlength{\stockheight}{\paperheight}
  \advance \stockwidth 2in
  \advance \stockheight 2in
\fi
```

■基準となる行送り

\n@baseline 基準となる行送りをポイント単位で表したものです。

```
%<slide>\def\n@baseline{13}%
%<!slide>\ifdim\bxjs@param@basefontsize<10pt \def\n@baseline{15}%
%<!slide>\else \def\n@baseline{16}\fi
```

■拡大率の設定

\bxjs@magstyle の値に応じてスイッチ jsc@mag と jsc@mag@xreal を設定する。

```
\ifx\bxjs@magstyle\bxjs@magstyle@@mag
  \jsc@magtrue
\else\ifx\bxjs@magstyle\bxjs@magstyle@@xreal
  \jsc@mag@xrealtrue
\fi\fi
```

サイズの変更は TeX のプリミティブ \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。

[2008-12-26] 1000 / \mag に相当する \inv@mag を定義しました。truein を使っていたところを \inv@mag in に直しましたので，geometry パッケージと共存できると思います。なお，新ドキュメントクラス側で 10pt 以外にする場合の注意：

- geometry 側でオプション truedimen を指定してください。

- geometry 側でオプション mag は使えません。

設定すべき \mag 値を (基底サイズ)/(10 pt) × 1000 と算出。BXJS クラスでは、\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。

```
\ifx\bxjs@param@mag\relax
  \@tempdima=\bxjs@param@basefontsize
  \advance\@tempdima.001pt \multiply\@tempdima25
  \divide\@tempdima16384\relax \@tempcnta\@tempdima\relax
  \edef\bxjs@param@mag{\the\@tempcnta}
\else
% mag 値が直接指定された場合
  \bxjs@gset@tempcnta{\bxjs@param@mag}
  \ifnum\@tempcnta<\z@ \@tempcnta=\z@ \fi
% 有効な mag 値の範囲は 1--32768
  \edef\bxjs@param@mag{\the\@tempcnta}
  \advance\@tempcnta100000
  \def\bxjs@tmpa#1#2#3#4#5\@nil{\@tempdima=#2#3#4.#5\p@}
  \expandafter\bxjs@tmpa\the\@tempcnta\@nil
  \edef\bxjs@param@basefontsize{\the\@tempdima}
\fi
\@tempcnta\bxjs@param@mag \advance\@tempcnta100000
\def\bxjs@tmpa#1#2#3#4\@nil{\@tempdima=#2#3.#4\p@}
\expandafter\bxjs@tmpa\the\@tempcnta\@nil
\edef\jsc@magscale{\strip@pt\@tempdima}
\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。

```
\newdimen\jsc@mpt
\newdimen\jsc@mmm
\ifjsc@mag
  \jsc@mpt=1\p@
  \jsc@mmm=1mm
\else
  \jsc@mpt=\jsc@magscale\p@
  \jsc@mmm=\jsc@magscale mm
\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 × (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。

```
\newdimen\jsZw
\jsZw=10\jsc@mpt \jsZw=\jsScale\jsZw
\ifbxjs@usezw
```

```
  \providecommand*\zw{\jsZw}
\fi
```

\zwspace 全角幅の水平空き。

```
\def\zwspace{\hskip\jsZw\relax}
```

そして、magstyle が nomag* の場合は、NFSS にパッチを当てる。

```
\ifjsc@mag@xreal
  \RequirePackage{type1cm}
  \let\jsc@invscale\bxjs@invscale
```

```
  \ifbxjs@TUenc
    \expandafter\let\csname TU/lmr/m/n/10\endcsname\relax
  \else
    \expandafter\let\csname OT1/cmr/m/n/10\endcsname\relax
  \fi
  \expandafter\let\csname OMX/cmex/m/n/10\endcsname\relax
  \let\jsc@get@external@font\get@external@font
  \def\get@external@font{%
    \jsc@preadjust@extract@font
    \jsc@get@external@font}
  \def\jsc@fstrunc#1{%
    \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 \jsc@smallskip を定義します。これは \maketitle で用いられます。\jsc@medskip と \jsc@bigskip は必要ないのでコメントアウトしています。

\jsc@smallskip
\jsc@medskip
\jsc@bigskip

```
\def\jsc@smallskip{\vspace\jsc@smallskipamount}
%\def\jsc@medskip{\vspace\jsc@medskipamount}
%\def\jsc@bigskip{\vspace\jsc@bigskipamount}
```

\jsc@smallskipamount
\jsc@medskipamount
\jsc@bigskipamount

```
\newskip\jsc@smallskipamount
\jsc@smallskipamount=3\jsc@mpt plus 1\jsc@mpt minus 1\jsc@mpt
%\newskip\jsc@medskipamount
%\jsc@medskipamount =6\jsc@mpt plus 2\jsc@mpt minus 2\jsc@mpt
%\newskip\jsc@bigskipamount
%\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>
%<*jsclasses>
%<*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）」を表す実数値マクロ \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，9.62216 pt ∗ 0.961/10 pt = 0.924690 です。

```
%</class>
%<*minijs>
%% min/goth -> jis/jisg (for pLaTeX only)
\ifnum\jis"2121="3000 \else
\@for\@tempa:=5,6,7,8,9,10,10.95,12,14.4,17.28,20.74,24.88\do{%
  \expandafter\let\csname JY1/mc/m/n/\@tempa\endcsname\relax
  \expandafter\let\csname JY1/gt/m/n/\@tempa\endcsname\relax
  \expandafter\let\csname JT1/mc/m/n/\@tempa\endcsname\relax
  \expandafter\let\csname JT1/gt/m/n/\@tempa\endcsname\relax
}
\def\Cjascale{0.924690}
\DeclareFontShape{JY1}{mc}{m}{n}{<-> s * [0.961] jis}{}
\DeclareFontShape{JY1}{gt}{m}{n}{<-> s * [0.961] jisg}{}
\DeclareFontShape{JT1}{mc}{m}{n}{<-> s * [0.961] tmin10}{}
\DeclareFontShape{JT1}{gt}{m}{n}{<-> s * [0.961] tgoth10}{}
\fi
%</minijs>
%<*class>
```

```
%<*!jspf>
\def\Cjascale{0.924690}
\ifmingoth
  \DeclareFontShape{\jsc@JYn}{mc}{m}{n}{<-> s * [0.961] \jsc@pfx@ min10}{}
  \DeclareFontShape{\jsc@JYn}{gt}{m}{n}{<-> s * [0.961] \jsc@pfx@ goth10}{}
  \DeclareFontShape{\jsc@JTn}{mc}{m}{n}{<-> s * [0.961] \jsc@pfx@ tmin10}{}
  \DeclareFontShape{\jsc@JTn}{gt}{m}{n}{<-> s * [0.961] \jsc@pfx@ tgoth10}{}
\else
  \ifjisfont
    \DeclareFontShape{\jsc@JYn}{mc}{m}{n}{<-> s * [0.961] \jsc@pfx@ jis}{}
    \DeclareFontShape{\jsc@JYn}{gt}{m}{n}{<-> s * [0.961] \jsc@pfx@ jisg}{}
    \DeclareFontShape{\jsc@JTn}{mc}{m}{n}{<-> s * [0.961] \jsc@pfx@ tmin10}{}
    \DeclareFontShape{\jsc@JTn}{gt}{m}{n}{<-> s * [0.961] \jsc@pfx@ tgoth10}{}
  \else
    \if@jsc@uplatex
      \DeclareFontShape{JY2}{mc}{m}{n}{<-> s * [0.924690] upjisr-h}{}
      \DeclareFontShape{JY2}{gt}{m}{n}{<-> s * [0.924690] upjisg-h}{}
      \DeclareFontShape{JT2}{mc}{m}{n}{<-> s * [0.924690] upjisr-v}{}
      \DeclareFontShape{JT2}{gt}{m}{n}{<-> s * [0.924690] upjisg-v}{}
    \else
      \DeclareFontShape{\jsc@JYn}{mc}{m}{n}{<-> s * [0.961] \jsc@pfx@ jis}{}
      \DeclareFontShape{\jsc@JYn}{gt}{m}{n}{<-> s * [0.961] \jsc@pfx@ jisg}{}
      \DeclareFontShape{\jsc@JTn}{mc}{m}{n}{<-> s * [0.961] \jsc@pfx@ tmin10}{}
      \DeclareFontShape{\jsc@JTn}{gt}{m}{n}{<-> s * [0.961] \jsc@pfx@ tgoth10}{}
    \fi
  \fi
\fi
%</!jspf>
```

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，$9/(9.62216 * 72/72.27) = 0.93885$ 倍します。

[2018-02-04] 和文スケール値 `\Cjascale` は $9.62216\,\mathrm{pt} * 0.93885/10\,\mathrm{pt} = 0.903375$ です。

```
%<*jspf>
\def\Cjascale{0.903375}
\ifmingoth
  \DeclareFontShape{\jsc@JYn}{mc}{m}{n}{<-> s * [0.93885] \jsc@pfx@ min10}{}
  \DeclareFontShape{\jsc@JYn}{gt}{m}{n}{<-> s * [0.93885] \jsc@pfx@ goth10}{}
  \DeclareFontShape{\jsc@JTn}{mc}{m}{n}{<-> s * [0.93885] \jsc@pfx@ tmin10}{}
  \DeclareFontShape{\jsc@JTn}{gt}{m}{n}{<-> s * [0.93885] \jsc@pfx@ tgoth10}{}
\else
  \ifjisfont
    \DeclareFontShape{\jsc@JYn}{mc}{m}{n}{<-> s * [0.93885] \jsc@pfx@ jis}{}
    \DeclareFontShape{\jsc@JYn}{gt}{m}{n}{<-> s * [0.93885] \jsc@pfx@ jisg}{}
    \DeclareFontShape{\jsc@JTn}{mc}{m}{n}{<-> s * [0.93885] \jsc@pfx@ tmin10}{}
    \DeclareFontShape{\jsc@JTn}{gt}{m}{n}{<-> s * [0.93885] \jsc@pfx@ tgoth10}{}
  \else
    \if@jsc@uplatex
      \DeclareFontShape{JY2}{mc}{m}{n}{<-> s * [0.903375] upjisr-h}{}
      \DeclareFontShape{JY2}{gt}{m}{n}{<-> s * [0.903375] upjisg-h}{}
```

```
      \DeclareFontShape{JT2}{mc}{m}{n}{<-> s * [0.903375] upjisr-v}{}
      \DeclareFontShape{JT2}{gt}{m}{n}{<-> s * [0.903375] upjisg-v}{}
    \else
      \DeclareFontShape{\jsc@JYn}{mc}{m}{n}{<-> s * [0.93885] \jsc@pfx@ jis}{}
      \DeclareFontShape{\jsc@JYn}{gt}{m}{n}{<-> s * [0.93885] \jsc@pfx@ jisg}{}
      \DeclareFontShape{\jsc@JTn}{mc}{m}{n}{<-> s * [0.93885] \jsc@pfx@ tmin10}{}
      \DeclareFontShape{\jsc@JTn}{gt}{m}{n}{<-> s * [0.93885] \jsc@pfx@ tgoth10}{}
    \fi
  \fi
\fi
%</jspf>
```

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々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 環境を手直ししてしのいでいましたが，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。

[2004-11-03] \rmfamily も和文対応にしました。

```
% \DeclareFontShape{\jsc@JYn}{mc}{bx}{n}{<->ssub*gt/m/n}{} % in \jsc@JYnmc
% \DeclareFontShape{\jsc@JYn}{gt}{bx}{n}{<->ssub*gt/m/n}{} % in \jsc@JYngt
\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}{mc}{bx}{it}{<->ssub*gt/m/n}{}
\DeclareFontShape{\jsc@JYn}{mc}{bx}{sl}{<->ssub*gt/m/n}{}
% \DeclareFontShape{\jsc@JTn}{mc}{bx}{n}{<->ssub*gt/m/n}{} % in \jsc@JTnmc
% \DeclareFontShape{\jsc@JTn}{gt}{bx}{n}{<->ssub*gt/m/n}{} % in \jsc@JTngt
\DeclareFontShape{\jsc@JTn}{mc}{m}{it}{<->ssub*mc/m/n}{}
\DeclareFontShape{\jsc@JTn}{mc}{m}{sl}{<->ssub*mc/m/n}{}
\DeclareFontShape{\jsc@JTn}{mc}{m}{sc}{<->ssub*mc/m/n}{}
\DeclareFontShape{\jsc@JTn}{gt}{m}{it}{<->ssub*gt/m/n}{}
\DeclareFontShape{\jsc@JTn}{gt}{m}{sl}{<->ssub*gt/m/n}{}
\DeclareFontShape{\jsc@JTn}{mc}{bx}{it}{<->ssub*gt/m/n}{}
\DeclareFontShape{\jsc@JTn}{mc}{bx}{sl}{<->ssub*gt/m/n}{}
```

[2020-02-02] LaTeX 2_ε 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いますので，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 2_ε 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 2_ε 2020-10-01 では \AddToHook を利用します。

```
%</class>
%<*class|minijs>
%% ad-hoc "relation font"
\@ifl@t@r\fmtversion{2020/10/01}
    {\jsc@needsp@tchfalse}{\jsc@needsp@tchtrue}
\ifjsc@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}
\DeclareRobustCommand\sffamily
        {\not@math@alphabet\sffamily\mathsf
         \romanfamily\sfdefault\kanjifamily\gtdefault\selectfont}
\DeclareRobustCommand\ttfamily
        {\not@math@alphabet\ttfamily\mathtt
         \romanfamily\ttdefault\kanjifamily\gtdefault\selectfont}
\AtBeginDocument{%
  \ifx\mweights@init\@undefined\else  % mweights.sty is loaded
    % my definitions above should have been overwritten, recover it!
    % \selectfont is executed twice but I don't care about speed...
    \expandafter\g@addto@macro\csname rmfamily \endcsname
      {\kanjifamily\mcdefault\selectfont}%
    \expandafter\g@addto@macro\csname sffamily \endcsname
      {\kanjifamily\gtdefault\selectfont}%
    \expandafter\g@addto@macro\csname ttfamily \endcsname
      {\kanjifamily\gtdefault\selectfont}%
  \fi}
\else                           % 2020-02-02
\g@addto@macro\@rmfamilyhook
  {\prepare@family@series@update@kanji{mc}\mcdefault}
\g@addto@macro\@sffamilyhook
  {\prepare@family@series@update@kanji{gt}\gtdefault}
\g@addto@macro\@ttfamilyhook
  {\prepare@family@series@update@kanji{gt}\gtdefault}
\fi
\else % --- for 2020-02-02 or older END & for 2020-10-01 BEGIN
\AddToHook{rmfamily}%
  {\prepare@family@series@update@kanji{mc}\mcdefault}
\AddToHook{sffamily}%
  {\prepare@family@series@update@kanji{gt}\gtdefault}
\AddToHook{ttfamily}%
  {\prepare@family@series@update@kanji{gt}\gtdefault}
\fi   % --- for 2020-10-01 END
%</class|minijs>
%<*class>
```

\textmc
\textgt

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。

```
\ifx\DeclareFixJFMCJKTextFontCommand\@undefined
\DeclareRobustCommand\textmc[1]{%
    \relax\ifmmode \expandafter\nfss@text \fi{\mcfamily #1}}
\DeclareRobustCommand\textgt[1]{%
    \relax\ifmmode \expandafter\nfss@text \fi{\gtfamily #1}}
\fi
```

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)。

[2010-03-14] http://oku.edu.mie-u.ac.jp/tex/mod/forum/discuss.php?d=411 での山本さんのご指摘に従って修正しました。

```
\def\reDeclareMathAlphabet#1#2#3{%
  \edef\@tempa{\expandafter\@gobble\string#2}%
  \edef\@tempb{\expandafter\@gobble\string#3}%
  \edef\@tempc{\string @\expandafter\@gobbletwo\string#2}%
  \ifx\@tempc\@tempa%
    \edef\@tempa{\expandafter\@gobbletwo\string#2}%
    \edef\@tempb{\expandafter\@gobbletwo\string#3}%
  \fi
  \begingroup
    \let\protect\noexpand
    \def\@tempaa{\relax}%
    \expandafter\ifx\csname RDMAorg@\@tempa\endcsname\relax
      \edef\@tempaa{\expandafter\def\expandafter\noexpand%
        \csname RDMAorg@\@tempa\endcsname{%
          \expandafter\noexpand\csname\@tempa\endcsname}}%
    \fi
    \def\@tempbb{\relax}%
    \expandafter\ifx\csname RDMAorg@\@tempb\endcsname\relax
      \edef\@tempbb{\expandafter\def\expandafter\noexpand%
        \csname RDMAorg@\@tempb\endcsname{%
```

```
          \expandafter\noexpand\csname\@tempb\endcsname}}%
    \fi
    \edef\@tempc{\@tempaa\@tempbb}%
  \expandafter\endgroup\@tempc%
  \edef#1{\noexpand\protect\expandafter\noexpand\csname%
    \expandafter\@gobble\string#1\space\space\endcsname}%
  \expandafter\edef\csname\expandafter\@gobble\string#1\space\space\endcsname%
    {\noexpand\DualLang@mathalph@bet%
      {\expandafter\noexpand\csname RDMAorg@\@tempa\endcsname}%
      {\expandafter\noexpand\csname RDMAorg@\@tempb\endcsname}%
  }%
}
\@onlypreamble\reDeclareMathAlphabet
\def\DualLang@mathalph@bet#1#2{%
  \relax\ifmmode
    \ifx\math@bgroup\bgroup%     2e normal style     (\mathrm{...})
      \bgroup\let\DualLang@Mfontsw\DLMfontsw@standard
    \else
      \ifx\math@bgroup\relax%    2e two letter style (\rm->\mathrm)
        \let\DualLang@Mfontsw\DLMfontsw@oldstyle
      \else
        \ifx\math@bgroup\@empty% 2.09 oldlfont style ({\mathrm ...})
          \let\DualLang@Mfontsw\DLMfontsw@oldlfont
        \else%                   panic! assume 2e normal style
          \bgroup\let\DualLang@Mfontsw\DLMfontsw@standard
        \fi
      \fi
    \fi
  \else
    \let\DualLang@Mfontsw\@firstoftwo
  \fi
  \DualLang@Mfontsw{#1}{#2}%
}
\def\DLMfontsw@standard#1#2#3{#1{#2{#3}}\egroup}
\def\DLMfontsw@oldstyle#1#2{#1\relax\@fontswitch\relax{#2}}
\def\DLMfontsw@oldlfont#1#2{#1\relax#2\relax}
\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}
  \AtBeginDocument{%
    \reDeclareMathAlphabet{\mathrm}{\@mathrm}{\@mathmc}
    \reDeclareMathAlphabet{\mathbf}{\@mathbf}{\@mathgt}}
\fi
```

\textsterling これは \pounds 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 \pounds 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，\pounds 以外で使われるとは思えないので，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。

```
% \DeclareTextCommand{\textsterling}{OT1}{{\itshape\char`\$}}
```

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，ピリオドの後に四分アキが入らないようにするために

```
% \xspcode`.=0
```

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。

```
\xspcode`+=3
\xspcode`\%=3
```

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。

```
\xspcode`^^80=3
\xspcode`^^81=3
\xspcode`^^82=3
\xspcode`^^83=3
\xspcode`^^84=3
\xspcode`^^85=3
\xspcode`^^86=3
\xspcode`^^87=3
\xspcode`^^88=3
\xspcode`^^89=3
\xspcode`^^8a=3
\xspcode`^^8b=3
\xspcode`^^8c=3
\xspcode`^^8d=3
\xspcode`^^8e=3
```

```
\xspcode`^^8f=3
\xspcode`^^90=3
\xspcode`^^91=3
\xspcode`^^92=3
\xspcode`^^93=3
\xspcode`^^94=3
\xspcode`^^95=3
\xspcode`^^96=3
\xspcode`^^97=3
\xspcode`^^98=3
\xspcode`^^99=3
\xspcode`^^9a=3
\xspcode`^^9b=3
\xspcode`^^9c=3
\xspcode`^^9d=3
\xspcode`^^9e=3
\xspcode`^^9f=3
\xspcode`^^a0=3
\xspcode`^^a1=3
\xspcode`^^a2=3
\xspcode`^^a3=3
\xspcode`^^a4=3
\xspcode`^^a5=3
\xspcode`^^a6=3
\xspcode`^^a7=3
\xspcode`^^a8=3
\xspcode`^^a9=3
\xspcode`^^aa=3
\xspcode`^^ab=3
\xspcode`^^ac=3
\xspcode`^^ad=3
\xspcode`^^ae=3
\xspcode`^^af=3
\xspcode`^^b0=3
\xspcode`^^b1=3
\xspcode`^^b2=3
\xspcode`^^b3=3
\xspcode`^^b4=3
\xspcode`^^b5=3
\xspcode`^^b6=3
\xspcode`^^b7=3
\xspcode`^^b8=3
\xspcode`^^b9=3
\xspcode`^^ba=3
\xspcode`^^bb=3
\xspcode`^^bc=3
\xspcode`^^bd=3
\xspcode`^^be=3
\xspcode`^^bf=3
```

```
\xspcode`^^c0=3
\xspcode`^^c1=3
\xspcode`^^c2=3
\xspcode`^^c3=3
\xspcode`^^c4=3
\xspcode`^^c5=3
\xspcode`^^c6=3
\xspcode`^^c7=3
\xspcode`^^c8=3
\xspcode`^^c9=3
\xspcode`^^ca=3
\xspcode`^^cb=3
\xspcode`^^cc=3
\xspcode`^^cd=3
\xspcode`^^ce=3
\xspcode`^^cf=3
\xspcode`^^d0=3
\xspcode`^^d1=3
\xspcode`^^d2=3
\xspcode`^^d3=3
\xspcode`^^d4=3
\xspcode`^^d5=3
\xspcode`^^d6=3
\xspcode`^^d7=3
\xspcode`^^d8=3
\xspcode`^^d9=3
\xspcode`^^da=3
\xspcode`^^db=3
\xspcode`^^dc=3
\xspcode`^^dd=3
\xspcode`^^de=3
\xspcode`^^df=3
\xspcode`^^e0=3
\xspcode`^^e1=3
\xspcode`^^e2=3
\xspcode`^^e3=3
\xspcode`^^e4=3
\xspcode`^^e5=3
\xspcode`^^e6=3
\xspcode`^^e7=3
\xspcode`^^e8=3
\xspcode`^^e9=3
\xspcode`^^ea=3
\xspcode`^^eb=3
\xspcode`^^ec=3
\xspcode`^^ed=3
\xspcode`^^ee=3
\xspcode`^^ef=3
\xspcode`^^f0=3
```

```
\xspcode`^^f1=3
\xspcode`^^f2=3
\xspcode`^^f3=3
\xspcode`^^f4=3
\xspcode`^^f5=3
\xspcode`^^f6=3
\xspcode`^^f7=3
\xspcode`^^f8=3
\xspcode`^^f9=3
\xspcode`^^fa=3
\xspcode`^^fb=3
\xspcode`^^fc=3
\xspcode`^^fd=3
\xspcode`^^fe=3
\xspcode`^^ff=3
%</class>
%</jsclasses>
%<*class>
```

`\@` 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。

[2016-07-14] 2015-01-01 の LaTeX で，auxiliary files に書き出されたときにスペースが食われないようにする修正が入りました。これに合わせて `{}` を補いました。

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。

```
\chardef\bxjs@periodchar=`\.
\bxjs@robust@def\bxjs@SE{%
  \ifnum\spacefactor<\@m \spacefactor\@m
  \else \spacefactor\sfcode\bxjs@periodchar
  \fi}
\ifbxjs@fix@at@cmd
  \def\@{\bxjs@SE{}}
\fi
```

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
\@viiipt   8         \@ixpt    9      \@xpt    10
\@xipt    10.95      \@xiipt  12      \@xivpt  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 pLaTeX 2_ε で 0pt 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。

\xkanjiskip 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ばすのが一般的ですが，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（1zw）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 に対してパッチを当てるように変更。

\bxjs@patch@set@fontsize \set@fontsize にパッチを当てる。

※\set@fontsize を書き換えるパッケージへの対策のため、クラス読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パッチを当てる。

TODO:3.0 新しい LaTeX カーネルで selectfont フックを利用する。

```
%\let\bxjs@prev@set@fontsize\@undefined
\@onlypreamble\bxjs@patch@set@fontsize
\def\bxjs@patch@set@fontsize{%
  \ifx\bxjs@prev@set@fontsize\set@fontsize\else
    \def\bxjs@tmpa{\def\set@fontsize####1####2####3}%
\expandafter\bxjs@tmpa\expandafter{%
  \set@fontsize{##1}{##2}{##3}%
% 末尾にコードを追加
```

```
  \expandafter\def\expandafter\size@update\expandafter{%
    \size@update
    \jsFontSizeChanged}%
}
    \let\bxjs@prev@set@fontsize\set@fontsize
  \fi}
```

この場とパッケージ末尾で作動させる。

```
\bxjs@patch@set@fontsize
\AtEndOfClass{\bxjs@patch@set@fontsize}
```

\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。

```
\newcommand*\jsFontSizeChanged{%
  \jsZw=\f@size\p@
  \jsZw=\jsScale \jsZw
  \ifdim\parindent>\z@
    \if@english \parindent=1em
    \else       \parindent=1\jsZw
    \fi
  \fi\relax
  \jsResetDimen}
```

\jsResetDimen ユーザ定義用のフック。

```
\providecommand*\jsResetDimen{}
```

\jsc@setfontsize 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。

```
\ifjsc@mag
  \let\jsc@setfontsize\@setfontsize
\else
  \def\jsc@setfontsize#1#2#3{%
    \@setfontsize#1{#2\jsc@mpt}{#3\jsc@mpt}}
% microtype 対策
  \ifjsWitheTeX\if j\jsEngine\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
\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
```

\ifnarrowbaselines
\narrowbaselines
\widebaselines

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。

[2003-06-30] 数式に入るところで \narrowbaselines を実行しているので \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，しっぽ愛好家さんが次の修正を教えてくださいました。

[2008-02-18] 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（\@currsize の実体）が呼ばれた」ことになり警告が出る。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も望ましくないので、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
\if@english
  \narrowbaselinestrue
\fi
\def\narrowbaselines{%
  \narrowbaselinestrue
  \skip0=\abovedisplayskip
  \skip2=\abovedisplayshortskip
  \skip4=\belowdisplayskip
  \skip6=\belowdisplayshortskip
% 一時的に警告を無効化する
  \let\bxjs@save@nomath\@nomath
  \let\@nomath\@gobble
  \@currsize\selectfont
  \let\@nomath\bxjs@save@nomath
  \abovedisplayskip=\skip0
  \abovedisplayshortskip=\skip2
  \belowdisplayskip=\skip4
  \belowdisplayshortskip=\skip6\relax}
\def\widebaselines{\narrowbaselinesfalse\@currsize\selectfont}
```

---

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 LaTeX 式条件文にしたもの。

```
\def\bxjs@if@narrowbaselines{%
  \ifnarrowbaselines\expandafter\@firstoftwo
  \else \expandafter\@secondoftwo
  \fi
}
```

\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，アスキー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（アスキーのものの 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，$16/9.25 \approx 1.73$ 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。

```
\renewcommand{\normalsize}{%
  \bxjs@if@narrowbaselines{%
    \jsc@setfontsize\normalsize\@xpt\@xiipt
  }{%else
    \jsc@setfontsize\normalsize\@xpt{\n@baseline}%
  }%
```

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逡巡していましたが，結局，微調節してみることにしました。

```
  \abovedisplayskip 11\jsc@mpt \@plus3\jsc@mpt \@minus4\jsc@mpt
  \abovedisplayshortskip \z@ \@plus3\jsc@mpt
  \belowdisplayskip 9\jsc@mpt \@plus3\jsc@mpt \@minus4\jsc@mpt
  \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
\normalsize
%</class|minijs>
%<*class>
```

\Cht
\Cdp
\Cwd
\Cvs
\Chs

基準となる長さの設定をします。pLaTeX 2_ε 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。

[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。

\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\Cht\@undefined \newdimen\Cht \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 \Cht と \Cdp は単純に \Cwd の 88% と 12% の値とする。

```
\setlength\Cht{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 の使われ方を考えて，ここでは和文 13 ポイント，欧文 11 ポイントとします。また，\topsep と \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（\z@）にしました。

```
\newcommand{\small}{%
  \bxjs@if@narrowbaselines{%
%<!kiyou>    \jsc@setfontsize\small\@ixpt{11}%
%<kiyou>    \jsc@setfontsize\small{8.8888}{11}%
  }{%else
%<!kiyou>    \jsc@setfontsize\small\@ixpt{13}%
%<kiyou>    \jsc@setfontsize\small{8.8888}{13.2418}%
  }%
  \abovedisplayskip 9\jsc@mpt \@plus3\jsc@mpt \@minus4\jsc@mpt
  \abovedisplayshortskip  \z@ \@plus3\jsc@mpt
  \belowdisplayskip \abovedisplayskip
  \belowdisplayshortskip \belowdisplayskip
  \def\@listi{\leftmargin\leftmargini
              \topsep \z@
              \parsep \z@
              \itemsep \parsep}}
```

\footnotesize

\footnotesize も同様です。\topsep と \parsep は，元はそれぞれ 3 ± 1，2 ± 1 ポイントでしたが，ここではゼロ（\z@）にしました。

```
\newcommand{\footnotesize}{%
  \bxjs@if@narrowbaselines{%
%<!kiyou>    \jsc@setfontsize\footnotesize\@viiipt{9.5}%
```

```
%<kiyou>    \jsc@setfontsize\footnotesize{8.8888}{11}%
  }{%else
%<!kiyou>    \jsc@setfontsize\footnotesize\@viiipt{11}%
%<kiyou>    \jsc@setfontsize\footnotesize{8.8888}{13.2418}%
  }%
  \abovedisplayskip 6\jsc@mpt \@plus2\jsc@mpt \@minus3\jsc@mpt
  \abovedisplayshortskip  \z@ \@plus2\jsc@mpt
  \belowdisplayskip \abovedisplayskip
  \belowdisplayshortskip \belowdisplayskip
  \def\@listi{\leftmargin\leftmargini
              \topsep \z@
              \parsep \z@
              \itemsep \parsep}}
```

\scriptsize
\tiny
\large
\Large
\LARGE
\huge
\Huge
\HUGE

それ以外のサイズは，本文に使うことがないので，単にフォントサイズと行送りだけ変更します。特に注意すべきは \large で，これは二段組のときに節見出しのフォントとして使い，行送りを \normalsize と同じにすることによって，節見出しが複数行にわたっても段間で行が揃うようにします。

[2004-11-03] \HUGE を追加。

