

# 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”)

v2.9d [2025/06/06]







---

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、ユーザーマニュアル `bxjscls-manual.pdf` を参照してください。

---

## 目次

<b>1</b>	<b>はじめに</b>	<b>4</b>
<b>2</b>	<b>オプション</b>	<b>13</b>
<b>3</b>	<b>和文フォントの変更</b>	<b>44</b>
<b>4</b>	<b>フォントサイズ</b>	<b>54</b>
<b>5</b>	<b>レイアウト</b>	<b>60</b>
5.1	ページレイアウト . . . . .	61
<b>6</b>	<b>改ページ (日本語 TeX 開発コミュニティ版のみ)</b>	<b>76</b>
<b>7</b>	<b>ページスタイル</b>	<b>77</b>
<b>8</b>	<b>文書のマークアップ</b>	<b>80</b>
8.1	表題 . . . . .	81
8.2	章・節 . . . . .	86
8.3	リスト環境 . . . . .	98
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 . . . . .	108
<b>9</b>	<b>フォントコマンド</b>	<b>110</b>


<b>10</b>	<b>相互参照</b>	<b>112</b>
10.1	目次の類 . . . . .	112
10.2	参考文献 . . . . .	117
10.3	索引 . . . . .	119
10.4	脚注 . . . . .	120
<b>11</b>	<b>段落の頭へのグルー挿入禁止</b>	<b>123</b>
<b>12</b>	<b>いろいろなロゴ</b>	<b>127</b>
<b>13</b>	<b>amsmath との衝突の回避</b>	<b>128</b>
<b>14</b>	<b>初期設定</b>	<b>128</b>
<b>15</b>	<b>実験的コード</b>	<b>133</b>
<b>16</b>	<b>BXJS 独自の追加処理 </b>	<b>133</b>
<b>付録 A</b>	<b>和文ドライバの仕様 </b>	<b>139</b>
<b>付録 B</b>	<b>和文ドライバ：minimal </b>	<b>140</b>
B.1	準備 . . . . .	140
B.2	(u)pTeX 用の設定 . . . . .	143
B.3	pdfTeX 用の処理 . . . . .	148
B.4	X <sub>Y</sub> TeX 用の処理 . . . . .	148
B.5	後処理（エンジン共通） . . . . .	149
<b>付録 C</b>	<b>和文ドライバ：standard </b>	<b>152</b>
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 + bxcjkatype . . . . .	164
C.6	X <sub>Y</sub> TeX 用設定：xeCJK + zxjatype . . . . .	166
C.7	LuaTeX 用設定：LuaTeX-ja . . . . .	169
C.8	共通処理 (2) . . . . .	173
<b>付録 D</b>	<b>和文ドライバ：modern </b>	<b>173</b>
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
<b>付録 E</b>	<b>和文ドライバ：pandoc </b>	<b>174</b>

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
<b>付録 F 補助パッケージ一覧</b>		<b>187</b>
<b>付録 G 補助パッケージ：bxjscompat</b>		<b>187</b>
G.1	準備	188
G.2	8bit 欧文 TeX	188
G.3	X <sub>g</sub> TeX	188
G.4	LuaTeX	190
G.5	完了	191
<b>付録 H 補助パッケージ：bxjscjcat</b>		<b>191</b>
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
<b>付録 I 補助パッケージ：bxjspandoc</b>		<b>200</b>
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  $\mathrm{T}_{\mathrm{E}}\mathrm{X}$  開発コミュニティによる「 $\mathrm{p}_{\mathrm{L}}\mathrm{T}_{\mathrm{E}}\mathrm{X}_{2_{\epsilon}}$  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。

<code>&lt;article&gt;</code>	<code>bxjsarticle.cls</code>	短いレポート（章なし）のクラス
<code>&lt;report&gt;</code>	<code>bxjsreport.cls</code>	長いレポート（章あり）のクラス
<code>&lt;book&gt;</code>	<code>bxjsbook.cls</code>	書籍用のクラス
<code>&lt;slide&gt;</code>	<code>bxjsslide.cls</code>	スライド用のクラス
<code>&lt;minimal&gt;</code>	<code>bxjsja-minimal.def</code>	minimal 和文ドライバ
<code>&lt;standard&gt;</code>	<code>bxjsja-standard.def</code>	standard 和文ドライバ
<code>&lt;modern&gt;</code>	<code>bxjsja-modern.def</code>	modern 和文ドライバ（未公開）
<code>&lt;pandoc&gt;</code>	<code>bxjsja-pandoc.def</code>	pandoc 和文ドライバ
<code>&lt;compat&gt;</code>	<code>bxjscompat.sty</code>	古いやつをどうにかする補助パッケージ
<code>&lt;cjkcat&gt;</code>	<code>bxjscjkcat.sty</code>	modern ドライバ用の補助パッケージ
<code>&lt;ancpandoc&gt;</code>	<code>bxjspandoc.sty</code>	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  $\mathrm{I}_{\mathrm{A}}\mathrm{T}_{\mathrm{E}}\mathrm{X}3$  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  $\mathrm{T}_{\mathrm{E}}\mathrm{X}$  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  $\mathrm{T}_{\mathrm{E}}\mathrm{X}$  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  $\mathrm{up}_{\mathrm{I}}\mathrm{A}\mathrm{T}_{\mathrm{E}}\mathrm{X}$  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。

```
1 %<*minijs>
2 %% if jsclasses loaded, abort loading this package
3 \ifx\@jsc@uplatextrue\@undefined\else
4   \PackageInfo{minijs}{jsclasses does not need minijs, exiting}
5   \expandafter\endinput
6 \fi
7 %% "fake" jsarticle
8 \expandafter\def\csname ver@jsarticle.cls\endcsname{}
9 %</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。

```
10 %<*class>
11 %% このファイルは日本語文字を含みます.
12 %<article>\def\bxjs@clsname{bxjsarticle}
13 %<book>\def\bxjs@clsname{bxjsbook}
14 %<report>\def\bxjs@clsname{bxjsreport}
15 %<slide>\def\bxjs@clsname{bxjsslide}
```

\ifjsc@needsptch [2016-08-22] 従来 jsclasses では、p $\text{\LaTeX}$  や  $\text{\LaTeX}$  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 p $\text{\LaTeX}$  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 p $\text{\LaTeX}$  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。

```
16 %</class>
17 %<*class|minijs>
18 \newif\ifjsc@needsptch
19 \jsc@needsptchfalse
20 %</class|minijs>
21 %<*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。

```

22 \let\jsArticle=a
23 \let\jsBook=b
24 \let\jsReport=r
25 \let\jsSlide=s
26 %<article>\let\jsDocClass\jsArticle
27 %<book>\let\jsDocClass\jsBook
28 %<report>\let\jsDocClass\jsReport
29 %<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。

```

30 \def\bxjs@test@engine#1#2{%
31   \edef\bxjs@tmpa{\string#1}%
32   \edef\bxjs@tmpb{\meaning#1}%
33   \ifx\bxjs@tmpa\bxjs@tmpb #2\fi}

```

`\jsEngine` [暗黙文字トークン] エンジン (T<sub>E</sub>X 処理系) の種別： `j` = pT<sub>E</sub>X 系、`x` = X<sub>Y</sub>T<sub>E</sub>X、`p` = pdfT<sub>E</sub>X、`l` = LuaT<sub>E</sub>X、`J` = NTT jT<sub>E</sub>X、`0` = Omega 系、`n` = 以上の何れでもない。

※ pdfT<sub>E</sub>X と LuaT<sub>E</sub>X については DVI モードの場合も含む。

```

34 \let\jsEngine=n
35 \bxjs@test@engine\kanjiskip{\let\jsEngine=j}
36 \bxjs@test@engine\jintercharskip{\let\jsEngine=J}
37 \bxjs@test@engine\Omegaversion{\let\jsEngine=0}
38 \bxjs@test@engine\XeTeXversion{\let\jsEngine=x}
39 \bxjs@test@engine\pdftexversion{\let\jsEngine=p}
40 \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。

- X<sub>Y</sub>T<sub>E</sub>X： 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。

- T<sub>E</sub>X： 3.0 版 [1990/03] 以上
- pT<sub>E</sub>X： 2.0 版 [1995/03] 以上
- upT<sub>E</sub>X： 0.10 版 [2007/07] 以上
- pdfT<sub>E</sub>X： 1.40 版 [2007/01] 以上
- LuaT<sub>E</sub>X： 0.60 版 [2010/04] 以上
- X<sub>Y</sub>T<sub>E</sub>X： 0.9994 版 [2009/06] 以上

※ Omega と NTT jT<sub>E</sub>X 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。

T<sub>E</sub>X 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。

```

41 \@tempwattrue

```

```

42 \if x\jsEngine
43 \ifdim\the\XeTeXversion\XeTeXrevision\p@<0.997\p@
44 \@tempswafalse \fi
45 \fi

```

非サポートのバージョン場合は強制終了させる。

```

46 \if@tempswa \expandafter\@gobble
47 \else
48 \ClassError\bxjs@clsname
49 {The engine in use is all too old}
50 {It's a fatal error. I'll quit right now.}
51 \expandafter\@firstofone
52 \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。

```

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

```

`\bxjs@if@format@at@least \bxjs@if@format@at@least{<日付>}{<真>}{<偽>}`: L<sup>A</sup>T<sub>E</sub>X カーネルの版が指定の日付以降であるか。

```

63 \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。

```

64 \bxjs@if@format@at@least{2017/04/15}{%
65 \let\bxjs@if@package@at@least\@ifpackagelater
66 }{%else
67 \def\bxjs@if@package@at@least#1#2{%
68 \@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 up<sub>T</sub>E<sub>X</sub>」であるか。

※つまり、`\jsEngine = j` である場合、このスイッチが真なら up<sub>L</sub>A<sub>T</sub>E<sub>X</sub>、偽なら p<sub>L</sub>A<sub>T</sub>E<sub>X</sub> である。2023 年 6 月に p<sub>L</sub>A<sub>T</sub>E<sub>X</sub> の T<sub>E</sub>X 処理系が「ε-p<sub>T</sub>E<sub>X</sub>」から「内部漢字コードが非 Unicode の ε-up<sub>T</sub>E<sub>X</sub>」に変わったが、これによる影響はない。

```

69 \newif\ifjsWithupTeX
70 \ifx\ucs\undefined\else \ifnum\ucs"3000="3000
71 \jsWithupTeXtrue

```

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

`\ifjsWithpTeXng` [スイッチ] エンジンが `pTeX-ng` であるか。

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

`\ifjsWitheTeX` [スイッチ] エンジンが `ε-TeX` 拡張をもつか。

※ `XYTeX` 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。

```
76 \newif\ifjsWitheTeX
77 \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。

```
78 \newif\ifjsInPdfMode
79 \@nameuse{jsInPdfMode\ifnum0%
80 \ifx\pdfoutput\undefined\else\the\pdfoutput\fi
81 \ifx\outputmode\undefined\else\the\outputmode\fi
82 >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。

```
83 \newif\ifbxjs@explIII
84 \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。

```
85 \newif\ifbxjs@brace@safe
86 \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 `XYTeX` 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。

```
87 \newif\ifbxjs@TUenc
88 \def\bxjs@tmpa{TU}\edef\bxjs@tmpb{\f@encoding}
89 \ifx\bxjs@tmpa\bxjs@tmpb
90 \bxjs@TUenctrue
91 \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。

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

### ■依存パッケージ読込

長さ値の指定で式を利用可能にするため calc を読み込む。

```
94 \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。

```
95 \RequirePackage{keyval}
```

PDF モードの判定を L<sup>A</sup>T<sub>E</sub>X 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。

```
96 \IfFileExists{iftex.sty}{%
97   \RequirePackage{iftex}
98 }{}
99 \begingroup\expandafter\endgroup
100 \expandafter\ifx\csname ifpdf\endcsname\@undefined\else
101   \expandafter\let\csname ifjsInPdfMode\expandafter\endcsname
102     \csname ifpdf\endcsname
103 \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。)

```
104 \def\jsAtEndOfClass{%
105   \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。

```
106 \IfFileExists{bxjscompat.sty}{%
107   \RequirePackage{bxjscompat}%
108 }{}
```

### ■BXJS クラス特有の設定

LuaT<sub>E</sub>X の場合、本クラス用の Lua モジュールを用意する。

```
109 \ifx l\jsEngine
110   \directlua{ bxjs = {} }
111 \fi
```

\bxjs@protected  $\varepsilon$ -T<sub>E</sub>X 拡張が有効な場合にのみ \protected の効果をもつ。

```

112 \ifjsWithTeX \let\bxjs@protected\protected
113 \else \let\bxjs@protected\empty
114 \fi

```

`\bxjs@robust@def` 無引数の頑強な命令を定義する。

```

115 \ifjsWithTeX
116 \def\bxjs@robust@def{\protected\def}
117 \else
118 \def\bxjs@robust@def{\DeclareRobustCommand*}
119 \fi

```

`\bxjs@CGHN` L<sup>A</sup>T<sub>E</sub>X 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 L<sup>A</sup>T<sub>E</sub>X 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 L<sup>A</sup>T<sub>E</sub>X では“package/after/PKG”に展開される。

```

120 \bxjs@if@format@at@least{2021/11/15}{%
121 \def\bxjs@CGHN#1{#1}%
122 }{%else
123 \def\bxjs@CGHN#1{\bxjs@CGHN@a#1//}%
124 \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。

```

125 \@gobbletwo\if\if \def\bxjs@cond#1\fi{%
126 #1\expandafter\@firstoftwo
127 \else\expandafter\@secondoftwo
128 \fi}

```

**TODO:2.9** `\bxjs@expanded` を定義する。

`\bxjs@cslet` `\bxjs@cslet{〈名前 1〉}\制御綴` :

```

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

```

`\bxjs@csletcs` `\bxjs@csletcs{〈名前 1〉}{〈名前 2〉}` :

```

131 \def\bxjs@csletcs#1#2{%
132 \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。

```

133 \def\bxjs@catopt#1#2{%
134 #1\if\relax#1\relax\else\if\relax#2\relax\else,\fi\fi#2}

```

`\bxjs@ifplus` `\@ifstar` の + 版。

```

135 \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置き換える。

```

136 \def\bxjs@trim#1{\expandafter\bxjs@trim@a#1\@nil#1}
137 \def\bxjs@trim@a{\futurelet\bxjs@tmpb\bxjs@trim@b}
138 \def\bxjs@trim@b{\bxjs@cond@ifx\bxjs@tmpb\@sptoken\fi
139   {\bxjs@trim@c\bxjs@trim@a}{\bxjs@trim@d}}
140 \def\bxjs@trim@c#1 {#1}
141 \def\bxjs@trim@d#1\@nil{\bxjs@trim@e#1\@nil: \@nil\@nnil}
142 \def\bxjs@trim@e#1 \@nil#2\@nnil{\bxjs@cond@ifx\@nnil#2\@nnil\fi
143   {\bxjs@trim@f#1\@nnil}{\bxjs@trim@e#1\@nil: \@nil\@nnil}}
144 \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。

```

145 \def\bxjs@set@array@from@clist#1#2{%
146   \@tempcnta\z@
147   \@for\bxjs@tmpa:=\@empty#2\do{%
148     \bxjs@trim\bxjs@tmpa \bxjs@cslet{#1/\the\@tempcnta}\bxjs@tmpa
149     \advance\@tempcnta\@ne}
150   \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。

```

151 \let\c@bxjs@tempcnta\@tempcnta
152 \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。

```

153 \def\bxjs@advance@qc#1#2{%
154   \begingroup
155     \@tempcnta=#1\relax \advance\@tempcnta by#2\relax
156     \global\chardef\bxjs@g@tmpa\@tempcnta
157   \endgroup \let#1\bxjs@g@tmpa}

```

`\bxjs@new@count`  $\varepsilon$ - $\mathrm{T}_{\mathrm{E}}\mathrm{X}$  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 158 \ifjsWithTeX
159   \let\bxjs@new@count\newcount
160   \def\bxjs@advance@count#1#2{\advance#1by#2\relax}
161 \else
162   \def\bxjs@new@count#1{\chardef#1\z@}
163   \let\bxjs@advance@count\bxjs@advance@qc
164 \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）。

```

165 \def\jsSetQHLength#1#2{%
166   \begingroup
167     \bxjs@parse@qh{#2}%
168     \ifx\bxjs@tmpb\relax
169       \setlength\@tempdima{#2}%

```

```

170     \xdef\bxjs@g@tmpa{\the\@tempdima}%
171     \else \global\let\bxjs@g@tmpa\bxjs@tmpb
172     \fi
173     \endgroup
174     #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)pL<sup>A</sup>T<sub>E</sub>X 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。

```

175 \if j\jsEngine \def\bxjs@parse@qh@units{zw,zh}
176 \else \def\bxjs@parse@qh@units{trueQ,trueH,Q,H,zw,zh}
177 \fi
178 \def\bxjs@parse@qh#1{%
179     \let\bxjs@tmpb\relax
180     \@for\bxjs@tmpa:=\bxjs@parse@qh@units\do{%
181         \ifx\bxjs@tmpb\relax
182             \edef\bxjs@next{\{ \bxjs@tmpa \} \{ #1 \} }%
183             \expandafter\bxjs@parse@qh@a\csname bxjs@unit@\bxjs@tmpa\expandafter
184                 \endcsname\bxjs@next
185         \fi}}
186 \def\bxjs@parse@qh@a#1#2#3{%
187     \def\bxjs@next##1#2\@nil##2\@nnil{\bxjs@parse@qh@b{##1}{##2}#1}%
188     \bxjs@next#3\@nil#2\@nil\@nnil}
189 \def\bxjs@parse@qh@b#1#2#3{%
190     \ifx\@nnil#2\@nnil\else
191         \ifx#3\relax
192             \ClassError\bxjs@clsname
193                 {You cannot use '\bxjs@tmpa' here}{\@ehc}%
194             \def\bxjs@tmpb{0pt}%
195         \else
196             \@tempdimb#3\relax \@tempdimb#1\@tempdimb
197             \edef\bxjs@tmpb{\the\@tempdimb}%
198         \fi
199     \fi}

```

今の段階では Q/H だけが使用可能。

```

200     \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。

```

201 \newif\ifbxjs@after@preamble

```

\bxjs@begin@document@hook BXJS クラス用の文書本体開始時フック。

```

202 \@onlypreamble\bxjs@begin@document@hook
203 \def\bxjs@begin@document@hook{\bxjs@after@preambletrue}
204 \AtBeginDocument{\bxjs@begin@document@hook}

```

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

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

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

```
207 \@onlypreamble\bxjs@pre@jadriver@hook
208 \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。

```
209 \@onlypreamble\bxjs@endpreamble@hook
210 \let\bxjs@endpreamble@hook\@empty
211 \AtEndOfClass{%
212   \ifx\AtEndPreamble\@undefined\else
213     \AtEndPreamble{\bxjs@endpreamble@hook}%
214   \fi}
```

一時的な手続き用の制御綴。

```
215 \@onlypreamble\bxjs@tmpdo
216 \@onlypreamble\bxjs@tmpdo@a
217 \@onlypreamble\bxjs@tmpdo@b
218 \@onlypreamble\bxjs@tmpdo@c
219 \@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。

```
220 \bxjs@robust@def\jsInhibitGlue{%
221   \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。

```
222 \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。

```
223 \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。

```
224 %<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。

```
225 %<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。

---

```
226 %<book|report>\newif\if@mainmatter \@mainmattertrue
```

`\if@enablejfam` 和文フォントを数式フォントとして登録するかどうかを示すスイッチです。

---

JS クラスと異なり、初期値は偽とする。

---

```
227 \newif\if@enablejfam \@enablejfamfalse
```

以下で各オプションを宣言します。

■用紙サイズ JIS や ISO の A0 判は面積  $1\text{m}^2$ 、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  $1.5\text{m}^2$  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  $1000\text{mm} \times 1414\text{mm}$  です。このため、 $\text{\LaTeX 2}_\epsilon$  の `b5paper` は  $250\text{mm} \times 176\text{mm}$  ですが、 $\text{p}\text{\LaTeX 2}_\epsilon$  の `b5paper` は  $257\text{mm} \times 182\text{mm}$  になっています。ここでは  $\text{p}\text{\LaTeX 2}_\epsilon$  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,  $182\text{mm} \times 230\text{mm}$ ), `a4var` (A4 変形,  $210\text{mm} \times 283\text{mm}$ ) を追加しました。

---

BXJS クラスではページレイアウト設定に `geometry` パッケージを用いる。用紙サイズ設定は `geometry` に渡すオプションの指定と扱われる。

```
228 \@onlypreamble\bxjs@setpaper
229 \def\bxjs@setpaper#1{\def\bxjs@param@paper{#1}}
230 \newif\ifbxjs@iso@bsize
231 \DeclareOption{iso-bsize}{\bxjs@iso@bsizetrue}
232 \@onlypreamble\bxjs@setpaper@bsize
233 \def\bxjs@setpaper@bsize#1{\edef\bxjs@param@paper{%
234   b#1\ifbxjs@iso@bsize paper\else j\fi}}
235 \DeclareOption{a3paper}{\bxjs@setpaper{a3paper}}
236 \DeclareOption{a4paper}{\bxjs@setpaper{a4paper}}
237 \DeclareOption{a5paper}{\bxjs@setpaper{a5paper}}
238 \DeclareOption{a6paper}{\bxjs@setpaper{a6paper}}
239 \DeclareOption{b4paper}{\bxjs@setpaper@bsize{4}}
240 \DeclareOption{b5paper}{\bxjs@setpaper@bsize{5}}
241 \DeclareOption{b6paper}{\bxjs@setpaper@bsize{6}}
242 \DeclareOption{a4j}{\bxjs@setpaper{a4paper}}
243 \DeclareOption{a5j}{\bxjs@setpaper{a5paper}}
244 \DeclareOption{b4j}{\bxjs@setpaper{b4j}}
245 \DeclareOption{b5j}{\bxjs@setpaper{b5j}}
```

```

246 \DeclareOption{a4var}{\bxjs@setpaper{{210truemm}{283truemm}}}
247 \DeclareOption{b5var}{\bxjs@setpaper{{182truemm}{230truemm}}}
248 \DeclareOption{letterpaper}{\bxjs@setpaper{letterpaper}}
249 \DeclareOption{legalpaper}{\bxjs@setpaper{legalpaper}}
250 \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。

```

251 \@for\bxjs@tmpa:={%
252   a0,a1,a2,c0,c1,c2,c3,c4,c5,c6,ansia,ansib,ansic,ansid,ansie%
253 } \do{\edef\bxjs@next{%
254   \noexpand\DeclareOption{\bxjs@tmpa paper}%
255   {\noexpand\bxjs@setpaper{\bxjs@tmpa paper}}%
256 } \bxjs@next}
257 \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。

```

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

```

Pandoc で用紙サイズを指定した場合は出力 L<sup>A</sup>T<sub>E</sub>X 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。

```

262 \DeclareOption{a4varpaper}{\bxjs@setpaper{{210truemm}{283truemm}}}
263 \DeclareOption{b5varpaper}{\bxjs@setpaper{{182truemm}{230truemm}}}
264 \DeclareOption{screenpaper}{\bxjs@setpaper{screen}}
265 \DeclareOption{custompaper}{\}

```

■横置き 用紙の縦と横の長さを入れ換えます。

```

266 \newif\if@landscape
267 \@landscapefalse
268 \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もので、簡単のため article のみで使えるオプションとしました。

```

269 \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。

```

270 %<!slide>\@slidefalse
271 %<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` の定義が変わったのでご迷惑をおかけしましたが, 標準的なドキュメントクラスと同様にポイント数から 10 を引いたものに直しました。

[2003-03-22] 14Q オプションを追加しました。

[2003-04-18] 12Q オプションを追加しました。

[2016-07-08] `\mag` を使わずに各種寸法をスケールさせるためのオプション `nomag` を新設しました。`usemag` オプションの指定で従来通りの動作となります。デフォルトは `usemag` です。

[2016-07-24] オプティカルサイズを調整するために NFSS ヘパッチを当てるオプション `nomag*` を新設しました。

---

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

```
272 \newcommand{\@ptsize}{0}
273 \def\bxjs@param@basefontsize{10pt}
274 \def\jsUnusualPtSize{-20}
```

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

```
275 \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、 $\text{\LaTeX}$  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。

```
276 \jsSetQHLength\@tempdima{#1}%
277 \edef\bxjs@param@basefontsize{\the\@tempdima}%
278 \ifdim\@tempdima=10pt \long\def\@ptsize{0}%
279 \else\ifdim\@tempdima=10.95pt \long\def\@ptsize{1}%
280 \else\ifdim\@tempdima=12pt \long\def\@ptsize{2}%
281 \else \long\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。

```
282 \def\bxjs@setjbasefontsize#1{%
283 \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。

```
284 \newif\ifjsc@mag
285 \newif\ifjsc@mag@xreal
286 %\let\jsc@magscale\undefined
287 \DeclareOption{8pt}{\bxjs@setbasefontsize{8pt}}
288 \DeclareOption{9pt}{\bxjs@setbasefontsize{9pt}}
```



```

289 \DeclareOption{10pt}{\bxjs@setbasefontsize{10pt}}
290 \DeclareOption{11pt}{\bxjs@setbasefontsize{10.95pt}}
291 \DeclareOption{12pt}{\bxjs@setbasefontsize{12pt}}
292 \DeclareOption{14pt}{\bxjs@setbasefontsize{14.4pt}}
293 \DeclareOption{17pt}{\bxjs@setbasefontsize{17.28pt}}
294 \DeclareOption{20pt}{\bxjs@setbasefontsize{20pt}}
295 \DeclareOption{21pt}{\bxjs@setbasefontsize{20.74pt}}
296 \DeclareOption{25pt}{\bxjs@setbasefontsize{24.88pt}}
297 \DeclareOption{30pt}{\bxjs@setbasefontsize{29.86pt}}
298 \DeclareOption{36pt}{\bxjs@setbasefontsize{35.83pt}}
299 \DeclareOption{43pt}{\bxjs@setbasefontsize{43pt}}
300 \DeclareOption{12Q}{\bxjs@setjbasefontsize{3mm}}
301 \DeclareOption{14Q}{\bxjs@setjbasefontsize{3.5mm}}
302 \DeclareOption{10ptj}{\bxjs@setjbasefontsize{10pt}}
303 \DeclareOption{10.5ptj}{\bxjs@setjbasefontsize{10.5pt}}
304 \DeclareOption{11ptj}{\bxjs@setjbasefontsize{11pt}}
305 \DeclareOption{12ptj}{\bxjs@setjbasefontsize{12pt}}

```

JS クラス互換の magstyle 設定オプション。

```

306 \DeclareOption{usemag}{\let\bxjs@magstyle\bxjs@magstyle@@usemag}
307 \DeclareOption{nomag}{\let\bxjs@magstyle\bxjs@magstyle@@nomag}
308 \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 p<sub>La</sub>T<sub>E</sub>X 2<sub>ε</sub> 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 p<sub>La</sub>T<sub>E</sub>X 2<sub>ε</sub> 本体で宣言されています。

---

取りあえず、p<sub>T</sub>E<sub>X</sub> 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。

---

```

309 \if j\jsEngine
310 \hour\time \divide\hour by 60\relax
311 \@tempcnta\hour \multiply\@tempcnta 60\relax
312 \minute\time \advance\minute-\@tempcnta
313 \DeclareOption{tombow}{%
314   \tombowtrue \tombowdatetrue
315   \setlength{\@tombowwidth}{.1\p@}%
316   \@bannertoken{%
317     \jobname\space(\number\year-\two@digits\month-\two@digits\day
318     \space\two@digits\hour:\two@digits\minute)}%
319   \maketombowbox}
320 \DeclareOption{tombo}{%
321   \tombowtrue \tombowdatefalse
322   \setlength{\@tombowwidth}{.1\p@}%
323   \maketombowbox}

```

324 \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。

```
325 \if j\jsEngine
326 \DeclareOption{mentuke}{%
327   \tombowtrue \tombowdatefalse
328   \setlength{\@tombowwidth}{\z@}%
329   \maketombowbox}
330 \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。

```
331 \DeclareOption{oneside}{\@twosidefalse \@mparswitchfalse}
332 \DeclareOption{twoside}{\@twosidetrue \@mparswitchtrue}
333 \DeclareOption{vartwoside}{\@twosidetrue \@mparswitchfalse}
```

■二段組 `twocolumn` で二段組になります。

```
334 \DeclareOption{onecolumn}{\@twocolumnfalse}
335 \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。

```
336 \DeclareOption{titlepage}{\@titlepagetrue}
337 \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  $\text{\LaTeX}$  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。

```
338 %<book|report>\DeclareOption{openright}{\@openrighttrue\@openleftfalse}
339 %<book|report>\DeclareOption{openleft}{\@openlefttrue\@openrightfalse}
340 %<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.*)  $\text{\LaTeX}$  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]  $\text{\LaTeX}$  2<sub>ε</sub> 2021-11-15 (ltmath.dtx 2021/10/14 v1.2j) で `\@currentcounter` が追加されましたので、追従します。

[2025-05-26]  $\text{\LaTeX}$  2<sub>ε</sub> 2025-06-01 に追従し、最初の `$$$` を `\dollar\dollar@begin` に変更

しました。 `\providecommand` で `\dollar\dollar@begin` を定義しているので、古い  $\mathrm{L}^{\mathrm{A}}\mathrm{T}_{\mathrm{E}}\mathrm{X} 2_{\epsilon}$  カーネルでも問題ありません。