```
\newcommand{\scriptsize}{\jsc@setfontsize\scriptsize\@viipt\@viiipt}
\newcommand{\tiny}{\jsc@setfontsize\tiny\@vpt\@vipt}
\if@twocolumn
%<!kiyou>  \newcommand{\large}{\jsc@setfontsize\large\@xiipt{\n@baseline}}
%<kiyou>  \newcommand{\large}{\jsc@setfontsize\large{11.111}{\n@baseline}}
\else
%<!kiyou>  \newcommand{\large}{\jsc@setfontsize\large\@xiipt{17}}
%<kiyou>  \newcommand{\large}{\jsc@setfontsize\large{11.111}{17}}
\fi
%<!kiyou>\newcommand{\Large}{\jsc@setfontsize\Large\@xivpt{21}}
%<kiyou>\newcommand{\Large}{\jsc@setfontsize\Large{12.222}{21}}
\newcommand{\LARGE}{\jsc@setfontsize\LARGE\@xviipt{25}}
\newcommand{\huge}{\jsc@setfontsize\huge\@xxpt{28}}
\newcommand{\Huge}{\jsc@setfontsize\Huge\@xxvpt{33}}
\newcommand{\HUGE}{\jsc@setfontsize\HUGE{30}{40}}
```

別行立て数式の中では \narrowbaselines にします。和文の行送りのままでは，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な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。

```
\everydisplay=\expandafter{\the\everydisplay \narrowbaselines}
```

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，もっとうまい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 2_ε 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。

```
% \newcommand{\headfont}{\bfseries}
\newcommand{\headfont}{\sffamily}
% \newcommand{\headfont}{\sffamily\fontseries{sbc}\selectfont}
```

5 レイアウト

■二段組

\columnsep
\columnseprule

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

```
%<!kiyou>\setlength\columnsep{2\Cwd}
%<kiyou>\setlength\columnsep{28truebp}
\setlength\columnseprule{\z@}
```

■段落

\lineskip
\normallineskip
\lineskiplimit
\normallineskiplimit

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。

```
\setlength\lineskip{1\jsc@mpt}
\setlength\normallineskip{1\jsc@mpt}
\setlength\lineskiplimit{1\jsc@mpt}
\setlength\normallineskiplimit{1\jsc@mpt}
```

\baselinestretch

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}{}
```

\parskip
\parindent

\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{\z@}
\if@slide
  \setlength\parindent{0\p@}
\else
  \setlength\parindent{1\Cwd}
\fi
```

\@lowpenalty
\@medpenalty
\@highpenalty

\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。

```
% \interlinepenalty 0
```

\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。

```
% \brokenpenalty 100
```

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

\bxjs@bd@pre@geometry@hook 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。

```
\@onlypreamble\bxjs@bd@pre@geometry@hook
\let\bxjs@bd@pre@geometry@hook\@empty
```

現状ではここで \mag を設定している。

\topskip も指定する。

```
\ifjsc@mag
\mag=\bxjs@param@mag
\fi
\setlength{\topskip}{10\jsc@mpt}
```

\jsSetQHLength のための和文単位の定義。

```
\def\bxjs@unit@trueQ{0.25truemm}\let\bxjs@unit@trueH\bxjs@unit@trueQ
\def\bxjs@unit@zw{\jsZw}\let\bxjs@unit@zh\bxjs@unit@zw
```

\bxjs@param@paper が長さ指定の場合、geometry の形式（papersize={W,H}）に変換する。{W}{H} の形式について。

```
\@tempswafalse
\def\bxjs@tmpdo{\@ifnextchar\bgroup\bxjs@tmpdo@a\remove@to@nnil}
\def\bxjs@tmpdo@a#1{\edef\bxjs@tmpa{#1}%
  \@ifnextchar\bgroup\bxjs@tmpdo@b\remove@to@nnil}
\def\bxjs@tmpdo@b#1{\edef\bxjs@tmpa{\bxjs@tmpa,#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
```

W,H の形式について。

```
\if@tempswa\else
```

```
  \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
```

W*H の形式について。

```
\if@tempswa\else
  \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
```

`\bxjs@layout@paper` geometry の用紙設定のオプション。

```
\edef\bxjs@layout@paper{%
  \ifjsc@mag truedimen,\fi
  \if@landscape landscape,\fi
  \bxjs@param@paper}
```

`\bxjs@layout` 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。

```
%<*article|report>
\def\bxjs@layout@base{%
  headheight=\topskip,footskip=0.03367\paperheight,%
  headsep=\footskip-\topskip,includeheadfoot,%
}
\edef\bxjs@layout{\bxjs@layout@base
  hscale=0.76,hmarginratio=1:1,%
  vscale=0.83,vmarginratio=1:1,%
}
%</article|report>
%<*book>
\def\bxjs@layout@base{%
  headheight=\topskip,headsep=6\jsc@mmm,nofoot,includeheadfoot,%
}
\ifbxjs@layout@buggyhmargin     %---
% アレ
\edef\bxjs@layout{\bxjs@layout@base
  hmargin=36\jsc@mmm,hmarginratio=1:1,%
  vscale=0.83,vmarginratio=1:1,%
}
\else                           %---
% 非アレ
\edef\bxjs@layout{\bxjs@layout@base
  hmargin=18\jsc@mmm,%
  vscale=0.83,vmarginratio=1:1,%
}
```

```
\fi                                %---
%</book>
%<*slide>
\def\bxjs@layout@base{%
  noheadfoot,%
}
\edef\bxjs@layout{\bxjs@layout@base
  hscale=0.9,hmarginratio=1:1,%
  vscale=0.944,vmarginratio=1:1,%
}
%</slide>
```

textwidth オプションの設定を反映する。

```
%<*!book>
\ifx\bxjs@textwidth@opt\@undefined\else
  \jsSetQHLength\@tempdima{\bxjs@textwidth@opt}
  \edef\bxjs@layout{\bxjs@layout width=\the\@tempdima,}
\fi
%</!book>
\ifx\bxjs@number@of@lines@opt\@undefined\else
  \bxjs@gset@tempcnta{\bxjs@number@of@lines@opt}
  \edef\bxjs@layout{\bxjs@layout lines=\the\@tempcnta,}
\fi
```

`\fullwidth`〔寸法レジスタ〕ヘッダ・フッタ領域の横幅。

```
\newdimen\fullwidth
```

`\bxjs@textwidth@limit`〔寸法値マクロ〕bxjsbook における、\textwidth 上限の値。

`\jsTextWidthLimit`〔実数値マクロ〕\bxjs@textwidth@limit の全角（\Cwd）単位での値。

```
%<*book>
\newcommand\jsTextWidthLimit{40}
\@tempdima=\jsTextWidthLimit\Cwd
\ifx\bxjs@textwidth@limit@opt\@undefined\else
  \bxjs@gset@tempcnta{\bxjs@textwidth@limit@opt}
  \@tempdima=\@tempcnta\Cwd
\fi
\ifx\bxjs@textwidth@opt\@undefined\else
  \jsSetQHLength\@tempdima{\bxjs@textwidth@opt}
\fi
\edef\bxjs@textwidth@limit{\the\@tempdima}
\ifdim\@tempdima=\jsTextWidthLimit\Cwd\else
  \bxjs@invscale\@tempdima{\strip@pt\Cwd}
  \bxjs@nclong\edef\jsTextWidthLimit{\strip@pt\@tempdima}
\fi
%</book>
```

`\bxjs@preproc@layout` geometry 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。

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

\bxjs@postproc@layout geometry の後処理。

```
\def\bxjs@postproc@layout{%
```

geometry のドライバを再設定する。

```
  \ifx\bxjs@geometry@driver\relax\else
    \let\Gm@driver\bxjs@geometry@driver
  \fi
```

\ht\strutbox の値を元に戻す。

```
  \ht\strutbox=\bxjs@save@ht@strutbox\relax
```

\textwidth の値を補正する。

```
  \ifbxjs@whole@zw@lines
    \@tempdimb=\textwidth
    \if@twocolumn \@tempdima=2\Cwd \else \@tempdima=1\Cwd \fi
    \advance\textwidth.005pt\relax
    \divide\textwidth\@tempdima \multiply\textwidth\@tempdima
    \advance\@tempdimb-\textwidth
    \advance\oddsidemargin 0.5\@tempdimb
    \advance\evensidemargin 0.5\@tempdimb
  \fi
  \fullwidth=\textwidth
```

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。

```
%<*book>
  \@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}
```

■geometry パッケージを読み込む

\xjs@apply@bd@pre@geometry@hook 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ャムニャ。

```
\def\bxjs@geometry@name{geometry}
\ifbxjs@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
```

geoemtry=class の場合に、実際に geometry パッケージを読みこむ。

```
\ifx\bxjs@geometry\bxjs@geometry@@class
```

geometry のドライバオプション指定。nopapersize 指定時は、special 命令出力を抑止するためにドライバを none にする。そうでない場合は、クラスで指定したドライバオプションが引き継がれるので何もしなくてよいが、例外として、ドライバが dvipdfmx の時は、現状の geometry は dvipdfm を指定する必要がある。

```
\ifbxjs@papersize
  \ifx\bxjs@driver@given\bxjs@driver@@dvipdfmx
    \PassOptionsToPackage{dvipdfm}{geometry}
  \else\ifx\bxjs@driver@given\bxjs@driver@@dvimode
```

```
    \PassOptionsToPackage{dvipdfm}{geometry}
  \fi\fi
  \let\bxPapersizeSpecialDone=t
\else
  \PassOptionsToPackage{driver=none}{geometry}
\fi
```

ここで geometry を読み込む。

※ geometry の begin-document フックにおいて、LuaTEX の旧版互換を有効にする。

```
\bxjs@apply@bd@pre@geometry@hook{%
  \bxjs@bd@pre@geometry@hook
  \@nameuse{ImposeOldLuaTeXBehavior}}
\bxjs@preproc@layout
\edef\bxjs@next{%
  \noexpand\RequirePackage[\bxjs@layout@paper,\bxjs@layout]{geometry}%
}\bxjs@next
\bxjs@apply@bd@pre@geometry@hook{\@nameuse{RevokeOldLuaTeXBehavior}}
```

\bxjs@geometry@driver geometry 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ほしくないので、\bxjs@postproc@layout において書き戻す処理を入れている。

```
\let\bxjs@geometry@driver\Gm@driver
\bxjs@postproc@layout
```

geometry のドライバ自動判別に対する前処理。

```
\g@addto@macro\bxjs@bd@pre@geometry@hook{%
```

BXJS の 2.0 版より、geometry の 4.x 版のサポートは廃止された。

```
    \@ifpackagelater{geometry}{2010/02/12}{}{%else
      \PackageError\bxjs@clsname
       {Your 'geometry' package is too old (< v5.0)}%
       {\@ehc}%
      \let\Gm@driver\relax}%
```

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

```
    \ifjsWithpTeXng
      \ifx\Gm@driver\@empty
        \def\Gm@driver{pdftex}%
      \fi
    \fi}
```

\setpagelayout ページレイアウト設定のためのユーザ命令。

```
\def\setpagelayout{%
  \bxjs@ifplus{\bxjs@setpagelayout@a\tw@}{%else
    \@ifstar{\bxjs@setpagelayout@a\@ne}{\bxjs@setpagelayout@a\z@}}}
\def\bxjs@setpagelayout@a#1#2{%
  \ifcase#1% modify
    \def\bxjs@next{\ifjsc@mag truedimen,\fi #2}%
  \or% reset(*)
```

```
    \def\bxjs@next{reset,\bxjs@layout@paper,#2}%
  \or% semireset(+)
    \def\bxjs@next{reset,\bxjs@layout@paper,\bxjs@layout@base,#2}%
  \fi
  \bxjs@preproc@layout
  \edef\bxjs@next{%
    \noexpand\geometry{\bxjs@next}%
  }\bxjs@next
  \bxjs@postproc@layout}
```

■geometry パッケージを読み込まない　geometry=user の場合の処理。

```
\else\ifx\bxjs@geometry\bxjs@geometry@@user
```

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。

```
\g@addto@macro\bxjs@begin@document@hook{%
  \ifdim\textwidth=.5\maxdimen
    \ClassError\bxjs@clsname
     {Page layout is not properly set}%
     {\@ehd}%
  \fi}
\def\jsUseMinimalPageLayout{%
  \setlength{\textwidth}{6.5in}%
  \setlength{\textheight}{8in}}
```

\setpagelayout 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。

```
%<*jsclasses>
```

\headheight
\topskip

\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 のままとします。

```
\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 判のときちょうど 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て，ゼロにしました。

```
%<*article|kiyou>
\if@slide
  \setlength\footskip{0pt}
\else
  \setlength\footskip{0.03367\paperheight}
  \ifdim\footskip<\baselineskip
    \setlength\footskip{\baselineskip}
  \fi
\fi
%</article|kiyou>
%<jspf>\setlength\footskip{9\jsc@mmm}
%<*book>
\if@report
  \setlength\footskip{0.03367\paperheight}
  \ifdim\footskip<\baselineskip
    \setlength\footskip{\baselineskip}
  \fi
\else
  \setlength\footskip{0pt}
\fi
%</book>
%<*report>
\setlength\footskip{0.03367\paperheight}
\ifdim\footskip<\baselineskip
  \setlength\footskip{\baselineskip}
\fi
```

```
%</report>
```

\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）になっていました。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 を減らしそこねていたのを修正しました（2016-08-17 での修正漏れ）。

```
%<*article>
\if@slide
  \setlength\headsep{0\jsc@mpt}
  \addtolength\headsep{-\topskip}%% added (2016-10-08)
  \addtolength\headsep{10\jsc@mpt}%% added (2016-10-08)
\else
  \setlength\headsep{\footskip}
  \addtolength\headsep{-\topskip}
\fi
%</article>
%<*book>
\if@report
  \setlength\headsep{\footskip}
  \addtolength\headsep{-\topskip}
\else
  \setlength\headsep{6\jsc@mmm}
  \addtolength\headsep{-\topskip}%% added (2016-10-08)
  \addtolength\headsep{10\jsc@mpt}%% added (2016-10-08)
\fi
%</book>
%<*report>
\setlength\headsep{\footskip}
\addtolength\headsep{-\topskip}
%</report>
%<*jspf>
\setlength\headsep{9\jsc@mmm}
\addtolength\headsep{-\topskip}
%</jspf>
%<*kiyou>
\setlength\headheight{0\jsc@mpt}
\setlength\headsep{0\jsc@mpt}
\addtolength\headsep{-\topskip}%% added (2016-10-08)
\addtolength\headsep{10\jsc@mpt}%% added (2016-10-08)
%</kiyou>
```

\maxdepth \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。

```
\setlength\maxdepth{.5\topskip}
```

■本文の幅と高さ

\fullwidth 本文の幅が全角 40 文字を超えると読みにくくなります。そこで，書籍の場合に限って，紙の幅が広いときは外側のマージンを余分にとって全角 40 文字に押え，ヘッダやフッタは本文領域より広く取ることにします。このときヘッダやフッタの幅を表す \fullwidth という長さを定義します。

```
\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。

```
%<*article>
\if@slide
  \setlength\fullwidth{0.9\paperwidth}
\else
  \setlength\fullwidth{0.76\paperwidth}
\fi
\if@twocolumn \@tempdima=2zw \else \@tempdima=1zw \fi
\divide\fullwidth\@tempdima \multiply\fullwidth\@tempdima
\setlength\textwidth{\fullwidth}
%</article>
%<*book>
\if@report
  \setlength\fullwidth{0.76\paperwidth}
\else
  \setlength\fullwidth{\paperwidth}
  \addtolength\fullwidth{-36\jsc@mmm}
\fi
\if@twocolumn \@tempdima=2zw \else \@tempdima=1zw \fi
\divide\fullwidth\@tempdima \multiply\fullwidth\@tempdima
\setlength\textwidth{\fullwidth}
\if@report \else
  \if@twocolumn \else
    \ifdim \fullwidth>40zw
      \setlength\textwidth{40zw}
    \fi
  \fi
\fi
%</book>
%<*report>
\setlength\fullwidth{0.76\paperwidth}
\if@twocolumn \@tempdima=2zw \else \@tempdima=1zw \fi
```

```
\divide\fullwidth\@tempdima \multiply\fullwidth\@tempdima
\setlength\textwidth{\fullwidth}
%</report>
%<*jspf>
\setlength\fullwidth{50zw}
\addtolength\fullwidth{8\jsc@mmm}
\setlength\textwidth{\fullwidth}
%</jspf>
%<*kiyou>
\setlength\fullwidth{48zw}
\addtolength\fullwidth{\columnsep}
\setlength\textwidth{\fullwidth}
%</kiyou>
```

\textheight 紙の高さ \paperheight は，1 インチと \topmargin と \headheight と \headsep と \textheight と \footskip とページ下部の余白を加えたものです。

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きます。0.83 倍という数値は，A4 縦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）。

```
%<*article|book|report>
\if@slide
  \setlength{\textheight}{0.95\paperheight}
\else
  \setlength{\textheight}{0.83\paperheight}
\fi
\addtolength{\textheight}{-10\jsc@mpt}%% from -\topskip (2016-10-08); from -
    \headheight (2003-06-26)
\addtolength{\textheight}{-\headsep}
\addtolength{\textheight}{-\footskip}
\addtolength{\textheight}{-\topskip}
\divide\textheight\baselineskip
\multiply\textheight\baselineskip
%</article|book|report>
%<jspf>\setlength{\textheight}{51\baselineskip}
%<kiyou>\setlength{\textheight}{47\baselineskip}
\addtolength{\textheight}{\topskip}
\addtolength{\textheight}{0.1\jsc@mpt}
```

```
%<jspf>\setlength{\mathindent}{10\jsc@mmm}
```

\flushbottom [2016-07-18] \textheight 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 LaTeX 2ε での完全な \flushbottom の定義は

```
\def\flushbottom{%
  \let\@textbottom\relax \let\@texttop\relax}
```

ですが，次のようにします。

```
\def\flushbottom{%
  \def\@textbottom{\vskip \z@ \@plus.1\jsc@mpt}%
  \let\@texttop\relax}
```

\marginparsep \marginparpush \marginparsep は欄外の書き込みと本文との間隔です。\marginparpush は欄外の書き込みどうしの最小の間隔です。

```
\setlength\marginparsep{\columnsep}
\setlength\marginparpush{\baselineskip}
```

\oddsidemargin \evensidemargin それぞれ奇数ページ，偶数ページの左マージンから 1 インチ引いた値です。片面印刷では \oddsidemargin が使われます。TeX は上・左マージンに 1truein を挿入しますが，トンボ関係のオプションが指定されると pLaTeX 2ε（plcore.ltx）はトンボの内側に 1in のスペース（1truein ではなく）を挿入するので，場合分けしています。

```
\setlength{\oddsidemargin}{\paperwidth}
\addtolength{\oddsidemargin}{-\fullwidth}
\setlength{\oddsidemargin}{.5\oddsidemargin}
\iftombow
  \addtolength{\oddsidemargin}{-1in}
\else
  \addtolength{\oddsidemargin}{-\inv@mag in}
\fi
\setlength{\evensidemargin}{\oddsidemargin}
\if@mparswitch
  \addtolength{\evensidemargin}{\fullwidth}
  \addtolength{\evensidemargin}{-\textwidth}
\fi
```

\marginparwidth \marginparwidth は欄外の書き込みの横幅です。外側マージンの幅（\evensidemargin + 1 インチ）から 1 センチを引き，さらに \marginparsep（欄外の書き込みと本文のアキ）を引いた値にしました。最後に 1 zw の整数倍に切り捨てます。

```
\setlength\marginparwidth{\paperwidth}
\addtolength\marginparwidth{-\oddsidemargin}
\addtolength\marginparwidth{-\inv@mag in}
\addtolength\marginparwidth{-\textwidth}
\addtolength\marginparwidth{-10\jsc@mmm}
\addtolength\marginparwidth{-\marginparsep}
\@tempdima=1zw
\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。

[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。

[2016-08-17] \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。…のつもりでしたが，\textheight を増やし忘れていたので変わってしまっていました（2016-08-26 修正済み）。

```
\setlength\topmargin{\paperheight}
\addtolength\topmargin{-\textheight}
\if@slide
  \addtolength\topmargin{-\headheight}
\else
  \addtolength\topmargin{-10\jsc@mpt}%% from -\topskip (2016-10-08); from -
  \headheight (2003-06-26)
\fi
\addtolength\topmargin{-\headsep}
\addtolength\topmargin{-\footskip}
\setlength\topmargin{0.5\topmargin}
%<kiyou>\setlength\topmargin{81truebp}
\iftombow
  \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。

\footnotesep 各脚注の頭に入る支柱（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。

```
2010 \footnotesep=11\jsc@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。

```
2011 \setlength{\skip\footins}{16\jsc@mpt \@plus 5\jsc@mpt \@minus 2\jsc@mpt}
```

■フロート関連　フロート（図，表）関連のパラメータは LaTeX 2_ε 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。
[2003-08-23] ちょっと増やしました。

```
2012 \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。

```
2013 \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。
[2003-08-23] ちょっと増やしました。

```
2014 \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。

```
2015 \renewcommand{\bottomfraction}{.8}
```

\c@totalnumber totalnumber 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。
[2003-08-23] ちょっと増やしました。

```
2016 \setcounter{totalnumber}{20}
```

\textfraction 本文ページに最低限入ら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。

```
2017 \renewcommand{\textfraction}{.1}
```

\floatpagefraction 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。

```
2018 \renewcommand{\floatpagefraction}{.8}
```

\c@dbltopnumber 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。
[2003-08-23] ちょっと増やしました。

```
2019 \setcounter{dbltopnumber}{9}
```

\dbltopfraction 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。

```
2020 \renewcommand{\dbltopfraction}{.8}
```

\dblfloatpagefraction 二段組のときフロートだけのページに入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。

```
\renewcommand{\dblfloatpagefraction}{.8}
```

\floatsep \floatsep はページ上部・下部のフロート間の距離です。\textfloatsep はページ上部・下部のフロートと本文との距離です。\intextsep は本文の途中に出力されるフロートと本文との距離です。

\textfloatsep

\intextsep

```
\setlength\floatsep    {12\jsc@mpt \@plus 2\jsc@mpt \@minus 2\jsc@mpt}
\setlength\textfloatsep{20\jsc@mpt \@plus 2\jsc@mpt \@minus 4\jsc@mpt}
\setlength\intextsep   {12\jsc@mpt \@plus 2\jsc@mpt \@minus 2\jsc@mpt}
```

\dblfloatsep 二段組のときの段抜きのフロートについての値です。

\dbltextfloatsep

```
\setlength\dblfloatsep    {12\jsc@mpt \@plus 2\jsc@mpt \@minus 2\jsc@mpt}
\setlength\dbltextfloatsep{20\jsc@mpt \@plus 2\jsc@mpt \@minus 4\jsc@mpt}
```

\@fptop 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。

\@fpsep

\@fpbot

```
\setlength\@fptop{0\jsc@mpt \@plus 1fil}
\setlength\@fpsep{8\jsc@mpt \@plus 2fil}
\setlength\@fpbot{0\jsc@mpt \@plus 1fil}
```

\@dblfptop 段抜きフロートについての値です。

\@dblfpsep

\@dblfpbot

```
\setlength\@dblfptop{0\jsc@mpt \@plus 1fil}
\setlength\@dblfpsep{8\jsc@mpt \@plus 2fil}
\setlength\@dblfpbot{0\jsc@mpt \@plus 1fil}
```

6 改ページ（日本語 TEX 開発コミュニティ版のみ）

\pltx@cleartorightpage [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。

\pltx@cleartoleftpage

\pltx@cleartooddpage

\pltx@cleartoevenpage

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
%    \iftdir
%      \hbox{}\thispagestyle{empty}\newpage
%      \if@twocolumn\hbox{}\newpage\fi
%    \fi
%  \else
%    \ifydir
%      \hbox{}\thispagestyle{empty}\newpage
```

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

BXJS クラスでは \iftdir 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
\toks@\expandafter{\clearpage}
\xdef\clearpage{\noexpand\ifbxjs@after@preamble\the\toks@\noexpand\fi}
\endgroup
\fi
```

\cleardoublepage [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，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 スタイルは 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

\@oddhead，\@oddfoot，\@evenhead，\@evenfoot 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 の中で呼び出されます。\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 \@mkboth 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。LaTeX 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。

```
% \def\ps@empty{%
%   \let\@mkboth\@gobbletwo
%   \let\@oddhead\@empty
%   \let\@oddfoot\@empty
%   \let\@evenhead\@empty
%   \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。

```
\def\ps@plainfoot{%
  \let\@mkboth\@gobbletwo
  \let\@oddhead\@empty
  \def\@oddfoot{\normalfont\hfil\thepage\hfil}%
  \let\@evenhead\@empty
```

```
  \let\@evenfoot\@oddfoot}
\def\ps@plainhead{%
  \let\@mkboth\@gobbletwo
  \let\@oddfoot\@empty
  \let\@evenfoot\@empty
  \def\@evenhead{%
    \if@mparswitch \hss \fi
    \hbox to \fullwidth{\textbf{\thepage}\hfil}%
    \if@mparswitch\else \hss \fi}%
  \def\@oddhead{%
    \hbox to \fullwidth{\hfil\textbf{\thepage}}\hss}}
%<book>\let\ps@plain\ps@plainhead
%<!book>\let\ps@plain\ps@plainfoot
```

\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。

```
%<*article|slide>
\if@twoside
  \def\ps@headings{%
    \let\@oddfoot\@empty
    \let\@evenfoot\@empty
    \def\@evenhead{\if@mparswitch \hss \fi
      \underline{\hbox to \fullwidth{\textbf{\thepage}\hfil\leftmark}}%
      \if@mparswitch\else \hss \fi}%
    \def\@oddhead{%
      \underline{%
        \hbox to \fullwidth{{\rightmark}\hfil\textbf{\thepage}}}\hss}%
    \let\@mkboth\markboth
    \def\sectionmark##1{\markboth{%
       \ifnum \c@secnumdepth >\z@ \bxjs@label@sect{section}\hskip1\jsZw\fi
       ##1}{}}%
    \def\subsectionmark##1{\markright{%
       \ifnum \c@secnumdepth >\@ne \bxjs@label@sect{subsection}\hskip1\jsZw\fi
       ##1}}%
  }
\else % if not twoside
  \def\ps@headings{%
    \let\@oddfoot\@empty
    \def\@oddhead{%
      \underline{%
        \hbox to \fullwidth{{\rightmark}\hfil\textbf{\thepage}}}\hss}%
    \let\@mkboth\markboth
    \def\sectionmark##1{\markright{%
        \ifnum \c@secnumdepth >\z@ \bxjs@label@sect{section}\hskip1\jsZw\fi
        ##1}}}
\fi
%</article|slide>
```

次は book および report の場合です。[2011-05-10] しっぽ愛好家さん [qa:6370] のパッチを取り込ませていただきました（北見さん [qa:55896] のご指摘ありがとうございます）。

```
%<*book|report>
```

\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。

```
\def\bxjs@maybe@autoxspacing{%
  \ifx\autoxspacing\@undefined\else \autoxspacing \fi}
```

```
\newif\if@omit@number
\def\ps@headings{%
  \let\@oddfoot\@empty
  \let\@evenfoot\@empty
  \def\@evenhead{%
    \if@mparswitch \hss \fi
    \underline{\hbox to \fullwidth{\bxjs@maybe@autoxspacing
        \textbf{\thepage}\hfil\leftmark}}%
    \if@mparswitch\else \hss \fi}%
  \def\@oddhead{\underline{\hbox to \fullwidth{\bxjs@maybe@autoxspacing
       {\if@twoside\rightmark\else\leftmark\fi}\hfil\textbf{\thepage}}}\hss}%
  \let\@mkboth\markboth
  \def\chaptermark##1{\markboth{%
    \ifnum \c@secnumdepth >\m@ne
      \if@mainmatter
        \if@omit@number\else
          \@chapapp\thechapter\@chappos\hskip1\jsZw
        \fi
      \fi
    \fi
    ##1}{}}%
  \def\sectionmark##1{\markright{%
    \ifnum \c@secnumdepth >\z@ \bxjs@label@sect{section}\hskip1\jsZw\fi
    ##1}}}%
%</book|report>
```

最後は学会誌の場合です。

```
%<*jspf>
\def\ps@headings{%
  \def\@oddfoot{\normalfont\hfil\thepage\hfil}
  \def\@evenfoot{\normalfont\hfil\thepage\hfil}
  \def\@oddhead{\normalfont\hfil \@title \hfil}
  \def\@evenhead{\normalfont\hfil プラズマ・核融合学会誌\hfil}}
%</jspf>
```

\ps@myheadings myheadings ページスタイルではユーザが \markboth や \markright で柱を設定するため，ここでの定義は非常に簡単です。

[2004-01-17] 渡辺徹さんのパッチを適用しました。

```
\def\ps@myheadings{%
  \let\@oddfoot\@empty\let\@evenfoot\@empty
  \def\@evenhead{%
    \if@mparswitch \hss \fi%
    \hbox to \fullwidth{\thepage\hfil\leftmark}%
    \if@mparswitch\else \hss \fi}%
  \def\@oddhead{%
    \hbox to \fullwidth{\rightmark\hfil\thepage}\hss}%
  \let\@mkboth\@gobbletwo
%<book|report>  \let\chaptermark\@gobble
  \let\sectionmark\@gobble
%<!book&!report>  \let\subsectionmark\@gobble
}
```

8 文書のマークアップ

\bxjs@phantomsection hyperref が読み込まれている場合に \phantomsection を実行する。

```
\let\bxjs@phantomsection\relax
\g@addto@macro\bxjs@begin@document@hook{%
  \@ifpackageloaded{hyperref}{%
    \let\bxjs@phantomsection\phantomsection
  }{}%
}
```

8.1 表題

\title \author \date 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。

```
% \newcommand*{\title}[1]{\gdef\@title{#1}}
% \newcommand*{\author}[1]{\gdef\@author{#1}}
% \newcommand*{\date}[1]{\gdef\@date{#1}}
% \date{\today}
```

\subtitle 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}
```

\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}
```

\etitle 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。

\eauthor

\keywords

```
%<*jspf>
\newcommand*{\etitle}[1]{\gdef\@etitle{#1}}
\newcommand*{\eauthor}[1]{\gdef\@eauthor{#1}}
\newcommand*{\keywords}[1]{\gdef\@keywords{#1}}
\newcommand*{\email}[1]{\gdef\authors@mail{#1}}
\newcommand*{\AuthorsEmail}[1]{\gdef\authors@mail{author's e-mail:\ #1}}
%</jspf>
```

\plainifnotempty 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す。

```
\def\plainifnotempty{%
  \ifx \@oddhead \@empty
    \ifx \@oddfoot \@empty
    \else
      \thispagestyle{plainfoot}%
    \fi
  \else
    \thispagestyle{plainhead}%
  \fi}
```

\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\jsc@smallskip に置き換えました。\smallskip

のままでは 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。

```
%<*article|book|report|slide>
\if@titlepage
  \newcommand{\maketitle}{%
    \begin{titlepage}%
      \let\footnotesize\small
      \let\footnoterule\relax
      \let\footnote\thanks
      \null\vfil
      \if@slide
        {\footnotesize \@date}%
        \begin{center}
          \mbox{} \\[1\jsZw]
          \large
          {\maybeblue\hrule height0\jsc@mpt depth2\jsc@mpt\relax}\par
          \jsc@smallskip
          \@title
          \ifx\bxjs@subtitle\@undefined\else
            \par\vskip\z@
            {\small \bxjs@subtitle\par}
          \fi
          \jsc@smallskip
          {\maybeblue\hrule height0\jsc@mpt depth2\jsc@mpt\relax}\par
          \vfill
          {\small \@author}%
        \end{center}
      \else
      \vskip 60\jsc@mpt
      \begin{center}%
        {\LARGE \@title \par}%
        \ifx\bxjs@subtitle\@undefined\else
          \vskip5\jsc@mpt
          {\normalsize \bxjs@subtitle\par}
        \fi
        \vskip 3em%
        {\large
          \lineskip .75em
          \begin{tabular}[t]{c}%
            \@author
          \end{tabular}\par}%
        \vskip 1.5em
        {\large \@date \par}%
      \end{center}%
      \fi
      \par
      \@thanks\vfil\null
    \end{titlepage}%
    \setcounter{footnote}{0}%
```

```
    \global\let\thanks\relax
    \global\let\maketitle\relax
    \global\let\@thanks\@empty
    \global\let\@author\@empty
    \global\let\@date\@empty
    \global\let\@title\@empty
    \global\let\title\relax
    \global\let\author\relax
    \global\let\date\relax
    \global\let\and\relax
    \bxjs@annihilate@subtitle
  }%
\else
  \newcommand{\maketitle}{\par
    \begingroup
      \renewcommand\thefootnote{\@fnsymbol\c@footnote}%
      \def\@makefnmark{\rlap{\@textsuperscript{\normalfont\@thefnmark}}}%
      \long\def\@makefntext##1{\advance\leftskip 3\jsZw
        \parindent 1\jsZw\noindent
        \llap{\@textsuperscript{\normalfont\@thefnmark}\hskip0.3\jsZw}##1}%
      \if@twocolumn
        \ifnum \col@number=\@ne
          \@maketitle
        \else
          \twocolumn[\@maketitle]%
        \fi
      \else
        \newpage
        \global\@topnum\z@  % Prevents figures from going at top of page.
        \@maketitle
      \fi
      \plainifnotempty
      \@thanks
    \endgroup
    \setcounter{footnote}{0}%
    \global\let\thanks\relax
    \global\let\maketitle\relax
    \global\let\@thanks\@empty
    \global\let\@author\@empty
    \global\let\@date\@empty
    \global\let\@title\@empty
    \global\let\title\relax
    \global\let\author\relax
    \global\let\date\relax
    \global\let\and\relax
    \bxjs@annihilate@subtitle
  }
```

\@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。

```
\def\@maketitle{%
  \newpage\null
  \vskip 2em
  \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}[t]{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\@makefnmark{\rlap{\@textsuperscript{\normalfont\@thefnmark}}}%
    \long\def\@makefntext##1{\advance\leftskip 3\jsZw
      \parindent 1\jsZw\noindent
      \llap{\@textsuperscript{\normalfont\@thefnmark}\hskip0.3\jsZw}##1}%
      \twocolumn[\@maketitle]%
    \plainifnotempty
    \@thanks
  \endgroup
  \setcounter{footnote}{0}%
  \global\let\thanks\relax
  \global\let\maketitle\relax
  \global\let\@thanks\@empty
  \global\let\@author\@empty
  \global\let\@date\@empty
% \global\let\@title\@empty % \@title は柱に使う
  \global\let\title\relax
  \global\let\author\relax
  \global\let\date\relax
  \global\let\and\relax
```

```
  \ifx\authors@mail\@undefined\else{%
    \def\@makefntext{\advance\leftskip 3\jsZw \parindent -3\jsZw}%
    \footnotetext[0]{\itshape\authors@mail}%
  }\fi
  \global\let\authors@mail\@undefined}
\def\@maketitle{%
  \newpage\null
  \vskip 6em % used to be 2em
  \begin{center}
    \let\footnote\thanks
    \ifx\@title\@undefined\else{\LARGE\headfont\@title\par}\fi
    \lineskip .5em
    \ifx\@author\@undefined\else
      \vskip 1em
      \begin{tabular}[t]{c}%
        \@author
      \end{tabular}\par
    \fi
    \ifx\@etitle\@undefined\else
      \vskip 1em
      {\large \@etitle \par}%
    \fi
    \ifx\@eauthor\@undefined\else
      \vskip 1em
      \begin{tabular}[t]{c}%
        \@eauthor
      \end{tabular}\par
    \fi
    \vskip 1em
    \@date
  \end{center}
  \vskip 1.5em
  \centerline{\box\@abstractbox}
  \ifx\@keywords\@undefined\else
    \vskip 1.5em
    \centerline{\parbox{157\jsc@mmm}{\textsf{Keywords:}\\ \small\@keywords}}
  \fi
  \vskip 1.5em}
%</jspf>
```

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{%
```

```
2404   \@ifundefined{label#1}{\@nameuse{the#1}}{\@nameuse{label#1}}}
2405 \def\@seccntformat#1{\bxjs@label@sect{#1}\quad}
```

`\@secapp` 節番号の接頭辞。

`\@secpos` 節番号の接尾辞。

```
2406 \ifnum\bxjs@label@section=\bxjs@label@section@@compat\else
2407 \def\@secapp{\presectionname}
2408 \def\@secpos{\postsectionname}
2409 \fi
```

`\labelsection` 節番号の出力書式。

```
2410 \ifnum\bxjs@label@section=\bxjs@label@section@@modern
2411 \def\labelsection{\@secapp\thesection\@secpos}
2412 \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　目次や柱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。

```
2413 \def\@startsection#1#2#3#4#5#6{%
2414   \if@noskipsec \leavevmode \fi
2415   \par
```

```
% 見出し上の空きを \@tempskipa にセットする
  \@tempskipa #4\relax
% \@afterindent は見出し直後の段落を字下げするかどうかを表すスイッチ
  \if@english \@afterindentfalse \else \@afterindenttrue \fi
% 見出し上の空きが負なら見出し直後の段落を字下げしない
  \ifdim \@tempskipa <\z@
    \@tempskipa -\@tempskipa \@afterindentfalse
  \fi
  \if@nobreak
%   \everypar{\everyparhook}% これは間違い
    \everypar{}%
  \else
    \addpenalty\@secpenalty
% 次の行は削除
%   \addvspace\@tempskipa
% 次の \noindent まで追加
    \ifdim \@tempskipa >\z@
      \if@slide\else
        \null
        \vspace*{-\baselineskip}%
      \fi
      \vskip\@tempskipa
    \fi
  \fi
  \noindent
% 追加終わり
  \@ifstar
    {\@ssect{#3}{#4}{#5}{#6}}%
    {\@dblarg{\@sect{#1}{#2}{#3}{#4}{#5}{#6}}}}
```

\@sect と \@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 も挿入しています。

\everyparhook の挿入は everyparhook=compat の時のみ行う。

\bxjs@if@ceph `\bxjs@if@ceph{⟨コード⟩}`： everyparhook=compat である場合にのみ ⟨コード⟩ を実行する。

```
\ifnum\bxjs@everyparhook=\bxjs@everyparhook@@compat
  \let\bxjs@if@ceph\@firstofone
\else \let\bxjs@if@ceph\@gobble
\fi
```

```
\def\@sect#1#2#3#4#5#6[#7]#8{%
  \ifnum #2>\c@secnumdepth
    \let\@svsec\@empty
  \else
    \refstepcounter{#1}%
    \protected@edef\@svsec{\@seccntformat{#1}\relax}%
```

```
  \fi
% 見出し後の空きを \@tempskipa にセット
  \@tempskipa #5\relax
% 条件判断の順序を入れ換えました
  \ifdim \@tempskipa<\z@
    \def\@svsechd{%
      #6{\hskip #3\relax
      \@svsec #8}%
      \csname #1mark\endcsname{#7}%
      \addcontentsline{toc}{#1}{%
        \ifnum #2>\c@secnumdepth \else
          \protect\numberline{\bxjs@label@sect{#1}}%
        \fi
        #7}}% 目次にフルネームを載せるなら #8
  \else
    \begingroup
      \interlinepenalty \@M % 下から移動
      #6{%
        \@hangfrom{\hskip #3\relax\@svsec}%
%       \interlinepenalty \@M % 上に移動
        #8\@@par}%
    \endgroup
    \csname #1mark\endcsname{#7}%
    \addcontentsline{toc}{#1}{%
      \ifnum #2>\c@secnumdepth \else
        \protect\numberline{\bxjs@label@sect{#1}}%
      \fi
      #7}% 目次にフルネームを載せるならここは #8
  \fi
  \@xsect{#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。

```
\def\@xsect#1{%
% 見出しの後ろの空きを \@tempskipa にセット
  \@tempskipa #1\relax
% 条件判断の順序を変えました
  \ifdim \@tempskipa<\z@
    \@nobreakfalse
    \global\@noskipsectrue
    \everypar{%
      \if@noskipsec
        \global\@noskipsecfalse
       {\setbox\z@\lastbox}%
        \clubpenalty\@M
        \begingroup \@svsechd \endgroup
```

```
        \unskip
        \@tempskipa #1\relax
        \hskip -\@tempskipa
      \else
        \clubpenalty \@clubpenalty
        \everypar\expandafter{\bxjs@if@ceph\everyparhook}%
```

TODO: ↑ナニコレ？

```
      \fi\bxjs@if@ceph\everyparhook}%
  \else
    \par \nobreak
    \vskip \@tempskipa
    \@afterheading
  \fi
  \if@slide
    {\vskip\if@twocolumn-5\jsc@mpt\else-6\jsc@mpt\fi
     \maybeblue\hrule height0\jsc@mpt depth1\jsc@mpt
     \vskip\if@twocolumn 4\jsc@mpt\else 7\jsc@mpt\fi\relax}%
  \fi
  \par  % 2000-12-18
  \ignorespaces}
\def\@ssect#1#2#3#4#5{%
  \@tempskipa #3\relax
  \ifdim \@tempskipa<\z@
    \def\@svsechd{#4{\hskip #1\relax #5}}%
  \else
    \begingroup
      #4{%
        \@hangfrom{\hskip #1}%
          \interlinepenalty \@M #5\@@par}%
    \endgroup
  \fi
  \@xsect{#3}}
```

■柱関係の命令

\chaptermark
\sectionmark
\subsectionmark
\subsubsectionmark
\paragraphmark
\subparagraphmark

\...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。

```
\newcommand*\chaptermark[1]{}
% \newcommand*{\sectionmark}[1]{}
% \newcommand*{\subsectionmark}[1]{}
% \newcommand*{\subsubsectionmark}[1]{}
% \newcommand*{\paragraphmark}[1]{}
% \newcommand*{\subparagraphmark}[1]{}
```

■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。

```
%<!book&!report>\setcounter{secnumdepth}{3}
%<book|report>\setcounter{secnumdepth}{2}
```

\c@chapter
\c@section
\c@subsection
\c@subsubsection
\c@paragraph
\c@subparagraph

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。

```
\newcounter{part}
%<book|report>\newcounter{chapter}
%<book|report>\newcounter{section}[chapter]
%<!book&!report>\newcounter{section}
\newcounter{subsection}[section]
\newcounter{subsubsection}[subsection]
\newcounter{paragraph}[subsubsection]
\newcounter{subparagraph}[paragraph]
```

\thepart
\thechapter
\thesection
\thesubsection
\thesubsubsection
\theparagraph
\thesubparagraph

カウンタの値を出力する命令 \the **何々** を定義します。

カウンタを出力するコマンドには次のものがあります。

\arabic{COUNTER}	1, 2, 3, …
\roman{COUNTER}	i, ii, iii, …
\Roman{COUNTER}	I, II, III, …
\alph{COUNTER}	a, b, c, …
\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。

```
\renewcommand{\thepart}{\@Roman\c@part}
%<*!book&!report>
\ifnum\bxjs@label@section=\bxjs@label@section@@compat
\renewcommand{\thesection}{\presectionname\@arabic\c@section\postsectionname}
\renewcommand{\thesubsection}{\@arabic\c@section.\@arabic\c@subsection}
\else
\renewcommand{\thesection}{\@arabic\c@section}
\renewcommand{\thesubsection}{\thesection.\@arabic\c@subsection}
\fi
%</!book&!report>
%<*book|report>
\renewcommand{\thechapter}{\@arabic\c@chapter}
\renewcommand{\thesection}{\thechapter.\@arabic\c@section}
\renewcommand{\thesubsection}{\thesection.\@arabic\c@subsection}
%</book|report>
\renewcommand{\thesubsubsection}{%
    \thesubsection.\@arabic\c@subsubsection}
\renewcommand{\theparagraph}{%
    \thesubsubsection.\@arabic\c@paragraph}
\renewcommand{\thesubparagraph}{%
    \theparagraph.\@arabic\c@subparagraph}
```

\@chapapp
\@chappos

\@chapapp の初期値は \prechaptername（第）です。

\@chappos の初期値は \postchaptername（章）です。

\appendix は \@chapapp を \appendixname に，\@chappos を空に再定義します。

[2003-03-02] \@secapp は外しました。

```
%<book|report>\newcommand{\@chapapp}{\prechaptername}
%<book|report>\newcommand{\@chappos}{\postchaptername}
```

■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 \mainmatter の 2 つの命令は，改丁または改ページした後で \pagenumbering{...} でノンブルを 1 にリセットします。長い間 \frontmatter は openany のときに単なる改ページとしていましたが，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。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 pLaTeX の標準クラス 2017/03/05 も参照）。

```
%<*book|report>
\newcommand\frontmatter{%
  \pltx@cleartooddpage
  \@mainmatterfalse
  \pagenumbering{roman}}
```

\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。

```
\newcommand\mainmatter{%
  \pltx@cleartooddpage
  \@mainmattertrue
  \pagenumbering{arabic}}
```

\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。

```
\newcommand\backmatter{%
  \if@openleft
    \cleardoublepage
  \else\if@openright
    \cleardoublepage
  \else
    \clearpage
  \fi\fi
  \@mainmatterfalse}
%</book|report>
```

■部

\part 新しい部を始めます。

\secdef 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。

\secdef{**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。

\secdef は次のようにして使います。

```
\def\chapter { ... \secdef \CMDA \CMDB }
\def\CMDA    [#1]#2{....} % \chapter[...]{...} の定義
\def\CMDB    #1{....}     % \chapter*{...} の定義
```

まず book と report のクラス以外です。

```
%<*!book&!report>
\newcommand\part{%
  \if@noskipsec \leavevmode \fi
  \par
  \addvspace{4ex}%
  \if@english \@afterindentfalse \else \@afterindenttrue \fi
  \secdef\@part\@spart}
%</!book&!report>
```

book および report クラスの場合は，少し複雑です。

```
%<*book|report>
\newcommand\part{%
  \if@openleft
    \cleardoublepage
  \else\if@openright
    \cleardoublepage
  \else
    \clearpage
  \fi\fi
  \thispagestyle{empty}% 欧文用標準スタイルでは plain
  \if@twocolumn
    \onecolumn
    \@restonecoltrue
  \else
    \@restonecolfalse
  \fi
  \null\vfil
  \secdef\@part\@spart}
%</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ます。

```
%<*!book&!report>
\def\@part[#1]#2{%
  \ifnum \c@secnumdepth >\m@ne
    \refstepcounter{part}%
    \addcontentsline{toc}{part}{%
      \prepartname\thepart\postpartname\hspace{1\jsZw}#1}%
  \else
```

```
    \addcontentsline{toc}{part}{#1}%
  \fi
  \markboth{}{}%
  {\parindent\z@
    \raggedright
    \interlinepenalty \@M
    \normalfont
    \ifnum \c@secnumdepth >\m@ne
      \Large\headfont\prepartname\thepart\postpartname
      \par\nobreak
    \fi
    \huge \headfont #2%
    \markboth{}{}\par}%
  \nobreak
  \vskip 3ex
  \@afterheading}
%</!book&!report>
```

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。

```
%<*book|report>
\def\@part[#1]#2{%
  \ifnum \c@secnumdepth >-2\relax
    \refstepcounter{part}%
    \addcontentsline{toc}{part}{%
      \prepartname\thepart\postpartname\hspace{1\jsZw}#1}%
  \else
    \addcontentsline{toc}{part}{#1}%
  \fi
  \markboth{}{}%
  {\centering
    \interlinepenalty \@M
    \normalfont
    \ifnum \c@secnumdepth >-2\relax
      \huge\headfont \prepartname\thepart\postpartname
      \par\vskip20\jsc@mpt
    \fi
    \Huge \headfont #2\par}%
  \@endpart}
%</book|report>
```

\@spart 番号を付けない部です。

```
%<*!book&!report>
\def\@spart#1{{%
    \parindent \z@ \raggedright
    \interlinepenalty \@M
    \normalfont
    \huge \headfont #1\par}%
  \nobreak
  \vskip 3ex
```

```
  \@afterheading}
%</!book&!report>
%<*book|report>
\def\@spart#1{{%
    \centering
    \interlinepenalty \@M
    \normalfont
    \Huge \headfont #1\par}%
  \@endpart}
%</book|report>
```

\@endpart \@part と \@spart 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。

```
%<*book|report>
\def\@endpart{\vfil\newpage
  \if@twoside
   \if@openleft %% added (2017/02/24)
    \null\thispagestyle{empty}\newpage
   \else\if@openright %% added (2016/12/13)
    \null\thispagestyle{empty}\newpage
   \fi\fi %% added (2016/12/13, 2017/02/24)
  \fi
  \if@restonecol
    \twocolumn
  \fi}
%</book|report>
```

■章

\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。

```
%<*book|report>
\newcommand{\chapter}{%
  \if@openleft\cleardoublepage\else
  \if@openright\cleardoublepage\else\clearpage\fi\fi
  \plainifnotempty % 元: \thispagestyle{plain}
  \global\@topnum\z@
  \if@english \@afterindentfalse \else \@afterindenttrue \fi
  \secdef
    {\@omit@numberfalse\@chapter}%
    {\@omit@numbertrue\@schapter}}
```

\@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。

```
\def\@chapter[#1]#2{%
  \ifnum \c@secnumdepth >\m@ne
    \if@mainmatter
      \refstepcounter{chapter}%
      \typeout{\@chapapp\thechapter\@chappos}%
      \addcontentsline{toc}{chapter}%
        {\protect\numberline
%       %{\if@english\thechapter\else\@chapapp\thechapter\@chappos\fi}%
        {\@chapapp\thechapter\@chappos}%
        #1}%
    \else\addcontentsline{toc}{chapter}{#1}\fi
  \else
    \addcontentsline{toc}{chapter}{#1}%
  \fi
  \chaptermark{#1}%
  \addtocontents{lof}{\protect\addvspace{10\jsc@mpt}}%
  \addtocontents{lot}{\protect\addvspace{10\jsc@mpt}}%
  \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\Cvs}% 欧文は 50pt
  {\parindent \z@ \raggedright \normalfont
    \ifnum \c@secnumdepth >\m@ne
      \if@mainmatter
        \huge\headfont \@chapapp\thechapter\@chappos
        \par\nobreak
        \vskip \Cvs % 欧文は 20pt
      \fi
    \fi
    \interlinepenalty\@M
    \Huge \headfont #1\par\nobreak
    \vskip 3\Cvs}} % 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{%
  \chaptermark{#1}%
  \if@twocolumn
    \@topnewpage[\@makeschapterhead{#1}]%
  \else
    \@makeschapterhead{#1}\@afterheading
  \fi}
```

\@makeschapterhead 番号なしの章見出しです。

```
\def\@makeschapterhead#1{%
  \vspace*{2\Cvs}% 欧文は 50pt
  {\parindent \z@ \raggedright
    \normalfont
    \interlinepenalty\@M
    \Huge \headfont #1\par\nobreak
    \vskip 3\Cvs}} % 欧文は 40pt
%</book|report>
```

■下位レベルの見出し

\section 欧文版では \@startsection の第 4 引数を負にして最初の段落の字下げを禁止していますが，和文版では正にして字下げするようにしています。

段組のときはなるべく左右の段が狂わないように工夫しています。

```
\if@twocolumn
  \newcommand{\section}{%
%<jspf>\ifx\maketitle\relax\else\maketitle\fi
    \@startsection{section}{1}{\z@}%
%<!kiyou>    {0.6\Cvs}{0.4\Cvs}%
%<kiyou>    {\Cvs}{0.5\Cvs}%
%   {\normalfont\large\headfont\@secapp}}
    {\normalfont\large\headfont\raggedright}}
\else
  \newcommand{\section}{%
    \if@slide\clearpage\fi
    \@startsection{section}{1}{\z@}%
    {\Cvs \@plus.5\Cdp \@minus.2\Cdp}% 前アキ
    {.5\Cvs \@plus.3\Cdp}% 後アキ
%   {\normalfont\Large\headfont\@secapp}}
    {\normalfont\Large\headfont\raggedright}}
\fi
```

\subsection 同上です。

```
\if@twocolumn
  \newcommand{\subsection}{\@startsection{subsection}{2}{\z@}%
    {\z@}{\if@slide .4\Cvs \else \z@ \fi}%
    {\normalfont\normalsize\headfont}}
\else
  \newcommand{\subsection}{\@startsection{subsection}{2}{\z@}%
    {\Cvs \@plus.5\Cdp \@minus.2\Cdp}% 前アキ
    {.5\Cvs \@plus.3\Cdp}% 後アキ
    {\normalfont\large\headfont}}
\fi
```

\subsubsection [2016-07-22] slide オプション指定時に \subsubsection の文字列と罫線が重なる問題に対処しました (forum:1982)。

```
\if@twocolumn
  \newcommand{\subsubsection}{\@startsection{subsubsection}{3}{\z@}%
```

```
    {\z@}{\if@slide .4\Cvs \else \z@ \fi}%
    {\normalfont\normalsize\headfont}}
\else
  \newcommand{\subsubsection}{\@startsection{subsubsection}{3}{\z@}%
    {\Cvs \@plus.5\Cdp \@minus.2\Cdp}%
    {\if@slide .5\Cvs \@plus.3\Cdp \else \z@ \fi}%
    {\normalfont\normalsize\headfont}}
\fi
```

\paragraph 見出しの後ろで改行されません。

\jsParagraphMark [2016-11-16] 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きません。

※ 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。

\jsJaChar standard 和文ドライバが読み込まれた場合は \jachar と同義で、それ以外は何もしない。

```
\let\jsJaChar\@empty
```

```
\newcommand\jsParagraphMark{\relax\jsJaChar{■}}
\let\bxjs@org@paragraph@mark\jsParagraphMark
\ifx\bxjs@paragraph@mark\@empty
  \let\jsParagraphMark\@empty
\else\ifx\bxjs@paragraph@mark\@undefined\else
  \bxjs@nclong\edef\jsParagraphMark{\noexpand\jsJaChar{\bxjs@paragraph@mark}}
\fi\fi
\if@twocolumn
  \newcommand{\paragraph}{\@startsection{paragraph}{4}{\z@}%
    {\z@}{\if@slide .4\Cvs \else -1\jsZw\fi}% 改行せず 1\jsZw のアキ
%<jspf>    {\normalfont\normalsize\headfont}}
%<!jspf>    {\normalfont\normalsize\headfont\jsParagraphMark}}
\else
  \newcommand{\paragraph}{\@startsection{paragraph}{4}{\z@}%
    {0.5\Cvs \@plus.5\Cdp \@minus.2\Cdp}%
    {\if@slide .5\Cvs \@plus.3\Cdp \else -1\jsZw\fi}% 改行せず 1\jsZw のアキ
%<jspf>    {\normalfont\normalsize\headfont}}
%<!jspf>    {\normalfont\normalsize\headfont\jsParagraphMark}}
\fi
```

\subparagraph 見出しの後ろで改行されません。

```
\if@twocolumn
```

```
  \newcommand{\subparagraph}{\@startsection{subparagraph}{5}{\z@}%
    {\z@}{\if@slide .4\Cvs \@plus.3\Cdp \else -1\jsZw\fi}%
    {\normalfont\normalsize\headfont}}
\else
  \newcommand{\subparagraph}{\@startsection{subparagraph}{5}{\z@}%
    {\z@}{\if@slide .5\Cvs \@plus.3\Cdp \else -1\jsZw\fi}%
    {\normalfont\normalsize\headfont}}
\fi
```

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。

```
\if@slide
  \setlength\leftmargini{1\jsZw}
\else
  \if@twocolumn
    \setlength\leftmargini{2\jsZw}
  \else
    \setlength\leftmargini{3\jsZw}
  \fi
\fi
```

\leftmarginii \leftmarginiii \leftmarginiv \leftmarginv \leftmarginvi ii，iii，iv は \labelsep とそれぞれ ‘(m)’，‘vii.’，‘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。

```
\if@slide
  \setlength\leftmarginii {1\jsZw}
  \setlength\leftmarginiii{1\jsZw}
  \setlength\leftmarginiv {1\jsZw}
  \setlength\leftmarginv  {1\jsZw}
  \setlength\leftmarginvi {1\jsZw}
\else
  \setlength\leftmarginii {2\jsZw}
  \setlength\leftmarginiii{2\jsZw}
  \setlength\leftmarginiv {2\jsZw}
  \setlength\leftmarginv  {1\jsZw}
  \setlength\leftmarginvi {1\jsZw}
\fi
```

\labelsep \labelwidth \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

```
\@beginparpenalty -\@lowpenalty
\@endparpenalty   -\@lowpenalty
\@itempenalty     -\@lowpenalty
```

\@listi \@listi は \leftmargin，\parsep，\topsep，\itemsep などのトップレベルの定義をします。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しまいました。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す。アスキー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。

\@listI

[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 0.5\baselineskip
  \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

\@listiv

\@listv

\@listvi

```
\def\@listii{\leftmargin\leftmarginii
  \labelwidth\leftmarginii \advance\labelwidth-\labelsep
  \topsep \z@
  \parsep \z@
  \itemsep\parsep}
\def\@listiii{\leftmargin\leftmarginiii
  \labelwidth\leftmarginiii \advance\labelwidth-\labelsep
  \topsep \z@
  \parsep \z@
  \itemsep\parsep}
\def\@listiv {\leftmargin\leftmarginiv
              \labelwidth\leftmarginiv
              \advance\labelwidth-\labelsep}
\def\@listv  {\leftmargin\leftmarginv
              \labelwidth\leftmarginv
              \advance\labelwidth-\labelsep}
\def\@listvi {\leftmargin\leftmarginvi
```

```
2868                \labelwidth\leftmarginvi
2869                \advance\labelwidth-\labelsep}
```

■enumerate 環境　enumerate 環境はカウンタ enumi，enumii，enumiii，enumiv を使います。enum$n$ は第 n レベルの番号です。

\theenumi
\theenumii
\theenumiii
\theenumiv

出力する番号の書式を設定します。これらは LaTeX 本体（ltlists.dtx 参照）で定義済みですが，ここでは表し方を変えています。\@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。

```
2870 \renewcommand{\theenumi}{\@arabic\c@enumi}
2871 \renewcommand{\theenumii}{\@alph\c@enumii}
2872 \renewcommand{\theenumiii}{\@roman\c@enumiii}
2873 \renewcommand{\theenumiv}{\@Alph\c@enumiv}
```

\labelenumi
\labelenumii
\labelenumiii
\labelenumiv

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欧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。

和文の括弧で囲むための補助命令 \jsInJaParen を定義して \labelenumii でそれを用いている。

```
2874 \newcommand*{\jsInJaParen}[1]{%
2875   \mbox{\jsInhibitGlue（#1）\jsInhibitGlue}}
2876 \newcommand{\labelenumi}{\theenumi.}
2877 \newcommand{\labelenumii}{\jsInJaParen{\theenumii}}
2878 \newcommand{\labelenumiii}{\theenumiii.}
2879 \newcommand{\labelenumiv}{\theenumiv.}
```

\p@enumii
\p@enumiii
\p@enumiv

\p@enum$n$ 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にしました。

```
2880 \renewcommand{\p@enumii}{\theenumi}
2881 \renewcommand{\p@enumiii}{\theenumi\jsInhibitGlue（\theenumii ）}
2882 \renewcommand{\p@enumiv}{\p@enumiii\theenumiii}
```

■itemize 環境

\labelitemi
\labelitemii
\labelitemiii
\labelitemiv

itemize 環境の第 n レベルのラベルを作るコマンドです。

```
2883 \newcommand\labelitemi{\textbullet}
2884 \newcommand\labelitemii{\normalfont\bfseries \textendash}
2885 \newcommand\labelitemiii{\textasteriskcentered}
2886 \newcommand\labelitemiv{\textperiodcentered}
```