```

341 \providecommand\dollar\dollar@begin{$$}
342 \def\eqnarray{%
343   \stepcounter{equation}%
344   \def\@currentlabel{\p@equation\theequation}%
345   \def\@currentcounter{equation}%
346   \global\@eqnswtrue
347   \m@th
348   \global\@eqcnt\z@
349   \tabskip\@centering
350   \let\@eqncr
351   \dollar\dollar@begin\everycr{}\halign to\displaywidth\bgroup
352     \hskip\@centering$\displaystyle\tabskip\z@skip{##}$\@eqnse
353     &\global\@eqcnt\@ne \hfil$\displaystyle{##}$\hfil
354     &\global\@eqcnt\tw@ $\displaystyle{##}$\hfil\tabskip\@centering
355     &\global\@eqcnt\thr@@ \hb@xt@\z@\bgroup\hss##\egroup
356     \tabskip\z@skip
357   \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]  $\mathrm{L}^{\mathrm{A}}\mathrm{T}_{\mathrm{E}}\mathrm{X} 2_{\epsilon}$  2021-11-15 (ltmath.dtx 2021/10/14 v1.2j) で `\@currentcounter` が追加されましたので、追従します。

```

358 \DeclareOption{leqno}{\input{leqno.clo}}
359 \DeclareOption{fleqn}{\input{fleqn.clo}}%
360 % fleqn 用の eqnarray 環境の再定義
361 \def\eqnarray{%
362   \stepcounter{equation}%
363   \def\@currentlabel{\p@equation\theequation}%
364   \def\@currentcounter{equation}%
365   \global\@eqnswtrue\m@th
366   \global\@eqcnt\z@
367   \tabskip\mathindent
368   \let\@eqncr
369   \setlength\abovedisplayskip{\topsep}%
370   \ifvmode
371     \addtolength\abovedisplayskip{\partopsep}%
372   \fi
373   \addtolength\abovedisplayskip{\parskip}%
374   \setlength\belowdisplayskip{\abovedisplayskip}%
375   \setlength\belowdisplayshortskip{\abovedisplayskip}%
376   \setlength\abovedisplayshortskip{\abovedisplayskip}%
377   \dollar\dollar@begin\everycr{}\halign to\linewidth% $$
378   \bgroup
379     \hskip\@centering$\displaystyle\tabskip\z@skip{##}$\@eqnse
380     &\global\@eqcnt\@ne \hfil$\displaystyle{##}$\hfil
381     &\global\@eqcnt\tw@

```

```

382      $\displaystyle{##}$\hfil \tabskip\@centering
383      &\global\@eqcnt\thr@@ \hb@xt@\z@\bgroup\hss##\egroup
384      \tabskip\z@skip\cr
385      }}

```

■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。

```

386 % \DeclareOption{openbib}{%
387 %   \AtEndOfPackage{%
388 %     \renewcommand\@openbib@code{%
389 %       \advance\leftmargin\bibindent
390 %       \itemindent -\bibindent
391 %       \listparindent \itemindent
392 %       \parsep \z@}%
393 %     \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：

```

394 %\let\bxjs@enablejfam\@undefined

enablejfam オプションの処理。
395 \def\bxjs@kv@enablejfam@true{\let\bxjs@enablejfam=t}
396 \def\bxjs@kv@enablejfam@false{\let\bxjs@enablejfam=f}
397 \def\bxjs@kv@enablejfam@default{\let\bxjs@enablejfam\@undefined}
398 \define@key{bxjs}{enablejfam}[true]{%
399   \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。
400 \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。

```
401 \newif\ifjsDraft
402 \DeclareOption{draft}{\jsDrafttrue \overfullrule=5pt }
403 \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。

```
404 \newif\ifbxjs@papersize
405 \bxjs@papersizetrue
406 \DeclareOption{nopapersize}{\bxjs@papersizefalse}
407 \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。

---

```
408 \newif\if@english
409 \@englishfalse
410 \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 IAT<sub>E</sub>X 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。

---

```
411 \newif\if@jslogo \@jslogofalse
412 \DeclareOption{jslogo}{\@jslogotrue}
413 \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 T<sub>E</sub>X 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。

```
414 \mathchardef\bxjs@isc@ll=128
415 \mathchardef\bxjs@isc@sl=259
416 \def\bxjs@isc@sl@h{65539 }
417 \def\bxjs@invscale#1#2{%
418   \begingroup \@tempdima=#1\relax \@tempdimb#2\p@\relax
419   \ifdim\@tempdima<\bxjs@isc@ll\p@
420     \@tempcnta\@tempdima \multiply\@tempcnta\@cclvi
421     \divide\@tempcnta\@tempdimb \multiply\@tempcnta\@cclvi
422   \else
423     \@tempcnta\@tempdima \divide\@tempcnta\@tempdimb
424     \multiply\@tempcnta\p@ \let\bxjs@isc@sl\bxjs@isc@sl@h
425   \fi
426   \@tempcntb\p@ \divide\@tempcntb\@tempdimb
427   \advance\@tempcnta-\@tempcntb \advance\@tempcnta-\tw@
428   \@tempdimb\@tempcnta\@ne
429   \advance\@tempcnta\@tempcntb \advance\@tempcnta\@tempcntb
430   \advance\@tempcnta\bxjs@isc@sl \@tempdimc\@tempcnta\@ne
431   \@whiledim\@tempdimb<\@tempdimc\do{%
432     \@tempcntb\@tempdimb \advance\@tempcntb\@tempdimc
433     \advance\@tempcntb\@ne \divide\@tempcntb\tw@
434   \ifdim #2\@tempcntb>\@tempdima
435     \advance\@tempcntb\m@ne \@tempdimc=\@tempcntb\@ne
```

```

436     \else \@tempdimb=\@tempcntb\@ne \fi}%
437     \xdef\bxjs@gtmpa{\the\@tempdimb}%
438     \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 L<sup>A</sup>T<sub>E</sub>X 用の既定テンプレートを用いて他形式から L<sup>A</sup>T<sub>E</sub>X (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。

```

439 \DeclareOption{pandoc}{%
440   \bxjs@apply@pandoc@opt}
441 \@onlypreamble\bxjs@apply@pandoc@opt
442 \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。

```

443   \g@addto@macro\bxjs@post@option@hook{%
444     \bxjs@oldfontcommandstrue
445     \setkeys{bxjs}{ja=pandoc}%
446     \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)。

```

447   \ifx\bxjs@driver@opt\@undefined
448     \def\bxjs@driver@opt{dvipdfmx}%
449     \bxjs@dvi@opttrue
450   \fi
451   \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。

```

452 \DeclareOption{pandoc+}{%
453   \g@addto@macro\bxjs@post@option@hook{%
454     \edef\jsJaParam{\bxjs@catopt{_plus}\jsJaParam}}%
455   \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。

```
456 %\let\bxjs@engine@given\@undefined
```

`\bxjs@engine@opt` 明示されたエンジンのオプション名。

```
457 %\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 L<sup>A</sup>T<sub>E</sub>X の慣習に従って「ドライバの指定」とみなすべきだから。

```

458 \DeclareOption{autodetect-engine}{%
459   \let\bxjs@engine@given=*}
460 \DeclareOption{latex}{%
461   \def\bxjs@engine@opt{latex}%
462   \let\bxjs@engine@given=n}
463 \DeclareOption{platex}{%
464   \def\bxjs@engine@opt{platex}%
465   \let\bxjs@engine@given=j}
466 \DeclareOption{uplatex}{%
467   \def\bxjs@engine@opt{uplatex}%
468   \let\bxjs@engine@given=u}
469 \DeclareOption{xelatex}{%
470   \def\bxjs@engine@opt{xelatex}%
471   \let\bxjs@engine@given=x}
472 \DeclareOption{pdflatex}{%
473   \def\bxjs@engine@opt{pdflatex}%
474   \let\bxjs@engine@given=p}
475 \DeclareOption{lualatex}{%
476   \def\bxjs@engine@opt{lualatex}%
477   \let\bxjs@engine@given=l}
478 \DeclareOption{platex-ng}{%
479   \def\bxjs@engine@opt{platex-ng}%
480   \let\bxjs@engine@given=g}
481 \DeclareOption{platex-ng*}{%
482   \def\bxjs@engine@opt{platex-ng*}%
483   \let\bxjs@platexng@nodrv=t%
484   \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。

```

485 %\let\bxjs@driver@given\@undefined
486 \let\bxjs@driver@@dvimode=0
487 \let\bxjs@driver@@dvipdfmx=1
488 \let\bxjs@driver@@pdfmode=2
489 \let\bxjs@driver@@xetex=3
490 \let\bxjs@driver@@dvips=4
491 \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。

```

492 %\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。

```

493 \DeclareOption{dvips}{%
494   \def\bxjs@driver@opt{dvips}%
495   \let\bxjs@driver@given\bxjs@driver@@dvips}
496 \DeclareOption{dviout}{%

```



```

497 \def\bxjs@driver@opt{dviout}%
498 \let\bxjs@driver@given\bxjs@driver@@dvimode}
499 \DeclareOption{xdvi}{%
500 \def\bxjs@driver@opt{xdvi}%
501 \let\bxjs@driver@given\bxjs@driver@@dvimode}
502 \DeclareOption{dvipdfmx}{%
503 \def\bxjs@driver@opt{dvipdfmx}%
504 \let\bxjs@driver@given\bxjs@driver@@dvipdfmx}
505 \DeclareOption{nodvidriver}{%
506 \def\bxjs@driver@opt{nodvidriver}%
507 \let\bxjs@driver@given\bxjs@driver@@none}
508 \DeclareOption{class-nodvidriver}{%
509 \def\bxjs@driver@opt{class-nodvidriver}%
510 \let\bxjs@driver@given\bxjs@driver@@none}
511 \DeclareOption{pdftex}{%
512 \def\bxjs@driver@opt{pdftex}%
513 \let\bxjs@driver@given\bxjs@driver@@pdfmode}
514 \DeclareOption{luatex}{%
515 \def\bxjs@driver@opt{luatex}%
516 \let\bxjs@driver@given\bxjs@driver@@pdfmode}
517 \DeclareOption{xetex}{%
518 \def\bxjs@driver@opt{xetex}%
519 \let\bxjs@driver@given\bxjs@driver@@xetex}

dvipdfmx-if-dvi は 2.0 版より非推奨となった。
520 \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 521 \@onlypreamble\bxjs@depre@opt
522 \def\bxjs@depre@opt#1#2{%
523 \ClassWarningNoLine\bxjs@clsname
524 {The old option '#1' is DEPRECATED\MessageBreak
525 and may be abolished in future!\MessageBreak
526 You should instead write:\MessageBreak
527 \space\space #2}}
528 \@onlypreamble\bxjs@depre@opt@do
529 \def\bxjs@depre@opt@do#1#2{%
530 \bxjs@depre@opt{#1}{#2}%
531 \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。

```

532 \newif\ifbxjs@bigcode \bxjs@bigcodetrue

```

nobigcode / bigcode オプションの定義。

```
533 \DeclareOption{nobigcode}{%
534   \bxjs@bigcodefalse}
535 \DeclareOption{bigcode}{%
536   \bxjs@bigcodetrue}
```

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

```
537 \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。

```
538 \DeclareOption{nooldfontcommands}{%
539   \bxjs@oldfontcommandsfalse}
540 \DeclareOption{oldfontcommands}{%
541   \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。

```
542 \ifbxjs@brace@safe
543   \@onlypreamble\bxjs@register@badopt
544   \def\bxjs@register@badopt#1{%
545     \expandafter\@onlypreamble\csname bxjs@badopt/#1\endcsname
546     \@namedef{bxjs@badopt/#1}}
547   \g@addto@macro\bxjs@begin@document@hook{%
548     \@for\bxjs@tmpa:=\@unusedoptionlist\do{%
549       \@nameuse{bxjs@badopt/\bxjs@tmpa}}}
550 \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。

```
551 \@onlypreamble\bxjs@invalid@opt
552 \ifbxjs@brace@safe
553   \def\bxjs@invalid@opt#1#2{%
554     \bxjs@register@badopt{#1}{\ClassError\bxjs@clsname{#2}\@ehc}}
555 \else
556   \def\bxjs@invalid@opt#1#2{%
557     \DeclareOption{#1}{\ClassWarningNoLine\bxjs@clsname{#2}}}
558 \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。

```
559 \bxjs@invalid@opt{winjis}{%
```

```

560 This class does not support 'winjis' option}
561 \bxjs@invalid@opt{mingoth}{%
562 This class does not support 'mingoth' option}
563 \bxjs@invalid@opt{jis}{%
564 This class does not support 'jis' option}
565 \if j\jsEngine\else
566 \bxjs@invalid@opt{tombo}{%
567 Option 'tombo' can be used only on (u)pLaTeX}
568 \bxjs@invalid@opt{tombow}{%
569 Option 'tombow' can be used only on (u)pLaTeX}
570 \bxjs@invalid@opt{mentuke}{%
571 Option 'mentuke' can be used only on (u)pLaTeX}
572 \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。

```

573 \DeclareOption*{%
574 \bxjs@check@ja@prefix \ifx\bxjs@next\relax
575 \def\bxjs@next{\bxjs@cls@setkeys{bxjs}}}%
576 \expandafter\bxjs@next\expandafter{\CurrentOption}%
577 \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。

```

578 \edef\bxjs@next{%
579 \noexpand\setkeys{bxjs}{japaram={\bxjs@next}}}%
580 }\bxjs@next
581 \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。

```

582 \def\bxjs@check@ja@prefix{%
583 \let\bxjs@next\relax
584 \expandafter\bxjs@check@ja@prefix@a\CurrentOption\@nil ja:\@nil\@nnil}
585 \def\bxjs@check@ja@prefix@a#1ja:#2\@nil#3\@nnil{%
586 \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。

```

587 \def\bxjs@safe@setkeys#1#2{%
588 \let\bxjs@save@KV@errx\KV@errx \let\KV@errx\@gobble
589 \setkeys{#1}{#2}%
590 \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。

```

591 \def\bxjs@cls@setkeys#1#2{%
592 \let\bxjs@save@KV@errx\KV@errx
593 \def\KV@errx##1{\OptionNotUsed}%
594 \setkeys{#1}{#2}%

```

```

595 \let\KV@errx\bxjs@save@KV@errx}
596 \ifbxjs@brace@safe\else
597 \let\bxjs@cls@setkeys\bxjs@safe@setkeys
598 \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。

```

599 \@onlypreamble\bxjs@declare@enum@option
600 \def\bxjs@declare@enum@option#1#2#3{%
601 \bxjs@csletcs{bxjs@#2}{bxjs@#2@@#3}%
602 \define@key{bxjs}{#1}{%
603 \ifundefined{bxjs@#2@@#1}{%
604 \bxjs@error@keyval{#1}{##1}%
605 }\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。

```

606 \@onlypreamble\bxjs@declare@bool@option
607 \def\bxjs@declare@bool@option#1#2#3{%
608 \csname newif\expandafter\endcsname\csname ifbxjs@#2\endcsname
609 \@nameuse{bxjs@#2#3}%
610 \define@key{bxjs}{#1}[true]{%
611 \ifundefined{bxjs@#2##1}{%
612 \bxjs@error@keyval{#1}{##1}%
613 }\@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。

```

614 \def\bxjs@set@keyval#1#2#3{%
615 \bxjs@csletcs{bxjs@next}{bxjs@kv@#1@#2}%
616 \ifx\bxjs@next\relax
617 \bxjs@error@keyval{#1}{#2}%
618 #3%
619 \else \bxjs@next
620 \fi}
621 \@onlypreamble\bxjs@error@keyval
622 \def\bxjs@error@keyval#1#2{%
623 \ClassError\bxjs@clsname
624 {Invalid value '#2' for option #1}\@ehc}

```

`\jsScale` [実数値マクロ] 和文スケール値。

```

625 \def\jsScale{0.924715}

```

`\bxjs@base@opt` 明示された base オプションの値。

```

626 %\let\bxjs@base@opt\@undefined

```

base オプションの処理。

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

\bxjs@jbase@opt 明示された jbase オプションの値。

```
631 %\let\bxjs@jbase@opt\@undefined
```

jbase オプションの処理。

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

\bxjs@scale@opt 明示された scale オプションの値。

```
634 %\let\bxjs@scale@opt\@undefined
```

scale オプションの処理。

```
635 \define@key{bxjs}{scale}{%
636   \edef\bxjs@scale@opt{#1}%
637   \let\jsScale\bxjs@scale@opt}
638 \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。

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

\bxjs@param@mag mag オプションの値。

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

mag オプションの処理。

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

paper オプションの処理。

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

\bxjs@jadriver 和文ドライバの名前。

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

\bxjs@jadriver@opt 明示された和文ドライバの名前。

```
644 %\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）。

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

`\jsJaFont` 和文フォント設定の名前。

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

`jafont` オプションの処理。

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

`\jsJaParam` 和文ドライバパラメタの文字列。

```
651 \let\jsJaParam\@empty
```

`japaram` オプションの処理。

```
652 \define@key{bxjs}{japaram}{%
```

```
653 \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。

```
654 \define@key{bxjs}{pandoc}[]{%
```

```
655 \ExecuteOptions{pandoc}%
```

```
656 \edef\jsJaParam{\bxjs@catopt\jsJaParam{#1}}}
```

```
657 \define@key{bxjs}{pandoc+}[]{%
```

```
658 \ExecuteOptions{pandoc+}%
```

```
659 \edef\jsJaParam{\bxjs@catopt\jsJaParam{#1}}}
```

`\bxjs@magstyle` `magstyle` 設定値。(古いイマイチな名前。)

```
660 \let\bxjs@magstyle@@mag=m
```

```
661 \let\bxjs@magstyle@@real=r
```

```
662 \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。

```
663 \let\bxjs@magstyle@@usemag\bxjs@magstyle@@mag
```

```
664 \let\bxjs@magstyle@@nomag\bxjs@magstyle@@real
```

```
665 \bxjs@cslet{bxjs@magstyle@@nomag*}\bxjs@magstyle@@xreal
```

`\bxjs@magstyle@@default` は既定の値を表す。

```
666 \let\bxjs@magstyle@@default\bxjs@magstyle@@usemag
```

```
667 \ifx l\jsEngine \ifnum\luatexversion>86
```

```
668 \let\bxjs@magstyle@@default\bxjs@magstyle@@xreal
```

```
669 \fi\fi
```

```
670 \ifjsWithpTeXng
```

```
671 \let\bxjs@magstyle@@default\bxjs@magstyle@@xreal
```

```
672 \fi
```

```
673 \let\bxjs@magstyle\bxjs@magstyle@@default
```

`magstyle` オプションの処理。

```
674 \define@key{bxjs}{magstyle}{%
```

```
675 \bxjs@csletcs{bxjs@magstyle}{bxjs@magstyle@@#1}%
```

```
676 \ifx\bxjs@magstyle\relax
```

```
677 \bxjs@error@keyval{magstyle}{#1}%
```

```
678 \let\bxjs@magstyle\bxjs@magstyle@@default
```

```
679 \fi}
```

`\bxjs@geometry geometry` オプションの指定値。

```
680 \let\bxjs@geometry@@class=c
681 \let\bxjs@geometry@@user=u
682 \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。

```
683 \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。

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

DVI モードのドライバとドライバ種別との対応。

```
685 \let\bxjs@dvidriver@@dvipdfmx=\bxjs@driver@@dvipdfmx
686 \let\bxjs@dvidriver@@dvips=\bxjs@driver@@dvips
687 \let\bxjs@dvidriver@@dviout=\bxjs@driver@@dvimode
688 \let\bxjs@dvidriver@@xdvi=\bxjs@driver@@dvimode
689 \let\bxjs@dvidriver@@nodvidriver=\bxjs@driver@@none
690 \bxjs@cslet{bxjs@dvidriver@@class-nodvidriver}\bxjs@driver@@none
```

`dvi` オプションの処理。

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

`\bxjs@driver@given` を未定義にしていることに注意。

```
696   \def\bxjs@driver@opt{#1}%
697   \let\bxjs@driver@given\@undefined
698   \bxjs@dvi@opttrue
699 \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。

```
700 \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。

```
701 \newif\ifbxjs@force@chapterabstract
702 %<book>\bxjs@force@chapterabstracttrue
```

`layout` オプションの処理。

```
703 \@namedef{bxjs@kv@layout@v1}{%
704 %<book>\bxjs@layout@buggyhmargintrue
705 %<report>\bxjs@force@chapterabstracttrue
706 }
707 \@namedef{bxjs@kv@layout@v2}{%
708 %<book>\bxjs@layout@buggyhmarginfalse
709 %<report>\bxjs@force@chapterabstractfalse
```

```

710 }
711 \define@key{bxjs}{layout}{%
712   \bxjs@set@keyval{layout}{#1}{}}

```

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

```

713 %\let\bxjs@textwidth@limit@opt\@undefined
714 \define@key{bxjs}{textwidth-limit}{%
715   \bxjs@depre@opt{textwidth-limit}{textwidth=#1zw}%
716   \edef\bxjs@textwidth@limit@opt{#1}}

```

`\bxjs@textwidth@opt` `textwidth` の指定値。

```

717 %\let\bxjs@textwidth@opt\@undefined
718 \define@key{bxjs}{textwidth}{\edef\bxjs@textwidth@opt{#1}}
719 \define@key{bxjs}{line_length}{\setkeys{bxjs}{textwidth=#1}}

```

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

```

720 %\let\bxjs@number@of@lines@opt\@undefined
721 \define@key{bxjs}{number-of-lines}{\edef\bxjs@number@of@lines@opt{#1}}
722 \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。

```

723 %\let\bxjs@paragraph@mark\@undefined
724 \define@key{bxjs}{paragraph-mark}{%
725   \edef\bxjs@paragraph@mark{#1}}

```

`\ifbxjs@whole@zw@lines` [スイッチ] `whole-zw-lines` の指定値。

```

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

```

`\ifbxjs@jaspace@cmd` [スイッチ] `jaspace-cmd` の指定値。

```

727 \bxjs@declare@bool@option{jaspace-cmd}{jaspace@cmd}{true}
728 \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。

```

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

```

`\ifbxjs@hyperref@enc` [スイッチ] `hyperref-enc` の指定値。

```

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

```

`\bxjs@everyparhook` `everyparhook` の指定値。

```

731 \chardef\bxjs@everyparhook@none=0
732 \chardef\bxjs@everyparhook@compat=1
733 \chardef\bxjs@everyparhook@modern=2
734 \bxjs@declare@enum@option{everyparhook}{everyparhook}{%
735   \if j\jsEngine compat\else modern\fi}

```

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

```

736 \chardef\bxjs@label@section@none=0
737 \chardef\bxjs@label@section@compat=1
738 \chardef\bxjs@label@section@modern=2
739 \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。

```
740 \bxjs@declare@bool@option{use-zw}{usezw}{true}
741 \DeclareOption{noz}{\bxjs@depre@opt@do{noz}{use-zw=false}}
742 \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。

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

`\ifbxjs@precisetext` [スイッチ] `precise-text` の指定値。

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

`\ifbxjs@simplejasetup` [スイッチ] `simple-ja-setup` の指定値。

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

`\ifbxjs@plautopatch` [スイッチ] `plautopatch` の指定値。

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

## ■ オプションの実行

---

L<sup>A</sup>T<sub>E</sub>X 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 引数を通るようにする。

※ 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。

```
755 \ifbxjs@brace@safe\else
756 \let\bxjs@org@removeelement\@removeelement
757 \def\@removeelement#1#2#3{%
758   \def\reserved@a{#2}%
759   \ifx\reserved@a\@empty \let#3\@empty
760   \else \bxjs@org@removeelement{#1}{#2}{#3}%
761   \fi}
762 \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。

```
763 %<article>\ExecuteOptions{a4paper,oneside,onecolumn,notitlepage,final}
764 %<book>\ExecuteOptions{a4paper,twoside,onecolumn,titlepage,openright,final}
765 %<report>\ExecuteOptions{a4paper,oneside,onecolumn,titlepage,openany,final}
766 %<slide>\ExecuteOptions{36pt,a4paper,landscape,oneside,onecolumn,titlepage,final}
767 \ProcessOptions\relax
768 \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。

```
769 \if@slide
770   \def\maybeblue{\ifundefined{ver@color.sty}{\color{blue}}{}}
771 \fi
772 %<jsclasses>
773 \if@landscape
774   \setlength\@tempdima {\paperheight}
775   \setlength\paperheight{\paperwidth}
776   \setlength\paperwidth {\@tempdima}
777 \fi
778 %</jsclasses>
```

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

**TODO: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。)

```
779 \def\bxjs@tmpdo{%
780   \def\bxjs@tmpa{\@gobble}%
781   \expandafter\bxjs@tmpdo@a\@classoptionslist,\@nil,%
782   \let\@classoptionslist\bxjs@tmpa}
783 \def\bxjs@tmpdo@a#1,{%
784   \ifx\@nil#1\relax\else
785     \bxjs@tmpdo@b#1{\@nil
786       \if@tempswa \edef\bxjs@tmpa{\bxjs@tmpa,#1}\fi
787       \expandafter\bxjs@tmpdo@a
788     \fi}
789 \def\bxjs@tmpdo@b#1#\bxjs@tmpdo@c}
790 \def\bxjs@tmpdo@c#1\@nil{%
791   \ifx\@nil#1\@nil \@tempwattrue \else \@tempwafalse \fi}
792 \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。

```

793 \@expandtwoargs\@removeelement
794 {papersize}\@classoptionslist\@classoptionslist
795 \@expandtwoargs\@removeelement
796 {10pt}\@classoptionslist\@classoptionslist
797 \@expandtwoargs\@removeelement
798 {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 p $\text{\LaTeX}$  か up $\text{\LaTeX}$  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] p $\text{\LaTeX}$ / up $\text{\LaTeX}$  を自動判別するオプション autodetect-engine を新設しました。up $\text{\LaTeX}$  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。

```

799 \ifx\bxjs@jadriver@opt\undefined\else
800   \let\bxjs@jadriver\bxjs@jadriver@opt
801 \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。

```

802 \let\bxjs@tmpb\jsEngine
803 \ifx j\bxjs@tmpb\ifjsWithpTeXng
804   \let\bxjs@tmpb=g
805 \fi\fi
806 \ifx j\bxjs@tmpb\ifjsWithupTeX
807   \let\bxjs@tmpb=u
808 \fi\fi
809 \ifx p\bxjs@tmpb\ifjsInPdfMode\else
810   \let\bxjs@tmpb=n
811 \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。)

```

812 \ifx *\bxjs@engine@given
813   \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)p $\text{\LaTeX}$  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。

```

814 \ifx j\bxjs@engine@given
815   \g@addto@macro\@classoptionslist{,platex}
816 \else\ifx u\bxjs@engine@given

```

```

817 \g@addto@macro\@classoptionslist{,uplatex}
818 \fi\fi
819 \fi
820 \ifx\bxjs@engine@given\@undefined\else
821 \ifx\bxjs@engine@given\bxjs@tmpb\else
822 \ClassError\bxjs@clsname
823 {Option '\bxjs@engine@opt' used on wrong engine}\@ehc
824 \fi
825 \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。

```

826 \ifjsWithpTeXng
827 \g@addto@macro\@classoptionslist{,uplatex}
828 \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。

```

829 \@tempwattrue
830 \ifx \bxjs@driver@given\@undefined\else
831 \ifjsInPdfMode
832 \ifx\bxjs@driver@given\bxjs@driver@@pdfmode\else
833 \@tempwafalse
834 \fi
835 \else\ifx x\jsEngine
836 \ifx\bxjs@driver@given\bxjs@driver@@xetex\else
837 \@tempwafalse
838 \fi
839 \else
840 \ifx\bxjs@driver@given\bxjs@driver@@pdfmode
841 \@tempwafalse
842 \else\ifx\bxjs@driver@given\bxjs@driver@@xetex
843 \@tempwafalse
844 \fi\fi
845 \ifjsWithpTeXng\ifx\bxjs@driver@given\bxjs@driver@@dvipdfmx\else
846 \@tempwafalse
847 \fi\fi
848 \fi\fi
849 \fi
850 \if@tempwa\else
851 \ClassError\bxjs@clsname
852 {Option '\bxjs@driver@opt' used on wrong engine}\@ehc
853 \fi

```

DVI 出力のエンジンである場合の追加処理。

```

854 \ifjsInPdfMode \@tempwafalse
855 \else\ifx x\jsEngine \@tempwafalse
856 \else\ifjsWithpTeXng \@tempwafalse
857 \else \@tempwattrue

```

```

858 \fi\fi\fi
859 \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)。

```

860 \ifx\bxjs@driver@opt\@undefined
861 \if \ifbxjs@explIIII T\else\ifx\bxjs@jadriver@opt\@undefined F\else T\fi\fi T%
862 \ClassWarningNoLine\bxjs@clsname
863 {A driver option is MISSING!!\MessageBreak
864 You should properly specify one of the valid\MessageBreak
865 driver options according to the DVI driver\MessageBreak
866 that is in use:\MessageBreak
867 \@spaces dvips, dvipdfmx, dviout, xdvi,\MessageBreak
868 \@spaces nodvidriver}
869 \fi
870 \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。)