■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。

```
\newenvironment{description}{%
  \list{}{%
    \labelwidth=\leftmargin
    \labelsep=1\jsZw
    \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\jsZw}）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 環境で作られていましたが，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：

- 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。
- 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。

```
%<*book|report>
\newenvironment{chapterabstract}{%
  \begin{list}{}{%
    \listparindent=1\jsZw
    \itemindent=\listparindent
    \rightmargin=0pt
    \leftmargin=5\jsZw}\item[]}{\end{list}\vspace{\baselineskip}}
%</book|report>
```

“普通の”abstract 環境の定義。

```
%<*article|report|slide>
\newbox\@abstractbox
\if@titlepage
  \newenvironment{abstract}{%
    \titlepage
    \null\vfil
```

```
    \@beginparpenalty\@lowpenalty
    \begin{center}%
      \headfont \abstractname
      \@endparpenalty\@M
    \end{center}%
```

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。

```
    \par}%
  {\par\vfil\null\endtitlepage}
\else
  \newenvironment{abstract}{%
    \if@twocolumn
      \ifx\maketitle\relax
        \section*{\abstractname}%
      \else
        \global\setbox\@abstractbox\hbox\bgroup
        \begin{minipage}[b]{\textwidth}
          \small\parindent1\jsZw
          \begin{center}%
            {\headfont \abstractname\vspace{-.5em}\vspace{\z@}}%
          \end{center}%
          \list{}{%
            \listparindent\parindent
            \itemindent \listparindent
            \rightmargin \leftmargin}%
          \item\relax
      \fi
    \else
      \small
      \begin{center}%
        {\headfont \abstractname\vspace{-.5em}\vspace{\z@}}%
      \end{center}%
      \list{}{%
        \listparindent\parindent
        \itemindent \listparindent
        \rightmargin \leftmargin}%
      \item\relax
    \fi}{\if@twocolumn
      \ifx\maketitle\relax
      \else
        \endlist\end{minipage}\egroup
      \fi
    \else
      \endlist
    \fi}
\fi
%</article|report|slide>
%<*jspf>
\newbox\@abstractbox
```

```
\newenvironment{abstract}{%
  \global\setbox\@abstractbox\hbox\bgroup
  \begin{minipage}[b]{157\jsc@mmm}{\sffamily Abstract}\par
    \small
    \if@english \parindent6\jsc@mmm \else \parindent1\jsZw \fi}%
  {\end{minipage}\egroup}
%</jspf>
```

bxjs@force@chapterabstract が真の場合は、abstract 環境を chapterabstract 環境と等価にする。

```
%<*book|report>
\ifbxjs@force@chapterabstract
  \let\abstract\chapterabstract
  \let\endabstract\endchapterabstract
\fi
%</book|report>
```

■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。

```
%<*jspf>
%\newbox\@keywordsbox
%\newenvironment{keywords}{%
%  \global\setbox\@keywordsbox\hbox\bgroup
%  \begin{minipage}[b]{1570\jsc@mmm}{\sffamily Keywords:}\par
%    \small\parindent0\jsZw}%
%  {\end{minipage}\egroup}
%</jspf>
```

■verse 環境

verse (*env.*) 詩のための verse 環境です。

```
\newenvironment{verse}{%
  \let \\=\@centercr
  \list{}{%
    \itemsep \z@
    \itemindent -2\jsZw % 元: -1.5em
    \listparindent\itemindent
    \rightmargin \z@
    \advance\leftmargin 2\jsZw}% 元: 1.5em
  \item\relax}{\endlist}
```

■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。

```
\newenvironment{quotation}{%
  \list{}{%
    \listparindent\parindent
    \itemindent\listparindent
    \rightmargin \z@}%
  \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。

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

■定理など ltthm.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，これでは和文がゴシック体になってしまうので，\itshape を削除しました。

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

```
\def\@begintheorem#1#2{\trivlist\labelsep=1\jsZw
   \item[\hskip \labelsep{\headfont #1\ #2}]}
\def\@opargbegintheorem#1#2#3{\trivlist\labelsep=1\jsZw
      \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 に合わせましたが，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。

```
\newenvironment{titlepage}{%
%<book>    \pltx@cleartooddpage %% 2017-02-24
    \if@twocolumn
      \@restonecoltrue\onecolumn
    \else
      \@restonecolfalse\newpage
    \fi
    \thispagestyle{empty}%
    \ifodd\c@page\setcounter{page}\@ne\else\setcounter{page}\z@\fi %% 2017-02-24
  }%
```

```
  {\if@restonecol\twocolumn \else \newpage \fi
    \if@twoside\else
      \setcounter{page}\@ne
    \fi}
```

■付録

\appendix 本文と付録を分離するコマンドです。

```
%<*!book&!report>
\newcommand{\appendix}{\par
  \setcounter{section}{0}%
  \setcounter{subsection}{0}%
  \ifnum\bxjs@label@section=\bxjs@label@section@@compat
  \gdef\presectionname{\appendixname}%
  \gdef\postsectionname{}%
% \gdef\thesection{\@Alph\c@section}% [2003-03-02]
  \gdef\thesection{\presectionname\@Alph\c@section\postsectionname}%
  \gdef\thesubsection{\@Alph\c@section.\@arabic\c@subsection}%
  \else
  \gdef\@secapp{\appendixname}%
  \gdef\@secpos{}%
  \gdef\thesection{\@Alph\c@section}%
  \fi}
%</!book&!report>
%<*book|report>
\newcommand{\appendix}{\par
  \setcounter{chapter}{0}%
  \setcounter{section}{0}%
  \gdef\@chapapp{\appendixname}%
  \gdef\@chappos{}%
  \gdef\thechapter{\@Alph\c@chapter}}
%</book|report>
```

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\jsc@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\jsc@mpt}
```

\arrayrulewidth array，tabular 環境内の罫線の幅です。

```
\setlength\arrayrulewidth{.4\p@}
```

\doublerulesep array，tabular 環境での二重罫線間のアキです。

```
\setlength\doublerulesep{2\p@}
```

■tabbing 環境

\tabbingsep \' コマンドで入るアキです。

```
3039 \setlength\tabbingsep{\labelsep}
```

■minipage 環境

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

```
3040 \skip\@mpfootins = \skip\footins
```

■framebox 環境

\fboxsep \fbox, \framebox で内側のテキストと枠との間の空きです。

\fboxrule \fbox, \framebox の罫線の幅です。

```
3041 \setlength\fboxsep{3\jsc@mpt}
3042 \setlength\fboxrule{.4\p@}
```

■equation と eqnarray 環境

\theequation 数式番号を出力するコマンドです。

```
3043 %<!book&!report>\renewcommand \theequation {\@arabic\c@equation}
3044 %<*book|report>
3045 \@addtoreset{equation}{chapter}
3046 \renewcommand\theequation
3047   {\ifnum \c@chapter>\z@ \thechapter.\fi \@arabic\c@equation}
3048 %</book|report>
```

\jot 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。

```
3049 % \setlength\jot{3pt}
```

\@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。

\jsInhibitGlue（\theequation ）\jsInhibitGlue のように和文かっこを使うことも可能です。

```
3050 % \def\@eqnnum{(\theequation)}
```

amsmath パッケージを使う場合は \tagform@ を次のように修正します。

```
3051 % \def\tagform@#1{\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⟨num⟩⟨text⟩　キャプションを出力するマクロです。⟨*num*⟩ は \fnum@... の生成する番号，⟨*text*⟩ 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。

```
%<*!book&!report>
\newcounter{figure}
\renewcommand \thefigure {\@arabic\c@figure}
%</!book&!report>
%<*book|report>
\newcounter{figure}[chapter]
\renewcommand \thefigure
     {\ifnum \c@chapter>\z@ \thechapter.\fi \@arabic\c@figure}
%</book|report>
```

\fps@figure
\ftype@figure
\ext@figure
\fnum@figure

figure のパラメータです。\figurename の直後に ~ が入っていましたが，ここでは外しました。

```
\def\fps@figure{tbp}
\def\ftype@figure{1}
\def\ext@figure{lof}
\def\fnum@figure{\figurename\nobreak\thefigure}
```

figure (*env.*)
figure* (*env.*)

* 形式は段抜きのフロートです。

```
\newenvironment{figure}%
               {\@float{figure}}%
               {\end@float}
\newenvironment{figure*}%
               {\@dblfloat{figure}}%
               {\end@dblfloat}
```

■table 環境

\c@table
\thetable

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 \thechapter. が \thechapter{}・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。

```
%<*!book&!report>
\newcounter{table}
\renewcommand\thetable{\@arabic\c@table}
%</!book&!report>
%<*book|report>
\newcounter{table}[chapter]
```

```
\renewcommand \thetable
     {\ifnum \c@chapter>\z@ \thechapter.\fi \@arabic\c@table}
%</book|report>
```

\fps@table
\ftype@table
\ext@table
\fnum@table

table のパラメータです。\tablename の直後に ~ が入っていましたが，ここでは外しました。

```
\def\fps@table{tbp}
\def\ftype@table{2}
\def\ext@table{lot}
\def\fnum@table{\tablename\nobreak\thetable}
```

table (*env.*)
table* (*env.*)

* は段抜きのフロートです。

```
\newenvironment{table}%
               {\@float{table}}%
               {\end@float}
\newenvironment{table*}%
               {\@dblfloat{table}}%
               {\end@dblfloat}
```

8.6 キャプション

\@makecaption

\caption 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
\belowcaptionskip

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 になっていましたので，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。

```
\newlength\abovecaptionskip
\newlength\belowcaptionskip
\setlength\abovecaptionskip{5\jsc@mpt} % 元: 10\p@
\setlength\belowcaptionskip{5\jsc@mpt} % 元: 0\p@
```

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 \small 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，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) に対処しました。

```
%<*!jspf>
% \long\def\@makecaption#1#2{{\small
%   \advance\leftskip10\jsc@mmm
%   \advance\rightskip10\jsc@mmm
%   \vskip\abovecaptionskip
%   \sbox\@tempboxa{#1\hskip1\jsZw\relax #2}%
%   \ifdim \wd\@tempboxa >\hsize
%     #1\hskip1\jsZw\relax #2\par
%   \else
```

```
%     \global \@minipagefalse
%     \hb@xt@\hsize{\hfil\box\@tempboxa\hfil}%
%   \fi
%   \vskip\belowcaptionskip}}
\long\def\@makecaption#1#2{{\small
  \advance\leftskip .0628\linewidth
  \advance\rightskip .0628\linewidth
  \vskip\abovecaptionskip
  \sbox\@tempboxa{#1\zwspace#2}%
  \ifdim \wd\@tempboxa <\hsize \centering \fi
  #1\zwspace#2\par
  \vskip\belowcaptionskip}}
%</!jspf>
%<*jspf>
\long\def\@makecaption#1#2{%
  \vskip\abovecaptionskip
  \sbox\@tempboxa{\small\sffamily #1\quad #2}%
  \ifdim \wd\@tempboxa >\hsize
    {\small\sffamily
      \list{#1}{%
        \renewcommand{\makelabel}[1]{##1\hfil}
        \itemsep    \z@
        \itemindent \z@
        \labelsep   \z@
        \labelwidth 11\jsc@mmm
        \listparindent\z@
        \leftmargin 11\jsc@mmm}\item\relax #2\endlist}
  \else
    \global \@minipagefalse
    \hb@xt@\hsize{\hfil\box\@tempboxa\hfil}%
  \fi
  \vskip\belowcaptionskip}
%</jspf>
```

9 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。

[2016-07-16] 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`

`\if@jsc@warnoldfontcmd` は 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
```

\jsc@DeclareOldFontCommand

```
\newcommand*{\jsc@DeclareOldFontCommand}[3]{%
  \g@addto@macro\bxjs@oldfontcmd@list{\do#1}%
  \DeclareOldFontCommand{#1}{%
    \bxjs@oldfontcmd{#1}#2%
  }{%
    \bxjs@oldfontcmd{#1}#3%
  }%
}
\DeclareRobustCommand*{\jsc@warnoldfontcmd}[1]{%
  \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@warnoldfontcmdexception\else
    \global\@jsc@warnoldfontcmdfalse
    \ifx#1\relax
      \global\let#1=t%
      \jsc@warnoldfontcmd{#2}%
    \fi
  \fi}
\def\bxjs@warnoldfontcmd@final{%
% \par
  \global\let\bxjs@warnoldfontcmd@final\@empty
  \let\@tempa\@empty
  \def\do##1{%
```

```
    \@ifundefined{bxjs@ofc/\string##1}{}{%else
      \edef\@tempa{\@tempa \space\string##1}}}%
  \bxjs@oldfontcmd@list
  \ifx\@tempa\@empty\else
    \ClassWarningNoLine\bxjs@clsname
     {Some old font commands were used in text:\MessageBreak
      \space\@tempa\MessageBreak
      You should note, that since 1994 LaTeX2e provides a\MessageBreak
      new font selection scheme called NFSS2 with several\MessageBreak
      new, combinable font commands. The
      class provides\MessageBreak
      the old font commands only for compatibility}
  \fi}
```

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。

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

\mc フォントファミリを変更します。

\gt
\rm
\sf
\tt

```
\jsc@DeclareOldFontCommand{\mc}{\normalfont\mcfamily}{\mathmc}
\jsc@DeclareOldFontCommand{\gt}{\normalfont\gtfamily}{\mathgt}
\jsc@DeclareOldFontCommand{\rm}{\normalfont\rmfamily}{\mathrm}
\jsc@DeclareOldFontCommand{\sf}{\normalfont\sffamily}{\mathsf}
\jsc@DeclareOldFontCommand{\tt}{\normalfont\ttfamily}{\mathtt}
```

\bf ボールドシリーズにします。通常のミーディアムシリーズに戻すコマンドは \mdseries です。

```
\jsc@DeclareOldFontCommand{\bf}{\normalfont\bfseries}{\mathbf}
```

\it フォントシェイプを変えるコマンドです。斜体とスモールキャップスは数式中では何もしません（警告メッセージを出力します）。通常のアップライト体に戻すコマンドは \upshape です。

\sl
\sc

```
\jsc@DeclareOldFontCommand{\it}{\normalfont\itshape}{\mathit}
\jsc@DeclareOldFontCommand{\sl}{\normalfont\slshape}{\@nomath\sl}
\jsc@DeclareOldFontCommand{\sc}{\normalfont\scshape}{\@nomath\sc}
```

\cal 数式モード以外では何もしません（警告を出します）。

\mit

```
\DeclareRobustCommand*{\cal}{\@fontswitch\relax\mathcal}
\DeclareRobustCommand*{\mit}{\@fontswitch\relax\mathnormal}
```

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}{\numberline{**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{...} は \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。これらの多くは \@dottedtocline コマンドを使って定義します。これは

\@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 でしたが，ここでは一つずつ減らしています。

```
\newcommand\@pnumwidth{1.55em}
\newcommand\@tocrmarg{2.55em}
\newcommand\@dotsep{4.5}
%<!book&!report>\setcounter{tocdepth}{2}
%<book|report>\setcounter{tocdepth}{1}
```

■目次

\tableofcontents 目次を生成します。

\jsc@tocl@width [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\jsc@tocl@width
\newcommand{\tableofcontents}{%
%<*book|report>
  \settowidth\jsc@tocl@width{\headfont\prechaptername\postchaptername}%
  \settowidth\@tempdima{\headfont\appendixname}%
  \ifdim\jsc@tocl@width<\@tempdima \setlength\jsc@tocl@width{\@tempdima}\fi
  \ifdim\jsc@tocl@width<2\jsZw \divide\jsc@tocl@width by 2 \advance\jsc@tocl@width 1\jsZw\fi
  \if@twocolumn
    \@restonecoltrue\onecolumn
  \else
    \@restonecolfalse
  \fi
  \chapter*{\contentsname}%
  \@mkboth{\contentsname}{}%
%</book|report>
%<*!book&!report>
  \settowidth\jsc@tocl@width{\headfont\presectionname\postsectionname}%
  \settowidth\@tempdima{\headfont\appendixname}%
  \ifdim\jsc@tocl@width<\@tempdima\relax\setlength\jsc@tocl@width{\@tempdima}\fi
  \ifdim\jsc@tocl@width<2\jsZw \divide\jsc@tocl@width by 2 \advance\jsc@tocl@width 1\jsZw\fi
  \section*{\contentsname}%
  \@mkboth{\contentsname}{\contentsname}%
%</!book&!report>
  \@starttoc{toc}%
%<book|report>  \if@restonecol\twocolumn\fi
}
```

\l@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\jsc@mpt}%
    \begingroup
      \parindent \z@
%     \@pnumwidth should be \@tocrmarg
%     \rightskip \@pnumwidth
      \rightskip \@tocrmarg
      \parfillskip -\rightskip
      {\leavevmode
        \large \headfont
        \setlength\@lnumwidth{4\jsZw}%
        #1\hfil \hb@xt@\@pnumwidth{\hss #2}}\par
      \nobreak
%<book|report>    \global\@nobreaktrue
%<book|report>    \everypar{\global\@nobreakfalse\everypar{}}%
```

```
    \endgroup
  \fi}
```

\l@chapter 章の目次です。\@lnumwidth を 4.683zw に増やしました。

[2013-12-30] \@lnumwidth を \jsc@tocl@width から決めるようにしてみました。(by ts)

```
%<*book|report>
\newcommand*{\l@chapter}[2]{%
  \ifnum \c@tocdepth >\m@ne
    \addpenalty{-\@highpenalty}%
    \addvspace{1.0em \@plus\jsc@mpt}
%   \vskip 1.0em \@plus\p@   % book.cls では↑がこうなっている
    \begingroup
      \parindent\z@
%     \rightskip\@pnumwidth
      \rightskip\@tocrmarg
      \parfillskip-\rightskip
      \leavevmode\headfont
%     % \if@english\setlength\@lnumwidth{5.5em}\else\setlength\@lnumwidth{4.683\jsZw}\fi
      \setlength\@lnumwidth{\jsc@tocl@width}\advance\@lnumwidth 2.683\jsZw
      \advance\leftskip\@lnumwidth \hskip-\leftskip
      #1\nobreak\hfil\nobreak\hbox to\@pnumwidth{\hss#2}\par
      \penalty\@highpenalty
    \endgroup
  \fi}
%</book|report>
```

\l@section 節の目次です。

```
%<*!book&!report>
\newcommand*{\l@section}[2]{%
  \ifnum \c@tocdepth >\z@
    \addpenalty{\@secpenalty}%
    \addvspace{1.0em \@plus\jsc@mpt}%
    \begingroup
      \parindent\z@
%     \rightskip\@pnumwidth
      \rightskip\@tocrmarg
      \parfillskip-\rightskip
      \leavevmode\headfont
%     % \setlength\@lnumwidth{4\jsZw}% 元 1.5em [2003-03-02]
      \setlength\@lnumwidth{\jsc@tocl@width}\advance\@lnumwidth 2\jsZw
      \advance\leftskip\@lnumwidth \hskip-\leftskip
      #1\nobreak\hfil\nobreak\hbox to\@pnumwidth{\hss#2}\par
    \endgroup
  \fi}
%</!book&!report>
```

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。

```
%<book|report> % \newcommand*{\l@section}{\@dottedtocline{1}{1\jsZw}{3.683\jsZw}}
```

[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)

\l@subsection
\l@subsubsection
\l@paragraph
\l@subparagraph

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，要修正かもしれません。

[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)

```
%<*!book&!report>
% \newcommand*{\l@subsection}   {\@dottedtocline{2}{1.5em}{2.3em}}
% \newcommand*{\l@subsubsection}{\@dottedtocline{3}{3.8em}{3.2em}}
% \newcommand*{\l@paragraph}    {\@dottedtocline{4}{7.0em}{4.1em}}
% \newcommand*{\l@subparagraph} {\@dottedtocline{5}{10em}{5em}}
%
% \newcommand*{\l@subsection}   {\@dottedtocline{2}{1zw}{3zw}}
% \newcommand*{\l@subsubsection}{\@dottedtocline{3}{2\jsZw}{3\jsZw}}
% \newcommand*{\l@paragraph}    {\@dottedtocline{4}{3\jsZw}{3\jsZw}}
% \newcommand*{\l@subparagraph} {\@dottedtocline{5}{4\jsZw}{3\jsZw}}
%
\newcommand*{\l@subsection}{%
          \@tempdima\jsc@tocl@width \advance\@tempdima -1\jsZw
          \@dottedtocline{2}{\@tempdima}{3\jsZw}}
\newcommand*{\l@subsubsection}{%
          \@tempdima\jsc@tocl@width \advance\@tempdima 0\jsZw
          \@dottedtocline{3}{\@tempdima}{4\jsZw}}
\newcommand*{\l@paragraph}{%
          \@tempdima\jsc@tocl@width \advance\@tempdima 1\jsZw
          \@dottedtocline{4}{\@tempdima}{5\jsZw}}
\newcommand*{\l@subparagraph}{%
          \@tempdima\jsc@tocl@width \advance\@tempdima 2\jsZw
          \@dottedtocline{5}{\@tempdima}{6\jsZw}}
%</!book&!report>
%<*book|report>
% \newcommand*{\l@subsection}   {\@dottedtocline{2}{3.8em}{3.2em}}
% \newcommand*{\l@subsubsection}{\@dottedtocline{3}{7.0em}{4.1em}}
% \newcommand*{\l@paragraph}    {\@dottedtocline{4}{10em}{5em}}
% \newcommand*{\l@subparagraph} {\@dottedtocline{5}{12em}{6em}}
\newcommand*{\l@section}{%
          \@tempdima\jsc@tocl@width \advance\@tempdima -1\jsZw
          \@dottedtocline{1}{\@tempdima}{3.683\jsZw}}
\newcommand*{\l@subsection}{%
          \@tempdima\jsc@tocl@width \advance\@tempdima 2.683\jsZw
          \@dottedtocline{2}{\@tempdima}{3.5\jsZw}}
\newcommand*{\l@subsubsection}{%
          \@tempdima\jsc@tocl@width \advance\@tempdima 6.183\jsZw
          \@dottedtocline{3}{\@tempdima}{4.5\jsZw}}
\newcommand*{\l@paragraph}{%
          \@tempdima\jsc@tocl@width \advance\@tempdima 10.683\jsZw
          \@dottedtocline{4}{\@tempdima}{5.5\jsZw}}
\newcommand*{\l@subparagraph}{%
          \@tempdima\jsc@tocl@width \advance\@tempdima 16.183\jsZw
```

```
          \@dottedtocline{5}{\@tempdima}{6.5\jsZw}}
%</book|report>
```

\numberline \@lnumwidth 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{\hb@xt@\@lnumwidth{#1\hfil}\hspace{0pt}}
```

\@dottedtocline \jsTocLine 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。

[2018-06-23] デフォルトでは . . . . のようにベースラインになります。これを変更可能にするため，`\jsTocLine` というマクロに切り出しました。例えば，仮想ボディの中央・・・ に変更したい場合は

```
\renewcommand{\jsTocLine}{\leaders \hbox {\hss ・\hss}\hfill}
```

とします。

```
\def\jsTocLine{\leaders\hbox{%
  $\m@th \mkern \@dotsep mu\hbox{.}\mkern \@dotsep mu$}\hfill}
\def\@dottedtocline#1#2#3#4#5{\ifnum #1>\c@tocdepth \else
  \vskip \z@ \@plus.2\jsc@mpt
  {\leftskip #2\relax \rightskip \@tocrmarg \parfillskip -\rightskip
    \parindent #2\relax\@afterindenttrue
   \interlinepenalty\@M
   \leavevmode
   \@lnumwidth #3\relax
   \advance\leftskip \@lnumwidth \null\nobreak\hskip -\leftskip
    {#4}\nobreak
    \jsTocLine \nobreak\hb@xt@\@pnumwidth{%
         \hfil\normalfont \normalcolor #5}\par}\fi}
```

■図目次と表目次

\listoffigures 図目次を出力します。

```
\newcommand{\listoffigures}{%
%<*book|report>
  \if@twocolumn\@restonecoltrue\onecolumn
  \else\@restonecolfalse\fi
  \chapter*{\listfigurename}%
  \@mkboth{\listfigurename}{}%
%</book|report>
%<*!book&!report>
  \section*{\listfigurename}%
  \@mkboth{\listfigurename}{\listfigurename}%
%</!book&!report>
  \@starttoc{lof}%
```

```
%<book|report>  \if@restonecol\twocolumn\fi
}
```

`\l@figure` 図目次の項目を出力します。

```
\newcommand*{\l@figure}{\@dottedtocline{1}{1\jsZw}{3.683\jsZw}}
```

`\listoftables` 表目次を出力します。

```
\newcommand{\listoftables}{%
%<*book|report>
  \if@twocolumn\@restonecoltrue\onecolumn
  \else\@restonecolfalse\fi
  \chapter*{\listtablename}%
  \@mkboth{\listtablename}{}%
%</book|report>
%<*!book&!report>
  \section*{\listtablename}%
  \@mkboth{\listtablename}{\listtablename}%
%</!book&!report>
  \@starttoc{lot}%
%<book|report>  \if@restonecol\twocolumn\fi
}
```

`\l@table` 表目次は図目次と同じです。

```
\let\l@table\l@figure
```

10.2 参考文献

`\bibindent` 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{2\jsZw}
```

`thebibliography` (*env.*) 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。

```
\newenvironment{thebibliography}[1]{%
  \@jsc@warnoldfontcmdexceptiontrue
  \global\let\presectionname\relax
  \global\let\postsectionname\relax
%<article|slide>  \section*{\refname}\@mkboth{\refname}{\refname}%
%<*kiyou>
  \vspace{1.5\baselineskip}
  \subsubsection*{\refname}\@mkboth{\refname}{\refname}%
  \vspace{0.5\baselineskip}
%</kiyou>
%<book|report>  \chapter*{\bibname}\@mkboth{\bibname}{}%
%<book|report>  \addcontentsline{toc}{chapter}{\bibname}%
```

```
    \list{\@biblabel{\@arabic\c@enumiv}}%
         {\settowidth\labelwidth{\@biblabel{#1}}%
          \leftmargin\labelwidth
          \advance\leftmargin\labelsep
          \@openbib@code
          \usecounter{enumiv}%
          \let\p@enumiv\@empty
          \renewcommand\theenumiv{\@arabic\c@enumiv}}%
%<kiyou>   \small
    \sloppy
    \clubpenalty4000
    \@clubpenalty\clubpenalty
    \widowpenalty4000%
    \sfcode`\.\@m}
   {\def\@noitemerr
     {\@latex@warning{Empty `thebibliography' environment}}%
    \endlist}
```

\newblock \newblock はデフォルトでは小さなスペースを生成します。

```
\newcommand{\newblock}{\hskip .11em\@plus.33em\@minus.07em}
```

\@openbib@code \@openbib@code はデフォルトでは何もしません。この定義は openbib オプションによって変更されます。

```
\let\@openbib@code\@empty
```

\@biblabel \bibitem[...] のラベルを作ります。ltbibl.dtx の定義の半角 [] を全角 ［］ に変え，余分なスペースが入らないように \jsInhibitGlue ではさみました。とりあえずコメントアウトしておきますので，必要に応じて生かしてください。

```
% \def\@biblabel#1{\jsInhibitGlue［#1］\jsInhibitGlue}
```

\cite \@cite \@citex 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。かっこの前後に入るグルーを \jsInhibitGlue で取っていますので，オリジナル同様，Knuth~\cite{knu}␣ のように半角空白で囲んでください。

```
% \def\@citex[#1]#2{\leavevmode
%   \let\@citea\@empty
%   \@cite{\@for\@citeb:=#2\do
%     {\@citea\def\@citea{, \inhibitglue\penalty\@m\ }%
%      \edef\@citeb{\expandafter\@firstofone\@citeb\@empty}%
%      \if@filesw\immediate\write\@auxout{\string\citation{\@citeb}}\fi
%      \@ifundefined{b@\@citeb}{\mbox{\normalfont\bfseries ?}%
%        \G@refundefinedtrue
%        \@latex@warning
%          {Citation `\@citeb' on page \thepage \space undefined}}%
%        {\@cite@ofmt{\csname b@\@citeb\endcsname}}}{#1}}
% \def\@cite#1#2{\jsInhibitGlue［{#1\if@tempswa , #2\fi}］\jsInhibitGlue}
```

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。

```
% \DeclareRobustCommand\cite{\unskip
%   \@ifnextchar [{\@tempswatrue\@citex}{\@tempswafalse\@citex[]}}
% \def\@cite#1#2{$^{\hbox{\scriptsize{#1\if@tempswa
%   , \jsInhibitGlue\ #2\fi}}}$}
```

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）。

```
\newenvironment{theindex}{% 索引を 3 段組で出力する環境
    \if@twocolumn
      \onecolumn\@restonecolfalse
    \else
      \clearpage\@restonecoltrue
    \fi
    \columnseprule.4pt \columnsep 2\jsZw
    \ifx\multicols\@undefined
```

---

hyperref 使用時に索引へのリンクが正常に作られるように、hyperref の説明書の解説に従って \phantomsection を配置した。

```
%<book|report>      \twocolumn[\bxjs@phantomsection
%<book|report>        \addcontentsline{toc}{chapter}{\indexname}%
%<book|report>        \@makeschapterhead{\indexname}]%
%<!book&!report>      \def\presectionname{}\def\postsectionname{}%
%<!book&!report>      \twocolumn[\section*{\indexname}]%
    \else
      \ifdim\textwidth<\fullwidth
        \setlength{\evensidemargin}{\oddsidemargin}
        \setlength{\textwidth}{\fullwidth}
        \setlength{\linewidth}{\fullwidth}
%<book|report>        \begin{multicols}{3}[\chapter*{\indexname}%
%<book|report>        \addcontentsline{toc}{chapter}{\indexname}]%
%<!book&!report>        \def\presectionname{}\def\postsectionname{}%
%<!book&!report>        \begin{multicols}{3}[\section*{\indexname}]%
      \else
%<book|report>        \begin{multicols}{2}[\chapter*{\indexname}%
%<book|report>        \addcontentsline{toc}{chapter}{\indexname}]%
%<!book&!report>        \def\presectionname{}\def\postsectionname{}%
%<!book&!report>        \begin{multicols}{2}[\section*{\indexname}]%
      \fi
    \fi
%<book|report>    \@mkboth{\indexname}{}%
%<!book&!report>    \@mkboth{\indexname}{\indexname}%
    \plainifnotempty % \thispagestyle{plain}
```

```
    \parindent\z@
    \parskip\z@ \@plus .3\jsc@mpt\relax
    \let\item\@idxitem
    \raggedright
    \footnotesize\narrowbaselines
  }{
    \ifx\multicols\@undefined
      \if@restonecol\onecolumn\fi
    \else
      \end{multicols}
    \fi
    \clearpage
  }
```

\@idxitem 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
\subsubitem

```
\newcommand{\@idxitem}{\par\hangindent 4\jsZw} % 元 40pt
\newcommand{\subitem}{\@idxitem \hspace*{2\jsZw}} % 元 20pt
\newcommand{\subsubitem}{\@idxitem \hspace*{3\jsZw}} % 元 30pt
```

\indexspace 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}
```

\seename
\alsoname

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 →\fi}
\newcommand\alsoname{\if@english see also\else →\fi}
```

10.4 脚注

\footnote
\footnotemark

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。pLaTeX の日付が 2016/09/03 より新しい場合は，このパッチが不要なのであてません。

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。

```
\ifx\pltx@foot@penalty\@undefined
  \let\footnotes@ve=\footnote
  \def\footnote{\jsInhibitGlue\footnotes@ve}
  \let\footnotemarks@ve=\footnotemark
  \def\footnotemark{\jsInhibitGlue\footnotemarks@ve}
\fi
```

\@makefnmark 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。

[2013-04-23] 新しい pTEX では脚注番号のまわりにスペースが入りすぎることを防ぐため，北川さんのパッチ [qa:57090] を取り込みました。

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

[2016-07-11] コミュニティ版 pLATEX の変更に追随しました（Thanks: 角藤さん）。pLATEX の日付が 2016/04/17 より新しい場合は，このパッチが不要なのであてません。

---

pTEX 依存のコードなので、minimal 和文ドライバ実装に移動。

`\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 `newtxtext` や `newpxtext` の使用時におかしくなってしまいます。これらのパッケージは内部で `\thefootnote` を再定義していますので，気になる場合はパッケージを読み込むときに `defaultsups` オプションを付けてください (qa:57284, qa:57287)。

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

「注 1」の形式にするには次のようにしてください。

```
% \def\thefootnote{\ifnum\c@footnote>\z@ 注\kern0.1\jsZw\@arabic\c@footnote\fi}
```

`\footnoterule` 本文と脚注の間の罫線です。

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

`\c@footnote` 脚注番号は章ごとにリセットされます。

```
%<book|report>\@addtoreset{footnote}{chapter}
```

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

[2016-08-25] コミュニティ版 pLATEX の「閉じ括弧類の直後に`\footnotetext` が続く場合に改行が起きることがある問題に対処」と同等のコードを追加しました。

[2016-09-08] コミュニティ版 pLATEX のバグ修正に追随しました。

[2016-11-29] 古い pLATEX で使用された場合を考慮してコードを改良。

[2018-03-11] `\next` などいくつかの内部命令を `\jsc@...` 付きのユニークな名前にしました。

[2022-09-13] LATEX 2_ε 2021-11-15 (ltfloat.dtx 2021/10/14 v1.2g) で`\@currentcounter` が追加されましたので，追随します。なお，LATEX 2_ε 2021-06-01 (ltfloat.dtx 2021/02/10 v1.2e) で parhook 対応として `\par` 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@prefix{\inhibitglue\ignorespaces}
\def\jsc@f@@t{\bgroup\aftergroup\jsc@@foot\afterassignment\jsc@f@@t@prefix\let\jsc@next}
\def\jsc@f@t#1{\jsc@f@@t@prefix#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}
```

\@makefntext 実際に脚注を出力する命令です。\@makefnmark 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]{%
  \advance\leftskip 3\jsZw
  \parindent 1\jsZw
  \noindent
  \llap{\@makefnmark\hskip0.3\jsZw}#1}
```

\@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。

```
% \def\@xfootnotenext[#1]{%
%   \begingroup
```

```
%       \ifnum#1>\z@
%         \csname c@\@mpfn\endcsname #1\relax
%         \unrestored@protected@xdef\@thefnmark{\thempfn}%
%       \else
%         \unrestored@protected@xdef\@thefnmark{}%
%       \fi
%    \endgroup
%    \@footnotetext}
```

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 1 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。

```
\ifx j\jsEngine
\def\@inhibitglue{%
  \futurelet\@let@token\@@inhibitglue}
\begingroup
```

```
\let\GDEF=\gdef
\let\CATCODE=\catcode
\let\ENDGROUP=\endgroup
\CATCODE`k=12
\CATCODE`a=12
\CATCODE`n=12
\CATCODE`j=12
\CATCODE`i=12
\CATCODE`c=12
\CATCODE`h=12
\CATCODE`r=12
\CATCODE`t=12
\CATCODE`e=12
\GDEF\KANJI@CHARACTER{kanji character }
\ENDGROUP
\def\@@inhibitglue{%
  \expandafter\expandafter\expandafter\jsc@inhibitglue\expandafter\meaning\expandafter\@let@
\expandafter\def\expandafter\jsc@inhibitglue\expandafter#\expandafter1\KANJI@CHARACTER#2#3\j
  \def\jsc@ig@temp{#1}%
  \ifx\jsc@ig@temp\@empty
    \ifnum\the\inhibitxspcode`#2=2\relax
      \inhibitglue
    \fi
  \fi}
\fi
```

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。

```
\ifnum\bxjs@everyparhook=\bxjs@everyparhook@@compat
```

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

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。

```
\def\@doendpe{%
  \@endpetrue
  \def\par{%
    \@restorepar\clubpenalty\@clubpenalty\everypar{\everyparhook}\par\@endpefalse}%
  \everypar{{\setbox\z@\lastbox}\everypar{\everyparhook}\@endpefalse\everyparhook}}
```

[2017-08-31] minipage 環境にも対策します。

```
\def\@setminipage{%
  \@minipagetrue
  \everypar{\@minipagefalse\everypar{\everyparhook}}%
}
```

\item 命令の直後です。

```
\def\@item[#1]{%
  \if@noparitem
    \@donoparitem
  \else
    \if@inlabel
      \indent \par
    \fi
    \ifhmode
      \unskip\unskip \par
    \fi
    \if@newlist
      \if@nobreak
        \@nbitem
      \else
        \addpenalty\@beginparpenalty
        \addvspace\@topsep
        \addvspace{-\parskip}%
      \fi
    \else
      \addpenalty\@itempenalty
      \addvspace\itemsep
    \fi
    \global\@inlabeltrue
  \fi
  \everypar{%
    \@minipagefalse
    \global\@newlistfalse
    \if@inlabel
      \global\@inlabelfalse
      {\setbox\z@\lastbox
       \ifvoid\z@
         \kern-\itemindent
       \fi}%
      \box\@labels
      \penalty\z@
    \fi
    \if@nobreak
      \@nobreakfalse
      \clubpenalty \@M
    \else
      \clubpenalty \@clubpenalty
      \everypar{\everyparhook}%
    \fi\everyparhook}%
  \if@noitemarg
    \@noitemargfalse
    \if@nmbrlist
      \refstepcounter\@listctr
```

```
    \fi
  \fi
  \sbox\@tempboxa{\makelabel{#1}}%
  \global\setbox\@labels\hbox{%
    \unhbox\@labels
    \hskip \itemindent
    \hskip -\labelwidth
    \hskip -\labelsep
    \ifdim \wd\@tempboxa >\labelwidth
      \box\@tempboxa
    \else
      \hbox to\labelwidth {\unhbox\@tempboxa}%
    \fi
    \hskip \labelsep}%
  \ignorespaces}
```

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。

```
\def\@afterheading{%
  \@nobreaktrue
  \everypar{%
    \if@nobreak
      \@nobreakfalse
      \clubpenalty \@M
      \if@afterindent \else
        {\setbox\z@\lastbox}%
      \fi
    \else
      \clubpenalty \@clubpenalty
      \everypar{\everyparhook}%
    \fi\everyparhook}}
```

「\everyparhook 用のパッチ処理」はここまで。

```
\fi
```

\@gnewline についてはちょっと複雑な心境です。もともとの pLaTeX 2_ε は段落の頭にグルーが入る方で統一されていました。しかし \\ の直後にはグルーが入らず，不統一でした。そこで \\ の直後にもグルーを入れるように直していただいた経緯があります。しかし，ここでは逆にグルーを入れない方で統一したいので，また元に戻してしまいました。

しかし単に戻すだけでも駄目みたいなので，ここでも最後にグルーを消しておきます。

※ luatexja を読み込んだ場合に lltjcore.sty によって上書きされるのを防ぐため遅延させる。

```
\ifnum\bxjs@everyparhook=\bxjs@everyparhook@@none\else
```

```
\AtEndOfClass{%
\def\@gnewline #1{%
  \ifvmode
    \@nolnerr
  \else
    \unskip \reserved@e {\reserved@f#1}\nobreak \hfil \break \null
    \jsInhibitGlue \ignorespaces
  \fi}
}
\fi
```

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、既定では読み込まない。
※\小、\上小 の制御綴は定義しない。

```
\if@jslogo
  \IfFileExists{jslogo.sty}{%
    \RequirePackage{jslogo}%
  }{%
    \ClassWarningNoLine\bxjs@clsname
     {The package 'jslogo' is not installed.\MessageBreak
      It is included in the recent release of\MessageBreak
      the 'jsclasses' bundle}
  }
\fi
```

13 amsmath との衝突の回避

\ltx@ifnextchar
\ProvidesFile

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 LaTeX 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きます。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 掲示板 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す（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。

```
%\let\ltx@ifnextchar\@ifnextchar
%\def\ProvidesFile#1{%
%  \begingroup
%    \catcode`\ 10 %
%    \ifnum \endlinechar<256 %
%      \ifnum \endlinechar>\m@ne
%        \catcode\endlinechar 10 %
%      \fi
%    \fi
%    \@makeother\/%
%    \@makeother\&%
%    \ltx@ifnextchar[{\@providesfile{#1}}{\@providesfile{#1}[]}}
```

14 初期設定

■いろいろな語

\prepartname
\postpartname
\prechaptername
\postchaptername
\presectionname
\postsectionname

```
\newcommand{\prepartname}{\if@english Part~\else 第\fi}
\newcommand{\postpartname}{\if@english\else 部\fi}
%<book|report>\newcommand{\prechaptername}{\if@english Chapter~\else 第\fi}
%<book|report>\newcommand{\postchaptername}{\if@english\else 章\fi}
\newcommand{\presectionname}{}%  第
\newcommand{\postsectionname}{}% 節
```

\contentsname
\listfigurename
\listtablename

```
\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}
```

\refname
\bibname
\indexname

```
\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}
```

\figurename
\tablename

```
%<!jspf>\newcommand{\figurename}{\if@english Fig.~\else 図\fi}
%<jspf>\newcommand{\figurename}{Fig.~}
%<!jspf>\newcommand{\tablename}{\if@english Table~\else 表\fi}
%<jspf>\newcommand{\tablename}{Table~}
```

\appendixname
\abstractname

```
% \newcommand{\appendixname}{\if@english Appendix~\else 付録\fi}
\newcommand{\appendixname}{\if@english \else 付録\fi}
%<!book>\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 **\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 \和暦 8 ビット欧文 TEX 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 TEX 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。

TEX エンジンの種類により処理を分ける。

```
\@onlypreamble\bxjs@decl@Seireki@cmds
\@tempswafalse
\if p\jsEngine \@tempswatrue \fi
\if n\jsEngine \@tempswatrue \fi
\bxjs@cond\if@tempswa\fi{%
```

8 ビット欧文 TEX の場合。

\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 本クラス用の **\西暦**/**\和暦** の命令を定義するためのマクロ。

※**\def\西暦** は実際には \^^e8 の定義文であることに注意。

```
\def\bxjs@decl@Seireki@cmds{%
  \def\西暦{\jsSeirekitrue}%
  \def\和暦{\jsSeirekifalse\bxjs@wareki@used}}
```

\Seireki \Wareki **\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。

```
\def\Seireki{\jsSeirekitrue}
\def\Wareki{\jsSeirekifalse\bxjs@wareki@used}

\def\bxjs@if@use@seireki{\bxjs@cond\ifjsSeireki\fi}
\def\bxjs@iai{\noexpand~}
}{%
```

8 ビット欧文 TEX 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
\def\bxjs@decl@Seireki@cmds{%
  \def\西暦{\西暦 true}%
  \def\和暦{\西暦 false\bxjs@wareki@used}}
\def\Seireki{\西暦 true}
\def\Wareki{\西暦 false\bxjs@wareki@used}
```

```
\def\bxjs@if@use@seireki{\bxjs@cond\if 西暦\fi}
\let\bxjs@iai\@empty
}
\bxjs@decl@Seireki@cmds
```

\ifbxjs@bxwareki@avail bxwareki パッケージが使用できるか。

※ 8 ビット欧文でかつ非 e-TeX なエンジン（現状ではサポート外だが）では bxwareki を読むだけでエラーが発生してしまうので、この場合は読込を回避する。

```
\newif\ifbxjs@bxwareki@avail
\IfFileExists{bxwareki.sty}{%
  \if \if n\jsEngine \ifjsWitheTeX T\else F\fi\else T\fi T%
    \RequirePackage{bxwareki}[2018/04/08]%v0.2
    \bxjs@bxwareki@availtrue
  \fi}{}
```

\bxjs@wareki@used bxwareki が利用できないのに和暦出力をしようとした場合に警告を出す。

```
\ifbxjs@bxwareki@avail \let\bxjs@wareki@used\@empty
\else
  \bxjs@robust@def\bxjs@wareki@used{%
    \global\let\bxjs@wareki@used\@empty
    \ClassWarning\bxjs@clsname
     {Wareki mode is not supported, since\MessageBreak
      'bxwareki' is unavailable, reported}}
  \g@addto@macro\bxjs@begin@document@hook{%
    \let\bxjs@wareki@used\@empty}
\fi
```

\jayear 和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 年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。

```
\ifbxjs@bxwareki@avail
  \let\jayear\warekiyear
  \def\bxjs@tmpa{H}\ifx\bxjs@tmpa\warekigengoinitial
    \newcount\heisei \heisei=\value{warekiyear}
  \fi
```

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。

```
\else
  \edef\jayear{\the\year \bxjs@iai}
\fi
```

\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。

```
%<*jsclasses>
\newif\if 西暦 \西暦 true
```

```
\def\西暦{\西暦 true}
\def\和暦{\西暦 false}
\newcount\heisei \heisei\year \advance\heisei-1988\relax
\def\pltx@today@year@#1{%
  \ifnum\numexpr\year-#1=1 元\else
    \ifnum1=\iftdir\ifmdir0\else1\fi\else0\fi
      \kansuji\numexpr\year-#1\relax
    \else
      \number\numexpr\year-#1\relax\nobreak
    \fi
  \fi 年
}
\def\pltx@today@year{%
  \ifnum\numexpr\year*10000+\month*100+\day<19890108
    昭和\pltx@today@year@{1925}%
  \else\ifnum\numexpr\year*10000+\month*100+\day<20190501
    平成\pltx@today@year@{1988}%
  \else
    令和\pltx@today@year@{2018}%
  \fi\fi}
%</jsclasses>
\begingroup
\let\bxjs@next\relax
\ifbxjs@bxwareki@avail \ifx\warekigengo\@empty\else
  \def\bxjs@next{\warekitoday}
  \bxjs@test@engine\unexpanded{%
    \def\bxjs@next{\unexpanded\expandafter{\warekitoday}}}
\fi\fi
\def\!#1#2#3{\noexpand#1\noexpand#2\noexpand#3}
\ifx\bxjs@iai\@empty \let\!\@empty \fi
\xdef\bxjs@today{%
  \if@english
    \ifcase\month\or
      January\or February\or March\or April\or May\or June\or
      July\or August\or September\or October\or November\or December\fi
      \space\number\day, \number\year
  \else
    \ifx\bxjs@next\relax \expandafter\@firstoftwo
    \else \noexpand\bxjs@if@use@seireki
    \fi {%
      \number\year\bxjs@iai\!年%
      \bxjs@iai\number\month\bxjs@iai\!月%
      \bxjs@iai\number\day\bxjs@iai\!日%
    }{\bxjs@next}%
  \fi}
\endgroup
\let\today\bxjs@today
```

texjporg 版の日本語用 Babel 定義ファイル（japanese.ldf）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：eng-lish）

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

■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 pLATEX で新設されたテスト用パッケージ（expl2e 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。expl2e パッケージが読みこ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 ☃

■\strong 命令の補填 fontspec で提供される \strong 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る。ここでは \js〜 の名前で定義することにする。

\jsStrongText 強調用の宣言型命令。

```
\bxjs@robust@def\jsStrongText{\bxjs@strong@text}%
```

fontspec と互換の \strongfontdeclare 命令も提供する。既定の設定は \bfseries（太字）である。

※\strongfontdeclare は試験的機能とする。

```
\chardef\bxjs@strong@level=0
\DeclareRobustCommand*\jsStrongDeclare[1]{%
  \bxjs@set@array@from@clist{bxjs@strong}{#1}%
  \chardef\bxjs@strong@level\z@}
\jsStrongDeclare{\bfseries}
\def\bxjs@strong@text{%
  \bxjs@csletcs{bxjs@tmpa}{bxjs@strong/\the\bxjs@strong@level}%
  \ifx\bxjs@tmpa\relax
    \bxjs@advance@qc\bxjs@strong@level\m@ne \bxjs@strong@text
  \else \bxjs@advance@qc\bxjs@strong@level\@ne \bxjs@tmpa
  \fi}
% \end{macro}
%
% \paragraph{共通命令の実装}
%
% |\jQ| 等の「単位」系の共通命令を実装する。
%
% \begin{macro}{\bxjs@const@unit}
% 固定値の単位として使える制御綴を定義する。
%
% {\eTeX}拡張が使える場合は、
%「|\dimexpr|外部寸法表記|\relax|」の形式
%（これは内部値なので単位として使える）
% に展開されるマクロとして定義する。
%    \begin{macrocode}
\@onlypreamble\bxjs@const@unit
\@onlypreamble\bxjs@const@unit@a
\ifjsWitheTeX
  \def\bxjs@const@unit#1#2#3{%
    \protected\edef#1{\dimexpr\the\dimexpr#3\relax\relax}}
```

ε-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 \fontdimen を使う」というテク

ニックを用いる（アレ）。

```
\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 0.25 mm に等しい。

\jH

```
\bxjs@const@unit\jQ{jQ}{0.25mm}
\let\jH\jQ
```

\trueQ \trueQ と \trueH はともに 0.25 true 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\bxjs@truept{truept}{\@tempdima}
\else \let\trueQ\jQ \let\bxjs@truept\p@
\fi
\let\trueH\trueQ
```

\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} とすると、和文が 12Q になる。

\ascpt 同様に、\ascpt は truept を和文スケールで割った値。

```
\@tempdima\trueQ \bxjs@invscale\@tempdima\jsScale
\bxjs@const@unit\ascQ{ascQ}{\@tempdima}
\@tempdima\bxjs@truept \bxjs@invscale\@tempdima\jsScale
\bxjs@const@unit\ascpt{ascpt}{\@tempdima}
```

\jafontsize \jafontsize{⟨フォントサイズ⟩}{⟨行送り⟩}： 和文フォント規準で、すなわち、1 zw が ⟨フォントサイズ⟩ に等しく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}
```

```
\def\bxjs@jafontsize@a#1{%
  \bxjs@parse@qh{#1}%
  \ifx\bxjs@tmpb\relax \def\bxjs@tmpb{#1}\fi
  \@defaultunits\@tempdima\bxjs@tmpb pt\relax\@nnil}
```

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@let@lenexpr{\edef}
```

`\bxjs@kanjiskip` 和文間空白の量を表すテキスト。

```
\def\bxjs@kanjiskip{0pt}
```

`\jsSetKanjiSkip` 和文間空白の量を設定する。

※`\setkanjiskip` の実体。

```
\DeclareRobustCommand*\jsSetKanjiSkip[1]{%
  \bxjs@let@lenexpr\bxjs@kanjiskip{#1}%
  \bxjs@reset@kanjiskip}
```

`\jsGetKanjiSkip` 和文間空白の量を表すテキストに展開する。

※`\getkanjiskip` の実体。

```
\newcommand*\jsGetKanjiSkip{%
  \bxjs@kanjiskip}
```

`\ifbxjs@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@robust@def\jsEnableKanjiSkip{%
  \bxjs@kanjiskip@enabledtrue
  \bxjs@reset@kanjiskip}
\bxjs@robust@def\jsDisableKanjiSkip{%
  \bxjs@kanjiskip@enabledfalse
  \bxjs@reset@kanjiskip}
```

`\bxjs@reset@kanjiskip` 現在の和文間空白の設定を実際に反映させる。

```
\bxjs@robust@def\bxjs@reset@kanjiskip{%
  \ifbxjs@kanjiskip@enabled
    \setlength{\@tempskipa}{\bxjs@kanjiskip}%
  \else \@tempskipa\z@
  \fi
  \jsApplyKanjiSkip\@tempskipa}
```

`\jsApplyKanjiSkip` `\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。

```
\let\jsApplyKanjiSkip\@gobble
```

\bxjs@xkanjiskip 和欧文間空白について同様のものを用意する。

\jsSetXKanjiSkip
\jsGetXKanjiSkip
\ifbxjs@xkanjiskip@enabled
\jsEnableXKanjiSkip
\jsDisableXKanjiSkip
\bxjs@reset@xkanjiskip
\jsApplyXKanjiSkip

```
\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
```

\jsResetDimen を用いて、フォントサイズが変更された時に空白の量が追随するようにする。

```
\g@addto@macro\jsResetDimen{%
  \bxjs@reset@kanjiskip
  \bxjs@reset@xkanjiskip}
```

和文・和欧文間空白の初期値。

```
\AtEndOfPackage{%
\jsSetKanjiSkip{0pt plus.1\jsZw minus.01\jsZw}%
\ifx\jsDocClass\jsSlide \jsSetXKanjiSkip{0.1em}%
\else \jsSetXKanjiSkip{0.25em plus 0.15em minus 0.06em}%
\fi
}
```

■和文空白命令

```
\ifbxjs@jaspace@cmd
```

\jaenspace 半角幅の水平空き。

```
  \def\jaenspace{\hskip.5\jsZw\relax}
```

\jathinspace 和欧文間空白を入れるユーザ命令。

```
  \def\jathinspace{\hskip\bxjs@xkanjiskip\relax}
```

_ 全角空白文字 1 つからなる名前の制御綴。\zwspace と等価になる。

```
  \def\  {\zwspace}
```

\> 非数式中では \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）。

```
\bxjs@protected\def\bxjs@choice@jathinspace{%
  \relax\ifmmode \mskip\medmuskip
  \else \jathinspace\ignorespaces
  \fi}
\jsAtEndOfClass{%
  \ifjsWitheTeX \let\>\bxjs@choice@jathinspace
  \else \def\>{\protect\bxjs@choice@jathinspace}%
  \fi}
```

\jaspace jlreq クラスと互換の命令。

```
\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}
```

終わり。

```
\fi
```

■和文ドライバ読込　フックを実行する。

```
\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。

```
%</class>
```

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`　XETEX
  - `j`　pTEX または upTEX
  - `n`　以上の何れでもない
- `\ifjsWithupTeX`［スイッチ］　使用されているエンジンが upTEX であるか。
- `\ifjsWitheTeX`［スイッチ］　使用されているエンジンが ε-TEX 拡張であるか。
- `\ifjsInPdfMode`［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 ×（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」はこの設定と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 の中で呼び出される、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。

```
%<*drv>
```

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る。

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 や LuaTEX-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く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

```
%<*minimal>
%% 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。

```
\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}
```

```
3988    {It's a fatal error. I'll quit right now.}
3989   \expandafter\@firstofone
3990 \fi{\endinput\@@end}
```

■補助マクロ

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

```
3991 \def\DeclareJaTextFontCommand#1#2{%
3992   \DeclareRobustCommand#1[1]{%
3993     \relax
3994     \ifmmode \expandafter\nfss@text \fi
3995     {#2##1}}%
3996 }
```

\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。

```
3997 \def\DeclareJaMathFontCommand#1#2{%
3998   \DeclareRobustCommand#1[1]{%
3999     \relax
4000     \ifmmode\else \non@alpherr{#1\space}\fi
4001     \nfss@text{\fontfamily\familydefault
4002         \fontseries{m}\fontshape{n}\selectfont\relax
4003         #2##1}%
4004   }%
4005 }
```

\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。

```
4006 \bxjs@nclong\def\bxjs@@CSsfdefault{\sfdefault}%
4007 \@onlypreamble\bxjs@if@sf@default
4008 \def\bxjs@if@sf@default#1{%
4009   \ifx\familydefault\bxjs@@CSsfdefault#1\fi
4010   \g@addto@macro\bxjs@begin@document@hook{%
4011     \ifx\familydefault\bxjs@@CSsfdefault#1\fi}%
4012 }
```

\jsInverseScale \jsScale の逆数。

※\CS=\jsInverseScale\CS は \bxjs@invscale\CS\jsScale よりも精度が劣るが処理が軽い。

```
4013 \@tempdima\p@ \bxjs@invscale\@tempdima\jsScale
4014 \edef\jsInverseScale{\strip@pt\@tempdima}
```

\jsLetHeadChar \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。

※先頭にあるのが制御綴やグループである場合は \CS は \relax に等置される。

※文字トークンは“\the-文字列”のカテゴリコードをもつ。

※非 Unicode エンジンの場合は文字列が UTF-8 で符号化されていると見なし、先頭が高位バイトの場合は 1 文字分のバイト列（のトークン列）を抽出する。この場合は元のカテゴリコードが保持される。

```
\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{%
  \bxjs@let@hchar@out\def{{#1}}}
\def\bxjs@let@hchar@out#1#2{%
  \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{%
```

```
    \bxjs@let@hchar@out\let\relax
  }{\bxjs@cond\ifnum\@tempcnta<\bxjs@let@hchar@cstc\fi{%
    \bxjs@let@hchar@chr@ue@b
  }{\bxjs@cond\ifnum\@tempcnta<\bxjs@let@hchar@cstd\fi{%
    \bxjs@let@hchar@chr@ue@c
  }{\bxjs@cond\ifnum\@tempcnta<\bxjs@let@hchar@cste\fi{%
    \bxjs@let@hchar@chr@ue@d
  }{%else
    \bxjs@let@hchar@out\let\relax
  }}}}}}
\def\bxjs@let@hchar@chr@ue@a#1{%
  \bxjs@let@hchar@out\def{{#1}}}
\def\bxjs@let@hchar@chr@ue@b#1#2{%
  \bxjs@let@hchar@out\def{{#1#2}}}
\def\bxjs@let@hchar@chr@ue@c#1#2#3{%
  \bxjs@let@hchar@out\def{{#1#2#3}}}
\def\bxjs@let@hchar@chr@ue@d#1#2#3#4{%
  \bxjs@let@hchar@out\def{{#1#2#3#4}}}
```

B.2 (u)pTEX 用の設定

```
\ifx j\jsEngine
```

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。

```
\def\bxjs@let@hchar@chr@pp#1#2{%
  \expandafter\bxjs@let@hchar@chr@pp@a\meaning#2\relax#1#2}
\def\bxjs@let@hchar@chr@pp@a#1#2\relax#3#4{%
%\message{(\meaning#3:\meaning#4)}%
  \bxjs@cond\if#1k\fi{%
    \bxjs@let@hchar@out\def{{#4}}%
  }{%else
    \bxjs@let@hchar@chr@ue#3#4%
  }}
\let\bxjs@let@hchar@chr\bxjs@let@hchar@chr@pp
```

■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}
```

`\bxjs@declarefontshape` は標準の和文フォント宣言である。後で `\bxjs@scale` を求

めるため一旦マクロにしておく。\bxjs@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。

```
\@onlypreamble\bxjs@declarefontshape
\ifjsWithupTeX
\def\bxjs@declarefontshape{%
\DeclareFontShape{JY2}{mc}{m}{n}{<->s*[\bxjs@scale]upjpnrm-h}{}%
\DeclareFontShape{JY2}{gt}{m}{n}{<->s*[\bxjs@scale]upjpngt-h}{}%
\DeclareFontShape{JT2}{mc}{m}{n}{<->s*[\bxjs@scale]upjpnrm-v}{}%
\DeclareFontShape{JT2}{gt}{m}{n}{<->s*[\bxjs@scale]upjpngt-v}{}%
}
\def\bxjs@sizereference{upjisr-h}
```

pTEX の場合の定義を示す。JS クラスのフォント種別オプション非指定時の定義と同じである。

```
\else
\def\bxjs@declarefontshape{%
\DeclareFontShape{JY1}{mc}{m}{n}{<->s*[\bxjs@scale]jis}{}%
\DeclareFontShape{JY1}{gt}{m}{n}{<->s*[\bxjs@scale]jisg}{}%
\DeclareFontShape{JT1}{mc}{m}{n}{<->s*[\bxjs@scale]tmin10}{}%
\DeclareFontShape{JT1}{gt}{m}{n}{<->s*[\bxjs@scale]tgoth10}{}%
}
\def\bxjs@sizereference{jis}
\fi
```

既に使用されている標準和文フォント定義がもしあれば取り消す。

```
\def\bxjs@next#1/#2/#3/#4/#5\relax{%
  \def\bxjs@tmpb{#5}}
\ifjsWithpTeXng \def\bxjs@tmpb{10}%
\else
\expandafter\expandafter\expandafter\bxjs@next
 \expandafter\string\the\jfont\relax
\fi
\@for\bxjs@tmpa:={\jsc@JYn/mc/m/n,\jsc@JYn/gt/m/n,%
                  \jsc@JTn/mc/m/n,\jsc@JTn/gt/m/n}\do
  {\expandafter\let\csname\bxjs@tmpa/10\endcsname=\@undefined
   \expandafter\let\csname\bxjs@tmpa/\bxjs@tmpb\endcsname=\@undefined}
```