```

871 \ifbxjs@dvi@opt
872 \edef\bxjs@next{%
873 \let\noexpand\bxjs@driver@given
874 \csname bxjs@dvidriver@@\bxjs@driver@opt\endcsname
875 \noexpand\g@addto@macro\noexpand\@classoptionslist
876 {,\bxjs@driver@opt}%
877 }\bxjs@next
878 \fi
879 \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。

```

880 \ifjsWithpTeXng
881 \ifx\bxjs@driver@given\bxjs@driver@@dvipdfmx
882 \let\bxjs@platexng@nodrv\@undefined
883 \else\ifx t\bxjs@platexng@nodrv\else
884 \g@addto@macro\@classoptionslist{,dvipdfmx}
885 \fi\fi
886 \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。

```

887 \ifx\bxjs@driver@given\bxjs@driver@@none
888 \bxjs@papersizefalse
889 \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。

```

890 \@ifpackageloaded{plautopatch}{%
891 \bxjs@plautopatchtrue

```

892 }{}

標準の和文ドライバの名前の定数。

```
893 \def\bxjs@@minimal{minimal}
894 \def\bxjs@@standard{standard}
895 \def\bxjs@@pandoc{pandoc}
896 \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。

```
897 \ifx\bxjs@jadriver\relax
898   \ifx j\jsEngine
899     \let\bxjs@jadriver\bxjs@@standard
900   \else
901     \ClassWarningNoLine\bxjs@clsname
902     {The option 'ja' is MISSING!!\MessageBreak
903     So 'ja=minimal' is assumed as fallback, but\MessageBreak
904     such implicit setting is now DEPRECATED!\MessageBreak
905     You should write 'ja=minimal' explicitly,\MessageBreak
906     if it is intended}
907     \let\bxjs@jadriver\bxjs@@minimal
908   \fi
909 \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。

```
910 \ifx\bxjs@jadriver\bxjs@@pandoc \ifx\bxjs@plautopatch@given\@undefined
911   \ifjsWithTeX
912     \bxjs@plautopatchtrue
913   \fi\fi\fi
914 \ifx j\jsEngine \ifbxjs@plautopatch
915   \RequirePackage{plautopatch}[2018/08/22]%v0.3
916 \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。

```
917 \ifx\bxjs@jadriver@opt\@undefined\else
918   \ifx\bxjs@engine@given\@undefined
919     \ClassError\bxjs@clsname
920     {An engine option must be explicitly given}%
921     {When you use a Japanese-driver you must specify a correct\MessageBreak
922     engine option.\MessageBreak\@ehc}
923 \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。)

```

924 \ifx\bxjs@magstyle@@default\bxjs@magstyle@@mag\else
925   \ifx\bxjs@magstyle\bxjs@magstyle@@mag
926     \let\bxjs@magstyle\bxjs@magstyle@@default
927     \ClassError\bxjs@clsname
928       {The engine does not support 'magstyle=usemag'%}
929       {LuaTeX v0.87 or later no longer supports the "mag" feature of TeX.\MessageBreak
930         The default value 'nomag*' is used instead.\MessageBreak \@ehc}
931   \fi
932 \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。

```

933 \ifx\bxjs@jbase@opt\@undefined\else
934   \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。

```

935     \jsSetQHLLength\@tempdima{\bxjs@jbase@opt}%
936     \bxjs@invscale\@tempdima\jsScale
937     \bxjs@setbasefontsize{\@tempdima}%
938   \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。

```

939     \ifx\bxjs@scale@opt\@undefined\else
940       \ClassWarningNoLine\bxjs@clsname
941         {Redundant 'scale' option is ignored}%
942     \fi
943     \jsSetQHLLength\@tempdima{\bxjs@jbase@opt}%
944     \@tempdimb=\bxjs@param@basefontsize\relax
945     \edef\jsScale{\strip@pt\@tempdimb}%
946     \bxjs@invscale\@tempdima\jsScale
947     \edef\jsScale{\strip@pt\@tempdima}%
948   \fi
949 \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。

```

950 \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。

```

951 \ifbxjs@disguise@js
952 %<book|report>\def\bxjs@js@clsname{jsbook}
953 %<!book&!report>\def\bxjs@js@clsname{jsarticle}
954   \@namedef{ver@\bxjs@js@clsname.cls}{2001/01/01 (bxjs)}
955 \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。

```
956 \g@addto@macro{\classoptionslist}{nosetpagesize}
```

`oldfontcommands` オプション指定時は `\allowoldfontcommands` 命令を実行する。

```
957 \ifbxjs@oldfontcommands
```

```
958 \AtEndOfClass{\allowoldfontcommands}
```

```
959 \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 p $\text{\LaTeX}$  2 $\epsilon$  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]  $\text{\LaTeX}$  2 $\epsilon$  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う (Issue #72)。

[2022-09-12] 次期  $\text{\LaTeX}$  2 $\epsilon$  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。

---

```
960 \ifx\stockwidth\@undefined\newdimen\stockwidth\fi
```

```
961 \ifx\stockheight\@undefined\newdimen\stockheight\fi
```

```
962 \begingroup\expandafter\expandafter\expandafter\endgroup
```

```
963 \expandafter\ifx\csname iftombow\expandafter\endcsname\csname iftrue\endcsname
```

```
964 \setlength{\stockwidth}{\paperwidth}
```

```
965 \setlength{\stockheight}{\paperheight}
```

```
966 \advance \stockwidth 2in
```

```
967 \advance \stockheight 2in
```



```
968 \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。

```
969 %<slide>\def\n@baseline{13}%
970 %<!slide>\ifdim\bxjs@param@basefontsize<10pt \def\n@baseline{15}%
971 %<!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。

```
972 \ifx\bxjs@magstyle\bxjs@magstyle@@mag
973   \jsc@magtrue
974 \else\ifx\bxjs@magstyle\bxjs@magstyle@@xreal
975   \jsc@mag@xrealtrue
976 \fi\fi
```

---

サイズの変更は  $\mathrm{T\!E\!X}$  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。

```
977 \ifx\bxjs@param@mag\relax
978   \@tempdima=\bxjs@param@basefontsize
979   \advance\@tempdima.001pt \multiply\@tempdima25
980   \divide\@tempdima16384\relax \@tempcnta\@tempdima\relax
981   \edef\bxjs@param@mag{\the\@tempcnta}
982 \else
983 % mag 値が直接指定された場合
984   \bxjs@gset@tempcnta{\bxjs@param@mag}
985   \ifnum\@tempcnta<\z@ \@tempcnta=\z@ \fi
986 % 有効な mag 値の範囲は 1--32768
987   \edef\bxjs@param@mag{\the\@tempcnta}
988   \advance\@tempcnta100000
989   \def\bxjs@tmpa#1#2#3#4#5\@nil{\@tempdima=#2#3#4.#5\p@}
990   \expandafter\bxjs@tmpa\the\@tempcnta\@nil
991   \edef\bxjs@param@basefontsize{\the\@tempdima}
992 \fi
```

---

```

993 \@tempcnta\bxjs@param@mag \advance\@tempcnta100000
994 \def\bxjs@tmpa#1#2#3#4\@nil{\@tempdima=#2#3.#4\p@}
995 \expandafter\bxjs@tmpa\the\@tempcnta\@nil
996 \edef\jsc@magscale{\strip@pt\@tempdima}
997 \let\jsBaseFontSize\bxjs@param@basefontsize

```

---

[2016-07-08] \jsc@empt 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。

```

998 \newdimen\jsc@empt
999 \newdimen\jsc@mmm
1000 \ifjsc@mag
1001   \jsc@empt=1\p@
1002   \jsc@mmm=1mm
1003 \else
1004   \jsc@empt=\jsc@magscale\p@
1005   \jsc@mmm=\jsc@magscale mm
1006 \fi

```

ここで p<sub>TEX</sub> 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。

```

1007 \newdimen\jsZw
1008 \jsZw=10\jsc@empt \jsZw=\jsScale\jsZw
1009 \ifbxjs@usezw
1010   \providecommand*\zw{\jsZw}
1011 \fi

```

\zwspace 全角幅の水平空き。

```

1012 \def\zwspace{\hskip\jsZw\relax}

```

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

```

1013 \ifjsc@mag@xreal
1014   \RequirePackage{type1cm}
1015   \let\jsc@invscale\bxjs@invscale

```

---

```

1016 \ifbxjs@TUenc
1017   \expandafter\let\csname TU/lmr/m/n/10\endcsname\relax
1018 \else
1019   \expandafter\let\csname OT1/cmr/m/n/10\endcsname\relax
1020 \fi
1021 \expandafter\let\csname OMX/cmex/m/n/10\endcsname\relax
1022 \let\jsc@get@external@font\get@external@font
1023 \def\get@external@font{%
1024   \jsc@preadjust@extract@font
1025   \jsc@get@external@font}

```

```

1026 \def\jsc@fstrunc#1{%
1027   \edef\jsc@tmpa{\strip@pt#1}%
1028   \expandafter\jsc@fstrunc@a\jsc@tmpa.****\@nil}
1029 \def\jsc@fstrunc@a#1.#2#3#4#5#6\@nil{%
1030   \if#5*\else
1031     \edef\jsc@tmpa{#1%
1032       \ifnum#2#3>\z@ .#2\ifnum#3>\z@ #3\fi\fi}%
1033   \fi}
1034 \def\jsc@preadjust@extract@font{%
1035   \let\jsc@req@size\f@size
1036   \dimen@f@size\p@ \jsc@invscale\dimen@\jsc@magscale
1037   \advance\dimen@.005pt\relax \jsc@fstrunc\dimen@
1038   \let\jsc@ref@size\jsc@tmpa
1039   \let\f@size\jsc@ref@size}
1040 \def\execute@size@function#1{%
1041   \let\jsc@cref@size\f@size
1042   \let\f@size\jsc@req@size
1043   \csname s@fct@#1\endcsname}
1044 \let\jsc@DeclareErrorFont\DeclareErrorFont
1045 \def\DeclareErrorFont#1#2#3#4#5{%
1046   \@tempdimc#5\p@ \@tempdimc\jsc@magscale\@tempdimc
1047   \edef\jsc@tmpa{{#1}{#2}{#3}{#4}{\strip@pt\@tempdimc}}
1048   \expandafter\jsc@DeclareErrorFont\jsc@tmpa}
1049 \def\gen@sfcnt{%
1050   \edef\mandatory@arg{\mandatory@arg\jsc@cref@size}%
1051   \empty@sfcnt}
1052 \def\genb@sfcnt{%
1053   \edef\mandatory@arg{%
1054     \mandatory@arg\expandafter\genb@x\jsc@cref@size..\@{}}%
1055   \empty@sfcnt}
1056 \ifbxjs@TUenc\else
1057   \DeclareErrorFont{OT1}{cmr}{m}{n}{10}
1058 \fi
1059 \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 1060 \def\jsc@smallskip{\vspace\jsc@smallskipamount}
\jsc@bigskip 1061 %\def\jsc@medskip{\vspace\jsc@medskipamount}
1062 %\def\jsc@bigskip{\vspace\jsc@bigskipamount}

```

`\jsc@smallskipamount`

```

\jsc@medskipamount 1063 \newskip\jsc@smallskipamount
\jsc@bigskipamount 1064 \jsc@smallskipamount=3\jsc@mpt plus 1\jsc@mpt minus 1\jsc@mpt
1065 %\newskip\jsc@medskipamount
1066 %\jsc@medskipamount =6\jsc@mpt plus 2\jsc@mpt minus 2\jsc@mpt

```

```
1067 %\newskip\jsc@bigskipamount
1068 %\jsc@bigskipamoun =12\jsc@empt plus 4\jsc@empt minus 4\jsc@empt
```

\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] トンボオブ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。

```
1069 % \ifpapersize
1070 %   \setlength{\@tempdima}{\paperwidth}
1071 %   \setlength{\@tempdimb}{\paperheight}
1072 %   \iftombow
1073 %     \advance \@tempdima 2truein
1074 %     \advance \@tempdimb 2truein
1075 %   \fi
1076 %   \AtBeginDvi{\special{pdf: pagesize width \the\@tempdima\space height \the\@tempdimb}}
1077 % \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。

---

```
1078 %</class>
1079 %<*jsclasses>
1080 %<*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] 上記のと通りの「クラスファイルが意図する和文スケール値 (1zw ÷ 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、 $9.62216 \text{ pt} * 0.961 / 10 \text{ pt} = 0.924690$  です。

```

1081 %</class>
1082 %<*minijs>
1083 %% min/goth -> jis/jisg (for pLaTeX only)
1084 \ifnum\jis"2121="3000 \else
1085 \@for\@tempa:=5,6,7,8,9,10,10.95,12,14.4,17.28,20.74,24.88\do{%
1086   \expandafter\let\csname JY1/mc/m/n/\@tempa\endcsname\relax
1087   \expandafter\let\csname JY1/gt/m/n/\@tempa\endcsname\relax
1088   \expandafter\let\csname JT1/mc/m/n/\@tempa\endcsname\relax
1089   \expandafter\let\csname JT1/gt/m/n/\@tempa\endcsname\relax
1090 }
1091 \def\Cjascale{0.924690}
1092 \DeclareFontShape{JY1}{mc}{m}{n}{<-> s * [0.961] jis}{}
1093 \DeclareFontShape{JY1}{gt}{m}{n}{<-> s * [0.961] jisg}{}
1094 \DeclareFontShape{JT1}{mc}{m}{n}{<-> s * [0.961] tmin10}{}
1095 \DeclareFontShape{JT1}{gt}{m}{n}{<-> s * [0.961] tgoth10}{}
1096 \fi
1097 %</minijs>
1098 %<*class>
1099 %<!*jspf>
1100 \def\Cjascale{0.924690}
1101 \ifmingoth
1102   \DeclareFontShape{\jsc@JYn}{mc}{m}{n}{<-> s * [0.961] \jsc@pfx@ min10}{}
1103   \DeclareFontShape{\jsc@JYn}{gt}{m}{n}{<-> s * [0.961] \jsc@pfx@ goth10}{}
1104   \DeclareFontShape{\jsc@JTn}{mc}{m}{n}{<-> s * [0.961] \jsc@pfx@ tmin10}{}
1105   \DeclareFontShape{\jsc@JTn}{gt}{m}{n}{<-> s * [0.961] \jsc@pfx@ tgoth10}{}
1106 \else
1107   \ifjisfont
1108     \DeclareFontShape{\jsc@JYn}{mc}{m}{n}{<-> s * [0.961] \jsc@pfx@ jis}{}
1109     \DeclareFontShape{\jsc@JYn}{gt}{m}{n}{<-> s * [0.961] \jsc@pfx@ jisg}{}
1110     \DeclareFontShape{\jsc@JTn}{mc}{m}{n}{<-> s * [0.961] \jsc@pfx@ tmin10}{}
1111     \DeclareFontShape{\jsc@JTn}{gt}{m}{n}{<-> s * [0.961] \jsc@pfx@ tgoth10}{}
1112   \else
1113     \if@jsc@uplatex
1114       \DeclareFontShape{JY2}{mc}{m}{n}{<-> s * [0.924690] upjisr-h}{}
1115       \DeclareFontShape{JY2}{gt}{m}{n}{<-> s * [0.924690] upjisg-h}{}
1116       \DeclareFontShape{JT2}{mc}{m}{n}{<-> s * [0.924690] upjisr-v}{}
1117       \DeclareFontShape{JT2}{gt}{m}{n}{<-> s * [0.924690] upjisg-v}{}
1118     \else
1119       \DeclareFontShape{\jsc@JYn}{mc}{m}{n}{<-> s * [0.961] \jsc@pfx@ jis}{}
1120       \DeclareFontShape{\jsc@JYn}{gt}{m}{n}{<-> s * [0.961] \jsc@pfx@ jisg}{}

```

```

1121      \DeclareFontShape{\jsc@JTn}{mc}{m}{n}{<-> s * [0.961] \jsc@pfx@ tmin10}{ }
1122      \DeclareFontShape{\jsc@JTn}{gt}{m}{n}{<-> s * [0.961] \jsc@pfx@ tgoth10}{ }
1123      \fi
1124      \fi
1125 \fi
1126 %</!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  $9.62216 \text{ pt} * 0.93885 / 10 \text{ pt} = 0.903375$  です。

```

1127 %<*jspf>
1128 \def\Cjascale{0.903375}
1129 \ifmingoth
1130   \DeclareFontShape{\jsc@JYn}{mc}{m}{n}{<-> s * [0.93885] \jsc@pfx@ min10}{ }
1131   \DeclareFontShape{\jsc@JYn}{gt}{m}{n}{<-> s * [0.93885] \jsc@pfx@ goth10}{ }
1132   \DeclareFontShape{\jsc@JTn}{mc}{m}{n}{<-> s * [0.93885] \jsc@pfx@ tmin10}{ }
1133   \DeclareFontShape{\jsc@JTn}{gt}{m}{n}{<-> s * [0.93885] \jsc@pfx@ tgoth10}{ }
1134 \else
1135   \ifjisfont
1136     \DeclareFontShape{\jsc@JYn}{mc}{m}{n}{<-> s * [0.93885] \jsc@pfx@ jis}{ }
1137     \DeclareFontShape{\jsc@JYn}{gt}{m}{n}{<-> s * [0.93885] \jsc@pfx@ jsig}{ }
1138     \DeclareFontShape{\jsc@JTn}{mc}{m}{n}{<-> s * [0.93885] \jsc@pfx@ tmin10}{ }
1139     \DeclareFontShape{\jsc@JTn}{gt}{m}{n}{<-> s * [0.93885] \jsc@pfx@ tgoth10}{ }
1140   \else
1141     \if@jsc@uplatex
1142       \DeclareFontShape{JY2}{mc}{m}{n}{<-> s * [0.903375] upjisr-h}{ }
1143       \DeclareFontShape{JY2}{gt}{m}{n}{<-> s * [0.903375] upjisg-h}{ }
1144       \DeclareFontShape{JT2}{mc}{m}{n}{<-> s * [0.903375] upjisr-v}{ }
1145       \DeclareFontShape{JT2}{gt}{m}{n}{<-> s * [0.903375] upjisg-v}{ }
1146     \else
1147       \DeclareFontShape{\jsc@JYn}{mc}{m}{n}{<-> s * [0.93885] \jsc@pfx@ jis}{ }
1148       \DeclareFontShape{\jsc@JYn}{gt}{m}{n}{<-> s * [0.93885] \jsc@pfx@ jsig}{ }
1149       \DeclareFontShape{\jsc@JTn}{mc}{m}{n}{<-> s * [0.93885] \jsc@pfx@ tmin10}{ }
1150       \DeclareFontShape{\jsc@JTn}{gt}{m}{n}{<-> s * [0.93885] \jsc@pfx@ tgoth10}{ }
1151     \fi
1152   \fi
1153 \fi
1154 %</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、 $\text{T}_{\text{E}}\text{X}$  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。

```

1155 % \DeclareFontShape{\jsc@JYn}{mc}{bx}{n}{<->ssub*gt/m/n}{ } % in \jsc@JYnmc
1156 % \DeclareFontShape{\jsc@JYn}{gt}{bx}{n}{<->ssub*gt/m/n}{ } % in \jsc@JYngt
1157 \DeclareFontShape{\jsc@JYn}{mc}{m}{it}{<->ssub*mc/m/n}{ }
1158 \DeclareFontShape{\jsc@JYn}{mc}{m}{sl}{<->ssub*mc/m/n}{ }
1159 \DeclareFontShape{\jsc@JYn}{mc}{m}{sc}{<->ssub*mc/m/n}{ }
1160 \DeclareFontShape{\jsc@JYn}{gt}{m}{it}{<->ssub*gt/m/n}{ }
1161 \DeclareFontShape{\jsc@JYn}{gt}{m}{sl}{<->ssub*gt/m/n}{ }
1162 \DeclareFontShape{\jsc@JYn}{mc}{bx}{it}{<->ssub*gt/m/n}{ }
1163 \DeclareFontShape{\jsc@JYn}{mc}{bx}{sl}{<->ssub*gt/m/n}{ }
1164 % \DeclareFontShape{\jsc@JTn}{mc}{bx}{n}{<->ssub*gt/m/n}{ } % in \jsc@JTnmc
1165 % \DeclareFontShape{\jsc@JTn}{gt}{bx}{n}{<->ssub*gt/m/n}{ } % in \jsc@JTngt
1166 \DeclareFontShape{\jsc@JTn}{mc}{m}{it}{<->ssub*mc/m/n}{ }
1167 \DeclareFontShape{\jsc@JTn}{mc}{m}{sl}{<->ssub*mc/m/n}{ }
1168 \DeclareFontShape{\jsc@JTn}{mc}{m}{sc}{<->ssub*mc/m/n}{ }
1169 \DeclareFontShape{\jsc@JTn}{gt}{m}{it}{<->ssub*gt/m/n}{ }
1170 \DeclareFontShape{\jsc@JTn}{gt}{m}{sl}{<->ssub*gt/m/n}{ }
1171 \DeclareFontShape{\jsc@JTn}{mc}{bx}{it}{<->ssub*gt/m/n}{ }
1172 \DeclareFontShape{\jsc@JTn}{mc}{bx}{sl}{<->ssub*gt/m/n}{ }

```

[2020-02-02] L<sup>A</sup>T<sub>E</sub>X 2<sub>ε</sub> 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 L<sup>A</sup>T<sub>E</sub>X 2<sub>ε</sub> 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] L<sup>A</sup>T<sub>E</sub>X 2<sub>ε</sub> 2020-10-01 では \AddToHook を利用します。

```

1173 %</class>
1174 %<*class|minijs>
1175 %% ad-hoc "relation font"
1176 \@ifl@t@r\fmtversion{2020/10/01}
1177   {\jsc@needspace@tchfalse}{\jsc@needspace@tchtrue}
1178 \ifjsc@needspace@tch      % --- for 2020-02-02 or older BEGIN
1179 \ifx\@rmfamilyhook\@undefined % old
1180 \DeclareRobustCommand\rmfamily
1181   {\not@math@alphabet\rmfamily\mathrm
1182     \romanfamily\rmdefault\kanjifamily\mcdefault\selectfont}
1183 \DeclareRobustCommand\sffamily
1184   {\not@math@alphabet\sffamily\mathsf
1185     \romanfamily\sfdefault\kanjifamily\gtdefault\selectfont}
1186 \DeclareRobustCommand\ttfamily
1187   {\not@math@alphabet\ttfamily\mathtt
1188     \romanfamily\ttdefault\kanjifamily\gtdefault\selectfont}
1189 \AtBeginDocument{%
1190   \ifx\mweights@init\@undefined\else % mweights.sty is loaded
1191     % my definitions above should have been overwritten, recover it!
1192     % \selectfont is executed twice but I don't care about speed...
1193     \expandafter\g@addto@macro\csname rmfamily \endcsname
1194       {\kanjifamily\mcdefault\selectfont}%

```

```

1195 \expandafter\g@addto@macro\csname sffamily \endcsname
1196 {\kanjifamily\gtdefault\selectfont}%
1197 \expandafter\g@addto@macro\csname ttfamily \endcsname
1198 {\kanjifamily\gtdefault\selectfont}%
1199 \fi}
1200 \else % 2020-02-02
1201 \g@addto@macro\@rmfamilyhook
1202 {\prepare@family@series@update@kanji{mc}\mcdefault}
1203 \g@addto@macro\@sffamilyhook
1204 {\prepare@family@series@update@kanji{gt}\gtdefault}
1205 \g@addto@macro\@ttfamilyhook
1206 {\prepare@family@series@update@kanji{gt}\gtdefault}
1207 \fi
1208 \else % --- for 2020-02-02 or older END & for 2020-10-01 BEGIN
1209 \AddToHook{rmfamily}%
1210 {\prepare@family@series@update@kanji{mc}\mcdefault}
1211 \AddToHook{sffamily}%
1212 {\prepare@family@series@update@kanji{gt}\gtdefault}
1213 \AddToHook{ttfamily}%
1214 {\prepare@family@series@update@kanji{gt}\gtdefault}
1215 \fi % --- for 2020-10-01 END
1216 %</class|minijs>
1217 %<*class>

```

`\textmc` 次のコマンドはイタリック補正なども含めて定義されていますが、和文ではイタリック補正  
`\textgt` 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。

```

1218 \ifx\DeclareFixJFMCJKTextFontCommand\@undefined
1219 \DeclareRobustCommand\textmc[1]{%
1220 \relax\ifmmode \expandafter\nfss@text \fi{\mcfamily #1}}
1221 \DeclareRobustCommand\textgt[1]{%
1222 \relax\ifmmode \expandafter\nfss@text \fi{\gtfamily #1}}
1223 \fi

```

新クラスでも `disablejfm` 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。

```
1224 \def\reDeclareMathAlphabet#1#2#3{%
1225   \edef\@tempa{\expandafter\@gobble\string#2}%
1226   \edef\@tempb{\expandafter\@gobble\string#3}%
1227   \edef\@tempc{\string \@expandafter\@gobbletwo\string#2}%
1228   \ifx\@tempc\@tempa%
1229     \edef\@tempa{\expandafter\@gobbletwo\string#2}%
1230     \edef\@tempb{\expandafter\@gobbletwo\string#3}%
1231   \fi
1232   \begingroup
1233     \let\protect\noexpand
1234     \def\@tempaa{\relax}%
1235     \expandafter\ifx\csname RDMAorg@\@tempa\endcsname\relax
1236       \edef\@tempaa{\expandafter\def\expandafter\noexpand%
1237         \csname RDMAorg@\@tempa\endcsname{%
1238           \expandafter\noexpand\csname\@tempa\endcsname}}%
1239     \fi
1240     \def\@tempbb{\relax}%
1241     \expandafter\ifx\csname RDMAorg@\@tempb\endcsname\relax
1242       \edef\@tempbb{\expandafter\def\expandafter\noexpand%
1243         \csname RDMAorg@\@tempb\endcsname{%
1244           \expandafter\noexpand\csname\@tempb\endcsname}}%
1245     \fi
1246     \edef\@tempc{\@tempaa\@tempbb}%
1247     \expandafter\endgroup\@tempc%
1248     \edef#1{\noexpand\protect\expandafter\noexpand\csname%
1249       \expandafter\@gobble\string#1\space\space\endcsname}%
1250     \expandafter\edef\csname\expandafter\@gobble\string#1\space\space\endcsname%
1251       {\noexpand\DualLang@mathalph@bet%
1252         {\expandafter\noexpand\csname RDMAorg@\@tempa\endcsname}%
1253         {\expandafter\noexpand\csname RDMAorg@\@tempb\endcsname}%
1254       }%
1255   }
1256 \onlypreamble\reDeclareMathAlphabet
1257 \def\DualLang@mathalph@bet#1#2{%
1258   \relax\ifmmode
1259     \ifx\math@bgroup\bgroup%      2e normal style      (\mathrm{...})
1260       \bgroup\let\DualLang@Mfontsw\DLMfontsw@standard
1261     \else
1262       \ifx\math@bgroup\relax%      2e two letter style (\rm->\mathrm)
1263         \let\DualLang@Mfontsw\DLMfontsw@oldstyle
1264       \else
1265         \ifx\math@bgroup@empty% 2.09 oldfont style ({\mathrm ...})
```

```

1266      \let\DualLang@Mfontsw\DLMfontsw@oldlfont
1267      \else%                               panic! assume 2e normal style
1268      \bgroup\let\DualLang@Mfontsw\DLMfontsw@standard
1269      \fi
1270    \fi
1271  \fi
1272 \else
1273   \let\DualLang@Mfontsw\@firstoftwo
1274   \fi
1275   \DualLang@Mfontsw{#1}{#2}%
1276 }
1277 \def\DLMfontsw@standard#1#2#3{#1{#2{#3}}\egroup}
1278 \def\DLMfontsw@oldstyle#1#2{#1\relax\@fontswitch\relax{#2}}
1279 \def\DLMfontsw@oldlfont#1#2{#1\relax#2\relax}
1280 \if@enablejfam
1281   \DeclareSymbolFont{mincho}{\jsc@JYn}{mc}{m}{n}
1282   \DeclareSymbolFontAlphabet{\mathmc}{\mincho}
1283   \SetSymbolFont{mincho}{bold}{\jsc@JYn}{gt}{m}{n}
1284   \jfam\symmincho
1285   \DeclareMathAlphabet{\mathgt}{\jsc@JYn}{gt}{m}{n}
1286   \AtBeginDocument{%
1287     \reDeclareMathAlphabet{\mathrm}{\@mathrm}{\@mathmc}
1288     \reDeclareMathAlphabet{\mathbf}{\@mathbf}{\@mathgt}}
1289 \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。

```
1290 % \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。

```

1291 \prebreakpenalty\jis"2147=10000      % 5000   '
1292 \postbreakpenalty\jis"2148=10000     % 5000   "
1293 \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。

```
1294 \inhibitxspcode`!=1
```

```
1295 \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