■和文フォントスケールの補正　実は、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 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 1 zw 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 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 s/f を求めてその値をマクロ \bxjs@scale に保存する。

```
\begingroup
% 参照用フォント (\bxjs@sizereference) の全角空白の幅を取得
  \font\bxjs@tmpa=\bxjs@sizereference\space at 10pt
```

```
  \setbox\z@\hbox{\bxjs@tmpa\char\jis"2121\relax}
% 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
    \edef\bxjs@tmpa{\strip@pt\wd\z@}
    \@tempdima=10pt \@tempdima=\jsScale\@tempdima
    \bxjs@invscale\@tempdima\bxjs@tmpa
    \xdef\bxjs@scale{\strip@pt\@tempdima}
  \fi
\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}{mc}{bx}{it}{<->ssub*gt/m/n}{}
\DeclareFontShape{\jsc@JYn}{mc}{bx}{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@JYn}{mc}{b}{n}{<->ssub*mc/bx/n}{}
\DeclareFontShape{\jsc@JYn}{mc}{b}{it}{<->ssub*mc/bx/n}{}
\DeclareFontShape{\jsc@JYn}{mc}{b}{sl}{<->ssub*mc/bx/n}{}
\DeclareFontShape{\jsc@JYn}{gt}{b}{n}{<->ssub*gt/bx/n}{}
\DeclareFontShape{\jsc@JYn}{gt}{b}{it}{<->ssub*gt/bx/n}{}
\DeclareFontShape{\jsc@JYn}{gt}{b}{sl}{<->ssub*gt/bx/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}{m}{sc}{<->ssub*mc/m/n}{}
\DeclareFontShape{\jsc@JTn}{gt}{m}{it}{<->ssub*gt/m/n}{}
\DeclareFontShape{\jsc@JTn}{gt}{m}{sl}{<->ssub*gt/m/n}{}
\DeclareFontShape{\jsc@JTn}{mc}{bx}{it}{<->ssub*gt/m/n}{}
\DeclareFontShape{\jsc@JTn}{mc}{bx}{sl}{<->ssub*gt/m/n}{}
\DeclareFontShape{\jsc@JTn}{gt}{bx}{it}{<->ssub*gt/m/n}{}
\DeclareFontShape{\jsc@JTn}{gt}{bx}{sl}{<->ssub*gt/m/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}{}
```

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。

```
\@ifl@t@r\fmtversion{2020/10/01}
    {\jsc@needsp@tchfalse}{\jsc@needsp@tchtrue}
\ifjsc@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}
\DeclareRobustCommand\sffamily
  {\not@math@alphabet\sffamily\mathsf
   \romanfamily\sfdefault\kanjifamily\gtdefault\selectfont}
\DeclareRobustCommand\ttfamily
  {\not@math@alphabet\ttfamily\mathtt
   \romanfamily\ttdefault\kanjifamily\gtdefault\selectfont}
\g@addto@macro\bxjs@begin@document@hook{%
  \ifx\mweights@init\@undefined\else  % mweights.sty is loaded
    % my definitions above should have been overwritten, recover it!
    % \selectfont is executed twice but I don't care about speed...
    \expandafter\g@addto@macro\csname rmfamily \endcsname
      {\kanjifamily\mcdefault\selectfont}%
    \expandafter\g@addto@macro\csname sffamily \endcsname
      {\kanjifamily\gtdefault\selectfont}%
    \expandafter\g@addto@macro\csname ttfamily \endcsname
      {\kanjifamily\gtdefault\selectfont}%
  \fi}
\else                          % 2020-02-02
\g@addto@macro\@rmfamilyhook
  {\prepare@family@series@update@kanji{mc}\mcdefault}
\g@addto@macro\@sffamilyhook
  {\prepare@family@series@update@kanji{gt}\gtdefault}
\g@addto@macro\@ttfamilyhook
  {\prepare@family@series@update@kanji{gt}\gtdefault}
\fi
\else % --- for 2020-02-02 or older END & for 2020-10-01 BEGIN
\AddToHook{rmfamily}%
  {\prepare@family@series@update@kanji{mc}\mcdefault}
\AddToHook{sffamily}%
  {\prepare@family@series@update@kanji{gt}\gtdefault}
\AddToHook{ttfamily}%
  {\prepare@family@series@update@kanji{gt}\gtdefault}
\fi   % --- for 2020-10-01 END
\ifx\DeclareFixJFMCJKTextFontCommand\@undefined
\DeclareJaTextFontCommand{\textmc}{\mcfamily}
\DeclareJaTextFontCommand{\textgt}{\gtfamily}
\fi
\bxjs@if@sf@default{%
```

```
  \renewcommand\kanjifamilydefault{\gtdefault}}
```

念のため。

```
\selectfont
```

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）。

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

■パラメタの設定

```
\prebreakpenalty\jis"2147=10000
\postbreakpenalty\jis"2148=10000
\prebreakpenalty\jis"2149=10000
\inhibitxspcode`！=1
\inhibitxspcode`〒=2
\xspcode`+=3
\xspcode`\%=3
```

"80〜"FF の範囲の \spcode を 3 に変更。

```
\@tempcnta="80 \@whilenum\@tempcnta<"100 \do{%
  \xspcode\@tempcnta=3\advance\@tempcnta\@ne}
```

\jsInhibitGlueAtParTop の定義。「JS クラスでの定義」を利用する。

```
\let\jsInhibitGlueAtParTop\@inhibitglue
```

\jsResetDimen は空のままでよい。

■組方向依存の処理　組方向判定の if-トークン（\if?dir）は pTEX 以外では未定義であるため、そのまま if 文に入れることができない。これを回避するため部分的に!をエスケープ文字に使う。

```
\begingroup
\catcode`\!=0
```

\bxjs@ptex@dir 現在の組方向：t=縦、y=横、?=その他。

```
\gdef\bxjs@ptex@dir{%
  !iftdir t%
  !else!ifydir y%
  !else ?%
  !fi!fi}
```

新版の pTEX で脚注番号の周囲の空きが過大になる現象への対処。
※現在の pLATEX カーネルでは対処が既に行われている。ここでは、\@makefnmark の定義が古いものであった場合に、新しいものに置き換える。

```
% 古い \@makefnmark の定義
\bxjs@nclong\def\bxjs@tmpa{\hbox{%
  !ifydir \@textsuperscript{\normalfont\@thefnmark}%
  !else\hbox{\yoko\@textsuperscript{\normalfont\@thefnmark}}!fi}}
```

```
\ifx\@makefnmark\bxjs@tmpa
\bxjs@nclong\gdef\@makefnmark{%
  !ifydir \hbox{}\hbox{\@textsuperscript{\normalfont\@thefnmark}}\hbox{}%
  !else\hbox{\yoko\@textsuperscript{\normalfont\@thefnmark}}!fi}
\fi
```

エスケープ文字の変更はここまで。

```
\endgroup
```

■minijs パッケージのブロック　やっておく。

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

B.3　pdfTEX 用の処理

```
\else\if \if p\jsEngine T\else\if n\jsEngine T\else F\fi\fi T
```

\jsLetHeadChar を UTF-8 バイト列に対応させる。

```
\let\bxjs@let@hchar@chr\bxjs@let@hchar@chr@ue
```

ムニャムニャ。

```
\@onlypreamble\bxjs@cjk@loaded
\def\bxjs@cjk@loaded{%
  \def\@footnotemark{%
    \leavevmode
    \ifhmode
      \edef\@x@sf{\the\spacefactor}%
      \ifdim\lastkern>\z@\ifdim\lastkern<5sp\relax
         \unkern\unkern
         \ifdim\lastskip>\z@ \unskip \fi
      \fi\fi
      \nobreak
    \fi
    \@makefnmark
    \ifhmode \spacefactor\@x@sf \fi
    \relax}%
  \let\bxjs@cjk@loaded\relax
}
\g@addto@macro\bxjs@begin@document@hook{%
  \@ifpackageloaded{CJK}{%
    \bxjs@cjk@loaded
  }{}%
}
```

B.4　XETEX 用の処理

```
\else\ifx x\jsEngine
```

\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
```

```
  \bxjs@cond\ifnum\@tempcnta=27 \fi{%
    \bxjs@let@hchar@chr@xe
  }{\bxjs@let@hchar@out\def{{#1}}}}
\def\bxjs@let@hchar@chr@xe#1{%
  \lccode`0=`#1\relax
  \lowercase{\bxjs@let@hchar@out\def{{0}}}}
```

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

```
\@onlypreamble\bxjs@do@precisetext
\ifx\XeTeXgenerateactualtext\@undefined\else
  \def\bxjs@do@precisetext{%
    \XeTeXgenerateactualtext=\@ne}
\fi
```

`\bxjs@do@simplejasetup` 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。

```
\@onlypreamble\bxjs@do@simplejasetup
\def\bxjs@do@simplejasetup{%
  \@namedef{bxjs@zeroglue/0.0pt}{T}%
  \ifnum\XeTeXinterchartokenstate>\z@
  \else\expandafter\ifx\csname bxjs@zeroglue/\the\XeTeXlinebreakskip\endcsname\relax\else
    \jsSimpleJaSetup
    \ClassInfo\bxjs@clsname
     {'\string\jsSimpleJaSetup' is applied\@gobble}%
  \fi\fi}
```

`\jsSimpleJaSetup` 日本語出力用の超簡易的な設定。

```
\newcommand*{\jsSimpleJaSetup}{%
  \XeTeXlinebreaklocale "ja"\relax
  \XeTeXlinebreakskip=0pt plus 1pt minus 0.1pt\relax
  \XeTeXlinebreakpenalty=0\relax}
```

B.5 後処理（エンジン共通）

```
\fi\fi\fi
```

simplejasetup オプションの処理。

```
\ifx\bxjs@do@simplejasetup\@undefined\else
  \g@addto@macro\bxjs@begin@document@hook{%
    \ifbxjs@simplejasetup
      \bxjs@do@simplejasetup
    \fi}
\fi
```

precisetext オプションの処理。

```
\ifbxjs@precisetext
  \ifx\bxjs@do@precisetext\@undefined
    \ClassWarning\bxjs@clsname
     {The current engine does not support the\MessageBreak
```

```
      'precise-text' option\@gobble}
  \else
    \bxjs@do@precisetext
  \fi
\fi
```

■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。

```
\g@addto@macro\bxjs@begin@document@hook{%
  \ifx\jsInhibitGlueAtParTop\@empty
    \def\bxjs@tmpa{\jsInhibitGlueAtParTop}%
    \ifx\everyparhook\bxjs@tmpa
      \let\everyparhook\@empty
    \fi
  \fi}
```

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。

```
\ifnum\bxjs@everyparhook=\bxjs@everyparhook@@modern
```

まず \everypar を“乗っ取る”処理を行う。

```
  \let\bxjs@everypar\everypar
  \newtoks\everypar
  \everypar\bxjs@everypar
```

そして本物の \everypar では、最後で常に \everyparhook が実行されるようにする。

```
  \bxjs@everypar{\the\expandafter\everypar\everyparhook}%
\fi
```

■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。

```
\ifbxjs@fancyhdr
```

\bxjs@adjust@fancyhdr　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。

```
\@onlypreamble\bxjs@adjust@fancyhdr
\def\bxjs@adjust@fancyhdr{%
```

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。

```
  \def\bxjs@tmpa{\fancyplain{}{\sl\rightmark}\strut}%
  \def\bxjs@tmpb{\fancyplain{}{\rightmark}\strut}%
  \ifx\f@ncyelh\bxjs@tmpa \global\let\f@ncyelh\bxjs@tmpb \fi
```

```
  \ifx\f@ncyerh\bxjs@tmpa \global\let\f@ncyerh\bxjs@tmpb \fi
  \ifx\f@ncyolh\bxjs@tmpa \global\let\f@ncyolh\bxjs@tmpb \fi
  \ifx\f@ncyorh\bxjs@tmpa \global\let\f@ncyorh\bxjs@tmpb \fi
  \def\bxjs@tmpa{\fancyplain{}{\sl\leftmark}\strut}%
  \def\bxjs@tmpb{\fancyplain{}{\leftmark}\strut}%
  \ifx\f@ncyelh\bxjs@tmpa \global\let\f@ncyelh\bxjs@tmpb \fi
  \ifx\f@ncyerh\bxjs@tmpa \global\let\f@ncyerh\bxjs@tmpb \fi
  \ifx\f@ncyolh\bxjs@tmpa \global\let\f@ncyolh\bxjs@tmpb \fi
  \ifx\f@ncyorh\bxjs@tmpa \global\let\f@ncyorh\bxjs@tmpb \fi
  \def\bxjs@tmpa{\rm\thepage\strut}%
  \def\bxjs@tmpb{\thepage\strut}%
  \ifx\f@ncyecf\bxjs@tmpa \global\let\f@ncyecf\bxjs@tmpb \fi
  \ifx\f@ncyocf\bxjs@tmpa \global\let\f@ncyocf\bxjs@tmpb \fi
```

\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。

```
  \ifx\fullwidth\@undefined\else \ifdim\textwidth<\fullwidth
    \setlength{\@tempdima}{\fullwidth-\textwidth}%
    \edef\bxjs@tmpa{\noexpand\fancyhfoffset[EL,OR]{\the\@tempdima}%
    }\bxjs@tmpa
  \fi\fi
  \PackageInfo\bxjs@clsname
   {Patch to fancyhdr is applied\@gobble}}
```

\bxjs@pagestyle@hook \pagestyle へのフックの本体。

```
\def\bxjs@pagestyle@hook{%
  \@ifpackageloaded{fancyhdr}{%
    \bxjs@adjust@fancyhdr
    \global\let\bxjs@adjust@fancyhdr\relax
  }{}}
```

\pagestyle にフックを入れ込む。

```
\let\bxjs@org@pagestyle\pagestyle
\def\pagestyle{%
  \bxjs@pagestyle@hook \bxjs@org@pagestyle}
```

begin-document フック。

※これ以降に fancyhdr が読み込まれることはあり得ない。

```
\g@addto@macro\bxjs@begin@document@hook{%
  \bxjs@pagestyle@hook
  \global\let\bxjs@pagestyle@hook\relax}
```

終わり。

```
\fi
```

以上で終わり。

```
%</minimal>
```

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
- XETEX：　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 ε-TEX 拡張を必須とする。
  ※ bxcjkjatype パッケージが ε-TEX 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。

```
\ifjsWitheTeX
  \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。

```
\bxjs@cslet{bxjs@kv@jis2004@true}\bxjs@jp@jismmivtrue
\bxjs@cslet{bxjs@kv@jis2004@false}\bxjs@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。

```
\let\bxjs@kv@units@true\bxjs@jp@unitstrue
\let\bxjs@kv@units@false\bxjs@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。

※ 2.9 版より、複数回指定した場合には累積させる。

```
\define@key{bxjsStd}{font}{%
  \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。

```
\let\bxjs@kv@strongcmd@true\bxjs@jp@strong@cmdtrue
\let\bxjs@kv@strongcmd@false\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)

```
\let\jafontsize\jsJaFontSize
```

■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。

```
\@onlypreamble\bxjs@apply@mmiv
\def\bxjs@apply@mmiv{%
  \bxjs@add@class@option{jis2004}
% \@ifpackagewith 判定への対策
  \PassOptionsToPackage{jis2004}{otf}
  \global\let\bxjs@apply@mmiv\relax}
\ifbxjs@jp@jismmiv \bxjs@apply@mmiv \fi
```

■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。

```
\if j\jsEngine
  \bxjs@jp@unitsfalse
\fi
```

units パラメタが有効の場合は、bxcalc パッケージの \usepTeXunits 命令を実行して和文用単位を有効化する。

```
\ifbxjs@jp@units
  \IfFileExists{bxcalc.sty}{%
    \RequirePackage{bxcalc}[2018/01/28]%v1.0a
    \ifx\usepTeXunits\@undefined
      \PackageWarningNoLine\bxjs@clsname
       {Cannot support pTeX units (zw etc.), since\MessageBreak
        the package 'bxcalc' is too old}%
      \bxjs@jp@unitsfalse
    \else \usepTeXunits
    \fi
  }{%else
    \PackageWarningNoLine\bxjs@clsname
     {Cannot support pTeX units (zw etc.), since\MessageBreak
      the package 'bxcalc' is unavailable}%
    \bxjs@jp@unitsfalse
  }
\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。

```
\ifbxjs@jp@units
\def\bxjs@parse@qh#1{\let\bxjs@tmpb\relax}
\let\bxjs@parse@qh@a\@undefined
\let\bxjs@parse@qh@b\@undefined
\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
strongenv (*env.*)

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 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
    \fi
  \else
    \let\bxjs@tmpa\jsJaFont
  \fi
  \if f#1\ifx\bxjs@tmpa\bxjs@@noEmbed
    \ClassWarningNoLine\bxjs@clsname
     {Option 'jafont=noEmbed' is ignored, because it is\MessageBreak
      not available on the current situation}%
```

```
    \let\bxjs@tmpa\@empty
  \fi\fi
}
\def\bxjs@@auto{auto}
\def\bxjs@@noEmbed{noEmbed}
\def\bxjs@@hziv{-04}
```

`\bxjs@jaEmbed` `\bxjs@jaVariant` 現在の updmap の jaEmbed・jaVariant パラメタの値。\bxjs@get@kanjiEmbed により実際の設定値が取得されてここに設定される。

※古い版の updmap では kanjiEmbed・kanjiVariant であった。

```
\let\bxjs@jaEmbed\relax
\let\bxjs@jaVariant\relax
```

`\bxjs@get@kanjiEmbed` 現在の updmap の jaEmbed・jaVariant パラメタの値を取得する。

```
\@onlypreamble\bxjs@get@kanjiEmbed
\def\bxjs@get@kanjiEmbed{%
  \begingroup\setbox\z@=\hbox{%
    \global\let\bxjs@tmpdo\@empty
    \def\bxjs@next##1##2##3{%
      \def##1####1##3 ####2\@nil####3\@nnil{%
        \ifx$####1$\gdef##2{####2}\fi}%
      \g@addto@macro\bxjs@tmpdo{%
        \expandafter##1\bxjs@tmpa\@nil##3 \@nil\@nnil}}%
    \bxjs@next\bxjs@tmpdo@a\bxjs@g@tmpa{kanjiEmbed}%
    \bxjs@next\bxjs@tmpdo@b\bxjs@g@tmpa{jaEmbed}%
    \bxjs@next\bxjs@tmpdo@c\bxjs@g@tmpb{kanjiVariant}%
    \bxjs@next\bxjs@tmpdo@d\bxjs@g@tmpb{jaVariant}%
%
    \global\let\bxjs@g@tmpa\relax
    \global\let\bxjs@g@tmpb\relax
    \endlinechar\m@ne
    \let\do\@makeother\dospecials
    \catcode32=10 \catcode12=10 %form-feed
    \let\bxjs@tmpa\@empty
    \openin\@inputcheck="|kpsewhich updmap.cfg"\relax
    \ifeof\@inputcheck\else
      \read\@inputcheck to\bxjs@tmpa
      \closein\@inputcheck
    \fi
    \ifx\bxjs@tmpa\@empty\else
      \openin\@inputcheck="\bxjs@tmpa"\relax
      \@tempswatrue
      \loop\if@tempswa
        \read\@inputcheck to\bxjs@tmpa
        \bxjs@tmpdo
        \ifeof\@inputcheck \@tempswafalse \fi
      \repeat
    \fi
  }\endgroup
```

```
  \let\bxjs@jaEmbed\bxjs@g@tmpa
  \let\bxjs@jaVariant\bxjs@g@tmpb
}
```

\bxjs@resolve@jafont@paren jafont パラメタ値内の () 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。

```
\@onlypreamble\bxjs@resolve@jafont@paren
\def\bxjs@resolve@jafont@paren#1{%
  \def\bxjs@tmpb{\let#1}%
  \expandafter\bxjs@resolve@jafont@paren@a#1\@nil()\@nil\@nnil#1}
\@onlypreamble\bxjs@resolve@jafont@paren@a
\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 で先頭の“文字”を拾ってそれを \bxjs@jachar に渡す。

```
    \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。

```
\let\jsJaChar\jachar
```

下請けの \bxjs@jachar の実装はエンジンにより異なる。

```
\let\bxjs@jachar\@firstofone
```

■hyperref 対策 出力ページサイズに館する処理は 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。

```
\@onlypreamble\bxjs@fix@hyperref@unicode
\def\bxjs@fix@hyperref@unicode#1{%
  \PassOptionsToPackage{bxjs/hook=#1}{hyperref}%
```

```
  \@namedef{KV@Hyp@bxjs/hook}##1{%
    \KV@Hyp@unicode{##1}%
    \def\KV@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\else
      \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
```

LaTeX 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。

```
\ifbxjs@old@hook@system
\def\bxjs@urgent@special#1{%
  \AtBeginDvi{\special{#1}}%
  \g@addto@macro\bxjs@begin@document@hook{%
    \@ifpackageloaded{atbegshi}{%
      \begingroup
```

```
        \toks\z@{\special{#1}}%
        \toks\tw@\expandafter{\AtBegShi@HookFirst}%
        \xdef\AtBegShi@HookFirst{\the\toks@\the\toks\tw@}%
      \endgroup
    }{}%
  }%
}
```

導入済の場合。

※自分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。

```
\else
  \def\bxjs@urgent@special#1{%
    \bxjs@urgent@special@a
    \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
```

■bm パッケージ対策　\reDeclareMathAlphabet を適用した数式英字フォント命令は通常の場合とは定義文の形が異なる。このため bm パッケージを利用して \bm{\mathrm{A}} を実行するとエラーが発生する。これを回避するため、「\bm の引数中では数式英字フォントの和文連動を無効にする」ことにする。

\reDeclareMathAlphabet の適用により例えば \mathrm は以下のように変更される。

- 適用前は \mathrm の一回展開は \protect\[mathrm␣] である。
- 適用後は \mathrm の一回展開が \protect\[mathrm␣␣] になる。
- \[mathrm␣␣] の一回展開は以下のようになる。
  \DualLang@mathalph@bet{\RDMAorg@mathrm}{\RDMAorg@mathmc}
- \RDMAorg@mathrm の一回展開は \[mathrm␣] になる。

\bxjs@patch@RDMA@for@bm　\reDeclareMathAlphabet の機能に対して bm パッケージ対策のパッチを当てる。

```
\@onlypreamble\bxjs@patch@RDMA@for@bm
\def\bxjs@patch@RDMA@for@bm{%
```

実際に改変するのは \DualLang@mathalph@bet である。

```
  \let\bxjs@org@DualLang@mathalph@bet\DualLang@mathalph@bet
  \def\DualLang@mathalph@bet{%
```

\bm の引数の中（そこでは \bm が \@firstofone に等置されているのでこれを判定に利用する）では、\DualLang@mathalph@bet を \@firstoftwo の動作に変える。これにより、\mathrm の（\protect を無視する場合の）先頭完全展開形が、「適用」前のものと一致する。このため \bm は「適用」の影響を受けずに正常動作できる。

```
    \ifx\bm\@firstofone \expandafter\@firstoftwo
    \else \expandafter\bxjs@org@DualLang@mathalph@bet
    \fi}%
```

```
}
```

C.4 pTEX 用設定

```
\if j\jsEngine
```

■共通命令の実装

```
\newcommand*{\setkanjiskip}{\jsSetKanjiSkip}
\newcommand*{\getkanjiskip}{\jsGetKanjiSkip}
\def\jsApplyKanjiSkip#1{%
  \kanjiskip=#1\relax}
\newcommand*{\setxkanjiskip}{\jsSetXKanjiSkip}
\newcommand*{\getxkanjiskip}{\jsGetXKanjiSkip}
\def\jsApplyXKanjiSkip#1{%
  \xkanjiskip=#1\relax}
```

\jaJaChar のサブマクロ。

```
\def\bxjs@jachar#1{%
  \bxjs@jachar@a#1....\@nil}
\def\bxjs@jachar@a#1#2#3#4#5\@nil{%
```

引数が単一トークンなら和文文字トークンが得られたと見なしてそれをそのまま出力する。

```
  \ifx.#2#1%
```

引数が複数トークンの場合は、UTF-8 のバイト列であると見なし、そのスカラー値を \@tempcnta に代入する。

```
  \else\ifx.#3%
    \@tempcnta`#1 \multiply\@tempcnta64
    \advance\@tempcnta`#2 \advance\@tempcnta-"3080
    \bxjs@jachar@b
  \else\ifx.#4%
    \@tempcnta`#1 \multiply\@tempcnta64
    \advance\@tempcnta`#2 \multiply\@tempcnta64
    \advance\@tempcnta`#3 \advance\@tempcnta-"E2080
    \bxjs@jachar@b
  \else
    \@tempcnta`#1 \multiply\@tempcnta64
    \advance\@tempcnta`#2 \multiply\@tempcnta64
    \advance\@tempcnta`#3 \multiply\@tempcnta64
    \advance\@tempcnta`#4 \advance\@tempcnta-"3C82080
    \bxjs@jachar@b
  \fi\fi\fi}
```

符号値が \@tempcnta の和文文字を出力する処理。

```
\ifjsWithupTeX
  \def\bxjs@jachar@b{\kchar\@tempcnta}
\else
  \def\bxjs@jachar@b{%
    \ifx\bxUInt\@undefined\else
      \bxUInt{\@tempcnta}%
```

```
    \fi}
\fi
```

和欧文間空白の命令 \jathinspace の実装。

```
\ifbxjs@jaspace@cmd
  \def\jathinspace{\hskip\xkanjiskip}
\fi
```

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

```
\ifbxjs@jp@jismmiv
  \PassOptionsToPackage{prefer2004jis}{pxchfon}
  \PassOptionsToPackage{prefer2004jis}{pxbabel}
\fi
```

■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 でもない場合は、\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タ font が指定されいる場合は、その値を pxchfon のオプションに追加する。

```
\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@jp@font}
\ifx\bxjs@tmpa\@empty\else
  \edef\bxjs@next{%
    \noexpand\RequirePackage[\bxjs@tmpa]{pxchfon}[2010/05/12]% v0.5
  }\bxjs@next
\fi
```

■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……。)

```
\begingroup
  \global\let\bxjs@g@tmpa\relax
  \catcode`\|=0 \catcode`\\=12
  |def|bxjs@tmpdo#1|@nil{%
    |bxjs@tmpdo@a#1|@nil\RequirePackage|@nnil}%
  |def|bxjs@tmpdo@a#1\RequirePackage#2|@nnil{%
    |ifx$#1$|bxjs@tmpdo@b#2|@nil keyval|@nnil |fi}%
```

```
  |catcode`|\=0 \catcode`\|=12
  \def\bxjs@tmpdo@b#1keyval#2\@nnil{%
    \ifx$#2$\else
      \xdef\bxjs@g@tmpa{%
        \noexpand\PassOptionsToPackage{scale=\jsScale}{otf}}%
    \fi}
\@firstofone{%
  \catcode10=12 \endlinechar\m@ne
  \let\do\@makeother \dospecials \catcode32=10
  \openin\@inputcheck=otf.sty\relax
  \@tempswatrue
  \loop\if@tempswa
    \ifeof\@inputcheck \@tempswafalse \fi
    \if@tempswa
      \read\@inputcheck to\bxjs@next
      \expandafter\bxjs@tmpdo\bxjs@next\@nil
    \fi
  \repeat
  \closein\@inputcheck
\endgroup}
\bxjs@g@tmpa
```

■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]：upLaTeX 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。

```
\ifbxjs@hyperref@enc
```

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
    \let\bxjs@plautopatch@new=t}{}
```

「unicode を有効にできるか」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こと」を検査する。
※ pxjahyper-uni.def をもつ pxjahyper の版であれば、upLaTeX 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 upLaTeX 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。

```
      \ifbxjs@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 pLaTeX である場合の処理。pxjahyper が「pLaTeX 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 も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 がカーネルが新しい」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}{\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}
      \PassOptionsToPackage{bigcode}{pxjahyper}
    \else
      \bxjs@urgent@special{pdf:tounicode UTF8-UCS2}
      \PassOptionsToPackage{nobigcode}{pxjahyper}
    \fi
  \fi\fi\fi
  \let\bxToUnicodeSpecialDone=t
  \fi
\fi
```

■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
```

実際に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{%
    \ifx\reDeclareMathAlphabet\@undefined\else
```

bm パッケージが読込済であればパッチを適用する。

```
      \@ifpackageloaded{bm}{\bxjs@patch@RDMA@for@bm}{}%
      \reDeclareMathAlphabet{\mathrm}{\@mathrm}{\@mathmc}%
      \reDeclareMathAlphabet{\mathbf}{\@mathbf}{\@mathgt}%
      \reDeclareMathAlphabet{\mathsf}{\@mathsf}{\@mathgt}%
    \fi}
\fi
```

C.5　pdfTEX 用設定：CJK + bxcjkjatype

```
\else\if \if p\jsEngine T\else\if n\jsEngine T\else F\fi\fi T
```

■bxcjkjatype パッケージの読込　\jsJaFont 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 \bxjs@get@kanjiEmbed を実行する。スケール値（\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。

```
\bxjs@adjust@jafont{f}
\let\bxjs@jafont@paren\@firstofone
```

```
\bxjs@resolve@jafont@paren\bxjs@tmpa
\edef\bxjs@tmpa{\bxjs@catopt\bxjs@tmpa\bxjs@jp@font}
\edef\bxjs@tmpa{\bxjs@catopt\bxjs@tmpa{whole}}
\ifx\bxjs@jadriver\bxjs@@pandoc\else
  \edef\bxjs@tmpa{\bxjs@catopt\bxjs@tmpa{autotilde}}
\fi
\edef\bxjs@next{%
  \noexpand\RequirePackage[\bxjs@tmpa]{bxcjkjatype}[2013/10/15]% v0.2c
}\bxjs@next
\bxjs@cjk@loaded
```

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

```
\ifbxjs@hyperref@enc
  \PassOptionsToPackage{unicode}{hyperref}
\fi
```

\hypersetup 命令で（CJK* 環境に入れなくても）日本語文字を含む文書情報を設定できるようにするための細工。
※ bxcjkjatype を whole 付きで使っていることが前提。
※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。

```
\ifx\bxcjkjatypeHyperrefPatchDone\@undefined
\begingroup
  \CJK@input{UTF8.bdg}
\endgroup
\g@addto@macro\pdfstringdefPreHook{%
  \@nameuse{CJK@UTF8Binding}%
}
\fi
```