```
1296 % \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。

1297 \xspcode`+=3

1298 \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。

1299 \xspcode`^^80=3

1300 \xspcode`^^81=3

1301 \xspcode`^^82=3

1302 \xspcode`^^83=3

1303 \xspcode`^^84=3

1304 \xspcode`^^85=3

1305 \xspcode`^^86=3

1306 \xspcode`^^87=3

1307 \xspcode`^^88=3

1308 \xspcode`^^89=3

1309 \xspcode`^^8a=3

1310 \xspcode`^^8b=3

1311 \xspcode`^^8c=3

1312 \xspcode`^^8d=3

1313 \xspcode`^^8e=3

1314 \xspcode`^^8f=3

1315 \xspcode`^^90=3

1316 \xspcode`^^91=3

1317 \xspcode`^^92=3

1318 \xspcode`^^93=3

1319 \xspcode`^^94=3

1320 \xspcode`^^95=3

1321 \xspcode`^^96=3

1322 \xspcode`^^97=3

1323 \xspcode`^^98=3

1324 \xspcode`^^99=3

1325 \xspcode`^^9a=3

1326 \xspcode`^^9b=3

1327 \xspcode`^^9c=3

1328 \xspcode`^^9d=3

1329 \xspcode`^^9e=3

1330 \xspcode`^^9f=3

1331 \xspcode`^^a0=3

1332 \xspcode`^^a1=3

1333 \xspcode`^^a2=3

1334 \xspcode`^^a3=3

1335 \xspcode`^^a4=3

1336 \xspcode`^^a5=3

1337 \xspcode`^^a6=3  
1338 \xspcode`^^a7=3  
1339 \xspcode`^^a8=3  
1340 \xspcode`^^a9=3  
1341 \xspcode`^^aa=3  
1342 \xspcode`^^ab=3  
1343 \xspcode`^^ac=3  
1344 \xspcode`^^ad=3  
1345 \xspcode`^^ae=3  
1346 \xspcode`^^af=3  
1347 \xspcode`^^b0=3  
1348 \xspcode`^^b1=3  
1349 \xspcode`^^b2=3  
1350 \xspcode`^^b3=3  
1351 \xspcode`^^b4=3  
1352 \xspcode`^^b5=3  
1353 \xspcode`^^b6=3  
1354 \xspcode`^^b7=3  
1355 \xspcode`^^b8=3  
1356 \xspcode`^^b9=3  
1357 \xspcode`^^ba=3  
1358 \xspcode`^^bb=3  
1359 \xspcode`^^bc=3  
1360 \xspcode`^^bd=3  
1361 \xspcode`^^be=3  
1362 \xspcode`^^bf=3  
1363 \xspcode`^^c0=3  
1364 \xspcode`^^c1=3  
1365 \xspcode`^^c2=3  
1366 \xspcode`^^c3=3  
1367 \xspcode`^^c4=3  
1368 \xspcode`^^c5=3  
1369 \xspcode`^^c6=3  
1370 \xspcode`^^c7=3  
1371 \xspcode`^^c8=3  
1372 \xspcode`^^c9=3  
1373 \xspcode`^^ca=3  
1374 \xspcode`^^cb=3  
1375 \xspcode`^^cc=3  
1376 \xspcode`^^cd=3  
1377 \xspcode`^^ce=3  
1378 \xspcode`^^cf=3  
1379 \xspcode`^^d0=3  
1380 \xspcode`^^d1=3  
1381 \xspcode`^^d2=3  
1382 \xspcode`^^d3=3  
1383 \xspcode`^^d4=3  
1384 \xspcode`^^d5=3  
1385 \xspcode`^^d6=3

```

1386 \xspcode`^^d7=3
1387 \xspcode`^^d8=3
1388 \xspcode`^^d9=3
1389 \xspcode`^^da=3
1390 \xspcode`^^db=3
1391 \xspcode`^^dc=3
1392 \xspcode`^^dd=3
1393 \xspcode`^^de=3
1394 \xspcode`^^df=3
1395 \xspcode`^^e0=3
1396 \xspcode`^^e1=3
1397 \xspcode`^^e2=3
1398 \xspcode`^^e3=3
1399 \xspcode`^^e4=3
1400 \xspcode`^^e5=3
1401 \xspcode`^^e6=3
1402 \xspcode`^^e7=3
1403 \xspcode`^^e8=3
1404 \xspcode`^^e9=3
1405 \xspcode`^^ea=3
1406 \xspcode`^^eb=3
1407 \xspcode`^^ec=3
1408 \xspcode`^^ed=3
1409 \xspcode`^^ee=3
1410 \xspcode`^^ef=3
1411 \xspcode`^^f0=3
1412 \xspcode`^^f1=3
1413 \xspcode`^^f2=3
1414 \xspcode`^^f3=3
1415 \xspcode`^^f4=3
1416 \xspcode`^^f5=3
1417 \xspcode`^^f6=3
1418 \xspcode`^^f7=3
1419 \xspcode`^^f8=3
1420 \xspcode`^^f9=3
1421 \xspcode`^^fa=3
1422 \xspcode`^^fb=3
1423 \xspcode`^^fc=3
1424 \xspcode`^^fd=3
1425 \xspcode`^^fe=3
1426 \xspcode`^^ff=3
1427 %</class>
1428 %</jsclasses>
1429 %<*class>

```

\@ 欧文といえば、 $\text{\LaTeX}$  の  $\text{\def\@{\spacefactor\@m}}$  という定義（ $\text{\@m}$  は 1000）では  $\text{I watch TV\@.}$  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、 $\text{I watch TV.\@}$  と書くことにします。

[2016-07-14] 2015-01-01 の L<sup>A</sup>T<sub>E</sub>X 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。

---

```
1430 \chardef\bxjs@periodchar=`\.  
1431 \bxjs@robust@def\bxjs@SE{%  
1432   \ifnum\spacefactor<\@m \spacefactor\@m  
1433   \else \spacefactor\sfcode\bxjs@periodchar  
1434   \fi}  
1435 \ifbxjs@fix@at@cmd  
1436   \def\@{\bxjs@SE{}}  
1437 \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 L<sup>A</sup>T<sub>E</sub>X 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、L<sup>A</sup>T<sub>E</sub>X 本体で定義されています。

<code>\@vpt</code>	5	<code>\@vipt</code>	6	<code>\@viipt</code>	7
<code>\@viipt</code>	8	<code>\@ixpt</code>	9	<code>\@xpt</code>	10
<code>\@xipt</code>	10.95	<code>\@xiipt</code>	12	<code>\@xivpt</code>	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 pL<sup>A</sup>T<sub>E</sub>X 2<sub>ε</sub> 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、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 L<sup>A</sup>T<sub>E</sub>X カーネルで `selectfont` フックを利用する。

```
1438 %\let\bxjs@prev@set@fontsize\@undefined
1439 \@onlypreamble\bxjs@patch@set@fontsize
1440 \def\bxjs@patch@set@fontsize{%
1441   \ifx\bxjs@prev@set@fontsize\set@fontsize\else
1442     \def\bxjs@tmpa{\def\set@fontsize####1####2####3}%
1443   \expandafter\bxjs@tmpa\expandafter{%
1444     \set@fontsize{##1}{##2}{##3}%
1445   % 末尾にコードを追加
1446   \expandafter\def\expandafter\size@update\expandafter{%
1447     \size@update
1448     \jsFontSizeChanged}%
1449 }
1450 \let\bxjs@prev@set@fontsize\set@fontsize
1451 \fi}
```

この場とパッケージ末尾で作動させる。

```
1452 \bxjs@patch@set@fontsize
1453 \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。

```
1454 \newcommand*\jsFontSizeChanged{%
1455   \jsZw=\f@size\p@
1456   \jsZw=\jsScale \jsZw
1457   \ifdim\parindent>\z@
1458     \if@english \parindent=1em
1459     \else       \parindent=1\jsZw
1460   \fi
1461   \fi\relax
1462   \jsResetDimen}
```

`\jsResetDimen` ユーザ定義用のフック。

```
1463 \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` の代わりに用いることにします。

```
1464 \ifjsc@mag
1465   \let\jsc@setfontsize\@setfontsize
1466 \else
1467   \def\jsc@setfontsize#1#2#3{%
1468     \@setfontsize#1{#2\jsc@mp\}{#3\jsc@mp\}}
1469 % microtype 対策
1470 \ifjswitheTeX\if j\jsEngine\else
1471   \def\jsc@setfontsize#1#2#3{%
1472     \edef\bxjs@sfs@next{%
1473       \unexpanded{\@setfontsize#1}%
1474       {\the\dimexpr#2\jsc@mp\relax}{\the\dimexpr#3\jsc@mp\relax}%
1475     }\bxjs@sfs@next}
1476 \fi\fi
1477 \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。

---

```
1478 \emergencystretch 3\jsZw
```

`\ifnarrowbaselines` 欧文用に行間を狭くする論理変数と、それを真・偽にするためのコマンドです。

`\narrowbaselines` [2003-06-30] 数式に入るところで `\narrowbaselines` を実行しているので

`\widebaselines` `\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、しばしば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。

---

```
1479 \newif\ifnarrowbaselines
```

```
1480 \if@english
```



```

1481 \narrowbaselinestrue
1482 \fi
1483 \def\narrowbaselines{%
1484 \narrowbaselinestrue
1485 \skip0=\abovedisplayskip
1486 \skip2=\abovedisplayshortskip
1487 \skip4=\belowdisplayskip
1488 \skip6=\belowdisplayshortskip
1489 % 一時的に警告を無効化する
1490 \let\bxjs@save@nomath\@nomath
1491 \let\@nomath\@gobble
1492 \@currsize\selectfont
1493 \let\@nomath\bxjs@save@nomath
1494 \abovedisplayskip=\skip0
1495 \abovedisplayshortskip=\skip2
1496 \belowdisplayskip=\skip4
1497 \belowdisplayshortskip=\skip6\relax}
1498 \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 L<sup>A</sup>T<sub>E</sub>X 式条件文にしたもの。

```

1499 \def\bxjs@if@narrowbaselines{%
1500 \ifnarrowbaselines\expandafter\@firstoftwo
1501 \else \expandafter\@secondoftwo
1502 \fi
1503 }

```

---

`\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。

---

```

1504 \renewcommand{\normalsize}{%
1505 \bxjs@if@narrowbaselines{%
1506 \jsc@setfontsize\normalsize\@xpt\@xiipt
1507 }{%else

```

```

1508 \jsc@setfontsize\normalsize\@xpt{\n@baseline}%
1509 }%

```

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] T<sub>E</sub>X 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。

```

1510 \abovedisplayskip 11\jsc@mpt \@plus3\jsc@mpt \@minus4\jsc@mpt
1511 \abovedisplayshortskip \z@ \@plus3\jsc@mpt
1512 \belowdisplayskip 9\jsc@mpt \@plus3\jsc@mpt \@minus4\jsc@mpt
1513 \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 にコピーしてきます。 \@listI の設定は後で出てきます。

```

1514 \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。

```

1515 %</class>
1516 %<*class|minijs>
1517 %% initialize
1518 \normalsize
1519 %</class|minijs>
1520 %<*class>

```

\Cht 基準となる長さの設定をします。pL<sub>A</sub>T<sub>E</sub>X 2<sub>ε</sub> カーネル (plfonts.dtx) で宣言されているパ  
\Cdp ラメータに実際の値を設定します。たとえば \Cwd は \normalfont の全角幅 (1zw) です。  
\Cwd [2017-08-31] 基準とする文字を「全角空白」(EUC コード 0xA1A1) から「漢」(JIS コー  
\Cvs ド 0x3441) へ変更しました。

---

\Chs

\Cwd 等の変数は pT<sub>E</sub>X 系以外では未定義なのでここで定義する。

```

1521 \ifx\Cht\undefined \newdimen\Cht \fi
1522 \ifx\Cdp\undefined \newdimen\Cdp \fi
1523 \ifx\Cwd\undefined \newdimen\Cwd \fi
1524 \ifx\Cvs\undefined \newdimen\Cvs \fi
1525 \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。

---

```

1526 \setlength\Cht{0.88\jsZw}
1527 \setlength\Cdp{0.12\jsZw}
1528 \setlength\Cwd{1\jsZw}
1529 \setlength\Cvs{\baselineskip}
1530 \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  $4 \pm 2$ ， $2 \pm 1$  ポイントでしたが，ここではゼロ (`\z@`) にしました。

```

1531 \newcommand{\small}{%
1532   \bxjs@if@narrowbaselines{%
1533     %<!kiyou>   \jsc@setfontsize\small\@ixpt{11}%
1534     %<kiyou>    \jsc@setfontsize\small{8.8888}{11}%
1535   }{%else
1536     %<!kiyou>   \jsc@setfontsize\small\@ixpt{13}%
1537     %<kiyou>    \jsc@setfontsize\small{8.8888}{13.2418}%
1538   }%
1539   \abovedisplayskip 9\jsc@empt \@plus3\jsc@empt \@minus4\jsc@empt
1540   \abovedisplayshortskip \z@ \@plus3\jsc@empt
1541   \belowdisplayskip \abovedisplayskip
1542   \belowdisplayshortskip \belowdisplayskip
1543   \def\@listif{\leftmargin\leftmargini
1544             \topsep \z@
1545             \parsep \z@
1546             \itemsep \parsep}}

```

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

```

1547 \newcommand{\footnotesize}{%
1548   \bxjs@if@narrowbaselines{%
1549     %<!kiyou>   \jsc@setfontsize\footnotesize\@viipt{9.5}%
1550     %<kiyou>    \jsc@setfontsize\footnotesize{8.8888}{11}%
1551   }{%else
1552     %<!kiyou>   \jsc@setfontsize\footnotesize\@viipt{11}%
1553     %<kiyou>    \jsc@setfontsize\footnotesize{8.8888}{13.2418}%
1554   }%
1555   \abovedisplayskip 6\jsc@empt \@plus2\jsc@empt \@minus3\jsc@empt
1556   \abovedisplayshortskip \z@ \@plus2\jsc@empt
1557   \belowdisplayskip \abovedisplayskip
1558   \belowdisplayshortskip \belowdisplayskip
1559   \def\@listif{\leftmargin\leftmargini
1560             \topsep \z@
1561             \parsep \z@
1562             \itemsep \parsep}}

```

`\scriptsize` 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，`\large` 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  
`\Large` 行が揃うようにします。

`\LARGE` [2004-11-03] `\HUGE` を追加。

```

\huge 1563 \newcommand{\scriptsize}{\jsc@setfontsize\scriptsize\@viipt\@viipt}
1564 \newcommand{\tiny}{\jsc@setfontsize\tiny\@vpt\@vpt}
\Huge 1565 \if@twocolumn
\HUGE 1566 %<!kiyou> \newcommand{\large}{\jsc@setfontsize\large\@xipt{\n@baseline}}

```

```

1567 %<kiyou> \newcommand{\large}{\jsc@setfontsize\large{11.111}{\n@baseline}}
1568 \else
1569 %<!kiyou> \newcommand{\large}{\jsc@setfontsize\large\xiip{17}}
1570 %<kiyou> \newcommand{\large}{\jsc@setfontsize\large{11.111}{17}}
1571 \fi
1572 %<!kiyou>\newcommand{\Large}{\jsc@setfontsize\Large\xivpt{21}}
1573 %<kiyou>\newcommand{\Large}{\jsc@setfontsize\Large{12.222}{21}}
1574 \newcommand{\LARGE}{\jsc@setfontsize\LARGE\xviip{25}}
1575 \newcommand{\huge}{\jsc@setfontsize\huge\xxpt{28}}
1576 \newcommand{\Huge}{\jsc@setfontsize\Huge\xxvpt{33}}
1577 \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り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。

```

1578 \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教えてください。

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。『*LaTeX 2<sub>ε</sub> 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。

```

1579 % \newcommand{\headfont}{\bfseries}
1580 \newcommand{\headfont}{\sffamily}
1581 % \newcommand{\headfont}{\sffamily\fontseries{sbc}\selectfont}

```

## 5 レイアウト

### ■二段組

`\columnsep` `\columnsep` は二段組のときの左右の段間の幅です。元は 10pt ですが、2zw にしました。  
`\columnseprule` このスペースの中央に `\columnseprule` の幅の罫線が引かれます。

```

1582 %<!kiyou>\setlength\columnsep{2\Cwd}
1583 %<kiyou>\setlength\columnsep{28truebp}
1584 \setlength\columnseprule{\z@}

```

### ■段落

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

```

\lineskiplimit

```

```

\normallineskiplimit

```

```

1585 \setlength\lineskip{1\jsc@mp}
1586 \setlength\normallineskip{1\jsc@mp}
1587 \setlength\lineskiplimit{1\jsc@mp}
1588 \setlength\normallineskiplimit{1\jsc@mp}

```

`\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。

```
1589 \renewcommand{\baselinestretch}{}
```

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

```

1590 \setlength\parskip{\z@}
1591 \if@slide
1592   \setlength\parindent{0\p@}
1593 \else
1594   \setlength\parindent{1\Cwd}
1595 \fi

```

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

```

\@highpenalty 1596 \@lowpenalty   51
1597 \@medpenalty 151
1598 \@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。

```
1599 % \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。

```
1600 % \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。

```

1601 \@onlypreamble\bxjs@bd@pre@geometry@hook
1602 \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。

```
1603 \ifjsc@mag
1604 \mag=\bxjs@param@mag
1605 \fi
1606 \setlength{\topskip}{10\jsc@empt}
```

`\jsSetQHLlength` のための和文単位の定義。

```
1607 \def\bxjs@unit@trueQ{0.25trueem}\let\bxjs@unit@trueH\bxjs@unit@trueQ
1608 \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。

```
1609 \@tempswafalse
1610 \def\bxjs@tmpdo{\@ifnextchar\bgroup\bxjs@tmpdo@a\remove@to@nnil}
1611 \def\bxjs@tmpdo@a#1{\edef\bxjs@tmpa{#1}%
1612   \@ifnextchar\bgroup\bxjs@tmpdo@b\remove@to@nnil}
1613 \def\bxjs@tmpdo@b#1{\edef\bxjs@tmpa{\bxjs@tmpa,#1}%
1614   \@ifnextchar\@nnil\bxjs@tmpdo@c\remove@to@nnil}
1615 \def\bxjs@tmpdo@c\@nnil{\@tempswatrue
1616   \edef\bxjs@param@paper{papersize={\bxjs@tmpa}}}
1617 \expandafter\bxjs@tmpdo\bxjs@param@paper\@nnil
```

`W,H` の形式について。

```
1618 \if@tempswa\else
1619   \def\bxjs@tmpa{\@nil,\@nil}
1620   \def\bxjs@tmpdo#1,#2,#3\@nnil{%
1621     \def\bxjs@tmpb{#3}\ifx\bxjs@tmpa\bxjs@tmpb
1622       \@tempswatrue\edef\bxjs@param@paper{papersize={#1,#2}}\fi
1623     \expandafter\bxjs@tmpdo\bxjs@param@paper,\@nil,\@nil\@nnil
1624   \fi
```

`W*H` の形式について。

```
1625 \if@tempswa\else
1626   \def\bxjs@tmpa{\@nil*\@nil}
1627   \def\bxjs@tmpdo#1*#2*#3\@nnil{%
1628     \def\bxjs@tmpb{#3}\ifx\bxjs@tmpa\bxjs@tmpb
1629       \@tempswatrue\edef\bxjs@param@paper{papersize={#1,#2}}\fi
1630     \expandafter\bxjs@tmpdo\bxjs@param@paper*\@nil*\@nil\@nnil
1631   \fi
```

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

```
1632 \edef\bxjs@layout@paper{%
1633   \ifjsc@mag truedimen,\fi
1634   \if@landscape landscape,\fi
1635   \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。

```
1636 %<*article|report>
1637 \def\bxjs@layout@base{%
```

```

1638 headheight=\topskip,footskip=0.03367\paperheight,%
1639 headsep=\footskip-\topskip,includeheadfoot,%
1640 }
1641 \edef\bxjs@layout{\bxjs@layout@base
1642 hscale=0.76,hmarginratio=1:1,%
1643 vscale=0.83,vmarginratio=1:1,%
1644 }
1645 %</article|report>
1646 %<*book>
1647 \def\bxjs@layout@base{%
1648 headheight=\topskip,headsep=6\jsc@mmm,nofoot,includeheadfoot,%
1649 }
1650 \ifbxjs@layout@buggyhmargin %---
1651 % アレ
1652 \edef\bxjs@layout{\bxjs@layout@base
1653 hmargin=36\jsc@mmm,hmarginratio=1:1,%
1654 vscale=0.83,vmarginratio=1:1,%
1655 }
1656 \else %---
1657 % 非アレ
1658 \edef\bxjs@layout{\bxjs@layout@base
1659 hmargin=18\jsc@mmm,%
1660 vscale=0.83,vmarginratio=1:1,%
1661 }
1662 \fi %---
1663 %</book>
1664 %<*slide>
1665 \def\bxjs@layout@base{%
1666 noheadfoot,%
1667 }
1668 \edef\bxjs@layout{\bxjs@layout@base
1669 hscale=0.9,hmarginratio=1:1,%
1670 vscale=0.944,vmarginratio=1:1,%
1671 }
1672 %</slide>

```

textwidth オプションの設定を反映する。

```

1673 %<!*book>
1674 \ifx\bxjs@textwidth@opt\undefined\else
1675 \jsSetQHLength\@tempdima{\bxjs@textwidth@opt}
1676 \edef\bxjs@layout{\bxjs@layout width=\the\@tempdima,}
1677 \fi
1678 %</!book>
1679 \ifx\bxjs@numberOf@lines@opt\undefined\else
1680 \bxjs@gset@tempcnta{\bxjs@numberOf@lines@opt}
1681 \edef\bxjs@layout{\bxjs@layout lines=\the\@tempcnta,}
1682 \fi

```

\fullwidth [寸法レジスタ] ヘッダ・フッタ領域の横幅。

```
1683 \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。

```
1684 %<*book>
1685 \newcommand\jsTextWidthLimit{40}
1686 \@tempdima=\jsTextWidthLimit\Cwd
1687 \ifx\bxjs@textwidth@limit@opt\undefined\else
1688   \bxjs@gset@tempcnta{\bxjs@textwidth@limit@opt}
1689   \@tempdima=\@tempcnta\Cwd
1690 \fi
1691 \ifx\bxjs@textwidth@opt\undefined\else
1692   \jsSetQHLength\@tempdima{\bxjs@textwidth@opt}
1693 \fi
1694 \edef\bxjs@textwidth@limit{\the\@tempdima}
1695 \ifdim\@tempdima=\jsTextWidthLimit\Cwd\else
1696   \bxjs@invscale\@tempdima{\strip@pt\Cwd}
1697   \long\edef\jsTextWidthLimit{\strip@pt\@tempdima}
1698 \fi
1699 %</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。

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

`\bxjs@postproc@layout geometry` の後処理。

```
1702 \def\bxjs@postproc@layout{%
```

`geometry` のドライバを再設定する。

```
1703 \ifx\bxjs@geometry@driver\relax\else
1704   \let\Gm@driver\bxjs@geometry@driver
1705 \fi
```

`\ht\strutbox` の値を元に戻す。

```
1706 \ht\strutbox=\bxjs@save@ht@strutbox\relax
```

`\textwidth` の値を補正する。

```
1707 \ifbxjs@whole@zw@lines
1708   \@tempdimb=\textwidth
1709   \if@twocolumn \@tempdima=2\Cwd \else \@tempdima=1\Cwd \fi
1710   \advance\textwidth.005pt\relax
1711   \divide\textwidth\@tempdima \multiply\textwidth\@tempdima
1712   \advance\@tempdimb-\textwidth
1713   \advance\oddsidemargin 0.5\@tempdimb
1714   \advance\evensidemargin 0.5\@tempdimb
```



```

1715 \fi
1716 \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。

```

1717 %<*book>
1718 \@tempdima=\bxjs@textwidth@limit\relax
1719 \ifbxjs@whole@zw@lines
1720   \advance\@tempdima.005pt\relax
1721   \divide\@tempdima\Cwd \multiply\@tempdima\Cwd
1722 \fi
1723 \ifdim\textwidth>\@tempdima
1724   \textwidth=\@tempdima
1725   \addtolength\evensidemargin{\fullwidth-\textwidth}
1726 \fi
1727 %</book>

```

\textheight 関連の調整。

```

1728 \@tempdimb=\textheight
1729 \advance\textheight-\topskip
1730 \advance\textheight.005pt\relax
1731 \divide\textheight\baselineskip \multiply\textheight\baselineskip
1732 \advance\textheight\topskip
1733 \advance\@tempdimb-\textheight
1734 \advance\topmargin0.5\@tempdimb

```

\headheight 関連の調整。

```

1735 \@tempdima=\topskip
1736 \advance\headheight\@tempdima
1737 \advance\topmargin-\@tempdima

```

marginpar 関連の調整。

```

1738 \setlength\marginparsep{\columnsep}
1739 \setlength\marginparpush{\baselineskip}
1740 \setlength\marginparwidth{\paperwidth-\oddsidemargin-1truein%
1741   -\textwidth-10\jsc@mmm-\marginparsep}
1742 \ifbxjs@whole@zw@lines
1743   \divide\marginparwidth\Cwd \multiply\marginparwidth\Cwd
1744 \fi

```

連動する変数。

```

1745 \maxdepth=.5\topskip
1746 \stockwidth=\paperwidth
1747 \stockheight=\paperheight
1748 }

```

\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。

```

1749 \edef\jsGeometryOptions{%
1750   \bxjs@layout@paper,\bxjs@layout}

```

## ■geometry パッケージを読み込む 🐙

`\bxjs@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。

※ L<sup>A</sup>T<sub>E</sub>X のフックシステムがある場合はムニャムニャ。

```
1751 \def\bxjs@geometry@name{geometry}
1752 \ifbxjs@old@hook@system
1753   \let\bxjs@apply@bd@pre@geometry@hook\AtBeginDocument
1754 \else
1755   \def\bxjs@apply@bd@pre@geometry@hook{%
1756     \AddToHook{begindocument}[\bxjs@geometry@name]}
1757 \fi
```

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

```
1758 \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。

```
1759 \ifbxjs@papersize
1760   \ifx\bxjs@driver@given\bxjs@driver@@dvipdfmx
1761     \PassOptionsToPackage{dvipdfm}{geometry}
1762   \else\ifx\bxjs@driver@given\bxjs@driver@@dvimode
1763     \PassOptionsToPackage{dvipdfm}{geometry}
1764   \fi\fi
1765   \let\bxPapersizeSpecialDone=t
1766 \else
1767   \PassOptionsToPackage{driver=none}{geometry}
1768 \fi
```

ここで geometry を読み込む。

※ geometry の begin-document フックにおいて、Lua<sub>T</sub>E<sub>X</sub> の旧版互換を有効にする。

```
1769 \bxjs@apply@bd@pre@geometry@hook{%
1770   \bxjs@bd@pre@geometry@hook
1771   \@nameuse{ImposeOldLuaTeXBehavior}}
1772 \bxjs@preproc@layout
1773 \edef\bxjs@next{%
1774   \noexpand\RequirePackage[\bxjs@layout@paper,\bxjs@layout]{geometry}%
1775 }\bxjs@next
1776 \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。

```
1777 \let\bxjs@geometry@driver\Gm@driver
1778 \bxjs@postproc@layout
```

geometry のドライバ自動判別に対する前処理。

```
1779 \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。

```
1780 \ifpackagelater{geometry}{2010/02/12}{\else
1781 \PackageError\bxjs@clsname
1782 {Your 'geometry' package is too old (< v5.0)}%
1783 {\@ehc}%
1784 \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。

```
1785 \ifjsWithpTeXng
1786 \ifx\Gm@driver\@empty
1787 \def\Gm@driver{pdftex}%
1788 \fi
1789 \fi}
```

`\setpagelayout` ページレイアウト設定のためのユーザ命令。

```
1790 \def\setpagelayout{%
1791 \bxjs@ifplus{\bxjs@setpagelayout@a\tw@}{\else
1792 \@ifstar{\bxjs@setpagelayout@a\@ne}{\bxjs@setpagelayout@a\z@}}
1793 \def\bxjs@setpagelayout@a#1#2{%
1794 \ifcase#1% modify
1795 \def\bxjs@next{\ifjsc@mag truedimen,\fi #2}%
1796 \or% reset(*)
1797 \def\bxjs@next{reset,\bxjs@layout@paper,#2}%
1798 \or% semireset(+)
1799 \def\bxjs@next{reset,\bxjs@layout@paper,\bxjs@layout@base,#2}%
1800 \fi
1801 \bxjs@preproc@layout
1802 \edef\bxjs@next{%
1803 \noexpand\geometry{\bxjs@next}%
1804 }\bxjs@next
1805 \bxjs@postproc@layout}
```

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

```
1806 \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。

```
1807 \g@addto@macro\bxjs@begin@document@hook{%
1808 \ifdim\textwidth=.5\maxdimen
1809 \ClassError\bxjs@clsname
1810 {Page layout is not properly set}%
1811 {\@ehd}%
1812 \fi}
1813 \def\jsUseMinimalPageLayout{%
1814 \setlength{\textwidth}{6.5in}%
```

```

1815 \setlength{\textheight}{8in}}

\setpagelayout はとりあえず無効にしておく。

1816 \let\bxjs@geometry@driver\relax
1817 \def\setpagelayout{%
1818 \bxjs@ifplus{\bxjs@pagelayout@a}{%else
1819 \@ifstar{\bxjs@pagelayout@a}{\bxjs@pagelayout@a}}
1820 \def\bxjs@pagelayout@a#1{%
1821 \ClassError\bxjs@clsname
1822 {Command '\string\setpagelayout' is not supported,\MessageBreak
1823 because 'geometry' value is not 'class'}\@eha}
1824 %
1825 \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。

---

```

1826 %<*jsclasses>

```

`\headheight` `\topskip` は本文領域上端と本文 1 行目のベースラインとの距離です。あまりぎりぎりの値 `\topskip` 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。

```

1827 \setlength\topskip{1.38zw}%% from 10\jsc@mpt (2016-08-17)
1828 \if@slide
1829 \setlength\headheight{0\jsc@mpt}
1830 \else
1831 \setlength\headheight{20\jsc@mpt}%% from 2\topskip (2016-08-17); from \topskip (2003-
    06-26)
1832 \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。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。

```

1833 %<*article|kiyou>
1834 \if@slide
1835 \setlength\footskip{0pt}
1836 \else

```

```

1837 \setlength\footskip{0.03367\paperheight}
1838 \ifdim\footskip<\baselineskip
1839 \setlength\footskip{\baselineskip}
1840 \fi
1841 \fi
1842 %</article|kiyou>
1843 %<jspf>\setlength\footskip{9\jsc@mmm}
1844 %<*book>
1845 \if@report
1846 \setlength\footskip{0.03367\paperheight}
1847 \ifdim\footskip<\baselineskip
1848 \setlength\footskip{\baselineskip}
1849 \fi
1850 \else
1851 \setlength\footskip{0pt}
1852 \fi
1853 %</book>
1854 %<*report>
1855 \setlength\footskip{0.03367\paperheight}
1856 \ifdim\footskip<\baselineskip
1857 \setlength\footskip{\baselineskip}
1858 \fi
1859 %</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)。

```

1860 %<*article>
1861 \if@slide
1862 \setlength\headsep{0\jsc@mpt}
1863 \addtolength\headsep{-\topskip}%% added (2016-10-08)
1864 \addtolength\headsep{10\jsc@mpt}%% added (2016-10-08)
1865 \else
1866 \setlength\headsep{\footskip}
1867 \addtolength\headsep{-\topskip}
1868 \fi
1869 %</article>
1870 %<*book>
1871 \if@report
1872 \setlength\headsep{\footskip}
1873 \addtolength\headsep{-\topskip}
1874 \else
1875 \setlength\headsep{6\jsc@mmm}
1876 \addtolength\headsep{-\topskip}%% added (2016-10-08)
1877 \addtolength\headsep{10\jsc@mpt}%% added (2016-10-08)
1878 \fi

```

```

1879 %</book>
1880 %<*report>
1881 \setlength\headsep{\footskip}
1882 \addtolength\headsep{-\topskip}
1883 %</report>
1884 %<*jspf>
1885 \setlength\headsep{9\jsc@mmm}
1886 \addtolength\headsep{-\topskip}
1887 %</jspf>
1888 %<*kiyou>
1889 \setlength\headheight{0\jsc@empt}
1890 \setlength\headsep{0\jsc@empt}
1891 \addtolength\headsep{-\topskip}%% added (2016-10-08)
1892 \addtolength\headsep{10\jsc@empt}%% added (2016-10-08)
1893 %</kiyou>

```

`\maxdepth` `\maxdepth` は本文最下行の最大の深さで、plain  $\mathrm{T}_{\mathrm{E}}\mathrm{X}$  や  $\mathrm{L}^{\mathrm{A}}\mathrm{T}_{\mathrm{E}}\mathrm{X}$  2.09 では 4pt に固定でした。 $\mathrm{L}^{\mathrm{A}}\mathrm{T}_{\mathrm{E}}\mathrm{X}2\mathrm{e}$  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。

```

1894 \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。

```

1895 \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。

```

1896 %<*article>
1897 \if@slide
1898   \setlength\fullwidth{0.9\paperwidth}
1899 \else
1900   \setlength\fullwidth{0.76\paperwidth}
1901 \fi
1902 \if@twocolumn \@tempdima=2zw \else \@tempdima=1zw \fi
1903 \divide\fullwidth\@tempdima \multiply\fullwidth\@tempdima
1904 \setlength\textwidth{\fullwidth}

```

```

1905 %</article>
1906 %<*book>
1907 \if@report
1908   \setlength\fullwidth{0.76\paperwidth}
1909 \else
1910   \setlength\fullwidth{\paperwidth}
1911   \addtolength\fullwidth{-36\jsc@mmm}
1912 \fi
1913 \if@twocolumn \@tempdima=2zw \else \@tempdima=1zw \fi
1914 \divide\fullwidth\@tempdima \multiply\fullwidth\@tempdima
1915 \setlength\textwidth{\fullwidth}
1916 \if@report \else
1917   \if@twocolumn \else
1918     \ifdim \fullwidth>40zw
1919       \setlength\textwidth{40zw}
1920     \fi
1921   \fi
1922 \fi
1923 %</book>
1924 %<*report>
1925 \setlength\fullwidth{0.76\paperwidth}
1926 \if@twocolumn \@tempdima=2zw \else \@tempdima=1zw \fi
1927 \divide\fullwidth\@tempdima \multiply\fullwidth\@tempdima
1928 \setlength\textwidth{\fullwidth}
1929 %</report>
1930 %<*jspf>
1931 \setlength\fullwidth{50zw}
1932 \addtolength\fullwidth{8\jsc@mmm}
1933 \setlength\textwidth{\fullwidth}
1934 %</jspf>
1935 %<*kiyou>
1936 \setlength\fullwidth{48zw}
1937 \addtolength\fullwidth{\columnsep}
1938 \setlength\textwidth{\fullwidth}
1939 %</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)。

```
1940 %<*article|book|report>
1941 \if@slide
1942   \setlength{\textheight}{0.95\paperheight}
1943 \else
1944   \setlength{\textheight}{0.83\paperheight}
1945 \fi
1946 \addtolength{\textheight}{-10\jsc@empt}%% from -\topskip (2016-10-08); from -
    \headheight (2003-06-26)
1947 \addtolength{\textheight}{-\headsep}
1948 \addtolength{\textheight}{-\footskip}
1949 \addtolength{\textheight}{-\topskip}
1950 \divide\textheight\baselineskip
1951 \multiply\textheight\baselineskip
1952 %</article|book|report>
1953 %<jspf>\setlength{\textheight}{51\baselineskip}
1954 %<kiyou>\setlength{\textheight}{47\baselineskip}
1955 \addtolength{\textheight}{\topskip}
1956 \addtolength{\textheight}{0.1\jsc@empt}
1957 %<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  $\text{L}^{\text{A}}\text{T}_{\text{E}}\text{X} 2_{\epsilon}$  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。

```
1958 \def\flushbottom{%
1959   \def\@textbottom{\vskip \z@ \@plus.1\jsc@empt}%
1960   \let\@texttop\relax}
```

`\marginparsep` `\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。

```
1961 \setlength\marginparsep{\columnsep}
1962 \setlength\marginparpush{\baselineskip}
```

`\oddsidemargin` 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 `\evensidemargin` `\oddsidemargin` が使われます。 $\text{T}_{\text{E}}\text{X}$  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  $\text{pL}^{\text{A}}\text{T}_{\text{E}}\text{X} 2_{\epsilon}$  (`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。

```
1963 \setlength{\oddsidemargin}{\paperwidth}
1964 \addtolength{\oddsidemargin}{-\fullwidth}
1965 \setlength{\oddsidemargin}{.5\oddsidemargin}
1966 \iftombow
```



```

1967 \addtolength{\oddsidemargin}{-1in}
1968 \else
1969 \addtolength{\oddsidemargin}{-\inv@mag in}
1970 \fi
1971 \setlength{\evensidemargin}{\oddsidemargin}
1972 \if@mparswitch
1973 \addtolength{\evensidemargin}{\fullwidth}
1974 \addtolength{\evensidemargin}{-\textwidth}
1975 \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 1zw の整数倍に切り捨てます。

```

1976 \setlength\marginparwidth{\paperwidth}
1977 \addtolength\marginparwidth{-\oddsidemargin}
1978 \addtolength\marginparwidth{-\inv@mag in}
1979 \addtolength\marginparwidth{-\textwidth}
1980 \addtolength\marginparwidth{-10\jsc@mmm}
1981 \addtolength\marginparwidth{-\marginparsep}
1982 \@tempdima=1zw
1983 \divide\marginparwidth\@tempdima
1984 \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)。

```

1985 \setlength\topmargin{\paperheight}
1986 \addtolength\topmargin{-\textheight}
1987 \if@slide
1988 \addtolength\topmargin{-\headheight}
1989 \else
1990 \addtolength\topmargin{-10\jsc@empt}%% from -\topskip (2016-10-08); from -
    \headheight (2003-06-26)
1991 \fi
1992 \addtolength\topmargin{-\headsep}
1993 \addtolength\topmargin{-\footskip}
1994 \setlength\topmargin{0.5\topmargin}
1995 %<kiyou>\setlength\topmargin{81truebp}
1996 \iftombow
1997 \addtolength\topmargin{-1in}
1998 \else
1999 \addtolength\topmargin{-\inv@mag in}
2000 \fi
2001 %</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。

---

```
2002 \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。

```
2003 \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 L<sup>A</sup>T<sub>E</sub>X 2<sub>ε</sub> 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。

```
2004 \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。

```
2005 \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。

```
2006 \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。

```
2007 \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。  
 2008 `\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。  
 2009 `\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。  
 2010 `\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。  
 2011 `\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。  
 2012 `\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。  
 2013 `\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・  
`\textfloatsep` 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  
`\intextsep` 文との距離です。  
 2014 `\setlength\floatsep {12\jsc@empt \@plus 2\jsc@empt \@minus 2\jsc@empt}`  
 2015 `\setlength\textfloatsep{20\jsc@empt \@plus 2\jsc@empt \@minus 4\jsc@empt}`  
 2016 `\setlength\intextsep {12\jsc@empt \@plus 2\jsc@empt \@minus 2\jsc@empt}`

`\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` 2017 `\setlength\dblfloatsep {12\jsc@empt \@plus 2\jsc@empt \@minus 2\jsc@empt}`  
 2018 `\setlength\dbltextfloatsep{20\jsc@empt \@plus 2\jsc@empt \@minus 4\jsc@empt}`

`\@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` `\@fpsep` はフロート間に入ります。  
`\@fpbot` 2019 `\setlength\@fptop{0\jsc@empt \@plus 1fil}`  
 2020 `\setlength\@fpsep{8\jsc@empt \@plus 2fil}`  
 2021 `\setlength\@fpbot{0\jsc@empt \@plus 1fil}`

`\@dblftop` 段抜きフロートについての値です。  
`\@dblfpsep` 2022 `\setlength\@dblftop{0\jsc@empt \@plus 1fil}`  
 2023 `\setlength\@dblfpsep{8\jsc@empt \@plus 2fil}`  
`\@dblfpbot` 2024 `\setlength\@dblfpbot{0\jsc@empt \@plus 1fil}`

## 6 改ページ（日本語 T<sub>E</sub>X 開発コミュニティ版のみ）

`\pltx@cleartorightpage` [2017-02-24] コミュニティ版 pL<sup>A</sup>T<sub>E</sub>X の標準クラス 2017/02/15 に合わせて、同じ命令を追  
`\pltx@cleartoleftpage` 加しました。

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

```
2025 %\def\pltx@cleartorightpage{\clearpage\if@twoside
2026 %  \ifodd\c@page
2027 %    \iftdir
2028 %      \hbox{}\thispagestyle{empty}\newpage
2029 %      \if@twocolumn\hbox{}\newpage\fi
2030 %    \fi
2031 %  \else
2032 %    \ifydir
2033 %      \hbox{}\thispagestyle{empty}\newpage
2034 %      \if@twocolumn\hbox{}\newpage\fi
2035 %    \fi
2036 %  \fi\fi}
2037 %\def\pltx@cleartoleftpage{\clearpage\if@twoside
2038 %  \ifodd\c@page
2039 %    \ifydir
2040 %      \hbox{}\thispagestyle{empty}\newpage
2041 %      \if@twocolumn\hbox{}\newpage\fi
2042 %    \fi
2043 %  \else
2044 %    \iftdir
2045 %      \hbox{}\thispagestyle{empty}\newpage
2046 %      \if@twocolumn\hbox{}\newpage\fi
2047 %    \fi
2048 %  \fi\fi}
2049 \def\pltx@cleartooddpage{\clearpage\if@twoside
2050   \ifodd\c@page\else
2051     \hbox{}\thispagestyle{empty}\newpage
2052     \if@twocolumn\hbox{}\newpage\fi
2053   \fi\fi}
2054 \def\pltx@cleartoevenpage{\clearpage\if@twoside
2055   \ifodd\c@page
2056     \hbox{}\thispagestyle{empty}\newpage
2057     \if@twocolumn\hbox{}\newpage\fi
2058   \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。

```
2059 \let\pltx@cleartorightpage\pltx@cleartooddpage
2060 \let\pltx@cleartoleftpage\pltx@cleartoevenpage

\vsiz の値がアレな場合は本体開始まで \clearpage を無効にする。
2061 \ifdim\vsiz=\z@
2062 \begingroup
2063 \toks@{\expandafter{\clearpage}}
2064 \xdef\clearpage{\noexpand\ifbxjs@after@preamble\the\toks@\noexpand\fi}
2065 \endgroup
2066 \fi
```

---

`\cleardoublepage` [2017-02-24] コミュニティ版 p $\text{\LaTeX}$  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。

```
2067 %<*book|report>
2068 \if@openleft
2069 \let\cleardoublepage\pltx@cleartoleftpage
2070 \else\if@openright
2071 \let\cleardoublepage\pltx@cleartorightpage
2072 \fi\fi
2073 %</book|report>
```

## 7 ページスタイル

ページスタイルとして、 $\text{\LaTeX}$  2<sub>ε</sub> (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  $\text{\LaTeX}$  2<sub>ε</sub> 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`, `\@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。  
`\@evenfoot` `\ps@...` の中で定義しておきます。

`\@oddfoot` 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。

<code>\markboth{左}{右}</code>	両方の柱を設定します。
<code>\markright{右}</code>	右の柱を設定します。
<code>\leftmark</code>	左の柱を出力します。
<code>\rightmark</code>	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。L<sup>A</sup>T<sub>E</sub>X 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。

```
2074 % \def\ps@empty{%
2075 %   \let\mkboth\gobbletwo
2076 %   \let\oddhead\empty
2077 %   \let\oddfoot\empty
2078 %   \let\evenhead\empty
2079 %   \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。

```
2080 \def\ps@plainfoot{%
2081   \let\mkboth\gobbletwo
2082   \let\oddhead\empty
2083   \def\oddfoot{\normalfont\hfil\thepage\hfil}%
2084   \let\evenhead\empty
2085   \let\evenfoot\oddfoot}
2086 \def\ps@plainhead{%
2087   \let\mkboth\gobbletwo
2088   \let\oddfoot\empty
2089   \let\evenfoot\empty
2090   \def\evenhead{%
2091     \ifmparswitch \hss \fi
2092     \hbox to \fullwidth{\textbf{\thepage}\hfil}%
2093     \ifmparswitch\else \hss \fi}%
2094   \def\oddhead{%
2095     \hbox to \fullwidth{\hfil\textbf{\thepage}}\hss}}
2096 %<book>\let\ps@plain\ps@plainhead
2097 %<!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。

```
2098 %<*article|slide>
2099 \if@twoside
2100   \def\ps@headings{%
2101     \let\oddfoot\empty
2102     \let\evenfoot\empty
2103     \def\evenhead{\ifmparswitch \hss \fi
2104       \underline{\hbox to \fullwidth{\textbf{\thepage}\hfil\leftmark}}}%
2105     \ifmparswitch\else \hss \fi}%
2106   \def\oddhead{\hss \fi}
```

```

2106 \def\@oddhead{%
2107 \underline{%
2108 \hbox to \fullwidth{\rightmark}\hfil\textbf{\thepage}}}\hss}%
2109 \let\@mkboth\markboth
2110 \def\sectionmark##1{\markboth{%
2111 \ifnum \c@secnumdepth >\z@ \bxjs@label@sect{section}\hskip1\jsZw\fi
2112 ##1}}}%
2113 \def\subsectionmark##1{\markright{%
2114 \ifnum \c@secnumdepth >\@ne \bxjs@label@sect{subsection}\hskip1\jsZw\fi
2115 ##1}}}%
2116 }
2117 \else % if not twoside
2118 \def\ps@headings{%
2119 \let\@oddfoot\@empty
2120 \def\@oddhead{%
2121 \underline{%
2122 \hbox to \fullwidth{\rightmark}\hfil\textbf{\thepage}}}\hss}%
2123 \let\@mkboth\markboth
2124 \def\sectionmark##1{\markright{%
2125 \ifnum \c@secnumdepth >\z@ \bxjs@label@sect{section}\hskip1\jsZw\fi
2126 ##1}}}%
2127 \fi
2128 %</article|slide>

```

次は book および report の場合です。[2011-05-10] しっぱ愛好家さん [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）。

```

2129 %<*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ので代わりに用いる。

```

2130 \def\bxjs@maybe@autoxspacing{%
2131 \ifx\autoxspacing\undefined\else \autoxspacing \fi}

```

---

```

2132 \newif\if@omit@number
2133 \def\ps@headings{%
2134 \let\@oddfoot\@empty
2135 \let\@evenfoot\@empty
2136 \def\@evenhead{%
2137 \if@mparswitch \hss \fi
2138 \underline{\hbox to \fullwidth{\bxjs@maybe@autoxspacing
2139 \textbf{\thepage}\hfil\leftmark}}}%
2140 \if@mparswitch\else \hss \fi}%
2141 \def\@oddhead{\underline{\hbox to \fullwidth{\bxjs@maybe@autoxspacing
2142 {\if@twoside\rightmark\else\leftmark\fi}\hfil\textbf{\thepage}}}\hss}%
2143 \let\@mkboth\markboth
2144 \def\chaptermark##1{\markboth{%

```

```

2145 \ifnum \c@secnumdepth >\m@ne
2146 \if@mainmatter
2147 \if@omit@number\else
2148 \@chapapp\thechapter\@chappos\hskip1\jsZw
2149 \fi
2150 \fi
2151 \fi
2152 ##1}{}}}%
2153 \def\sectionmark##1{\markright{%
2154 \ifnum \c@secnumdepth >\z@ \bxjs@label@sect{section}\hskip1\jsZw\fi
2155 ##1}}}%
2156 %</book|report>

```

最後は学会誌の場合です。

```

2157 %<*jspf>
2158 \def\ps@headings{%
2159 \def\@oddfoot{\normalfont\hfil\thepage\hfil}
2160 \def\@evenfoot{\normalfont\hfil\thepage\hfil}
2161 \def\@oddhead{\normalfont\hfil \@title \hfil}
2162 \def\@evenhead{\normalfont\hfil プラズマ・核融合学会誌\hfil}}
2163 %</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。

```

2164 \def\ps@myheadings{%
2165 \let\@oddfoot\@empty\let\@evenfoot\@empty
2166 \def\@evenhead{%
2167 \if@mparswitch \hss \fi%
2168 \hbox to \fullwidth{\thepage\hfil\leftmark}%
2169 \if@mparswitch\else \hss \fi}%
2170 \def\@oddhead{%
2171 \hbox to \fullwidth{\rightmark\hfil\thepage}\hss}%
2172 \let\@mkboth\@gobbletwo
2173 %<book|report> \let\chaptermark\@gobble
2174 \let\sectionmark\@gobble
2175 %<!book&!report> \let\subsectionmark\@gobble
2176 }

```

## 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。

```

2177 \let\bxjs@phantomsection\relax
2178 \g@addto@macro\bxjs@begin@document@hook{%
2179 \@ifpackageloaded{hyperref}{%
2180 \let\bxjs@phantomsection\phantomsection

```



```

2181 }{}%
2182 }

```

---

## 8.1 表題

`\title` これらは L<sup>A</sup>T<sub>E</sub>X 本体で次のように定義されています。ここではコメントアウトした形で示します。

```

\author
\date 2183 % \newcommand*{\title}[1]{\gdef\@title{#1}}
      2184 % \newcommand*{\author}[1]{\gdef\@author{#1}}
      2185 % \newcommand*{\date}[1]{\gdef\@date{#1}}
      2186 % \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。

```

2187 \newcommand*{\jsSubtitle}[1]{\gdef\bxjs@subtitle{#1}}
2188 %\let\bxjs@subtitle\@undefined

\title にフックを入れる。

2189 \renewcommand*{\title}[1]{\bxjs@decl@subtitle\gdef\@title{#1}}
2190 \g@addto@macro\bxjs@begin@document@hook{\bxjs@decl@subtitle}
2191 \def\bxjs@decl@subtitle{%
2192   \global\let\bxjs@decl@subtitle\relax
2193   \ifx\subtitle\@undefined
2194     \global\let\subtitle\jsSubtitle
2195   \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。

```

2196 \def\bxjs@annihilate@subtitle{%
2197   \ifx\subtitle\jsSubtitle \global\let\subtitle\relax \fi
2198   \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。

```

\author 2199 %<*jspf>
\keywords 2200 \newcommand*{\etitle}[1]{\gdef\@etitle{#1}}
          2201 \newcommand*{\eauthor}[1]{\gdef\@eauthor{#1}}
          2202 \newcommand*{\keywords}[1]{\gdef\@keywords{#1}}
          2203 \newcommand*{\email}[1]{\gdef\@authors@mail{#1}}

```

```

2204 \newcommand*{\AuthorsEmail}[1]{\gdef\authors@mail{author's e-mail:\ #1}}
2205 %</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なります。

```

2206 \def\plainifnotempty{%
2207   \ifx \@oddhead \@empty
2208     \ifx \@oddfoot \@empty
2209       \else
2210         \thispagestyle{plainfoot}%
2211       \fi
2212     \else
2213       \thispagestyle{plainhead}%
2214     \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。

```

2215 %<*article|book|report|slide>
2216 \if@titlepage
2217   \newcommand{\maketitle}{%
2218     \begin{titlepage}%
2219       \let\footnotesize\small
2220       \let\footnoterule\relax
2221       \let\footnote\thanks
2222       \null\vfil
2223       \if@slide
2224         {\footnotesize \@date}%
2225       \begin{center}
2226         \mbox{} \\\[1\jsZw]
2227         \large
2228         {\maybeblue\hrule height0\jsc@mpt depth2\jsc@mpt\relax}\par
2229         \jsc@smallskip
2230         \@title
2231         \ifx\bxjs@subtitle\@undefined\else
2232           \par\vskip\z@
2233           {\small \bxjs@subtitle\par}
2234         \fi
2235         \jsc@smallskip
2236         {\maybeblue\hrule height0\jsc@mpt depth2\jsc@mpt\relax}\par
2237         \vfill
2238         {\small \@author}%

```

```

2239         \end{center}
2240     \else
2241         \vskip 60\jsc@empt
2242         \begin{center}%
2243             {\LARGE \@title \par}%
2244             \ifx\bxjs@subtitle\@undefined\else
2245                 \vskip5\jsc@empt
2246                 {\normalsize \bxjs@subtitle\par}
2247             \fi
2248             \vskip 3em%
2249             {\large
2250                 \lineskip .75em
2251                 \begin{tabular}[t]{c}%
2252                     \@author
2253                 \end{tabular}\par}%
2254             \vskip 1.5em
2255             {\large \@date \par}%
2256         \end{center}%
2257     \fi
2258     \par
2259     \@thanks\vfil\null
2260 \end{titlepage}%
2261 \setcounter{footnote}{0}%
2262 \global\let\thanks\relax
2263 \global\let\maketitle\relax
2264 \global\let\@thanks\@empty
2265 \global\let\@author\@empty
2266 \global\let\@date\@empty
2267 \global\let\@title\@empty
2268 \global\let\title\relax
2269 \global\let\author\relax
2270 \global\let\date\relax
2271 \global\let\and\relax
2272 \bxjs@annihilate@subtitle
2273 }%
2274 \else
2275     \newcommand{\maketitle}{\par
2276         \begingroup
2277             \renewcommand\thefootnote{\@fnsymbol\c@footnote}%
2278             \def\@makefnmark{\rlap{\@textsuperscript{\normalfont\@thefnmark}}}%
2279             \long\def\@makefntext##1{\advance\leftskip 3\jsZw
2280                 \parindent 1\jsZw\noindent
2281                 \llap{\@textsuperscript{\normalfont\@thefnmark}\hskip0.3\jsZw}##1}%
2282             \if@twocolumn
2283                 \ifnum \col@number=\@ne
2284                     \@maketitle
2285                 \else
2286                     \twocolumn[\@maketitle]%
2287                 \fi

```

```

2288     \else
2289     \newpage
2290     \global\@topnum\z@ % Prevents figures from going at top of page.
2291     \@maketitle
2292     \fi
2293     \plainifnotempty
2294     \@thanks
2295 \endgroup
2296 \setcounter{footnote}{0}%
2297 \global\let\thanks\relax
2298 \global\let\maketitle\relax
2299 \global\let\@thanks\@empty
2300 \global\let\@author\@empty
2301 \global\let\@date\@empty
2302 \global\let\@title\@empty
2303 \global\let\title\relax
2304 \global\let\author\relax
2305 \global\let\date\relax
2306 \global\let\and\relax
2307 \bxjs@annihilate@subtitle
2308 }

```

`\@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。

```

2309 \def\@maketitle{%
2310     \newpage\null
2311     \vskip 2em
2312     \begin{center}%
2313         \let\footnote\thanks
2314         {\LARGE \@title \par}%
2315         \ifx\bxjs@subtitle\@undefined\else
2316             \vskip3\jsc@mpt
2317             {\normalsize \bxjs@subtitle\par}
2318         \fi
2319         \vskip 1.5em
2320         {\large
2321             \lineskip .5em
2322             \begin{tabular}[t]{c}%
2323                 \@author
2324             \end{tabular}\par}%
2325         \vskip 1em
2326         {\large \@date}%
2327     \end{center}%
2328     \par\vskip 1.5em
2329 %<article|slide> \ifvoid\@abstractbox\else\centerline{\box\@abstractbox}\vskip1.5em\fi
2330 }
2331 \fi
2332 %</article|book|report|slide>
2333 %<*jspf>
2334 \newcommand{\maketitle}{\par

```

```

2335 \begin{group
2336   \renewcommand\thefootnote{\@fnsymbol\c@footnote}%
2337   \def\@makefnmark{\rlap{\@textsuperscript{\normalfont\@thefnmark}}}%
2338   \long\def\@makefntext##1{\advance\leftskip 3\jsZw
2339     \parindent 1\jsZw\noindent
2340     \llap{\@textsuperscript{\normalfont\@thefnmark}\hskip0.3\jsZw}##1}%
2341   \twocolumn[\@maketitle]%
2342   \plainifnotempty
2343   \@thanks
2344 \end{group}
2345 \setcounter{footnote}{0}%
2346 \global\let\thanks\relax
2347 \global\let\maketitle\relax
2348 \global\let\@thanks\@empty
2349 \global\let\@author\@empty
2350 \global\let\@date\@empty
2351 % \global\let\@title\@empty % \@title は柱に使う
2352 \global\let\title\relax
2353 \global\let\author\relax
2354 \global\let\date\relax
2355 \global\let\and\relax
2356 \ifx\authors@mail\@undefined\else{%
2357   \def\@makefntext{\advance\leftskip 3\jsZw \parindent -3\jsZw}%
2358   \footnotetext[0]{\itshape\authors@mail}%
2359 } \fi
2360 \global\let\authors@mail\@undefined}
2361 \def\@maketitle{%
2362   \newpage\null
2363   \vskip 6em % used to be 2em
2364   \begin{center}
2365     \let\footnote\thanks
2366     \ifx\@title\@undefined\else{\LARGE\headfont\@title\par} \fi
2367     \lineskip .5em
2368     \ifx\@author\@undefined\else
2369       \vskip 1em
2370       \begin{tabular}[t]{c}%
2371         \@author
2372       \end{tabular}\par
2373     \fi
2374     \ifx\@etitle\@undefined\else
2375       \vskip 1em
2376       {\large \@etitle \par}%
2377     \fi
2378     \ifx\@eauthor\@undefined\else
2379       \vskip 1em
2380       \begin{tabular}[t]{c}%
2381         \@eauthor
2382       \end{tabular}\par
2383     \fi

```

```

2384 \vskip 1em
2385 \@date
2386 \end{center}
2387 \vskip 1.5em
2388 \centerline{\box\@abstractbox}
2389 \ifx\@keywords\@undefined\else
2390 \vskip 1.5em
2391 \centerline{\parbox{157\jsc@mmm}{\textsf{Keywords:}}\ \small\@keywords}}
2392 \fi
2393 \vskip 1.5em}
2394 %</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。

```

2395 \def\bxjs@label@sect#1{%
2396 \ifundefined{label#1}{\@nameuse{the#1}}{\@nameuse{label#1}}}
2397 \def\@seccntformat#1{\bxjs@label@sect{#1}\quad}

```

`\@secapp` 節番号の接頭辞。

`\@secpos` 節番号の接尾辞。

```

2398 \ifnum\bxjs@label@section=\bxjs@label@section@compat\else
2399 \def\@secapp{\presectionname}
2400 \def\@secpos{\postsectionname}
2401 \fi

```

`\labelsection` 節番号の出力書式。

```

2402 \ifnum\bxjs@label@section=\bxjs@label@section@modern
2403 \def\labelsection{\@secapp\thesection\@secpos}
2404 \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、見出しの下側の空きです。負の場合は、絶対値が見出しの右側の空きです (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。