~ 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。

```
\ifx\bxcjkjatypeHyperrefPatchDone\@undefined
\g@addto@macro\pdfstringdefPreHook{%
  \ifx~\bxjs@@CJKtilde
    \let\bxjs@org@LetUnexpandableSpace\HyPsd@LetUnexpandableSpace
    \let\HyPsd@LetUnexpandableSpace\bxjs@LetUnexpandableSpace
    \let~\@empty
  \fi
}
\def\bxjs@@CJKtilde{\CJKecglue\ignorespaces}
\def\bxjs@@tildecmd{~}
\def\bxjs@LetUnexpandableSpace#1{%
  \def\bxjs@tmpa{#1}\ifx\bxjs@tmpa\bxjs@@tildecmd\else
    \bxjs@org@LetUnexpandableSpace#1%
  \fi}
\fi
```

■共通命令の実装

```
\newskip\jsKanjiSkip
\newskip\jsXKanjiSkip
\ifx\CJKecglue\@undefined
  \def\CJKtilde{\CJK@global\def~{\CJKecglue\ignorespaces}}
\fi
\newcommand*{\setkanjiskip}{\jsSetKanjiSkip}
\newcommand*{\getkanjiskip}{\jsGetKanjiSkip}
\newcommand*{\autospacing}{\jsEnableKanjiSkip}
\newcommand*{\noautospacing}{\jsDisableKanjiSkip}
\protected\def\bxjs@CJKglue{\hskip\jsKanjiSkip}
\def\jsApplyKanjiSkip#1{%
  \jsKanjiSkip=#1\relax
  \let\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
  \let\CJKecglue\bxjs@CJKecglue}
```

\jachar のサブマクロの実装。

```
\def\bxjs@jachar#1{%
  \CJKforced{#1}}
```

和欧文間空白の命令 \jathinspace の実装。

```
\ifbxjs@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 @enablejfam は常に偽になる。

```
\ifx t\bxjs@enablejfam
  \ClassWarningNoLine\bxjs@clsname
   {You cannot use 'enablejfam=true', since the\MessageBreak
    CJK package does not support Japanese math}
\fi
```

C.6　XeTeX 用設定：xeCJK + zxjatype

```
\else\if x\jsEngine
```

■zxjatype パッケージの読込　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
```

```
4824   \ClassError\bxjs@clsname
4825   {xeCJK or zxjatype is too old}\@ehc
4826 \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。

```
4827 \bxjs@adjust@jafont{f}
4828 \let\bxjs@jafont@paren\@gobble
4829 \bxjs@resolve@jafont@paren\bxjs@tmpa
4830 \edef\bxjs@tmpa{\bxjs@catopt\bxjs@tmpa\bxjs@jp@font}
4831 \ifx\bxjs@tmpa\@empty
4832   \setCJKmainfont[BoldFont=HaranoAjiGothic-Medium.otf]{HaranoAjiMincho-
     Regular.otf}
4833   \setCJKsansfont[BoldFont=HaranoAjiGothic-Medium.otf]{HaranoAjiGothic-
     Medium.otf}
4834 \else
4835   \edef\bxjs@next{%
4836     \noexpand\RequirePackage[\bxjs@tmpa]{zxjafont}[2013/01/28]% v0.2a
4837   }\bxjs@next
4838 \fi
```

■hyperref 対策　unicode オプションの指定に関する話。

XeTeX 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、XeTeX 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、「XeTeX 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。

```
4839 \ifx\strcmp\@undefined\else % 未定義なら条件を満たさない
4840 \ifnum\strcmp{\the\XeTeXversion\XeTeXrevision}{0.99992}>\m@ne
4841   \ifbxjs@hyperref@enc
4842     \PassOptionsToPackage{unicode}{hyperref}
4843   \fi
4844 \fi
4845 \fi
```

■段落頭でのグルー挿入禁止　どうやら、zxjatype の \inhibitglue の実装が極めて杜撰なため、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する。

強制改行直後のグルー禁止処理、のような怪しげな何か。

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

■共通命令の実装

```
\newskip\jsKanjiSkip
\newskip\jsXKanjiSkip
\ifx\CJKecglue\@undefined
  \def\CJKtilde{\CJK@global\def~{\CJKecglue\ignorespaces}}
\fi
\newcommand*{\setkanjiskip}{\jsSetKanjiSkip}
\newcommand*{\getkanjiskip}{\jsGetKanjiSkip}
\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
  \protected\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
  \protected\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。

```
\ifx t\bxjs@enablejfam
  \@enablejfamtrue
\fi
```

実際に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\if l\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[scale=\jsScale]{luatexja-fontspec}[2015/08/26]%
}\bxjs@next
```

\set@fontsize へのパッチ適用を再度行う。

```
\bxjs@patch@set@fontsize
```

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。

```
\bxjs@adjust@jafont{t}
\ifx\bxjs@tmpa\bxjs@@noEmbed
  \def\bxjs@tmpa{noembed}
\fi
\let\bxjs@jafont@paren\@gobble
\bxjs@resolve@jafont@paren\bxjs@tmpa
\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。

```
\@ifpackagelater{luatexja}{2016/03/31}{}{%else
```

```
\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}
}
\bxjs@nclong\def\jttdefault{\gtdefault}
\unless\ifx\@ltj@match@familytrue\@undefined
  \@ltj@match@familytrue
\fi
\g@addto@macro\bxjs@begin@document@hook{%
  \@ifpackageloaded{bm}{\bxjs@patch@RDMA@for@bm}{}%
  \reDeclareMathAlphabet{\mathrm}{\mathrm}{\mathmc}%
  \reDeclareMathAlphabet{\mathbf}{\mathbf}{\mathgt}%
  \reDeclareMathAlphabet{\mathsf}{\mathsf}{\mathgt}}%
\bxjs@if@sf@default{%
  \renewcommand\kanjifamilydefault{\gtdefault}}
```

■和文パラメタの設定

```
% 次の 3 つは既定値の通り
%\ltjsetparameter{prebreakpenalty={`’,10000}}
%\ltjsetparameter{postbreakpenalty={`“,10000}}
%\ltjsetparameter{prebreakpenalty={`”,10000}}
\ltjsetparameter{jaxspmode={`！,1}}
\ltjsetparameter{jaxspmode={`〒,2}}
\ltjsetparameter{alxspmode={`+,3}}
\ltjsetparameter{alxspmode={`\%,3}}
```

■段落頭でのグルー挿入禁止　基本的に現状の `ltjs*` 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に備えて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
```

`ltjs*` 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{}\fi}}
  \ifnum\bxjs@everyparhook=\bxjs@everyparhook@@compat
    \@gobble\if\def\bxjs@tmpa{\@{\everypar{\everyparhook}\fi}}\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(\@\@xsect, [[\meaning\@xsect]],
      \@{\hskip-\@tempskipa}, \@\ltjfakeparbegin)
    patchcmd(\@\@item, [[\meaning\@item]],
      \bxjs@tmpa, \@\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}
\newcommand*{\getkanjiskip}{\jsGetKanjiSkip}
\newcommand*{\setxkanjiskip}{\jsSetXKanjiSkip}
\newcommand*{\getxkanjiskip}{\jsGetXKanjiSkip}
\protected\def\autospacing{%
  \ltjsetparameter{autospacing=true}}
\protected\def\noautospacing{%
  \ltjsetparameter{autospacing=false}}
\protected\def\autoxspacing{%
  \ltjsetparameter{autoxspacing=true}}
\protected\def\noautoxspacing{%
  \ltjsetparameter{autoxspacing=false}}
\def\jsApplyKanjiSkip#1{%
  \ltjsetparameter{kanjiskip={#1}}}
\def\jsApplyXKanjiSkip#1{%
  \ltjsetparameter{xkanjiskip={#1}}}
```

\jachar のサブマクロの実装。

```
\def\bxjs@jachar#1{%
  \ltjjachar`#1\relax}
```

\jathinspace の実装。

```
\ifbxjs@jaspace@cmd
  \protected\def\jathinspace{%
    \hskip\ltjgetparameter{xkanjiskip}\relax}
```

```
\fi
```

■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な設定は済んでいる。従って @enablejfam は常に真になる。

```
\ifx f\bxjs@enablejfam
  \ClassWarningNoLine\bxjs@clsname
   {You cannot use 'enablejfam=false', since the\MessageBreak
    LuaTeX-ja always provides Japanese math families}
\fi
```

C.8　共通処理 (2)

```
\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 を利用する。

\textgt
```
\ifx\DeclareFixJFMCJKTextFontCommand\@undefined
\DeclareJaTextFontCommand{\textmc}{\mcfamily}
\DeclareJaTextFontCommand{\textgt}{\gtfamily}
\fi
```

\mathmc この時点で未定義である場合に限り、\DeclareJaMathFontCommand を利用したフォールバックの定義を行う。

\mathgt
```
\ifx\mathmc\@undefined
  \DeclareJaMathFontCommand{\mathmc}{\mcfamily}
  \DeclareJaMathFontCommand{\mathgt}{\gtfamily}
\fi
```

以上で終わり。

```
%</standard>
```

付録 D　和文ドライバ：modern

モダーンな設定。

standard ドライバの設定を引き継ぐ。

```
%<*modern>
\input{bxjsja-standard.def}
```

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。

```
\ifnum0\if x\jsEngine1\fi\if l\jsEngine1\fi=\z@
\def\encodingdefault{T1}%
\input{t1enc.def}%
```

```
\fontencoding\encodingdefault\selectfont
\fi
```

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。

```
\ifnum0\if x\jsEngine1\fi\if l\jsEngine1\fi=\z@
\renewcommand{\rmdefault}{lmr}
\renewcommand{\sfdefault}{lmss}
\renewcommand{\ttdefault}{lmtt}
\fi
```

大型演算子用の数式フォントの設定。
※ `amsfonts` パッケージと同等にする。

```
\DeclareFontShape{OMX}{cmex}{m}{n}{%
  <-7.5>cmex7<7.5-8.5>cmex8%
  <8.5-9.5>cmex9<9.5->cmex10}{}%
\expandafter\let\csname OMX/cmex/m/n/10\endcsname\relax
```

`amsmath` 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。

```
\ifx\@IncludeInRelease\@undefined
\RequirePackage{fixltx2e}
\fi
```

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。

```
\RequirePackage{bxjscjkcat}
```

D.4 完了

おしまい。

```
%</modern>
```

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

```
%<*pandoc>
```

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。

```
\if x\jsEngine
  \PassOptionsToPackage{nospace}{xeCJK}
\fi
```

standard ドライバの設定を引き継ぐ。

```
\input{bxjsja-standard.def}
```

■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。

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 ε-TEX 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}
```

ε-TEX ではない場合に警告を出す。

```
\ifjsWitheTeX\else
  \ClassWarningNoLine\bxjs@clsname
   {!!!!!!!! WARNING !!!!!!!!\MessageBreak
    This engine does not support e-TeX extension!\MessageBreak
    Some feature might not work properly}
\fi
```

`\ifbxjs@bxghost@available` 〔スイッチ〕 bxghost パッケージが利用できるか。

```
\newif\ifbxjs@bxghost@available
\ifjsWitheTeX
  \RequirePackage{pdftexcmds}[2009/09/22]% v0.5
  \IfFileExists{bxghost.sty}{%
    \bxjs@bxghost@availabletrue
    \@namedef{bxjs@bgbv/79E70A0991967E27981832C84DB5DF99}{1}%v0.2.0
    \ifx\pdf@filemdfivesum\@undefined\else
      \expandafter\ifx\csname bxjs@bgbv/\pdf@filemdfivesum{bxghost.sty}%
          \endcsname\relax\else \bxjs@bxghost@availablefalse \fi
    \fi
  }{}
\fi
```

その他の依存パッケージを読み込む。

```
\RequirePackage{iftex}[2013/04/04]% v0.2
\ifjsWitheTeX
  \RequirePackage{etoolbox}[2010/08/21]% v2.0
  \RequirePackage{filehook}[2011/10/12]% v0.5d
\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 重要要素を補正するか。

```
\newif\ifbxjs@jp@fix@strong \bxjs@jp@fix@strongtrue
```

fix-strong オプションの処理。

```
\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 インラインコード要素を補正するか。

```
\newif\ifbxjs@jp@fix@code \bxjs@jp@fix@codetrue
```

fix-code オプションの処理。

```
\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}{}}
```

\bxjs@jp@strong 重要要素に適用される書体変更の種類。

```
\chardef\bxjs@jp@strong=0
```

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}{%
  \bxjs@set@keyval{strong}{#1}{}}
```

\ifbxjs@jp@or@indent プレアンブルでのレイアウト上書きを許可するか。既定値は真。
\ifbxjs@jp@or@secnumdepth
\ifbxjs@jp@or@block@heading

```
\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
```

クラスで pandoc+ が指定された場合、内部和文パラメタ _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 は必ずパラメタ列の先頭にあるので、個別のパラメ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]{%
    \bxjs@set@keyval{orindent}{#1}{}}
\let\bxjs@kv@orsecnumdepth@true\bxjs@jp@or@secnumdepthtrue
\let\bxjs@kv@orsecnumdepth@false\bxjs@jp@or@secnumdepthfalse
\define@key{bxjsPan}{or-secnumdepth}[true]{%
    \bxjs@set@keyval{orsecnumdepth}{#1}{}}
\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]{%
    \bxjs@set@keyval{orblockheading}{#1}{}}
```

実際の japaram の値を適用する。

```
\def\bxjs@next#1{\bxjs@safe@setkeys{bxjsPan}{#1}}
\expandafter\bxjs@next\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”の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。

\bxjs@set@dupload@proc `\bxjs@set@dupload@proc{⟨ファイル名⟩}{⟨定義本体⟩}`： 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。

```
\@onlypreamble\bxjs@set@dupload@proc
\def\bxjs@set@dupload@proc#1{%
    \expandafter\bxjs@set@dupload@proc@a\csname bxjs@dlp/#1\endcsname}
\@onlypreamble\bxjs@set@dupload@proc@a
\def\bxjs@set@dupload@proc@a#1{%
    \@onlypreamble#1\def#1##1}
\def\bxjs@unset@dupload@proc#1{%
    \bxjs@cslet{bxjs@dlp/#1}\@undefined}
```

\@if@ptions \@if@ptions の再定義。

```
\@onlypreamble\bxjs@org@if@ptions
\let\bxjs@org@if@ptions\@if@ptions
\@onlypreamble\bxjs@org@reset@ptions
\let\bxjs@org@reset@ptions\relax
```

```
\def\@if@ptions#1#2#3{%
  \let\bxjs@next\@secondoftwo
  \def\bxjs@tmpa{#1}\def\bxjs@tmpb{\@currext}%
  \ifx\bxjs@tmpa\bxjs@tmpb
    \expandafter\ifx\csname bxjs@dlp/#2.#1\endcsname\relax\else
      \let\bxjs@next\@firstoftwo \fi
  \fi
  \bxjs@next\bxjs@do@dupload@proc\bxjs@org@if@ptions{#1}{#2}{#3}}
\g@addto@macro\bxjs@begin@document@hook{%
  \let\@if@ptions\bxjs@org@if@ptions}
\@onlypreamble\bxjs@do@dupload@proc
\def\bxjs@do@dupload@proc#1#2#3{%
  \ifx\bxjs@org@reset@ptions\relax
    \let\bxjs@org@reset@ptions\@reset@ptions
  \fi
  \bxjs@csletcs{bxjs@next}{bxjs@dlp/#2.#1}%
  \def\@reset@ptions{%
    \let\@reset@ptions\bxjs@org@reset@ptions
    \@reset@ptions
    \bxjs@next{#3}}%
  \@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 LaTeX 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。
※そもそも BXJS クラスは日本語用の文書クラスであるため、もし言語設定が行われているのであれば「メイン言語は日本語である」であるはずなので、「サブ言語が日本語である」ことは考慮しない。

■Polyglossia について　現在 CTAN に登録されている日本語用の gloss ファイルは超絶アレでかつ有害な設定を行うため、これの読込を避ける必要がある。そのため、メイン言語が `japanese` である場合（古い Pandoc ではこの場合に引数が空の `\setmainlanguage{}` が実行されるがこのパターンも同様に扱う）には、Polyglossia の処理を無効化してしまうことにする。つまり、Polyglossia が提供する命令について、何もしないダミーの定義を与える。
※ Polyglossia は古い Pandoc のテンプレートにおいて、エンジンが XeTeX か LuaTeX の場合に利用されていた。

`\bxjs@polyglossia@options` Polyglossia 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@polyglossia@options\relax
```

エンジンが XeTeX か LuaTeX 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。

```
\ifnum0\if x\jsEngine1\fi\if l\jsEngine1\fi>0
```

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}%
      \bxjs@pandoc@polyglossia@ja
```

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。

```
    }{%else
      \ClassWarning\bxjs@clsname
       {Main language is '##2',\MessageBreak
        thus polyglossia will be loaded}%
      \csundef{ver@polyglossia.sty}%
      \edef\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。

```
\@onlypreamble\bxjs@pandoc@polyglossia@ja
\def\bxjs@pandoc@polyglossia@ja{%
```

```
  \renewcommand*\setmainlanguage[2][]{}%
  \newcommand*\setotherlanguage[2][]{%
    \ifblank{##2}{}{%else
      \cslet{##2}\@empty \cslet{end##2}\@empty
      \cslet{text##2}\@firstofone}}%
  \newcommand*\setotherlanguages[2][]{%
    \@for\bxjs@tmpa:={##2}\do{%
      \setotherlangauge{\bxjs@tmpa}}}%
```

Polyglossia の読込済マークは外れるようにしておく。

```
  \let\bxjs@polyglossia@options\relax}%
\fi
```

■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。
※ XeTeX では 2.15 版で、LuaTeX は 2.6 版で Polyglossia から Babel に変更されている。

`\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
```

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{babel}
\bxjs@set@dupload@proc{babel.sty}{%
  \bxjs@unset@dupload@proc{babel.sty}%
  \ClassWarning\bxjs@clsname
   {Package babel is requested}%
```

パ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。

```
  \@tempswafalse \let\bxjs@babel@options\@empty
  \def\bxjs@tmpb{main=}%
  \@for\bxjs@tmpa:=#1\do{%
    \ifx\bxjs@tmpa\bxjs@tmpb \def\bxjs@tmpa{main=japanese}\fi
    \edef\bxjs@babel@options{\bxjs@babel@options,\bxjs@tmpa}}%
  \bxjs@cslet{ver@babel.sty}\@undefined
  \edef\bxjs@next{%
    \noexpand\RequirePackage[\bxjs@babel@options]{babel}\relax
  }\bxjs@next
  \RequirePackage{bxorigcapt}\relax}
```

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

```
\g@addto@macro\bxjs@endpreamble@hook{%
  \ifx\bxjs@babel@options\relax
    \bxjs@cslet{ver@babel.sty}\@undefined
  \fi}
```

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 ＋ bxorigrcapt パッケージ」の状態にしておきたい。

```
\ifjsWitheTeX
```

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

```
\AtBeginOfFile{japanese.ldf}{\bxjs@begin@japanese@ldf@hook}
\def\bxjs@begin@japanese@ldf@hook{%
  \let\bxjs@begin@japanese@ldf@hook\relax
  \let\bxjs@save@ProvidesLanguage\ProvidesLanguage
  \let\bxjs@save@LdfInit\LdfInit
  \def\ProvidesLanguage##1[##2]{\bxjs@do@japanese@ldf{##2}}%
  \def\LdfInit##1##2{\bxjs@do@japanese@ldf{0000/00/00}}}
```

バージョンを判定する部分。

※\LdfInit にも細工を入れている理由は、初期の japanese.ldf には \ProvidesLanguage が記述されていないため。

```
\def\bxjs@do@japanese@ldf#1{\bxjs@do@japanese@ldf@a#1\@nil}
\def\bxjs@do@japanese@ldf@a#1/#2/#3#4#5\@nil{%
  \let\LdfInit\bxjs@save@LdfInit
  \ClassInfo\bxjs@clsname
   {Release date of japanese.ldf is:\MessageBreak
    \@spaces #1/#2/#3#4\@gobble}%
  \ifnum#1#2#3#4<20201206 % v3.0
    \let\bxjs@japanese@ldf@skipped=t\csuse{endinput}%
  \fi}
\AtEndOfFile{japanese.ldf}{\bxjs@end@japanese@ldf@hook}
\def\bxjs@end@japanese@ldf@hook{%
  \let\bxjs@end@japanese@ldf@hook\relax
  \let\ProvidesLanguage\bxjs@save@ProvidesLanguage
  \let\LdfInit\bxjs@save@LdfInit
  \ifx t\bxjs@japanese@ldf@skipped
    \ClassWarningNoLine\bxjs@clsname
     {Loading japanese.ldf is skipped}%
```

ダミーの言語定義。

```
    \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。

```
\ifjsWitheTeX
\@onlypreamble\bxjs@info@or@ban
\def\bxjs@info@or@ban#1{%
  \PackageInfo\bxjs@clsname
   {Freeze layout on '#1',\MessageBreak reported}}
```

■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。

```
\unless\ifbxjs@jp@or@indent
  \bxjs@info@or@ban{indent}
```

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

```
  \IfFileExists{parskip.sty}{%
    \pandocSkipLoadPackage{parskip}%
```

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。

```
  }{%else
    \eappto\bxjs@endpreamble@hook{%
      \parindent=\the\parindent\relax
      \parskip=\the\parskip\relax}}
\fi
```

■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。

```
\ifbxjs@jp@or@secnumdepth\else
  \bxjs@info@or@ban{secnumdepth}
  \eappto\bxjs@endpreamble@hook{%
    \c@secnumdepth=\the\c@secnumdepth\relax}
\fi
```

■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。

```
\ifbxjs@jp@or@block@heading\else
  \let\bxjs@frozen@paragraph\paragraph
  \let\bxjs@frozen@subparagraph\subparagraph
  \bxjs@info@or@ban{block-heading}
  \appto\bxjs@endpreamble@hook{%
  \let\oldparagraph\@undefined
  \let\paragraph\bxjs@frozen@paragraph
  \let\subparagraph\bxjs@frozen@subparagraph}
\fi
```

以上。

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 がプレアンブ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。

```
\g@addto@macro\bxjs@begin@document@hook{%
  \@tempswafalse
```

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。

```
\ifx\oldparagraph\@undefined\else
  \@tempswatrue
\fi
```

\paragraph が番号付きの場合は、マーク変更が必要である。

```
\ifnum\c@secnumdepth>3
  \@tempswatrue
\fi
```

「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。

```
\if@tempswa\ifx\jsParagraphMark\bxjs@org@paragraph@mark
  \let\jsParagraphMark\@empty
\fi\fi}
```

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)pLaTeX 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。

```
\ifnum0\if x\jsEngine1\fi\if l\jsEngine1\fi>\z@
  \catcode"3000=\active
  \begingroup \catcode`\!=7
  \protected\gdef!!!!3000{\pandocZWSpace}
  \endgroup
\else\ifx\DeclareUnicodeCharacter\@undefined\else
  \DeclareUnicodeCharacter{3000}{\bxjs@zsp@char}
  \bxjs@protected\def\bxjs@zsp@char{\pandocZWSpace}
\fi\fi
```

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。

```
\if j\jsEngine
  \bxjs@fix@hyperref@unicode{false}
\else
  \bxjs@fix@hyperref@unicode{true}
\fi
```

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 `\strong` を使いたいため、`\textbf` を書き換えてしまう（うわぁ）。

```
\ifbxjs@jp@fix@strong\ifbxjs@jp@strong@cmd
  \let\orgtextbf\textbf
  \DeclareRobustCommand\pandocTextbf[1]{%
    \begingroup
      \let\textbf\orgtextbf
      \strong{#1}%
    \endgroup}%
  \g@addto@macro\bxjs@begin@document@hook{%
    \let\textbf\pandocTextbf}
\fi\fi
```

`\strong` の書体を設定する。

```
\jsAtEndOfClass{%
  \ifx\strongfontdeclare\@undefined\else
    \ifcase\bxjs@jp@strong
    \or \strongfontdeclare{\sffamily}%
    \or \strongfontdeclare{\sffamily\bfseries}%
    \fi
  \fi}
```

■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。

```
\ifbxjs@jp@fix@code
```

bxghost パッケージが利用できる場合はその機能を利用する。使えない場合は自前実装を用いる。

```
  \ifbxjs@bxghost@available
    \RequirePackage[verb]{bxghost}[2020/01/31]% v0.3.0
    \let\bxjs@eghostguarded\eghostguarded
  \else
  \chardef\bxjs@eghost@c=23
  \ifx j\jsEngine \xspcode\bxjs@eghost@c=3
  \else\ifx l\jsEngine \ltjsetparameter{alxspmode={\bxjs@eghost@c,3}}
  \else\ifx x\jsEngine %no-op
  \else \let\bxjs@eghost@c\@undefined
  \fi\fi\fi
  \ifx\bxjs@eghost@c\@undefined\else
    \font\bxjs@eghost@f=ec-lmr10 at 1.23456pt
    \def\bxjs@pan@eghost{\bgroup\bxjs@eghost@f\bxjs@eghost@c\egroup}
    \def\bxjs@eghostguarded#1{%
      \bxjs@pan@eghost\null#1\null\bxjs@pan@eghost}
  \fi
```

```
  \fi
  \ifx\bxjs@eghostguarded\@undefined\else
    \let\orgtexttt\texttt
    \DeclareRobustCommand\pandocTexttt[1]{%
      \ifmmode \nfss@text{\ttfamily #1}%
      \else
        \ifvmode \leavevmode \fi
        \bxjs@eghostguarded{\begingroup\ttfamily#1\endgroup}%
      \fi}
    \g@addto@macro\bxjs@begin@document@hook{%
      \let\texttt\pandocTexttt}
```

bxghost を使わない場合の \verb の処理。

※ bxghost の実装を参考にした。

```
    \ifbxjs@bxghost@available\else
    \expandafter\def\expandafter\verb\expandafter{%
      \expandafter\bxjs@pan@eghost\verb}
    \g@addto@macro\verb@egroup{\bxjs@pan@eghost}
    \fi
  \fi
\fi
```

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 iftex パッケージが導入されて「pdfTEX の判定を直接 \ifPDFTeX で行う」ように改修された。このため、(u)pTEX 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ブル実行中に限って \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「pdfTEX 用の処理が実行される」前提が維持されるようにする。

```
\if j\jsEngine
```

\bxjs@check@frontier \bxjs@check@frontier\CS は現在のパッケージ読込ネストレベルが丁度 1 であるときにのみ \CS を実行する。

```
  \def\bxjs@check@frontier{%
    \expandafter\bxjs@check@frontier@a\@currnamestack\noindent...\@nil}
  \def\bxjs@check@frontier@a#1#2#3#4#5\@nil#6{%
    \ifx\noindent#4#6\fi}
```

\bxjs@unforge@ifPDFTeX \ifPDFTeX を偽（正しい値）にする。

```
\@onlypreamble\bxjs@unforge@ifPDFTeX
\def\bxjs@unforge@ifPDFTeX{\global\bxjs@csletcs{ifPDFTeX}{iffalse}}
```

`\bxjs@forge@ifPDFTeX` `\ifPDFTeX` を真（偽装した値）にする。

```
\@onlypreamble\bxjs@forge@ifPDFTeX
\def\bxjs@forge@ifPDFTeX{\global\bxjs@csletcs{ifPDFTeX}{iftrue}}
```

`\bxjs@unload@forge@ifPDFTeX` `\ifPDFTeX` に対する細工を無効化する。

```
\def\bxjs@unload@forge@ifPDFTeX{%
  \bxjs@unforge@ifPDFTeX
  \global\let\bxjs@check@frontier\@gobble}
```

プレアンブル開始時は `\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。

```
\jsAtEndOfClass{\bxjs@forge@ifPDFTeX}
\ifjsWitheTeX
  \AtBeginOfEveryFile{\bxjs@check@frontier\bxjs@unforge@ifPDFTeX}%
  \AtEndOfEveryFile{\bxjs@check@frontier\bxjs@forge@ifPDFTeX}%
  \g@addto@macro\bxjs@endpreamble@hook{\bxjs@unload@forge@ifPDFTeX}
\else
  \g@addto@macro\bxjs@begin@document@hook{\bxjs@unload@forge@ifPDFTeX}
\fi
\fi
```

E.14　完了

おしまい。

```
%</pandoc>
```

和文ドライバ実装はここまで。

```
%</drv>
```

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。
- bxjscjkcat：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。

```
%<*anc>
```

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

```
%<*compat>
\def\bxac@pkgname{bxjscompat}
```

`\bxjx@engine` エンジンの種別。

```
\let\bxac@engine=n
\def\bxac@do#1#2{%
  \edef\bxac@tmpa{\string#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}
```

`\bxac@delayed@if@bxjs` もし BXJS クラスの読込中で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。

```
\ifx\jsAtEndOfClass\@undefined
  \let\bxac@delayed@if@bxjs\@firstofone
\else \let\bxac@delayed@if@bxjs\jsAtEndOfClass
\fi
```

`\ImposeOldLuaTeXBehavior` `\RevokeOldLuaTeXBehavior` `\ImposeOldLuaTeXBehavior` は 0.85 版以降の LuaTeX を**一時的に** pdfTeX と互換であるように見せかける。`\RevokeOldLuaTeXBehavior` で元に戻すことができる。
※エンジンが LuaTeX 以外の場合は何もしない。

```
\newif\ifbxac@in@old@behavior
\let\ImposeOldLuaTeXBehavior\relax
\let\RevokeOldLuaTeXBehavior\relax
```

G.2 8bit 欧文 TeX

```
\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。

```
\@tempcnta="80 \loop \ifnum\@tempcnta<"100
  \catcode\@tempcnta\active
  \advance\@tempcnta\@ne
\repeat
```

以上。

```
\fi
```

G.3 XeTeX

```
\ifx x\bxac@engine
```

■文字クラスの設定 XeTeX 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 LaTeX 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、XeTeX 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。