```
2405 \def\@startsection#1#2#3#4#5#6{%
2406   \if@noskipsec \leavevmode \fi
2407   \par
2408   % 見出し上の空きを \@tempskipa にセットする
2409   \@tempskipa #4\relax
2410   % \@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
2411   \if@english \@afterindentfalse \else \@afterindenttrue \fi
2412   % 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
2413   \ifdim \@tempskipa <\z@
2414     \@tempskipa -\@tempskipa \@afterindentfalse
2415   \fi
2416   \if@nobreak
2417     % \everypar{\everyparhook}% これは間違い
2418     \everypar{}%
2419   \else
2420     \addpenalty\@secpenalty
2421     % 次の行は削除
2422     % \addvspace\@tempskipa
2423     % 次の \noindent まで追加
2424     \ifdim \@tempskipa >\z@
2425       \if@slide\else
2426         \null
2427         \vspace*{-\baselineskip}%
2428       \fi
2429       \vskip\@tempskipa
2430     \fi
2431   \fi
2432   \noindent
2433   % 追加終わり
2434   \@ifstar
```

```

2435     {\@ssect{#3}{#4}{#5}{#6}}%
2436     {\@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。

```

2437 \ifnum\bxjs@everyparhook=\bxjs@everyparhook@@compat
2438   \let\bxjs@if@ceph\@firstofone
2439 \else \let\bxjs@if@ceph\@gobble
2440 \fi

```

---

```

2441 \def\@sect#1#2#3#4#5#6[#7]#8{%
2442   \ifnum #2>\c@secnumdepth
2443     \let\@svsec\@empty
2444   \else
2445     \refstepcounter{#1}%
2446     \protected@edef\@svsec{\@secntformat{#1}\relax}%
2447   \fi
2448 % 見出し後の空きを \@tempskipa にセット
2449 \@tempskipa #5\relax
2450 % 条件判断の順序を入れ替えました
2451 \ifdim \@tempskipa<\z@
2452   \def\@svsechd{%
2453     #6{\hskip #3\relax
2454       \@svsec #8}%
2455     \csname #1mark\endcsname{#7}%
2456     \addcontentsline{toc}{#1}{%
2457       \ifnum #2>\c@secnumdepth \else
2458         \protect\numberline{\bxjs@label@sect{#1}}%
2459       \fi
2460       #7}}% 目次にフルネームを載せるなら #8
2461   \else
2462     \begingroup
2463     \interlinepenalty \@M % 下から移動
2464     #6{%
2465       \@hangfrom{\hskip #3\relax\@svsec}%
2466     % \interlinepenalty \@M % 上に移動
2467       #8\@par}%
2468     \endgroup
2469     \csname #1mark\endcsname{#7}%
2470     \addcontentsline{toc}{#1}{%
2471       \ifnum #2>\c@secnumdepth \else
2472         \protect\numberline{\bxjs@label@sect{#1}}%
2473       \fi

```



```

2474      #7}% 目次にフルネームを載せるならここは #8
2475 \fi
2476 \@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。

```

2477 \def\@xsect#1{%
2478 % 見出しの後ろの空きを \@tempskipa にセット
2479 \@tempskipa #1\relax
2480 % 条件判断の順序を変えました
2481 \ifdim \@tempskipa<\z@
2482   \nobreakfalse
2483   \global\@noskipsectrue
2484   \everypar{%
2485     \if@noskipsec
2486       \global\@noskipsecfalse
2487       {\setbox\z@\lastbox}%
2488       \clubpenalty\@M
2489       \begingroup \@svsechd \endgroup
2490       \unskip
2491       \@tempskipa #1\relax
2492       \hskip -\@tempskipa
2493     \else
2494       \clubpenalty \@clubpenalty
2495       \everypar\expandafter{\bxjs@if@ceph\everyparhook}%

```

**TODO:** ↑ ナニコレ？

```

2496     \fi\bxjs@if@ceph\everyparhook}%
2497 \else
2498   \par \nobreak
2499   \vskip \@tempskipa
2500   \@afterheading
2501 \fi
2502 \if@slide
2503   {\vskip\if@twocolumn-5\jsc@mpt\else-6\jsc@mpt\fi
2504     \maybeblue\hrule height0\jsc@mpt depth1\jsc@mpt
2505     \vskip\if@twocolumn 4\jsc@mpt\else 7\jsc@mpt\fi\relax}%
2506 \fi
2507 \par % 2000-12-18
2508 \ignorespaces}
2509 \def\@ssect#1#2#3#4#5{%
2510 \@tempskipa #3\relax
2511 \ifdim \@tempskipa<\z@
2512   \def\@svsechd{#4{\hskip #1\relax #5}}%
2513 \else
2514   \begingroup
2515   #4{%

```

```

2516      \@hangfrom{\hskip #1}%
2517      \interlinepenalty \@M #5\@@par}%
2518  \endgroup
2519  \fi
2520  \@xsect{#3}}

```

## ■柱関係の命令

`\chaptermark` `\...mark` の形の命令を初期化します (第 7 節参照)。`\chaptermark` 以外は L<sup>A</sup>T<sub>E</sub>X 本体で定義済みです。

```

\subsectionmark 2521 \newcommand*\chaptermark[1]{ }
\subsubsectionmark 2522 % \newcommand*\sectionmark[1]{ }
2523 % \newcommand*\subsectionmark[1]{ }
\paragraphmark 2524 % \newcommand*\subsubsectionmark[1]{ }
\subparagraphmark 2525 % \newcommand*\paragraphmark[1]{ }
2526 % \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定めるカウンタです。

```

2527 %<!book&!report>\setcounter{secnumdepth}{3}
2528 %<book|report>\setcounter{secnumdepth}{2}

```

`\c@chapter` 見出し番号のカウンタです。`\newcounter` の第 1 引数が新たに作るカウンタです。これは

`\c@section` 第 2 引数が増加するたびに 0 に戻されます。第 2 引数は定義済みのカウンタです。

```

\c@subsection 2529 \newcounter{part}
2530 %<book|report>\newcounter{chapter}
\c@subsubsection 2531 %<book|report>\newcounter{section}[chapter]
\c@paragraph 2532 %<!book&!report>\newcounter{section}
\c@subparagraph 2533 \newcounter{subsection}[section]
2534 \newcounter{subsubsection}[subsection]
2535 \newcounter{paragraph}[subsubsection]
2536 \newcounter{subparagraph}[paragraph]

```

`\thepart` カウンタの値を出力する命令 `\the 何々` を定義します。

`\thechapter` カウンタを出力するコマンドには次のものがあります。

<code>\thesection</code>	<code>\arabic{COUNTER}</code>	1, 2, 3, ...
<code>\thesubsection</code>	<code>\roman{COUNTER}</code>	i, ii, iii, ...
<code>\thesubsubsection</code>	<code>\Roman{COUNTER}</code>	I, II, III, ...
<code>\theparagraph</code>	<code>\alph{COUNTER}</code>	a, b, c, ...
<code>\thesubparagraph</code>	<code>\Alph{COUNTER}</code>	A, B, C, ...
	<code>\kansuji{COUNTER}</code>	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。

```

2537 \renewcommand{\thepart}{\@Roman\c@part}
2538 %<*&!book&!report>

```

```

2539 \ifnum\bxjs@label@section=\bxjs@label@section@@compat
2540 \renewcommand{\thesection}{\presectionname\@arabic\c@section\postsectionname}
2541 \renewcommand{\thesubsection}{\@arabic\c@section.\@arabic\c@subsection}
2542 \else
2543 \renewcommand{\thesection}{\@arabic\c@section}
2544 \renewcommand{\thesubsection}{\thesection.\@arabic\c@subsection}
2545 \fi
2546 %</!book&!report>
2547 %<*book|report>
2548 \renewcommand{\thechapter}{\@arabic\c@chapter}
2549 \renewcommand{\thesection}{\thechapter.\@arabic\c@section}
2550 \renewcommand{\thesubsection}{\thesection.\@arabic\c@subsection}
2551 %</book|report>
2552 \renewcommand{\thesubsubsection}{\%
2553 \thesubsection.\@arabic\c@subsubsection}
2554 \renewcommand{\theparagraph}{\%
2555 \thesubsubsection.\@arabic\c@paragraph}
2556 \renewcommand{\thesubparagraph}{\%
2557 \theparagraph.\@arabic\c@subparagraph}

```

\@chapapp \@chapapp の初期値は \prechaptername (第) です。

\@chappos \@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。

```

2558 %<book|report>\newcommand{\@chapapp}{\prechaptername}
2559 %<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す。openany かどうかに関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、L<sup>A</sup>T<sub>E</sub>X 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 pL<sup>A</sup>T<sub>E</sub>X の標準クラス 2017/03/05 も参照)。

```

2560 %<*book|report>
2561 \newcommand\frontmatter{%
2562 \pltx@cleartooddpage
2563 \@mainmatterfalse
2564 \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。

```

2565 \newcommand\mainmatter{%
2566 \pltx@cleartooddpage
2567 \@mainmattertrue
2568 \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。

```
2569 \newcommand\backmatter{%
2570   \if@openleft
2571     \cleardoublepage
2572   \else\if@openright
2573     \cleardoublepage
2574   \else
2575     \clearpage
2576   \fi\fi
2577   \@mainmatterfalse}
2578 %</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 \CMDDB }
\def\CMDA      [#1]#2{...} % \chapter[...]{...} の定義
\def\CMDDB     #1{...}     % \chapter*{...} の定義
```

まず `book` と `report` のクラス以外です。

```
2579 %<!*book&!report>
2580 \newcommand\part{%
2581   \if@noskipsec \leavevmode \fi
2582   \par
2583   \addvspace{4ex}%
2584   \if@english \@afterindentfalse \else \@afterindenttrue \fi
2585   \secdef\@part\@spart}
2586 %</!book&!report>
```

`book` および `report` クラスの場合は、少し複雑です。

```
2587 %<*book|report>
2588 \newcommand\part{%
2589   \if@openleft
2590     \cleardoublepage
2591   \else\if@openright
2592     \cleardoublepage
2593   \else
2594     \clearpage
2595   \fi\fi
2596   \thispagestyle{empty}% 欧文用標準スタイルでは plain
```

```

2597 \if@twocolumn
2598 \onecolumn
2599 \@restonecoltrue
2600 \else
2601 \@restonecolfalse
2602 \fi
2603 \null\vfil
2604 \secdef\@part\@spart}
2605 %</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。

```

2606 %<*&book&!report>
2607 \def\@part[#1]#2{%
2608 \ifnum \c@secnumdepth >\m@ne
2609 \refstepcounter{part}%
2610 \addcontentsline{toc}{part}{%
2611 \prepartname\thepart\postpartname\hspace{1\jsZw}#1}%
2612 \else
2613 \addcontentsline{toc}{part}{#1}%
2614 \fi
2615 \markboth{}{}%
2616 {\parindent\z@
2617 \raggedright
2618 \interlinepenalty \@M
2619 \normalfont
2620 \ifnum \c@secnumdepth >\m@ne
2621 \Large\headfont\prepartname\thepart\postpartname
2622 \par\nobreak
2623 \fi
2624 \huge \headfont #2%
2625 \markboth{}{}\par}%
2626 \nobreak
2627 \vskip 3ex
2628 \@afterheading}
2629 %</!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。

```

2630 %<*&book|report>
2631 \def\@part[#1]#2{%
2632 \ifnum \c@secnumdepth >-2\relax
2633 \refstepcounter{part}%
2634 \addcontentsline{toc}{part}{%
2635 \prepartname\thepart\postpartname\hspace{1\jsZw}#1}%
2636 \else
2637 \addcontentsline{toc}{part}{#1}%
2638 \fi
2639 \markboth{}{}%

```

```

2640 {\centering
2641 \interlinepenalty \@M
2642 \normalfont
2643 \ifnum \c@secnumdepth >-2\relax
2644 \huge\headfont \prepartname\thepart\postpartname
2645 \par\vskip20\jsc@empt
2646 \fi
2647 \Huge \headfont #2\par}%
2648 \@endpart}
2649 %</book|report>

```

\@spart 番号を付けない部です。

```

2650 %<*!book&!report>
2651 \def\@spart#1{%
2652 \parindent \z@ \raggedright
2653 \interlinepenalty \@M
2654 \normalfont
2655 \huge \headfont #1\par}%
2656 \nobreak
2657 \vskip 3ex
2658 \@afterheading}
2659 %</!book&!report>
2660 %<*book|report>
2661 \def\@spart#1{%
2662 \centering
2663 \interlinepenalty \@M
2664 \normalfont
2665 \Huge \headfont #1\par}%
2666 \@endpart}
2667 %</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 L<sup>A</sup>T<sub>E</sub>X では classes.dtx v1.4b (2000/05/19) で修正されています。

```

2668 %<*book|report>
2669 \def\@endpart{\vfil\newpage
2670 \if@twoside
2671 \if@openleft %% added (2017/02/24)
2672 \null\thispagestyle{empty}\newpage
2673 \else\if@openright %% added (2016/12/13)
2674 \null\thispagestyle{empty}\newpage
2675 \fi\fi %% added (2016/12/13, 2017/02/24)
2676 \fi
2677 \if@restonecol
2678 \twocolumn
2679 \fi}

```

```
2680 %</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。

```
2681 %<*book|report>
2682 \newcommand{\chapter}{%
2683   \ifopenleft\cleardoublepage\else
2684   \ifopenright\cleardoublepage\else\clearpage\fi\fi
2685   \plainifnotempty % 元: \thispagestyle{plain}
2686   \global\@topnum\z@
2687   \ifenglish \@afterindentfalse \else \@afterindenttrue \fi
2688   \secdef
2689     {\@omit@numberfalse\@chapter}%
2690     {\@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。

```
2691 \def\@chapter[#1]#2{%
2692   \ifnum \c@secnumdepth >\m@ne
2693     \if@mainmatter
2694       \refstepcounter{chapter}%
2695       \typeout{\@chapapp\thechapter\@chappos}%
2696       \addcontentsline{toc}{chapter}%
2697         {\protect\numberline
2698 %       %{\ifenglish\thechapter\else\@chapapp\thechapter\@chappos\fi}%
2699         {\@chapapp\thechapter\@chappos}%
2700         #1}%
2701     \else\addcontentsline{toc}{chapter}{#1}\fi
2702   \else
2703     \addcontentsline{toc}{chapter}{#1}%
2704   \fi
2705   \chaptermark{#1}%
2706   \addtocontents{lof}{\protect\addvspace{10\jsc@empt}}%
2707   \addtocontents{lot}{\protect\addvspace{10\jsc@empt}}%
2708   \if@twocolumn
2709     \@topnewpage[\@makechapterhead{#2}]%
2710   \else
2711     \@makechapterhead{#2}%
2712     \@afterheading
2713   \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。

```
2714 \def\@makechapterhead#1{%
2715   \vspace*{2\Cvs}% 欧文は 50pt
2716   {\parindent \z@ \raggedright \normalfont
2717     \ifnum \c@secnumdepth >\m@ne
```

```

2718      \if@mainmatter
2719      \huge\headfont \@chapapp\thechapter\@chappos
2720      \par\nobreak
2721      \vskip \Cvs % 欧文は 20pt
2722      \fi
2723      \fi
2724      \interlinepenalty\@M
2725      \Huge \headfont #1\par\nobreak
2726      \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。

```

2727 \def\@schapter#1{%
2728   \chaptermark{#1}%
2729   \if@twocolumn
2730     \@topnewpage[\@makeschapterhead{#1}]%
2731   \else
2732     \@makeschapterhead{#1}\@afterheading
2733   \fi}

```

`\@makeschapterhead` 番号なしの章見出しです。

```

2734 \def\@makeschapterhead#1{%
2735   \vspace*{2\Cvs}% 欧文は 50pt
2736   {\parindent \z@ \raggedright
2737     \normalfont
2738     \interlinepenalty\@M
2739     \Huge \headfont #1\par\nobreak
2740     \vskip 3\Cvs}} % 欧文は 40pt
2741 %</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。

```

2742 \if@twocolumn
2743   \newcommand{\section}{%
2744     %<jspf>\ifx\maketitle\relax\else\maketitle\fi
2745     \@startsection{section}{1}{\z@}%
2746     %<!kiyou> {0.6\Cvs}{0.4\Cvs}%
2747     %<kiyou> { \Cvs}{0.5\Cvs}%
2748     % {\normalfont\large\headfont\@secapp}}
2749     {\normalfont\large\headfont\raggedright}}
2750 \else
2751   \newcommand{\section}{%
2752     \if@slide\clearpage\fi
2753     \@startsection{section}{1}{\z@}%
2754     {\Cvs \@plus.5\Cdp \@minus.2\Cdp}% 前アキ
2755     {.5\Cvs \@plus.3\Cdp}% 後アキ

```



```

2756 %    {\normalfont\Large\headfont\@secapp}}
2757      {\normalfont\Large\headfont\raggedright}}
2758 \fi

```

`\subsection` 同上です。

```

2759 \if@twocolumn
2760   \newcommand{\subsection}{\@startsection{subsection}{2}{\z@}%
2761     {\z@}{\if@slide .4\Cvs \else \z@ \fi}%
2762     {\normalfont\normalsize\headfont}}
2763 \else
2764   \newcommand{\subsection}{\@startsection{subsection}{2}{\z@}%
2765     {\Cvs \@plus.5\Cdp \@minus.2\Cdp}% 前アキ
2766     {.5\Cvs \@plus.3\Cdp}% 後アキ
2767     {\normalfont\large\headfont}}
2768 \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)。

```

2769 \if@twocolumn
2770   \newcommand{\subsubsection}{\@startsection{subsubsection}{3}{\z@}%
2771     {\z@}{\if@slide .4\Cvs \else \z@ \fi}%
2772     {\normalfont\normalsize\headfont}}
2773 \else
2774   \newcommand{\subsubsection}{\@startsection{subsubsection}{3}{\z@}%
2775     {\Cvs \@plus.5\Cdp \@minus.2\Cdp}%
2776     {\if@slide .5\Cvs \@plus.3\Cdp \else \z@ \fi}%
2777     {\normalfont\normalsize\headfont}}
2778 \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。

```
2779 \let\jsJaChar\@empty
```

---

```

2780 \newcommand\jsParagraphMark{\relax\jsJaChar{■}}
2781 \let\bxjs@org@paragraph@mark\jsParagraphMark

```

```

2782 \ifx\bxjs@paragraph@mark\@empty
2783   \let\jsParagraphMark\@empty
2784 \else\ifx\bxjs@paragraph@mark\@undefined\else
2785   \long\edef\jsParagraphMark{\noexpand\jsJaChar{\bxjs@paragraph@mark}}
2786 \fi\fi
2787 \if@twocolumn
2788   \newcommand{\paragraph}{\@startsection{paragraph}{4}{\z@}%
2789     {\z@}{\if@slide .4\Cvs \else -1\jsZw\fi}% 改行せず 1\jsZw のアキ
2790 %<jspf>   {\normalfont\normalsize\headfont}}
2791 %<!jspf>   {\normalfont\normalsize\headfont\jsParagraphMark}}
2792 \else
2793   \newcommand{\paragraph}{\@startsection{paragraph}{4}{\z@}%
2794     {0.5\Cvs \@plus.5\Cdp \@minus.2\Cdp}%
2795     {\if@slide .5\Cvs \@plus.3\Cdp \else -1\jsZw\fi}% 改行せず 1\jsZw のアキ
2796 %<jspf>   {\normalfont\normalsize\headfont}}
2797 %<!jspf>   {\normalfont\normalsize\headfont\jsParagraphMark}}
2798 \fi

```

\subparagraph 見出しの後ろで改行されません。

```

2799 \if@twocolumn
2800   \newcommand{\subparagraph}{\@startsection{subparagraph}{5}{\z@}%
2801     {\z@}{\if@slide .4\Cvs \@plus.3\Cdp \else -1\jsZw\fi}%
2802     {\normalfont\normalsize\headfont}}
2803 \else
2804   \newcommand{\subparagraph}{\@startsection{subparagraph}{5}{\z@}%
2805     {\z@}{\if@slide .5\Cvs \@plus.3\Cdp \else -1\jsZw\fi}%
2806     {\normalfont\normalsize\headfont}}
2807 \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。

```

2808 \if@slide
2809   \setlength\leftmargini{1\jsZw}
2810 \else
2811   \if@twocolumn
2812     \setlength\leftmargini{2\jsZw}
2813   \else
2814     \setlength\leftmargini{3\jsZw}
2815   \fi
2816 \fi

```

`\leftmarginii` 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  
`\leftmarginiii` になっています。ここでは全角幅の整数倍に丸めました。

```
\leftmarginiv 2817 \if@slide
\leftmarginv 2818 \setlength\leftmarginii {1\jsZw}
2819 \setlength\leftmarginiii{1\jsZw}
\leftmarginvi 2820 \setlength\leftmarginiv {1\jsZw}
2821 \setlength\leftmarginv {1\jsZw}
2822 \setlength\leftmarginvi {1\jsZw}
2823 \else
2824 \setlength\leftmarginii {2\jsZw}
2825 \setlength\leftmarginiii{2\jsZw}
2826 \setlength\leftmarginiv {2\jsZw}
2827 \setlength\leftmarginv {1\jsZw}
2828 \setlength\leftmarginvi {1\jsZw}
2829 \fi
```

`\labelsep` `\labelsep` はラベルと本文の間の距離です。`\labelwidth` はラベルの幅です。これは二分  
`\labelwidth` に変えました。

```
2830 \setlength \labelsep {0.5\jsZw} % .5em
2831 \setlength \labelwidth{\leftmargini}
2832 \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。

```
2833 \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 2834 \@beginparpenalty -\@lowpenalty
2835 \@endparpenalty -\@lowpenalty
\itempenalty 2836 \@itempenalty -\@lowpenalty
```

`\@listi` `\@listi` は `\leftmargin`, `\parsep`, `\topsep`, `\itemsep` などのトップレベルの定義を  
`\@listI` します。この定義は、フォントサイズコマンドによって変更されます（たとえば `\small` の  
中では小さい値に設定されます）。このため、`\normalsize` がすべてのパラメータを戻せる  
ように、`\@listI` で `\@listi` のコピーを保存します。元の値はかなり複雑ですが、ここ  
では簡素化してしまいました。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。

[2004-09-27] `\topsep` のグルー  $\pm_{0.1}^{0.2}$  `\baselineskip` を思い切って外しました。

```
2837 \def\@listi{\leftmargin\leftmargini
2838 \parsep \z@
2839 \topsep 0.5\baselineskip
2840 \itemsep \z@ \relax}
2841 \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）。

```
2842 \@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 2843 \def\@listii{\leftmargin\leftmarginii
\@listiv 2844 \labelwidth\leftmarginii \advance\labelwidth-\labelsep
2845 \topsep \z@
\@listv 2846 \parsep \z@
\@listvi 2847 \itemsep\parsep}
2848 \def\@listiii{\leftmargin\leftmarginiii
2849 \labelwidth\leftmarginiii \advance\labelwidth-\labelsep
2850 \topsep \z@
2851 \parsep \z@
2852 \itemsep\parsep}
2853 \def\@listiv {\leftmargin\leftmarginiv
2854 \labelwidth\leftmarginiv
2855 \advance\labelwidth-\labelsep}
2856 \def\@listv {\leftmargin\leftmarginv
2857 \labelwidth\leftmarginv
2858 \advance\labelwidth-\labelsep}
2859 \def\@listvi {\leftmargin\leftmarginvi
2860 \labelwidth\leftmarginvi
2861 \advance\labelwidth-\labelsep}
```

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

`\theenumi` 出力する番号の書式を設定します。これらは L<sup>A</sup>T<sub>E</sub>X 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。

```
2862 \renewcommand{\theenumi}{\@arabic\c@enumi}
2863 \renewcommand{\theenumii}{\@alph\c@enumii}
2864 \renewcommand{\theenumiii}{\@roman\c@enumiii}
2865 \renewcommand{\theenumiv}{\@Alph\c@enumiv}
```

`\labelenumi` `enumerate` 環境の番号を出力する命令です。第 2 レベル以外は最後に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  
`\labelenumiii` 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。

`\labelenumiv`

---

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。

---

```
2866 \newcommand*{\jsInJaParen}[1]{%
2867 \mbox{\jsInhibitGlue (#1) \jsInhibitGlue}}
2868 \newcommand{\labelenumi}{\theenumi.}
2869 \newcommand{\labelenumii}{\jsInJaParen{\theenumii}}
2870 \newcommand{\labelenumiii}{\theenumiii.}
2871 \newcommand{\labelenumiv}{\theenumiv.}
```

`\p@enumii \p@enumn` 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に書  
`\p@enumiii` 式です。これも第 2 レベルは和文用かっこにしました。

```
\p@enumiv 2872 \renewcommand{\p@enumii}{\theenumi}
2873 \renewcommand{\p@enumiii}{\theenumi\jsInhibitGlue (\theenumii )}
2874 \renewcommand{\p@enumiv}{\p@enumiii\theenumiii}
```

## ■itemize 環境

`\labelitemi` `itemize` 環境の第  $n$  レベルのラベルを作るコマンドです。

```
\labelitemii 2875 \newcommand\labelitemi{\textbullet}
2876 \newcommand\labelitemii{\normalfont\bfseries \textendash}
\labelitemiii 2877 \newcommand\labelitemiii{\textasteriskcentered}
\labelitemiv 2878 \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してしまいます。これを解決した新しい `description` の実装です。

```
2879 \newenvironment{description}{%
2880   \list{}{%
2881     \labelwidth=\leftmargin
2882     \labelsep=1\jsZw
2883     \advance \labelwidth by -\labelsep
2884     \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。

```
2885 \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。

```

2886 %<*book|report>
2887 \newenvironment{chapterabstract}{%
2888   \begin{list}{}{%
2889     \listparindent=1\jsZw
2890     \itemindent=\listparindent
2891     \rightmargin=0pt
2892     \leftmargin=5\jsZw}\item[]{}\end{list}\vspace{\baselineskip}}
2893 %</book|report>

```

“普通の” abstract 環境の定義。

```

2894 %<*article|report|slide>
2895 \newbox\@abstractbox
2896 \if@titlepage
2897   \newenvironment{abstract}{%
2898     \titlepage
2899     \null\vfil
2900     \@beginparpenalty\@lowpenalty
2901     \begin{center}%
2902       \headfont \abstractname
2903       \@endparpenalty\@M
2904     \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。

```

2905   \par}%
2906   {\par\vfil\null\endtitlepage}
2907 \else
2908   \newenvironment{abstract}{%
2909     \if@twocolumn
2910       \ifx\maketitle\relax
2911         \section*{\abstractname}%
2912       \else
2913         \global\setbox\@abstractbox\hbox\bgroup
2914         \begin{minipage}[b]{\textwidth}
2915           \small\parindent1\jsZw
2916           \begin{center}%
2917             {\headfont \abstractname\vspace{-.5em}\vspace{\z@}}%
2918           \end{center}%
2919           \list{}{%
2920             \listparindent\parindent
2921             \itemindent \listparindent
2922             \rightmargin \leftmargin}%
2923           \item\relax
2924         \fi
2925       \else
2926         \small
2927         \begin{center}%
2928           {\headfont \abstractname\vspace{-.5em}\vspace{\z@}}%
2929         \end{center}%
2930       \list{}{%

```

```

2931         \listparindent\parindent
2932         \itemindent \listparindent
2933         \rightmargin \leftmargin}%
2934     \item\relax
2935 \fi}{\if@twocolumn
2936     \ifx\maketitle\relax
2937     \else
2938         \endlist\end{minipage}\egroup
2939     \fi
2940 \else
2941     \endlist
2942 \fi}
2943 \fi
2944 %</article|report|slide>
2945 %<*jspf>
2946 \newbox\@abstractbox
2947 \newenvironment{abstract}{%
2948     \global\setbox\@abstractbox\hbox\bgroup
2949     \begin{minipage}[b]{157\jsc@mmm}\sffamily Abstract}\par
2950     \small
2951     \if@english \parindent6\jsc@mmm \else \parindent1\jsZw \fi}%
2952 {\end{minipage}\egroup}
2953 %</jspf>

```

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

```

2954 %<*book|report>
2955 \ifbxjs@force@chapterabstract
2956     \let\abstract\chapterabstract
2957     \let\endabstract\endchapterabstract
2958 \fi
2959 %</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。

```

2960 %<*jspf>
2961 %\newbox\@keywordsbox
2962 %\newenvironment{keywords}{%
2963 %     \global\setbox\@keywordsbox\hbox\bgroup
2964 %     \begin{minipage}[b]{1570\jsc@mmm}\sffamily Keywords:}\par
2965 %         \small\parindent0\jsZw}%
2966 %     {\end{minipage}\egroup}
2967 %</jspf>

```

## ■verse 環境

`verse (env.)` 詩のための `verse` 環境です。

```
2968 \newenvironment{verse}{%
2969   \let \=\@centercr
2970   \list{}{%
2971     \itemsep \z@
2972     \itemindent -2\jsZw % 元: -1.5em
2973     \listparindent\itemindent
2974     \rightmargin \z@
2975     \advance\leftmargin 2\jsZw}% 元: 1.5em
2976   \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。

```
2977 \newenvironment{quotation}{%
2978   \list{}{%
2979     \listparindent\parindent
2980     \itemindent\listparindent
2981     \rightmargin \z@}%
2982   \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。