```
\ifx\XeTeXcharclass\@undefined\else
\bxac@delayed@if@bxjs{%
  \@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。

```
    \ifx\xe@alloc@intercharclass\@undefined\else
        \ifnum\xe@alloc@intercharclass=\z@
      \PackageInfo\bxac@pkgname
        {Setting up interchar class for CJK...\@gobble}%
      \InputIfFileExists{load-unicode-xetex-classes.tex}{%
        \xe@alloc@intercharclass=3
      }{%else
        \PackageWarning\bxac@pkgname
          {Cannot find file 'load-unicode-xetex-classes.tex'%
           \@gobble}%
      }%
    \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。

```
    \ifnum\XeTeXcharclass"3041=\z@
      \PackageInfo\bxac@pkgname
        {Adjusting interchar class for CJK...\@gobble}%
      \@for\bxac@tmpb:={%
        3041,3043,3045,3047,3049,3063,3083,3085,3087,308E,%
        3095,3096,30A1,30A3,30A5,30A7,30A9,30C3,30E3,30E5,%
        30E7,30EE,30F5,30F6,30FC,31F0,31F1,31F2,31F3,31F4,%
        31F5,31F6,31F7,31F8,31F9,31FA,31FB,31FC,31FD,31FE,%
        31FF%
      }\do{\XeTeXcharclass"\bxac@tmpb=\@ne}%
    \fi
  }%
}
\fi
```

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。

```
\fi
```

G.4 LuaTEX

```
\ifx l\bxac@engine
```

0.82〜0.84 版の LuaTEX を（0.81 版以前と同様に）「pdfTEX の拡張である」ように見せかける処理。

※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

0.85 版以降であるかを検査する。

```
\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}\pdfhorigin{{\pdfvariable horigin}}%
  \do{\protected\edef}\pdfvorigin{{\pdfvariable vorigin}}}
\def\bxac@ob@do#1#2{\begingroup
  \expandafter\bxac@ob@do@a\csname bxac@\string#2\endcsname{#1}#2}
\def\bxac@ob@do@a#1#2#3#4{\endgroup
  \ifbxac@in@old@behavior \let#1#3\relax #2#3#4\relax
  \else \let#3#1\relax \let#1\@undefined
  \fi}
\protected\def\ImposeOldLuaTeXBehavior{%
  \unless\ifbxac@in@old@behavior
    \bxac@in@old@behaviortrue
    \let\do\bxac@ob@do \bxac@ob@list
  \fi}
\protected\def\RevokeOldLuaTeXBehavior{%
  \ifbxac@in@old@behavior
    \bxac@in@old@behaviorfalse
    \let\do\bxac@ob@do \bxac@ob@list
  \fi}
\fi
```

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(0x2A700, 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。

```
\fi
```

G.5 完了

おしまい。

```
%</compat>
```

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

```
%<*cjkcat>
\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{%
  \ifx\ucs\@undefined\else \ifnum\ucs"3000="3000
    \let\bxjx@engine=u\fi\fi}
\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。

```
\if u\bxjx@engine
\@for\bxjx@tmpa:={%
0080,0100,0180,0250,02B0,0300,0500,0530,0590,0600,%
0700,0750,0780,07C0,0800,0840,0860,08A0,0900,0980,%
0A00,0A80,0B00,0B80,0C00,0C80,0D00,0D80,0E00,0E80,%
0F00,1000,10A0,1200,1380,13A0,1400,1680,16A0,1700,%
1720,1740,1760,1780,1800,18B0,1900,1950,1980,19E0,%
1A00,1A20,1AB0,1B00,1B80,1BC0,1C00,1C50,1C80,1CC0,%
1CD0,1D00,1D80,1DC0,1E00,1F00,2440,27C0,27F0,2800,%
2A00,2C00,2C60,2C80,2D00,2D30,2D80,2DE0,2E00,4DC0,%
A4D0,A500,A640,A6A0,A700,A720,A800,A830,A840,A880,%
A8E0,A900,A930,A980,A9E0,AA00,AA60,AA80,AAE0,AB00,%
AB30,AB70,ABC0,D800,DB80,DC00,E000,FB00,FB50,FE00,%
FE70,FFF0,%
10000,10080,10100,10140,10190,101D0,10280,102A0,%
```

```
102E0,10300,10330,10350,10380,103A0,10400,10450,%
10480,104B0,10500,10530,10600,10800,10840,10860,%
10880,108E0,10900,10920,10980,109A0,10A00,10A60,%
10A80,10AC0,10B00,10B40,10B60,10B80,10C00,10C80,%
10E60,11000,11080,110D0,11100,11150,11180,111E0,%
11200,11280,112B0,11300,11400,11480,11580,11600,%
11660,11680,11700,118A0,11A00,11A50,11AC0,11C00,%
11C70,11D00,12000,12400,12480,13000,14400,16800,%
16A40,16AD0,16B00,16F00,1BC00,1BCA0,1D000,1D100,%
1D200,1D300,1D360,1D400,1D800,1E000,1E800,1E900,%
1EE00,1F000,1F030,1F0A0,1F300,1F600,1F650,1F680,%
1F700,1F780,1F800,1F900,E0000,E0100,F0000,100000,%
00C0%
}\do{%
\@tempcnta="\bxjx@tmpa\relax
\@tempcntb\@tempcnta \advance\@tempcntb\m@ne
\chardef\bxjx@tmpb\kcatcode\@tempcntb
\kcatcode\@tempcnta=15 \kcatcode\@tempcntb\bxjx@tmpb}
\fi
```

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」**とは以下に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 曖昧文字」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⟩`}{`⟨対象 fontenc⟩`}{`⟨テキスト LICR⟩`}{`⟨数式 LICR⟩`}`
※数式で使わない文字は ⟨数式 LICR⟩ を空にする。

```
\@onlypreamble\bxjx@grkcyr@list
\def\bxjx@grkcyr@list{%
\do{0391}{LGR}{\textAlpha}{A}%          % GR. C. L. ALPHA
\do{0392}{LGR}{\textBeta}{B}%           % GR. C. L. BETA
\do{0393}{LGR}{\textGamma}{\Gamma}%     % GR. C. L. GAMMA
\do{0394}{LGR}{\textDelta}{\Delta}%     % GR. C. L. DELTA
\do{0395}{LGR}{\textEpsilon}{E}%        % GR. C. L. EPSILON
\do{0396}{LGR}{\textZeta}{Z}%           % GR. C. L. ZETA
\do{0397}{LGR}{\textEta}{H}%            % GR. C. L. ETA
\do{0398}{LGR}{\textTheta}{\Theta}%     % GR. C. L. THETA
\do{0399}{LGR}{\textIota}{I}%           % GR. C. L. IOTA
\do{039A}{LGR}{\textKappa}{K}%          % GR. C. L. KAPPA
\do{039B}{LGR}{\textLambda}{\Lambda}%   % GR. C. L. LAMDA
\do{039C}{LGR}{\textMu}{M}%             % GR. C. L. MU
\do{039D}{LGR}{\textNu}{N}%             % GR. C. L. NU
```

```
\do{039E}{LGR}{\textXi}{\Xi}%             % GR. C. L. XI
\do{039F}{LGR}{\textOmicron}{O}%          % GR. C. L. OMICRON
\do{03A0}{LGR}{\textPi}{\Pi}%             % GR. C. L. PI
\do{03A1}{LGR}{\textRho}{P}%              % GR. C. L. RHO
\do{03A3}{LGR}{\textSigma}{\Sigma}%       % GR. C. L. SIGMA
\do{03A4}{LGR}{\textTau}{T}%              % GR. C. L. TAU
\do{03A5}{LGR}{\textUpsilon}{\Upsilon}%   % GR. C. L. UPSILON
\do{03A6}{LGR}{\textPhi}{\Phi}%           % GR. C. L. PHI
\do{03A7}{LGR}{\textChi}{X}%              % GR. C. L. CHI
\do{03A8}{LGR}{\textPsi}{\Psi}%           % GR. C. L. PSI
\do{03A9}{LGR}{\textOmega}{\Omega}%       % GR. C. L. OMEGA
\do{03B1}{LGR}{\textalpha}{\alpha}%       % GR. S. L. ALPHA
\do{03B2}{LGR}{\textbeta}{\beta}%         % GR. S. L. BETA
\do{03B3}{LGR}{\textgamma}{\gamma}%       % GR. S. L. GAMMA
\do{03B4}{LGR}{\textdelta}{\delta}%       % GR. S. L. DELTA
\do{03B5}{LGR}{\textepsilon}{\epsilon}%   % GR. S. L. EPSILON
\do{03B6}{LGR}{\textzeta}{\zeta}%         % GR. S. L. ZETA
\do{03B7}{LGR}{\texteta}{\eta}%           % GR. S. L. ETA
\do{03B8}{LGR}{\texttheta}{\theta}%       % GR. S. L. THETA
\do{03B9}{LGR}{\textiota}{\iota}%         % GR. S. L. IOTA
\do{03BA}{LGR}{\textkappa}{\kappa}%       % GR. S. L. KAPPA
\do{03BB}{LGR}{\textlambda}{\lambda}%     % GR. S. L. LAMDA
\do{03BC}{LGR}{\textmu}{\mu}%             % GR. S. L. MU
\do{03BD}{LGR}{\textnu}{\nu}%             % GR. S. L. NU
\do{03BE}{LGR}{\textxi}{\xi}%             % GR. S. L. XI
\do{03BF}{LGR}{\textomicron}{o}%          % GR. S. L. OMICRON
\do{03C0}{LGR}{\textpi}{\pi}%             % GR. S. L. PI
\do{03C1}{LGR}{\textrho}{\rho}%           % GR. S. L. RHO
\do{03C2}{LGR}{\textvarsigma}{\varsigma}% % GR. S. L. FINAL SIGMA
\do{03C3}{LGR}{\textsigma}{\sigma}%       % GR. S. L. SIGMA
\do{03C4}{LGR}{\texttau}{\tau}%           % GR. S. L. TAU
\do{03C5}{LGR}{\textupsilon}{\upsilon}%   % GR. S. L. UPSILON
\do{03C6}{LGR}{\textphi}{\phi}%           % GR. S. L. PHI
\do{03C7}{LGR}{\textchi}{\chi}%           % GR. S. L. CHI
\do{03C8}{LGR}{\textpsi}{\psi}%           % GR. S. L. PSI
\do{03C9}{LGR}{\textomega}{\omega}%       % GR. S. L. OMEGA
\do{0401}{T2A}{\CYRYO}{}%                 % CY. C. L. IO
\do{0410}{T2A}{\CYRA}{}%                  % CY. C. L. A
\do{0411}{T2A}{\CYRB}{}%                  % CY. C. L. BE
\do{0412}{T2A}{\CYRV}{}%                  % CY. C. L. VE
\do{0413}{T2A}{\CYRG}{}%                  % CY. C. L. GHE
\do{0414}{T2A}{\CYRD}{}%                  % CY. C. L. DE
\do{0415}{T2A}{\CYRE}{}%                  % CY. C. L. IE
\do{0416}{T2A}{\CYRZH}{}%                 % CY. C. L. ZHE
\do{0417}{T2A}{\CYRZ}{}%                  % CY. C. L. ZE
\do{0418}{T2A}{\CYRI}{}%                  % CY. C. L. I
\do{0419}{T2A}{\CYRISHRT}{}%              % CY. C. L. SHORT I
\do{041A}{T2A}{\CYRK}{}%                  % CY. C. L. KA
\do{041B}{T2A}{\CYRL}{}%                  % CY. C. L. EL
```

```
\do{041C}{T2A}{\CYRM}{}%                % CY. C. L. EM
\do{041D}{T2A}{\CYRN}{}%                % CY. C. L. EN
\do{041E}{T2A}{\CYRO}{}%                % CY. C. L. O
\do{041F}{T2A}{\CYRP}{}%                % CY. C. L. PE
\do{0420}{T2A}{\CYRR}{}%                % CY. C. L. ER
\do{0421}{T2A}{\CYRS}{}%                % CY. C. L. ES
\do{0422}{T2A}{\CYRT}{}%                % CY. C. L. TE
\do{0423}{T2A}{\CYRU}{}%                % CY. C. L. U
\do{0424}{T2A}{\CYRF}{}%                % CY. C. L. EF
\do{0425}{T2A}{\CYRH}{}%                % CY. C. L. HA
\do{0426}{T2A}{\CYRC}{}%                % CY. C. L. TSE
\do{0427}{T2A}{\CYRCH}{}%               % CY. C. L. CHE
\do{0428}{T2A}{\CYRSH}{}%               % CY. C. L. SHA
\do{0429}{T2A}{\CYRSHCH}{}%             % CY. C. L. SHCHA
\do{042A}{T2A}{\CYRHRDSN}{}%            % CY. C. L. HARD SIGN
\do{042B}{T2A}{\CYRERY}{}%              % CY. C. L. YERU
\do{042C}{T2A}{\CYRSFTSN}{}%            % CY. C. L. SOFT SIGN
\do{042D}{T2A}{\CYREREV}{}%             % CY. C. L. E
\do{042E}{T2A}{\CYRYU}{}%               % CY. C. L. YU
\do{042F}{T2A}{\CYRYA}{}%               % CY. C. L. YA
\do{0430}{T2A}{\cyra}{}%                % CY. S. L. A
\do{0431}{T2A}{\cyrb}{}%                % CY. S. L. BE
\do{0432}{T2A}{\cyrv}{}%                % CY. S. L. VE
\do{0433}{T2A}{\cyrg}{}%                % CY. S. L. GHE
\do{0434}{T2A}{\cyrd}{}%                % CY. S. L. DE
\do{0435}{T2A}{\cyre}{}%                % CY. S. L. IE
\do{0436}{T2A}{\cyrzh}{}%               % CY. S. L. ZHE
\do{0437}{T2A}{\cyrz}{}%                % CY. S. L. ZE
\do{0438}{T2A}{\cyri}{}%                % CY. S. L. I
\do{0439}{T2A}{\cyrishrt}{}%            % CY. S. L. SHORT I
\do{043A}{T2A}{\cyrk}{}%                % CY. S. L. KA
\do{043B}{T2A}{\cyrl}{}%                % CY. S. L. EL
\do{043C}{T2A}{\cyrm}{}%                % CY. S. L. EM
\do{043D}{T2A}{\cyrn}{}%                % CY. S. L. EN
\do{043E}{T2A}{\cyro}{}%                % CY. S. L. O
\do{043F}{T2A}{\cyrp}{}%                % CY. S. L. PE
\do{0440}{T2A}{\cyrr}{}%                % CY. S. L. ER
\do{0441}{T2A}{\cyrs}{}%                % CY. S. L. ES
\do{0442}{T2A}{\cyrt}{}%                % CY. S. L. TE
\do{0443}{T2A}{\cyru}{}%                % CY. S. L. U
\do{0444}{T2A}{\cyrf}{}%                % CY. S. L. EF
\do{0445}{T2A}{\cyrh}{}%                % CY. S. L. HA
\do{0446}{T2A}{\cyrc}{}%                % CY. S. L. TSE
\do{0447}{T2A}{\cyrch}{}%               % CY. S. L. CHE
\do{0448}{T2A}{\cyrsh}{}%               % CY. S. L. SHA
\do{0449}{T2A}{\cyrshch}{}%             % CY. S. L. SHCHA
\do{044A}{T2A}{\cyrhrdsn}{}%            % CY. S. L. HARD SIGN
\do{044B}{T2A}{\cyrery}{}%              % CY. S. L. YERU
\do{044C}{T2A}{\cyrsftsn}{}%            % CY. S. L. SOFT SIGN
```

```
\do{044D}{T2A}{\cyrerev}{}%               % CY. S. L. E
\do{044E}{T2A}{\cyryu}{}%                 % CY. S. L. YU
\do{044F}{T2A}{\cyrya}{}%                 % CY. S. L. YA
\do{0451}{T2A}{\cyryo}{}%                 % CY. S. L. IO
\do{00A7}{TS1}{\textsection}{\mathsection}% SECTION SYMBOL
\do{00A8}{TS1}{\textasciidieresis}{}%      % DIAERESIS
\do{00B0}{TS1}{\textdegree}{\mathdegree}% % DEGREE SIGN
\do{00B1}{TS1}{\textpm}{\pm}%             % PLUS-MINUS SIGN
\do{00B4}{TS1}{\textasciiacute}{}%        % ACUTE ACCENT
\do{00B6}{TS1}{\textparagraph}{\mathparagraph}% PILCROW SIGN
\do{00D7}{TS1}{\texttimes}{\times}%       % MULTIPLICATION SIGN
\do{00F7}{TS1}{\textdiv}{\div}%           % DIVISION SIGN
}
```

`\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}
```

`\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{B}{\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。

```
\@ifpackageloaded{inputenc}{}{%else
  \RequirePackage[utf8]{inputenc}}
\def\bxjx@tmpa{utf8}
\ifx\bxjx@tmpa\inputencdoingname
  \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
  \expandafter\bxjx@tmpdo@a\csname bxjx@KC/\the\@tempcnta\endcsname{#1}}
\def\bxjx@tmpdo@a#1#2#3#4#5{%
```

引数＝\[bxjx@KC/⟨符号値⟩]{⟨符号値⟩}{⟨fontenc⟩}{⟨LICR⟩}{⟨数式 LICR⟩}
“数式中の動作”を決定する。⟨数式 LICR⟩ が空（数式非対応）なら警告を出す。

```
  \ifx\\#5\\%
    \def\bxjx@tmpa{\@inmathwarn#4}%
```

⟨数式 LICR⟩ が英字である場合は \bxjx@fake@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。

```
  \else\ifcat A\noexpand#5%
    \edef\bxjx@tmpa{\noexpand\bxjx@fake@grk{#5}%
      {\ifnum\uccode`#5=`#5\noexpand\Pi\else\noexpand\pi\fi}}%
```

それ以外は ⟨数式 LICR⟩ をそのまま実行する。

```
  \else \def\bxjx@tmpa{#5}%
  \fi\fi
  \def\bxjx@tmpb{\bxjx@tmpdo@b{#1}{#2}{#3}{#4}}%
  \expandafter\bxjx@tmpb\expandafter{\bxjx@tmpa}}
```

以降はエンジン種別で分岐する。upTeX の場合。

```
\if u\bxjx@engine
\def\bxjx@tmpdo@b#1#2#3#4#5{%
```

引数＝\[bxjx@KC/⟨符号値⟩]{⟨符号値⟩}{⟨fontenc⟩}{⟨LICR⟩}{⟨数式中の動作⟩}

当該の Unicode 文字の動作は「テキストでは ⟨LICR⟩、数式では ⟨数式中の動作⟩」となる。LICR は現在エンコーディングで有効な定義がある場合はそれが実行されるはずである。（つまり、現在が LGR である場合はギリシャ文字は常に欧文扱いになる。）それ以外の場合は LICR を \bxjx@ja@or@not に帰着させる。この際に、和文用の定義として当該の kchardef を使用し、その制御綴として \[bxjx@KC/...] を流用している。

```
\kchardef#1=\@tempcnta
\DeclareTextCommandDefault{#4}{\bxjx@ja@or@not{#1}{#3}{#4}}%
\DeclareUnicodeCharacter{#2}{\TextOrMath{#4}{#5}}}
```

pdfTEX の場合も処理はほとんど同じ。ただし、和文用の定義として \UTF{⟨符号値⟩} を使う（\UTF は bxcjkjatype の命令）。\[bxjx@KC/...] は使わないが定義済にする必要がある。

```
\else\if p\bxjx@engine
\def\bxjx@tmpdo@b#1#2#3#4#5{%
  \mathchardef#1=\@tempcnta
  \DeclareTextCommandDefault{#4}{\bxjx@ja@or@not{\UTF{#2}}{#3}{#4}}%
  \DeclareUnicodeCharacter{#2}{\TextOrMath{#4}{#5}}}
\fi\fi
```

以上の処理を「特定 CJK 曖昧文字」の各々に適用する。

```
\let\do\bxjx@tmpdo \bxjx@grkcyr@list
```

\bxjx@DeclareUnicodeCharacter　\bxjx@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。

```
\@onlypreamble\bxjx@org@DeclareUnicodeCharacter
\let\bxjx@org@DeclareUnicodeCharacter\DeclareUnicodeCharacter
\@onlypreamble\bxjx@DeclareUnicodeCharacter
\def\bxjx@DeclareUnicodeCharacter#1#2{%
  \count@="#1\relax
  \expandafter\ifx\csname bxjx@KC/\the\count@\endcsname\relax
    \bxjx@org@DeclareUnicodeCharacter{#1}{#2}%
  \else
    \wlog{ \space\space skipped defining Unicode char U+#1}%
  \fi}
```

\bxjx@ja@or@not　\bxjx@ja@or@not{⟨和文用定義⟩}{⟨対象 fontenc⟩}{⟨LICR⟩}：　\[no]greekasCJK の状態に応じて和文または欧文で文字を出力する。

```
\def\bxjx@ja@or@not#1#2#3{%
```

\greekasCJK の場合は、無条件に ⟨和文用定義⟩ を実行する。

```
  \ifbxjx@gcc@cjk #1%
```

\nogreekasCJK の場合は、対象のエンコーディングに変更して LICR を実行するが、そのエンコーディングが未定義の場合は（フォールバックとして）和文用定義を使う。

```
  \else\expandafter\ifx\csname T@#2\endcsname\relax #1%
  \else \UseTextSymbol{#2}{#3}%
  \fi\fi}
```

\DeclareFontEncoding@ \DeclareFontEncoding@ にパッチを当てて、\DeclareFontEncoding の実行中だけ改変後の \DeclareUnicodeCharacter が使われるようにする。

```
\begingroup
\toks@\expandafter{\DeclareFontEncoding@{#1}{#2}{#3}}
\xdef\next{\def\noexpand\DeclareFontEncoding@##1##2##3{%
  \noexpand\bxjx@swap@DUC@cmd
  \the\toks@
  \noexpand\bxjx@swap@DUC@cmd}}
\endgroup\next
\def\bxjx@swap@DUC@cmd{%
  \let\bxjx@tmpa\DeclareUnicodeCharacter
  \let\DeclareUnicodeCharacter\bxjx@DeclareUnicodeCharacter
  \let\bxjx@DeclareUnicodeCharacter\bxjx@tmpa
  \let\bxjx@tmpa\relax}
```

以上。

■XeTeX・LuaTeX の場合

```
\else\ifnum0\if x\bxjx@engine1\fi\if l\bxjx@engine1\fi>0
```

各文字について、数式中の動作を定義する。

```
\def\bxjx@tmpdo#1{%
  \bxjx@cnta="#1\relax
  \begingroup
    \lccode`~=\bxjx@cnta
  \lowercase{\endgroup
    \bxjx@tmpdo@a{~}}{#1}}
\def\bxjx@tmpdo@a#1#2#3#4#5{%
```

⟨数式 LICR⟩ が空なら何もしない。空でない場合、upLaTeX 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。

```
  \ifx\\#5\\\let\bxjx@tmpa\relax
  \else\ifcat A\noexpand#5%
    \edef\bxjx@tmpa{\noexpand\bxjx@fake@grk{#5}%
      {\ifnum\uccode`#5=`#5\noexpand\Pi\else\noexpand\pi\fi}}%
  \else \def\bxjx@tmpa{#5}%
  \fi\fi
  \ifx\bxjx@tmpa\relax\else
    \mathcode\bxjx@cnta"8000 \let#1\bxjx@tmpa
  \fi}
```

「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。

```
\mathchardef\bxjx@tmpa="119
\ifx\bxjx@tmpa\pi \let\do\bxjx@tmpdo \bxjx@grkcyr@list \fi
```

次に、テキストにおいて「特定 CJK 曖昧文字」の扱いが \[no]greekasCJK で切り替わるようにする。

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。

```
\if l\bxjx@engine
  \protected\def\greekasCJK{%
    \bxjx@gcc@cjktrue
    \ltjsetparameter{jacharrange={+2, +8}}}
  \protected\def\nogreekasCJK{%
    \bxjx@gcc@cjkfalse
    \ltjsetparameter{jacharrange={-2, -8}}}
\fi
```

XeTeX の場合、xeCJK は XeTeX の文字クラス定義を参照しているので、対象文字の文字クラスを変更する。

```
\if x\bxjx@engine
  \let\bxjx@gcc@cjk@list\@empty
  \def\do#1#2#3#4{%
    \edef\bxjx@gcc@cjk@list{\bxjx@gcc@cjk@list
      \noexpand\XeTeXcharclass"#1\bxjx@cnta}}
  \bxjx@grkcyr@list
  \protected\def\greekasCJK{%
    \bxjx@gcc@cjktrue
    \bxjx@cnta=\@ne \bxjx@gcc@cjk@list}
  \protected\def\nogreekasCJK{%
    \bxjx@gcc@cjkfalse
    \bxjx@cnta=\z@ \bxjx@gcc@cjk@list}
\fi
```

以上。

```
\fi\fi
```

H.4　初期設定

「特定 CJK 曖昧文字」を欧文扱いにする。

```
\nogreekasCJK
```

H.5　完了

おしまい。

```
%</cjkcat>
```

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こむ。
※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込に限られる。

I.1 準備

```
%<*ancpandoc>
%% このファイルは日本語文字を含みます.
\def\bxjsp@pkgname{bxjspandoc}
```

\bxjsp@engine エンジンの種別。

```
\let\bxjsp@engine=n
\@onlypreamble\bxjsp@do
\def\bxjsp@do#1#2{%
  \edef\bxjsp@tmpa{\string#1}%
  \edef\bxjsp@tmpb{\meaning#1}%
  \ifx\bxjsp@tmpa\bxjsp@tmpb #2\fi}
\bxjsp@do\kanjiskip{\let\bxjsp@engine=j}
\bxjsp@do\XeTeXversion{\let\bxjsp@engine=x}
\bxjsp@do\pdftexversion{\let\bxjsp@engine=p}
\bxjsp@do\luatexversion{\let\bxjsp@engine=l}
```

\bxjsp@begin@document@hook 文書本体開始時フック。

```
\@onlypreamble\bxjsp@begin@document@hook
\let\bxjsp@begin@document@hook\@empty
\AtBeginDocument{\bxjsp@begin@document@hook}
```

\ifbxjsp@babel@used〔スイッチ〕Babel が読み込まれたか。

```
\newif\ifbxjsp@babel@used
\g@addto@macro\bxjsp@begin@document@hook{%
  \@ifpackageloaded{babel}{\bxjsp@babel@usedtrue}{}}
```

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。

```
\newif\ifbxjsp@english
\DeclareOption{english}{\bxjsp@englishtrue}
```

オプション定義はおしまい。

```
\ProcessOptions*
```

I.3 パッケージ読込の阻止

\pandocSkipLoadFile \pandocSkipLoadFile{⟨ファイル名⟩}： 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。

```
\@onlypreamble\pandocSkipLoadFile
\newcommand*\pandocSkipLoadFile[1]{%
```

```
  \expandafter\bxjsp@skip@load@file@a\csname ver@#1\endcsname{#1}}
\def\bxjsp@skip@load@file@a#1#2{%
  \ifx#1\relax
    \def#1{2001/01/01}%
    \PackageInfo\bxjsp@pkgname
     {File '#2' marked as loaded\@gobble}%
  \fi}
```

\pandocSkipLoadPackage \pandocSkipLoadPackage{⟨パッケージ名⟩}： \pandocSkipLoadFile の機能を用いてパッケージの読込を阻止する。

```
\@onlypreamble\pandocSkipLoadPackage
\newcommand*\pandocSkipLoadPackage[1]{%
  \pandocSkipLoadFile{#1.sty}}
```

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。

```
\ifx\@IncludeInRelease\@undefined\else
  \pandocSkipLoadPackage{fixltx2e}
\fi
```

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。）

```
\if j\bxjsp@engine
  \pandocSkipLoadPackage{cmap}
\fi
```

I.6 microtype パッケージ

警告が多すぎなので消す。

```
\if j\bxjsp@engine \else
  \PassOptionsToPackage{verbose=silent}{microtype}
\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。

```
\if j\bxjsp@engine
  \pandocSkipLoadPackage{microtype}
  \newcommand*\UseMicrotypeSet[2][]{}
```

```
\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。

```
\DeclareRobustCommand{\pandocLdots}{%
  \let\bxjsp@do\bxjsp@ja@ellipsis
  \ifmmode \let\bxjsp@do\bxjsp@org@ldots
  \else\ifbxjsp@babel@used
    \expandafter\ifx\csname bxjsp@ld/\languagename\endcsname\relax
      \let\bxjsp@do\bxjsp@org@ldots \fi
  \fi\fi \bxjsp@do}
\@namedef{bxjsp@ld/japanese}{1}
\def\bxjsp@ja@ellipsis{…}
\let\bxjsp@org@ldots\ldots
```

`\ldots` の実装を `\pandocLdots` に置き換える。

```
\g@addto@macro\bxjsp@begin@document@hook{%
  \let\bxjsp@org@ldots\ldots
```

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。

```
  \bxjs@nclong\def\bxjsp@tmpa{\ldots}%
  \ifx\pandocLdots\bxjsp@tmpa\else
```

`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。

```
    \ifnum0\ifbxjsp@english\ifbxjsp@babel@used\else1\fi\fi=0
      \let\ldots\pandocLdots
    \fi
  \fi}
```

`\ldots` の直後の文字が非英字の場合、Pandoc は「`\ldots`。」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 XeTeX・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。

```
\chardef\bxjsp@cc@other=12
\@onlypreamble\bxjsp@makeother@range
\def\bxjsp@makeother@range#1#2{%
  \@tempcnta"#1\relax \@tempcntb"#2\relax
  \loop\ifnum\@tempcnta<\@tempcntb
    \catcode\@tempcnta\bxjsp@cc@other
    \advance\@tempcnta\@ne
  \repeat}
\ifnum0\if x\bxjsp@engine1\fi\if l\bxjsp@engine1\fi>0
  \catcode"1F23B=\bxjsp@cc@other
  \bxjsp@makeother@range{9FCD}{A000}
  \bxjsp@makeother@range{1B002}{1B170}
  \bxjsp@makeother@range{2B820}{2EBF0}
\fi
```

I.8 PandoLa モジュール

インストール済であれば読み込む。

```
\IfFileExists{bxpandola.sty}{%
  \RequirePackage{bxpandola}\relax
  \PackageInfo\bxjsp@pkgname
   {PandoLa module is loaded\@gobble}
}{}
```

I.9 完了

おしまい。

```
%</ancpandoc>
```

補助パッケージ実装はここまで。

```
%</anc>
```