```
2983 \newenvironment{quote}%
2984   {\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 1zw にし、括弧を全角にしました。

```
2985 \def\@begintheorem#1#2{\trivlist\labelsep=1\jsZw
2986   \item[\hskip \labelsep{\headfont #1\ #2}]}
2987 \def\@opargbegintheorem#1#2#3{\trivlist\labelsep=1\jsZw
2988   \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 pL<sup>A</sup>T<sub>E</sub>X 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 of 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 pL<sup>A</sup>T<sub>E</sub>X 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。

```

2989 \newenvironment{titlepage}{%
2990 %<book>      \pltx@cleartooddpage %% 2017-02-24
2991     \if@twocolumn
2992         \@restonecoltrue\onecolumn
2993     \else
2994         \@restonecolfalse\newpage
2995     \fi
2996     \thispagestyle{empty}%
2997     \ifodd\c@page\setcounter{page}\@ne\else\setcounter{page}\z@\fi %% 2017-
    02-24
2998 }%
2999 {\if@restonecol\twocolumn \else \newpage \fi
3000     \if@twoside\else
3001         \setcounter{page}\@ne
3002     \fi}

```

## ■付録

\appendix 本文と付録を分離するコマンドです。

```

3003 %<!*book&!report>
3004 \newcommand{\appendix}{\par
3005     \setcounter{section}{0}%
3006     \setcounter{subsection}{0}%
3007     \ifnum\bxjs@label@section=\bxjs@label@section@@compat
3008     \gdef\presectionname{\appendixname}%
3009     \gdef\postsectionname{}%
3010 % \gdef\thesection{\@Alph\c@section}% [2003-03-02]
3011     \gdef\thesection{\presectionname\@Alph\c@section\postsectionname}%
3012     \gdef\thesubsection{\@Alph\c@section.\@arabic\c@subsection}%
3013     \else
3014     \gdef\@secapp{\appendixname}%
3015     \gdef\@secpos{}%
3016     \gdef\thesection{\@Alph\c@section}%
3017     \fi}
3018 %</!*book&!report>
3019 %<*book|report>
3020 \newcommand{\appendix}{\par
3021     \setcounter{chapter}{0}%
3022     \setcounter{section}{0}%
3023     \gdef\@chapapp{\appendixname}%
3024     \gdef\@chappos{}%
3025     \gdef\thechapter{\@Alph\c@chapter}}
3026 %</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 倍の幅の空が入ります。

```
3027 \setlength\arraycolsep{5\jsc@mp}
```

`\tabcolsep` tabular 環境の列間には `\tabcolsep` の 2 倍の幅の空が入ります。

```
3028 \setlength\tabcolsep{6\jsc@mp}
```

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

```
3029 \setlength\arrayrulewidth{.4\p@}
```

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

```
3030 \setlength\doublerulesep{2\p@}
```

### ■tabbing 環境

`\tabbingsep` `\'` コマンドで入るアキです。

```
3031 \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。

```
3032 \skip\@mpfootins = \skip\footins
```

### ■framebox 環境

`\fbxsep` `\fbx`, `\framebox` で内側のテキストと枠との間の空です。

`\fbxrule` `\fbx`, `\framebox` の罫線の幅です。

```
3033 \setlength\fbxsep{3\jsc@mp}
```

```
3034 \setlength\fbxrule{.4\p@}
```

### ■equation と eqnarray 環境

`\theequation` 数式番号を出力するコマンドです。

```
3035 %<!book&!report>\renewcommand \theequation {\@arabic\c@equation}
```

```
3036 %<*book|report>
```

```
3037 \@addtoreset{equation}{chapter}
```

```
3038 \renewcommand\theequation
```

```
3039 {\ifnum \c@chapter>\z@ \thechapter.\fi \@arabic\c@equation}
```

```
3040 %</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。

```
3041 % \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。

```
3042 % \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。

```
3043 % \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。

```
3044 %<!*book&!report>
3045 \newcounter{figure}
3046 \renewcommand \thefigure {\@arabic\c@figure}
3047 %<!/!book&!report>
3048 %<*book|report>
3049 \newcounter{figure}[chapter]
3050 \renewcommand \thefigure
3051     {\ifnum \c@chapter>\z@ \thechapter.\fi \@arabic\c@figure}
3052 %</book|report>
```

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

```
\ext@figure 3053 \def\fps@figure{tbp}
3054 \def\ftype@figure{1}
\fnum@figure 3055 \def\ext@figure{lof}
3056 \def\fnum@figure{\figurename\nobreak\thefigure}
```

`figure (env.)` \* 形式は段抜きのフロートです。

```
figure* (env.) 3057 \newenvironment{figure}%
3058                 {\@float{figure}}%
3059                 {\end@float}
```

```

3060 \newenvironment{figure*}%
3061             {\@dblfloat{figure}}%
3062             {\end@dblfloat}

```

## ■table 環境

`\c@table` 表番号カウンタと表番号を出力するコマンドです。アスキー版では `\thechapter.` が `\thetable \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。

```

3063 %<!*book&!report>
3064 \newcounter{table}
3065 \renewcommand\thetable{\@arabic\c@table}
3066 %</!*book&!report>
3067 %<*book|report>
3068 \newcounter{table}[chapter]
3069 \renewcommand \thetable
3070         {\ifnum \c@chapter>\z@ \thechapter.\fi \@arabic\c@table}
3071 %</book|report>

```

`\fps@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。

```

\ext@table 3072 \def\fps@table{tbp}
3073 \def\ftype@table{2}
\fnun@table 3074 \def\ext@table{lot}
3075 \def\fnun@table{\tablename\nobreak\thetable}

```

`table (env.)` \* は段抜きのフロートです。

```

table* (env.) 3076 \newenvironment{table}%
3077             {\@float{table}}%
3078             {\end@float}
3079 \newenvironment{table*}%
3080             {\@dblfloat{table}}%
3081             {\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` それぞれキャプションの前後に挿入されるスペースです。`\belowcaptionskip` が 0 になっていたので、キャプションを表の上につけた場合にキャプションと表がくっついてしまうのを直しました。

```

3082 \newlength\abovecaptionskip
3083 \newlength\belowcaptionskip
3084 \setlength\abovecaptionskip{5\jsc@empt} % 元: 10\p@
3085 \setlength\belowcaptionskip{5\jsc@empt} % 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。

```
3086 %<!*jspf>
3087 % \long\def\@makecaption#1#2{\small
3088 %   \advance\leftskip10\jsc@mmm
3089 %   \advance\rightskip10\jsc@mmm
3090 %   \vskip\abovcaptionskip
3091 %   \sbox\@tempboxa{#1\hskip1\jsZw\relax #2}%
3092 %   \ifdim \wd\@tempboxa >\hsize
3093 %     #1\hskip1\jsZw\relax #2\par
3094 %   \else
3095 %     \global \@minipagefalse
3096 %     \hb@xt@\hsize{\hfil\box\@tempboxa\hfil}%
3097 %   \fi
3098 %   \vskip\belowcaptionskip}}
3099 \long\def\@makecaption#1#2{\small
3100   \advance\leftskip .0628\linewidth
3101   \advance\rightskip .0628\linewidth
3102   \vskip\abovcaptionskip
3103   \sbox\@tempboxa{#1\zwspace#2}%
3104   \ifdim \wd\@tempboxa <\hsize \centering \fi
3105   #1\zwspace#2\par
3106   \vskip\belowcaptionskip}}
3107 %</!jspf>
3108 %<*jspf>
3109 \long\def\@makecaption#1#2{%
3110   \vskip\abovcaptionskip
3111   \sbox\@tempboxa{\small\sffamily #1\quad #2}%
3112   \ifdim \wd\@tempboxa >\hsize
3113     {\small\sffamily
3114       \list{#1}{%
3115         \renewcommand{\makelabel}[1]{##1\hfil}
3116         \itemsep \z@
3117         \itemindent \z@
3118         \labelsep \z@
3119         \labelwidth 11\jsc@mmm
3120         \listparindent\z@
3121         \leftmargin 11\jsc@mmm}\item\relax #2\endlist}
3122   \else
3123     \global \@minipagefalse
3124     \hb@xt@\hsize{\hfil\box\@tempboxa\hfil}%
3125   \fi
3126   \vskip\belowcaptionskip}
3127 %</jspf>
```

## 9 フォントコマンド

ここでは L<sup>A</sup>T<sub>E</sub>X 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
\jsc@DeclareOldFontCommand
```

---

```
3128 \newif\if@jsc@warnoldfontcmd
3129 \@jsc@warnoldfontcmdtrue
3130 \newif\if@jsc@warnoldfontcmdexception
3131 \@jsc@warnoldfontcmdexceptionfalse

3132 \newcommand*{\jsc@DeclareOldFontCommand}[3]{%
3133   \g@addto@macro\bxjs@oldfontcmd@list{\do#1}%
3134   \DeclareOldFontCommand{#1}{%
3135     \bxjs@oldfontcmd{#1}#2%
3136   }{%
3137     \bxjs@oldfontcmd{#1}#3%
3138   }%
3139 }
3140 \DeclareRobustCommand*{\jsc@warnoldfontcmd}[1]{%
3141   \ClassInfo\bxjs@clsname
3142   {Old font command '\string#1' is used!!\MessageBreak
3143     The first occurrence is}%
3144 }
```

---

```
\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。

3145 \DeclareRobustCommand*{\allowoldfontcommands}{%
3146   \@jsc@warnoldfontcmdexceptiontrue}
3147 \DeclareRobustCommand*{\disallowoldfontcommands}{%
3148   \@jsc@warnoldfontcmdexceptionfalse}

3149 \let\bxjs@oldfontcmd@list\@empty
```

---

```

3150 \def\bxjs@oldfontcmd#1{%
3151   \expandafter\bxjs@oldfontcmd@a\csname bxjs@ofc/\string#1\endcsname#1}
3152 \def\bxjs@oldfontcmd@a#1#2{%
3153   \if@jsc@warnoldfontcmdexception\else
3154     \global\@jsc@warnoldfontcmdfalse
3155     \ifx#1\relax
3156       \global\let#1=t%
3157       \jsc@warnoldfontcmd{#2}%
3158     \fi
3159   \fi}
3160 \def\bxjs@warnoldfontcmd@final{%
3161 % \par
3162 \global\let\bxjs@warnoldfontcmd@final\@empty
3163 \let\@tempa\@empty
3164 \def\do##1{%
3165   \@ifundefined{bxjs@ofc/\string##1}{\}{%else
3166     \edef\@tempa{\@tempa \space\string##1}}}%
3167 \bxjs@oldfontcmd@list
3168 \ifx\@tempa\@empty\else
3169   \ClassWarningNoLine\bxjs@clsname
3170   {Some old font commands were used in text:\MessageBreak
3171     \space\@tempa\MessageBreak
3172     You should note, that since 1994 LaTeX2e provides a\MessageBreak
3173     new font selection scheme called NFSS2 with several\MessageBreak
3174     new, combinable font commands. The
3175     class provides\MessageBreak
3176     the old font commands only for compatibility}
3177   \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 L<sup>A</sup>T<sub>E</sub>X ではフックシステムの機能を利用する。

```

3178 \ifbxjs@old@hook@system
3179   \AtEndDocument{%
3180     \g@addto@macro\clearpage{\bxjs@warnoldfontcmd@final}}
3181 \else
3182   \AddToHook{enddocument/afterlastpage}{\bxjs@warnoldfontcmd@final}
3183 \fi

```

---

`\mc` フォントファミリーを変更します。

```

\gt 3184 \jsc@DeclareOldFontCommand{\mc}{\normalfont\mcfamily}{\mathmc}
3185 \jsc@DeclareOldFontCommand{\gt}{\normalfont\gtfamily}{\mathgt}
\rm 3186 \jsc@DeclareOldFontCommand{\rm}{\normalfont\rmfamily}{\mathrm}
\sff 3187 \jsc@DeclareOldFontCommand{\sf}{\normalfont\sffamily}{\mathsf}
\tt 3188 \jsc@DeclareOldFontCommand{\tt}{\normalfont\ttfamily}{\mathtt}

```

`\bf` ボールドシリーズにします。通常のミディアムシリーズに戻るコマンドは `\mdseries` です。

```
3189 \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るコマンドは `\upshape` `\sc` です。

```
3190 \jsc@DeclareOldFontCommand{\it}{\normalfont\itshape}{\mathit}
```

```
3191 \jsc@DeclareOldFontCommand{\sl}{\normalfont\slshape}{\@nomath\sl}
```

```
3192 \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 3193 \DeclareRobustCommand*{\cal}{\@fontswitch\relax\mathcal}
```

```
3194 \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。

```
3195 \newcommand\@pnumwidth{1.55em}
3196 \newcommand\@tocrmarg{2.55em}
3197 \newcommand\@dotsep{4.5}
3198 %<!book&!report>\setcounter{tocdepth}{2}
3199 %<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)

```
3200 \newdimen\jsc@tocl@width
3201 \newcommand{\tableofcontents}{%
3202 %<*book|report>
3203   \settowidth\jsc@tocl@width{\headfont\prechaptername\postchaptername}%
3204   \settowidth\@tempdima{\headfont\appendixname}%
3205   \ifdim\jsc@tocl@width<\@tempdima \setlength\jsc@tocl@width{\@tempdima}\fi
3206   \ifdim\jsc@tocl@width<2\jsZw \divide\jsc@tocl@width by 2 \advance\jsc@tocl@width 1\jsZw\fi
3207   \if@twocolumn
3208     \@restonecoltrue\onecolumn
3209   \else
3210     \@restonecolfalse
3211   \fi
3212   \chapter*{\contentsname}%
3213   \@mkboth{\contentsname}{}%
3214 %</book|report>
3215 %<!*book&!report>
3216   \settowidth\jsc@tocl@width{\headfont\presectionname\postsectionname}%
3217   \settowidth\@tempdima{\headfont\appendixname}%
3218   \ifdim\jsc@tocl@width<\@tempdima\relax\setlength\jsc@tocl@width{\@tempdima}\fi
3219   \ifdim\jsc@tocl@width<2\jsZw \divide\jsc@tocl@width by 2 \advance\jsc@tocl@width 1\jsZw\fi
3220   \section*{\contentsname}%
3221   \@mkboth{\contentsname}{\contentsname}%
3222 %</!book&!report>
3223   \@starttoc{toc}%
3224 %<book|report> \if@restonecol\twocolumn\fi
3225 }
```

`\l@part` 部の目次です。

```
3226 \newcommand*{\l@part}[2]{%
3227   \ifnum \c@tocdepth >-2\relax
```

```

3228 %<!book&!report> \addpenalty\@secpenalty
3229 %<book|report> \addpenalty{-\@highpenalty}%
3230 \addvspace{2.25em \@plus\jsc@empt}%
3231 \begingroup
3232 \parindent \z@
3233 % \@pnumwidth should be \@tocrmarg
3234 % \rightskip \@pnumwidth
3235 \rightskip \@tocrmarg
3236 \parfillskip -\rightskip
3237 {\leavevmode
3238 \large \headfont
3239 \setlength\@lnumwidth{4\jsZw}%
3240 #1\hfil \hb@xt@\@pnumwidth{\hss #2}}\par
3241 \nobreak
3242 %<book|report> \global\@nobreaktrue
3243 %<book|report> \everypar{\global\@nobreakfalse\everypar{}}%
3244 \endgroup
3245 \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)

```

3246 %<*book|report>
3247 \newcommand*{\l@chapter}[2]{%
3248 \ifnum \c@tocdepth >\m@ne
3249 \addpenalty{-\@highpenalty}%
3250 \addvspace{1.0em \@plus\jsc@empt}
3251 % \vskip 1.0em \@plus\p@ % book.cls では↑がこうなっている
3252 \begingroup
3253 \parindent\z@
3254 % \rightskip\@pnumwidth
3255 \rightskip\@tocrmarg
3256 \parfillskip-\rightskip
3257 \leavevmode\headfont
3258 % % \if@english\setlength\@lnumwidth{5.5em}\else\setlength\@lnumwidth{4.683\jsZw}\fi
3259 \setlength\@lnumwidth{\jsc@tocl@width}\advance\@lnumwidth 2.683\jsZw
3260 \advance\leftskip\@lnumwidth \hskip-\leftskip
3261 #1\nobreak\hfil\nobreak\hbox to\@pnumwidth{\hss#2}\par
3262 \penalty\@highpenalty
3263 \endgroup
3264 \fi}
3265 %</book|report>

```

\l@section 節の目次です。

```

3266 %<*!book&!report>
3267 \newcommand*{\l@section}[2]{%
3268 \ifnum \c@tocdepth >\z@
3269 \addpenalty{\@secpenalty}%
3270 \addvspace{1.0em \@plus\jsc@empt}%

```

```

3271 \begingroup
3272 \parindent\z@
3273 % \rightskip\@pnumwidth
3274 \rightskip\@tocrmarg
3275 \parfillskip-\rightskip
3276 \leavevmode\headfont
3277 % % \setlength\@lnumwidth{4\jsZw}% 元 1.5em [2003-03-02]
3278 \setlength\@lnumwidth{\jsc@tocl@width}\advance\@lnumwidth 2\jsZw
3279 \advance\leftskip\@lnumwidth \hskip-\leftskip
3280 #1\nobreak\hfil\nobreak\hbox to\@pnumwidth{\hss#2}\par
3281 \endgroup
3282 \fi}
3283 %<book&!report>

```

インデントと幅はそれぞれ 1.5em, 2.3em ですが, 1zw, 3.683zw に変えました。

```

3284 %<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@section 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  
 \l@subsection しれません。

\l@paragraph [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)

```

\l@subparagraph 3285 %<*!book&!report>
3286 % \newcommand*{\l@section} {\@dottedtocline{2}{1.5em}{2.3em}}
3287 % \newcommand*{\l@subsection}{\@dottedtocline{3}{3.8em}{3.2em}}
3288 % \newcommand*{\l@paragraph} {\@dottedtocline{4}{7.0em}{4.1em}}
3289 % \newcommand*{\l@subparagraph} {\@dottedtocline{5}{10em}{5em}}
3290 %
3291 % \newcommand*{\l@section} {\@dottedtocline{2}{1zw}{3zw}}
3292 % \newcommand*{\l@subsection}{\@dottedtocline{3}{2\jsZw}{3\jsZw}}
3293 % \newcommand*{\l@paragraph} {\@dottedtocline{4}{3\jsZw}{3\jsZw}}
3294 % \newcommand*{\l@subparagraph} {\@dottedtocline{5}{4\jsZw}{3\jsZw}}
3295 %
3296 \newcommand*{\l@section}{%
3297 \tempdima\jsc@tocl@width \advance\tempdima -1\jsZw
3298 \@dottedtocline{2}{\tempdima}{3\jsZw}}
3299 \newcommand*{\l@subsection}{%
3300 \tempdima\jsc@tocl@width \advance\tempdima 0\jsZw
3301 \@dottedtocline{3}{\tempdima}{4\jsZw}}
3302 \newcommand*{\l@paragraph}{%
3303 \tempdima\jsc@tocl@width \advance\tempdima 1\jsZw
3304 \@dottedtocline{4}{\tempdima}{5\jsZw}}
3305 \newcommand*{\l@subparagraph}{%
3306 \tempdima\jsc@tocl@width \advance\tempdima 2\jsZw
3307 \@dottedtocline{5}{\tempdima}{6\jsZw}}
3308 %<book&!report>
3309 %<*book|report>
3310 % \newcommand*{\l@section} {\@dottedtocline{2}{3.8em}{3.2em}}
3311 % \newcommand*{\l@subsection}{\@dottedtocline{3}{7.0em}{4.1em}}

```

```

3312 % \newcommand*{\l@paragraph} {\@dottedtocline{4}{10em}{5em}}
3313 % \newcommand*{\l@subparagraph} {\@dottedtocline{5}{12em}{6em}}
3314 \newcommand*{\l@section}{%
3315     \@tempdima\jsc@tocl@width \advance\@tempdima -1\jsZw
3316     \@dottedtocline{1}{\@tempdima}{3.683\jsZw}}
3317 \newcommand*{\l@subsection}{%
3318     \@tempdima\jsc@tocl@width \advance\@tempdima 2.683\jsZw
3319     \@dottedtocline{2}{\@tempdima}{3.5\jsZw}}
3320 \newcommand*{\l@subsubsection}{%
3321     \@tempdima\jsc@tocl@width \advance\@tempdima 6.183\jsZw
3322     \@dottedtocline{3}{\@tempdima}{4.5\jsZw}}
3323 \newcommand*{\l@paragraph}{%
3324     \@tempdima\jsc@tocl@width \advance\@tempdima 10.683\jsZw
3325     \@dottedtocline{4}{\@tempdima}{5.5\jsZw}}
3326 \newcommand*{\l@subparagraph}{%
3327     \@tempdima\jsc@tocl@width \advance\@tempdima 16.183\jsZw
3328     \@dottedtocline{5}{\@tempdima}{6.5\jsZw}}
3329 %</book|report>

```

`\numberline` 欧文版 L<sup>A</sup>T<sub>E</sub>X では `\numberline{...}` は幅 `\@tempdima` の箱に左詰めで出力する命令で  
`\@lnumwidth` 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。

```

3330 \newdimen\@lnumwidth
3331 \def\numberline#1{\hb@xt@\@lnumwidth{#1\hfil}\hspace{0pt}}

```

`\@dottedtocline` L<sup>A</sup>T<sub>E</sub>X 本体 (ltsect.dtx 参照) での定義と同じですが、`\@tempdima` を `\@lnumwidth` に  
`\jsTocLine` 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 \cdot\hss}\hfill}
```

とします。

```

3332 \def\jsTocLine{\leaders\hbox{%
3333     $\m@th \mkern \@dotsep mu\hbox{.}\mkern \@dotsep mu$\}\hfill}
3334 \def\@dottedtocline#1#2#3#4#5{\ifnum #1>\c@tocdepth \else
3335     \vskip \z@ \@plus.2\jsc@mpt
3336     {\leftskip #2\relax \rightskip \@tocrmarg \parfillskip -\rightskip
3337     \parindent #2\relax\@afterindenttrue
3338     \interlinepenalty\@M
3339     \leavevmode
3340     \@lnumwidth #3\relax
3341     \advance\leftskip \@lnumwidth \null\nobreak\hskip -\leftskip
3342     {#4}\nobreak
3343     \jsTocLine \nobreak\hb@xt@\@pnumwidth{%
3344         \hfil\normalfont \normalcolor #5}\par}\fi}

```

## ■ 図目次と表目次

`\listoffigures` 図目次を出力します。

```
3345 \newcommand{\listoffigures}{%
3346 %<*book|report>
3347 \if@twocolumn\@restonecoltrue\onecolumn
3348 \else\@restonecolfalse\fi
3349 \chapter*{\listfigurename}%
3350 \mkboth{\listfigurename}{}%
3351 %</book|report>
3352 %<!*book&!report>
3353 \section*{\listfigurename}%
3354 \mkboth{\listfigurename}{\listfigurename}%
3355 %</!book&!report>
3356 \@starttoc{lof}%
3357 %<book|report> \if@restonecol\twocolumn\fi
3358 }
```

`\l@figure` 図目次の項目を出力します。

```
3359 \newcommand*{\l@figure}{\@dottedtocline{1}{1\jsZw}{3.683\jsZw}}
```

`\listoftables` 表目次を出力します。

```
3360 \newcommand{\listoftables}{%
3361 %<*book|report>
3362 \if@twocolumn\@restonecoltrue\onecolumn
3363 \else\@restonecolfalse\fi
3364 \chapter*{\listtablename}%
3365 \mkboth{\listtablename}{}%
3366 %</book|report>
3367 %<!*book&!report>
3368 \section*{\listtablename}%
3369 \mkboth{\listtablename}{\listtablename}%
3370 %</!book&!report>
3371 \@starttoc{lot}%
3372 %<book|report> \if@restonecol\twocolumn\fi
3373 }
```

`\l@table` 表目次は図目次と同じです。

```
3374 \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。

```
3375 \newdimen\bibindent
3376 \setlength\bibindent{2\jsZw}
```

`thebibliography (env.)` 参考文献リストを出力します。

[2016-07-16] L<sup>A</sup>T<sub>E</sub>X 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。

```

3377 \newenvironment{thebibliography}[1]{%
3378   \@jsc@warnoldfontcmdexceptiontrue
3379   \global\let\presectionname\relax
3380   \global\let\postsectionname\relax
3381   %<article|slide> \section*{\refname}\mkboth{\refname}{\refname}%
3382   %<*kiyou>
3383   \vspace{1.5\baselineskip}
3384   \subsubsection*{\refname}\mkboth{\refname}{\refname}%
3385   \vspace{0.5\baselineskip}
3386   %</kiyou>
3387   %<book|report> \chapter*{\bibname}\mkboth{\bibname}{}%
3388   %<book|report> \addcontentsline{toc}{chapter}{\bibname}%
3389   \list{\@biblabel{\@arabic{c@enumiv}}}%
3390     {\settowidth\labelwidth{\@biblabel{#1}}%
3391      \leftmargin\labelwidth
3392      \advance\leftmargin\labelsep
3393      \@openbib@code
3394      \usecounter{enumiv}%
3395      \let\p@enumiv\@empty
3396      \renewcommand\theenumiv{\@arabic{c@enumiv}}}%
3397   %<kiyou> \small
3398   \sloppy
3399   \clubpenalty4000
3400   \@clubpenalty\clubpenalty
3401   \widowpenalty4000%
3402   \sfcode`\.\@m}
3403   {\def\@noitemerr
3404     {\@latex@warning{Empty `thebibliography' environment}}}%
3405   \endlist}

```

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

```

3406 \newcommand{\newblock}{\hspace .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。

```

3407 \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。

```

3408 % \def\@biblabel#1{\jsInhibitGlue [#1] \jsInhibitGlue}

```

\cite 文献の番号を出力する部分は ltbibl.dtx で定義されていますが、コンマとカッコを和文

\cite 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

```

\@citex

```

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` で取っていただきますので、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。

```

3409 % \def\@citex[#1]#2{\leavevmode
3410 %   \let\@citea\@empty
3411 %   \@cite{\@for\@citeb:=#2\do
3412 %     {\@citea\def\@citea{, \inhibitglue\penalty\@m\ }%
3413 %     \edef\@citeb{\expandafter\@firstofone\@citeb\@empty}}%
3414 %     \if@filesw\immediate\write\@auxout{\string\citation{\@citeb}}\fi
3415 %     \ifundefined{b@\@citeb}{\mbox{\normalfont\bfseries ?}}%
3416 %     \G@refundefinedtrue
3417 %     \@latex@warning
3418 %       {Citation `~\@citeb' on page \thepage \space undefined}}%
3419 %     {\@cite\@ofmt{\csname b@\@citeb\endcsname}}}{#1}}
3420 % \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。

```

3421 % \DeclareRobustCommand\cite{\unskip
3422 %   \@ifnextchar [{\@tempswatrue\@citex}{\@tempswafalse\@citex[]}]
3423 % \def\@cite#1#2{${\hbox{\scriptsize{#1\if@tempswa
3424 %   , \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)。

```

3425 \newenvironment{theindex}{% 索引を 3 段組で出力する環境
3426   \if@twocolumn
3427     \onecolumn\@restonecolfalse
3428   \else
3429     \clearpage\@restonecoltrue
3430   \fi
3431   \columnseprule.4pt \columnsep 2\jsZw
3432   \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。

---

```

3433 %<book|report>   \twocolumn[\bxjs@phantomsection
3434 %<book|report>   \addcontentsline{toc}{chapter}{\indexname}%
3435 %<book|report>   \@makeschapterhead{\indexname}}%
3436 %<!book&!report> \def\presectionname{}\def\postsectionname{}%
3437 %<!book&!report> \twocolumn[\section*{\indexname}}%
3438   \else
3439     \ifdim\textwidth<\fullwidth
3440       \setlength{\evensidemargin}{\oddsidemargin}

```

```

3441      \setlength{\textwidth}{\fullwidth}
3442      \setlength{\linewidth}{\fullwidth}
3443 %<book|report>      \begin{multicols}{3}[\chapter*{\indexname}]%
3444 %<book|report>      \addcontentsline{toc}{chapter}{\indexname}]%
3445 %<!book&!report>    \def\presectionname{}\def\postsectionname{}%
3446 %<!book&!report>    \begin{multicols}{3}[\section*{\indexname}]%
3447      \else
3448 %<book|report>      \begin{multicols}{2}[\chapter*{\indexname}]%
3449 %<book|report>      \addcontentsline{toc}{chapter}{\indexname}]%
3450 %<!book&!report>    \def\presectionname{}\def\postsectionname{}%
3451 %<!book&!report>    \begin{multicols}{2}[\section*{\indexname}]%
3452      \fi
3453      \fi
3454 %<book|report>      \@mkboth{\indexname}{}%
3455 %<!book&!report>    \@mkboth{\indexname}{\indexname}%
3456      \plainifnotempty % \thispagestyle{plain}
3457      \parindent\z@
3458      \parskip\z@ \@plus .3\jsc@mpt\relax
3459      \let\item\@idxitem
3460      \raggedright
3461      \footnotesize\narrowbaselines
3462    }{
3463      \ifx\multicols\@undefined
3464        \if@restonecol\onecolumn\fi
3465      \else
3466        \end{multicols}
3467      \fi
3468      \clearpage
3469    }

```

`\@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` 3470 `\newcommand{\@idxitem}{\par\hangindent 4\jsZw}` % 元 40pt

`\subsubitem` 3471 `\newcommand{\subitem}{\@idxitem \hspace*{2\jsZw}}` % 元 20pt

3472 `\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。

3473 `\newcommand{\indexspace}{\par \vskip 10\jsc@mpt \@plus5\jsc@mpt \@minus3\jsc@mpt\relax}`

`\seename` 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*

`\alsoname` 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。

3474 `\newcommand\seename{\if@english see\else  $\rightarrow$ \fi}`

3475 `\newcommand\alsoname{\if@english see also\else  $\rightarrow$ \fi}`

## 10.4 脚注

`\footnote` 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,

`\footnotemark` `\inhibitglue` を入れることにします。pL<sup>A</sup>T<sub>E</sub>X 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。

---

```
3476 \ifx\pltx@foot@penalty\undefined
3477   \let\footnotes@ve=\footnote
3478   \def\footnote{\jsInhibitGlue\footnotes@ve}
3479   \let\footnotemarks@ve=\footnotemark
3480   \def\footnotemark{\jsInhibitGlue\footnotemarks@ve}
3481 \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 p<sub>T</sub>E<sub>X</sub> 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 p<sub>L</sub>A<sub>T</sub>E<sub>X</sub> の変更に追従しました (Thanks: 角藤さん)。p<sub>L</sub>A<sub>T</sub>E<sub>X</sub> 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。

---

p<sub>T</sub>E<sub>X</sub> 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う。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)。

```
3482 \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。

```
3483 % \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。

```
3484 \renewcommand{\footnoterule}{%
3485   \kern-2.6\jsc@mpt \kern-.4\p@
3486   \hrule width .4\columnwidth
3487   \kern 2.6\jsc@mpt}
```

`\c@footnote` 脚注番号は章ごとにリセットされます。

```
3488 %<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, *T<sub>E</sub>X and TUG NEWS*, Vol. 2, No. 4 (1993), p. 9)

[2016-08-25] コミュニティ版 p<sub>A</sub>T<sub>E</sub>X 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 p<sub>A</sub>T<sub>E</sub>X のバグ修正に追随しました。

[2016-11-29] 古い p<sub>A</sub>T<sub>E</sub>X 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] L<sup>A</sup>T<sub>E</sub>X 2<sub>ε</sub> 2021-11-15 (lfloat.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、L<sup>A</sup>T<sub>E</sub>X 2<sub>ε</sub> 2021-06-01 (lfloat.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。

```
3489 \long\def\@footnotetext{%
3490   \insert\footins\bgroup
3491     \normalfont\footnotesize
3492     \interlinepenalty\interfootnotelinepenalty
3493     \splittopskip\footnotesep
3494     \splitmaxdepth \dp\strutbox \floatingpenalty \@MM
3495     \hsize\columnwidth \@parboxrestore
3496     \def\@currentcounter{footnote}%
3497     \protected@edef\@currentlabel{%
3498       \csname p@footnote\endcsname\@thefnmark
3499     }%
3500     \color@begingroup
3501     \@makefnmark{%
3502       \rule{z@footnotesep}{\ignorespaces}%
3503       \futurelet\jsc@next\jsc@fo@t}
3504 \def\jsc@fo@t{\ifcat\bgroup\noexpand\jsc@next \let\jsc@next\jsc@fo@t
3505               \else \let\jsc@next\jsc@fo@t\fi \jsc@next}
3506 \def\jsc@fo@t@prefix{\inhibitglue\ignorespaces}
3507 \def\jsc@fo@t{\bgroup\aftergroup\jsc@@foot\afterassignment\jsc@fo@t@prefix\let\jsc@next}
3508 \def\jsc@fo@t#1{\jsc@fo@t@prefix#1\jsc@@foot}
3509 \def\jsc@@foot{\@finalstrut\strutbox\color@endgroup\egroup
3510   \ifx\pltx@foot@penalty\@undefined\else
3511     \ifhmode\null\fi
3512     \ifnum\pltx@foot@penalty=z@\else
3513       \penalty\pltx@foot@penalty
3514       \pltx@foot@penalty\z@
3515     \fi
3516   \fi}
```

`\@makefnmark` 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。

```
3517 \newcommand\@makefntext[1]{%
3518   \advance\leftskip 3\jsZw
3519   \parindent 1\jsZw
3520   \noindent
3521   \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。

```
3522 % \def\@xfootnotenext[#1]{%
3523 %   \begingroup
3524 %     \ifnum#1>\z@
3525 %       \csname c@\@mpfn\endcsname #1\relax
3526 %       \unrestored@protected@xdef\@thefnmark{\thempfn}%
3527 %     \else
3528 %       \unrestored@protected@xdef\@thefnmark{}%
3529 %     \fi
3530 %   \endgroup
3531 %   \@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

段落頭のかぎカッコなどを見かけ 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。

```
3532 \let\jsInhibitGlueAtParTop\@empty
```

`\everyparhook` 全ての段落の冒頭で実行されるフック。この初期値を先述の `\jsInhibitGlueAtParTop` とする。

```
3533 \def\everyparhook{\jsInhibitGlueAtParTop}
3534 \ifnum\bxjs@everyparhook=\bxjs@everyparhook@@compat
3535 \g@addto@macro\bxjs@begin@document@hook{\everypar{\everyparhook}}
3536 \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。

---

```
3537 \ifx j\jsEngine
3538 \def\@inhibitglue{%
3539   \futurelet\@let@token\@@inhibitglue}
3540 \begingroup
3541 \let\GDEF=\gdef
3542 \let\CATCODE=\catcode
3543 \let\ENDGROUP=\endgroup
3544 \CATCODE`k=12
3545 \CATCODE`a=12
3546 \CATCODE`n=12
3547 \CATCODE`j=12
3548 \CATCODE`i=12
3549 \CATCODE`c=12
3550 \CATCODE`h=12
3551 \CATCODE`r=12
3552 \CATCODE`t=12
3553 \CATCODE`e=12
3554 \GDEF\KANJI@CHARACTER{kanji character }
3555 \ENDGROUP
3556 \def\@@inhibitglue{%
3557   \expandafter\expandafter\expandafter\jsc@inhibitglue\expandafter\meaning\expandafter\@let@
3558   \expandafter\def\expandafter\jsc@inhibitglue\expandafter#\expandafter1KANJI@CHARACTER#2#3\j
3559   \def\jsc@ig@temp{#1}%
3560   \ifx\jsc@ig@temp\@empty
3561     \ifnum\the\inhibitxspcode`#2=2\relax
3562       \inhibitglue
3563     \fi
3564   \fi}
3565 \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。

---

```
3566 \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 の変更に従って `\clubpenalty` のリセットを追加しました。

```
3567 \def\@doendpe{%
3568   \@endpetrue
3569   \def\par{%
3570     \@restorepar\clubpenalty\@clubpenalty\everypar{\everyparhook}\par\@endpefalse}%
3571   \everypar{\setbox\z@\lastbox}\everypar{\everyparhook}\@endpefalse\everyparhook}}
```

[2017-08-31] `minipage` 環境にも対策します。

```
3572 \def\@setminipage{%
3573   \@minipagetrue
3574   \everypar{\@minipagefalse\everypar{\everyparhook}}%
3575 }
```

`\item` 命令の直後です。

```
3576 \def\@item[#1]{%
3577   \if@nparitem
3578     \@donoparitem
3579   \else
3580     \if@inlabel
3581       \indent \par
3582     \fi
3583     \ifhmode
3584       \unskip\unskip \par
3585     \fi
3586     \if@newlist
3587       \if@nobreak
3588         \nbitem
3589       \else
3590         \addpenalty\@beginparpenalty
3591         \addvspace\@topsep
3592         \addvspace{-\parskip}%
3593       \fi
3594     \else
3595       \addpenalty\@itempenalty
3596       \addvspace\itemsep
3597     \fi
3598     \global\@inlabeltrue
3599   \fi
3600   \everypar{%
3601     \@minipagefalse
3602     \global\@newlistfalse
3603     \if@inlabel
3604       \global\@inlabelfalse
3605     {\setbox\z@\lastbox
```

```

3606     \ifvoid\z@
3607     \kern-\itemindent
3608     \fi}%
3609     \box\@labels
3610     \penalty\z@
3611 \fi
3612 \if@nobreak
3613     \@nobreakfalse
3614     \clubpenalty \@M
3615 \else
3616     \clubpenalty \@clubpenalty
3617     \everypar{\everyparhook}%
3618 \fi\everyparhook}%
3619 \if@noitemarg
3620     \@noitemargfalse
3621 \if@nmbrlist
3622     \refstepcounter\@listctr
3623 \fi
3624 \fi
3625 \sbox\@tempboxa{\makelabel{#1}}%
3626 \global\setbox\@labels\hbox{%
3627     \unhbox\@labels
3628     \hskip \itemindent
3629     \hskip -\labelwidth
3630     \hskip -\labelsep
3631     \ifdim \wd\@tempboxa >\labelwidth
3632         \box\@tempboxa
3633     \else
3634         \hbox to\labelwidth {\unhbox\@tempboxa}%
3635     \fi
3636     \hskip \labelsep}%
3637 \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。

```

3638 \def\@afterheading{%
3639     \@nobreaktrue
3640     \everypar{%
3641         \if@nobreak
3642             \@nobreakfalse
3643             \clubpenalty \@M
3644             \if@afterindent \else
3645                 {\setbox\z@\lastbox}%
3646             \fi
3647         \else
3648             \clubpenalty \@clubpenalty
3649             \everypar{\everyparhook}%
3650         \fi\everyparhook}}

```

---

「\everyparhook 用のパッチ処理」はここまで。

---

3651 \fi

\@gnewline についてはちょっと複雑な心境です。もともとの p $\text{\LaTeX}$  2<sub>ε</sub> 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。

---

```
3652 \ifnum\bxjs@everyparhook=\bxjs@everyparhook@@none\else
3653 \AtEndOfClass{%
3654 \def\@gnewline #1{%
3655   \ifvmode
3656     \@nolnerr
3657   \else
3658     \unskip \reserved@e {\reserved@f#1}\nobreak \hfil \break \null
3659     \jsInhibitGlue \ignorespaces
3660   \fi}
3661 }
3662 \fi
```

## 12 いろいろなロゴ

$\text{\LaTeX}$  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。

---

```
3663 \if@jslogo
3664   \IfFileExists{jslogo.sty}{%
3665     \RequirePackage{jslogo}%
3666   }{%
3667     \ClassWarningNoLine\bxjs@clsname
3668     {The package 'jslogo' is not installed.\MessageBreak
3669     It is included in the recent release of\MessageBreak
```

```

3670         the 'jsclasses' bundle}
3671     }
3672 \fi

```

## 13 amsmath との衝突の回避

`\ltx@ifnextchar` `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 L<sup>A</sup>T<sub>E</sub>X の `\ProvidesFile` `\ProvidesFile` で悪さをする例が F<sub>T</sub>E<sub>X</sub> 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: しっぱ愛好家さん)。

[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。

```

3673 %\let\ltx@ifnextchar\@ifnextchar
3674 %\def\ProvidesFile#1{%
3675 %   \begingroup
3676 %     \catcode`\ 10 %
3677 %     \ifnum \endlinechar<256 %
3678 %       \ifnum \endlinechar>\m@ne
3679 %         \catcode\endlinechar 10 %
3680 %     \fi
3681 %   \fi
3682 %   \@makeother\/%
3683 %   \@makeother\&%
3684 %   \ltx@ifnextchar[{\@providesfile{#1}}{\@providesfile{#1}[]}]

```

## 14 初期設定

### ■いろいろな語

```

\prepartname
\postpartname 3685 \newcommand{\prepartname}{\if@english Part~\else 第\fi}
\prechaptername 3686 \newcommand{\postpartname}{\if@english\else 部\fi}
\postchaptername 3687 %<book|report>\newcommand{\prechaptername}{\if@english Chapter~\else 第\fi}
\postchaptername 3688 %<book|report>\newcommand{\postchaptername}{\if@english\else 章\fi}
\presectionname 3689 \newcommand{\presectionname}{}% 第
\postsectionname 3690 \newcommand{\postsectionname}{}% 節

\contentsname

\listfigurename 3691 \newcommand{\contentsname}{\if@english Contents\else 目次\fi}
\listtablename 3692 \newcommand{\listfigurename}{\if@english List of Figures\else 図目次\fi}
\listtablename 3693 \newcommand{\listtablename}{\if@english List of Tables\else 表目次\fi}

\refname
\bibname
\indexname

```



```

3694 \newcommand{\refname}{\if@english References\else 参考文献\fi}
3695 \newcommand{\bibname}{\if@english Bibliography\else 参考文献\fi}
3696 \newcommand{\indexname}{\if@english Index\else 索引\fi}

\figurename

\tablename 3697 %<!jspf>\newcommand{\figurename}{\if@english Fig.~\else 図\fi}
3698 %<jspf>\newcommand{\figurename}{Fig.~}
3699 %<!jspf>\newcommand{\tablename}{\if@english Table~\else 表\fi}
3700 %<jspf>\newcommand{\tablename}{Table~}

\appendixname

\abstractname 3701 % \newcommand{\appendixname}{\if@english Appendix~\else 付録\fi}
3702 \newcommand{\appendixname}{\if@english \else 付録\fi}
3703 %<!book>\newcommand{\abstractname}{\if@english Abstract\else 概要\fi}

```

■今日の日付 L<sup>A</sup>T<sub>E</sub>X 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` 8 ビット欧文 T<sub>E</sub>X ではそもそも非 ASCII の制御綴は使えないのであるが、JS クラスのユーザー `\和暦` 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 T<sub>E</sub>X 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 つのユーザー命令以外の非 ASCII の制御綴は使わないようにする。

T<sub>E</sub>X エンジンの種類により処理を分ける。

```

3704 \@onlypreamble\bxjs@decl@Seireki@cmds
3705 \@tempswafalse
3706 \if p\jsEngine \@tempwattrue \fi
3707 \if n\jsEngine \@tempwattrue \fi
3708 \bxjs@cond\if@tempswa\fi{%

```

8 ビット欧文 T<sub>E</sub>X 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。

```

3709 \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。

```

3710 \def\bxjs@decl@Seireki@cmds{%

```

```

3711 \def\西暦{\jsSeirekitrue}%
3712 \def\和暦{\jsSeirekifalse\bxjs@wareki@used}}

```

\Seireki \西暦/\和暦 の代わりにになる ASCII 名の命令も（念のため）用意しておく。

```

\Wareki 3713 \def\Seireki{\jsSeirekitrue}
3714 \def\Wareki{\jsSeirekifalse\bxjs@wareki@used}

3715 \def\bxjs@if@use@seireki{\bxjs@cond\ifjsSeireki\fi}
3716 \def\bxjs@iaif{\noexpand~}
3717 }{%

```

8 ビット欧文 T<sub>E</sub>X 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。

```

3718 \newif\if 西暦 \西暦 true
3719 \def\bxjs@decl@Seireki@cmds{%
3720 \def\西暦{\西暦 true}%
3721 \def\和暦{\西暦 false\bxjs@wareki@used}}
3722 \def\Seireki{\西暦 true}
3723 \def\Wareki{\西暦 false\bxjs@wareki@used}
3724 \def\bxjs@if@use@seireki{\bxjs@cond\if 西暦\fi}
3725 \let\bxjs@iaif\@empty
3726 }
3727 \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。

```

3728 \newif\ifbxjs@bxwareki@avail
3729 \IfFileExists{bxwareki.sty}{%
3730 \if \if n\jsEngine \ifjsWitheTeX T\else F\fi\else T\fi T%
3731 \RequirePackage{bxwareki}[2018/04/08]v0.2
3732 \bxjs@bxwareki@availtrue
3733 \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。

```

3734 \ifbxjs@bxwareki@avail \let\bxjs@wareki@used\@empty
3735 \else
3736 \bxjs@robust@def\bxjs@wareki@used{%
3737 \global\let\bxjs@wareki@used\@empty
3738 \ClassWarning\bxjs@clsname
3739 {Wareki mode is not supported, since\MessageBreak
3740 'bxwareki' is unavailable, reported}}
3741 \g@addto@macro\bxjs@begin@document@hook{%
3742 \let\bxjs@wareki@used\@empty}
3743 \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。

```
3744 \ifbxjs@bxwareki@avail
3745   \let\jyear\warekiyear
3746   \def\bxjs@tmpa{H}\ifx\bxjs@tmpa\warekigengoinitial
3747     \newcount\heisei \heisei=\value{warekiyear}
3748   \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。

```
3749 \else
3750   \edef\jyear{\the\year \bxjs@iai}
3751 \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。

```
3752 %<jsclasses>
3753 \newif\if 西暦 \西暦 true
3754 \def\西暦{\西暦 true}
3755 \def\和暦{\西暦 false}
3756 \newcount\heisei \heisei\year \advance\heisei-1988\relax
3757 \def\pltx@today@year@#1{%
3758   \ifnum\numexpr\year-#1=1 元\else
3759     \ifnum1=\iftdir\ifmdir0\else1\fi\else0\fi
3760     \kansuji\numexpr\year-#1\relax
3761   \else
3762     \number\numexpr\year-#1\relax\nobreak
3763   \fi
3764 \fi 年
3765 }
3766 \def\pltx@today@year{%
3767   \ifnum\numexpr\year*10000+\month*100+\day<19890108
3768     昭和\pltx@today@year@{1925}%
3769   \else\ifnum\numexpr\year*10000+\month*100+\day<20190501
3770     平成\pltx@today@year@{1988}%
3771   \else
3772     令和\pltx@today@year@{2018}%
3773   \fi\fi}
3774 %</jsclasses>
3775 \begingroup
3776 \let\bxjs@next\relax
3777 \ifbxjs@bxwareki@avail \ifx\warekigengo\@empty\else
3778   \def\bxjs@next{\warekitoday}
3779   \bxjs@test@engine\unexpanded{%
3780     \def\bxjs@next{\unexpanded\expandafter{\warekitoday}}
3781   \fi\fi
3782 \def\!#1#2#3{\noexpand#1\noexpand#2\noexpand#3}
3783 \ifx\bxjs@iai\@empty \let\!\@empty \fi
3784 \xdef\bxjs@today{%
```

```

3785 \if@english
3786 \ifcase\month\or
3787     January\or February\or March\or April\or May\or June\or
3788     July\or August\or September\or October\or November\or December\fi
3789     \space\number\day, \number\year
3790 \else
3791     \ifx\bxjs@next\relax \expandafter\@firstoftwo
3792     \else \noexpand\bxjs@if@use@seireki
3793     \fi {%
3794         \number\year\bxjs@iai\!年 %
3795         \bxjs@iai\number\month\bxjs@iai\!月 %
3796         \bxjs@iai\number\day\bxjs@iai\!日 %
3797     }\bxjs@next}%
3798 \fi}
3799 \endgroup
3800 \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。

```

3801 \g@addto@macro\bxjs@begin@document@hook{%
3802     \ifx\bb1\jpn@maybekansuji\@undefined\else
3803         \bxjs@decl@Seireki@cmds
3804         \g@addto@macro\datejapanese{%
3805             \let\today\bxjs@today}%
3806     \fi}

```

---

■ハイフネーション例外  $\mathrm{T}_{\mathrm{E}}\mathrm{X}$  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)

```

3807 \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。

```

3808 %<slide>\pagestyle{empty}%
3809 %<article|report>\pagestyle{plain}%
3810 %<book>\pagestyle{headings}%
3811 \pagenumbering{arabic}
3812 \if@twocolumn
3813     \twocolumn
3814     \sloppy
3815     \flushbottom
3816 \else
3817     \onecolumn
3818     \raggedbottom
3819 \fi
3820 %<*slide>

```

```

3821 \renewcommand\familydefault{\sfdefault}
3822 \raggedright
3823 %</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 pL<sup>A</sup>T<sub>E</sub>X で新設されたテスト用パッケージ (expp12e 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。expp12e パッケージが読み込まれていない場合は通常版として動作しますので、ここで終了します。

以上です。

## 16 BXJS 独自の追加処理

■**\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。

```

3824 \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。

```

3825 \chardef\bxjs@strong@level=0
3826 \DeclareRobustCommand*\jsStrongDeclare[1]{%
3827   \bxjs@set@array@from@clist{\bxjs@strong}{#1}%
3828   \chardef\bxjs@strong@level\z@}
3829 \jsStrongDeclare{\bfseries}
3830 \def\bxjs@strong@text{%
3831   \bxjs@csletcs{\bxjs@tmpa}{\bxjs@strong/\the\bxjs@strong@level}%
3832   \ifx\bxjs@tmpa\relax
3833     \bxjs@advance@qc\bxjs@strong@level\m@ne \bxjs@strong@text
3834   \else \bxjs@advance@qc\bxjs@strong@level\@ne \bxjs@tmpa
3835   \fi}
3836 % \end{macro}
3837 %
3838 % \paragraph{共通命令の実装}

```

```

3839 %
3840 % |\jQ| 等の「単位」系の共通命令を実装する。
3841 %
3842 % \begin{macro}{\bxjs@const@unit}
3843 % 固定値の単位として使える制御綴を定義する。
3844 %
3845 % {\eTeX}拡張が使える場合は、
3846 % 「|\dimexpr|外部寸法表記|\relax|」の形式
3847 % (これは内部値なので単位として使える)
3848 % に展開されるマクロとして定義する。
3849 % \begin{macrocode}
3850 \@onlypreamble\bxjs@const@unit
3851 \@onlypreamble\bxjs@const@unit@a
3852 \ifjsWitheTeX
3853 \def\bxjs@const@unit#1#2#3{%
3854 \protected\edef#1{\dimexpr\the\dimexpr#3\relax\relax}}

```

$\epsilon$ -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)。

```

3855 \else
3856 \let\bxjs@Ct\fontdimen \font\bxjs@Ut=cmtex9 at 0.98245pt
3857 \bxjs@Ct8\bxjs@Ut=8sp \bxjs@Ct16\bxjs@Ut=\z@
3858 \def\bxjs@const@unit#1#2{%
3859 \expandafter\bxjs@const@unit@a\csname bxjs@#2@\endcsname#1}
3860 \def\bxjs@const@unit@a#1#2#3{%
3861 \chardef#1\bxjs@Ct8\bxjs@Ut \bxjs@advance@qc#1\@ne \bxjs@Ct8\bxjs@Ut#1sp
3862 \bxjs@Ct#1\bxjs@Ut=#3\relax \def#2{\bxjs@Ct#1\bxjs@Ut}}
3863 \fi

```

`\jQ` `\jQ` と `\jH` はともに 0.25 mm に等しい。

```

\jH 3864 \bxjs@const@unit\jQ{\jQ}{0.25mm}
3865 \let\jH\jQ

```

`\trueQ` `\trueQ` と `\trueH` はともに 0.25 true mm に等しい。

```

\trueH 3866 \ifjsc@mag
3867 \@tempdimb=\jsBaseFontSize\relax
3868 \edef\bxjs@tmpa{\strip@pt\@tempdimb}%
3869 \@tempdima=2.5mm
3870 \bxjs@invscale\@tempdima\bxjs@tmpa
3871 \bxjs@const@unit\trueQ{\trueQ}{\@tempdima}
3872 \@tempdima=10pt
3873 \bxjs@invscale\@tempdima\bxjs@tmpa
3874 \bxjs@const@unit\bxjs@truept{\truept}{\@tempdima}
3875 \else \let\trueQ\jQ \let\bxjs@truept\p@
3876 \fi
3877 \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}`

`\ascpt`

とすると、和文が 12Q になる。

同様に、`\ascpt` は `truept` を和文スケールで割った値。

```
3878 \@tempdima\trueQ \bxjs@invscale\@tempdima\jsScale
3879 \bxjs@const@unit\ascQ{ascQ}{\@tempdima}
3880 \@tempdima\bxjs@truept \bxjs@invscale\@tempdima\jsScale
3881 \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、1zw 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。

```
3882 \DeclareRobustCommand*\jsJaFontSize[2]{%
3883   \begingroup
3884     \bxjs@jafontsize@a{#1}%
3885     \@tempdimb\jsInverseScale\@tempdima
3886     \bxjs@jafontsize@a{#2}%
3887     \xdef\bxjs@g@tmpa{%
3888       \noexpand\fontsize{\the\@tempdimb}{\the\@tempdima}}%
3889   \endgroup\bxjs@g@tmpa}
3890 \def\bxjs@jafontsize@a#1{%
3891   \bxjs@parse@qh{#1}%
3892   \ifx\bxjs@tmpb\relax \def\bxjs@tmpb{#1}\fi
3893   \@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。

```
3894 \def\bxjs@let@lenexpr{\edef}
```

`\bxjs@kanjiskip` 和文間空白の量を表すテキスト。

```
3895 \def\bxjs@kanjiskip{0pt}
```

`\jsSetKanjiSkip` 和文間空白の量を設定する。

※`\setkanjiskip` の実体。

```
3896 \DeclareRobustCommand*\jsSetKanjiSkip[1]{%
3897   \bxjs@let@lenexpr\bxjs@kanjiskip{#1}%
3898   \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。

```
3899 \newcommand*\jsGetKanjiSkip{%
3900   \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。

```
3901 \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`

※ p<sub>T</sub>E<sub>X</sub> 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。

```
3902 \bxjs@robust@def\jsEnableKanjiSkip{%
3903   \bxjs@kanjiskip@enabledtrue
3904   \bxjs@reset@kanjiskip}
3905 \bxjs@robust@def\jsDisableKanjiSkip{%
3906   \bxjs@kanjiskip@enabledfalse
3907   \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。

```
3908 \bxjs@robust@def\bxjs@reset@kanjiskip{%
3909   \ifbxjs@kanjiskip@enabled
3910     \setlength{\@tempskipa}{\bxjs@kanjiskip}%
3911   \else \@tempskipa\z@
3912   \fi
3913   \jsApplyKanjiSkip\@tempskipa}
```

`\jsApplyKanjiSkip` `\jsApplyKanjiSkip{⟨グループ値⟩}`： 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。

```
3914 \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 3915 \def\bxjs@xkanjiskip{0pt}
\jsGetXKanjiSkip 3916 \DeclareRobustCommand*\jsSetXKanjiSkip[1]{%
\ifbxjs@xkanjiskip@enabled 3917   \bxjs@let@lenexpr\bxjs@xkanjiskip{#1}%
3918   \bxjs@reset@xkanjiskip}
\jsEnableXKanjiSkip 3919 \newcommand*\jsGetXKanjiSkip{%
\jsDisableXKanjiSkip 3920   \bxjs@xkanjiskip}
3921 \newif\ifbxjs@xkanjiskip@enabled \bxjs@xkanjiskip@enabledtrue
\bxjs@reset@xkanjiskip 3922 \bxjs@robust@def\jsEnableXKanjiSkip{%
\jsApplyXKanjiSkip 3923   \bxjs@xkanjiskip@enabledtrue
3924   \bxjs@reset@xkanjiskip}
3925 \bxjs@robust@def\jsDisableXKanjiSkip{%
3926   \bxjs@xkanjiskip@enabledfalse
3927   \bxjs@reset@xkanjiskip}
3928 \bxjs@robust@def\bxjs@reset@xkanjiskip{%
3929   \ifbxjs@xkanjiskip@enabled
3930     \setlength{\@tempskipa}{\bxjs@xkanjiskip}%
3931   \else \@tempskipa\z@
3932   \fi
3933   \jsApplyXKanjiSkip\@tempskipa}
3934 \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従するようにする。

```
3935 \g@addto@macro\jsResetDimen{%
3936   \bxjs@reset@kanjiskip
3937   \bxjs@reset@xkanjiskip}
```



和文・和欧文間空白の初期値。

```
3938 \AtEndOfPackage{%
3939 \jsSetKanjiSkip{0pt plus.1\jsZw minus.01\jsZw}%
3940 \ifx\jsDocClass\jsSlide \jsSetXKanjiSkip{0.1em}%
3941 \else \jsSetXKanjiSkip{0.25em plus 0.15em minus 0.06em}%
3942 \fi
3943 }
```

## ■和文空白命令

```
3944 \ifbxjs@jaspace@cmd
```

`\jaenspace` 半角幅の水平空き。

```
3945 \def\jaenspace{\hskip.5\jsZw\relax}
```

`\jathinspace` 和欧文間空白を入れるユーザ命令。

```
3946 \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。

```
3947 \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)。

```
3948 \bxjs@protected\def\bxjs@choice@jathinspace{%
3949 \relax\ifmmode \mskip\medmuskip
3950 \else \jathinspace\ignorespaces
3951 \fi}
3952 \jsAtEndOfClass{%
3953 \ifjsWitheTeX \let\>\bxjs@choice@jathinspace
3954 \else \def\>{\protect\bxjs@choice@jathinspace}%
3955 \fi}
```

`\jaspace` `jlreq` クラスと互換の命令。

```
3956 \DeclareRobustCommand*\jaspace[1]{%
3957 \expandafter\ifx\csname bxjs@jaspace@@#1\endcsname\relax
3958 \ClassError\bxjs@clsname
3959 {Unknown jaspac: #1}{\@eha}%
3960 \else
3961 \csname bxjs@jaspace@@#1\endcsname
3962 \fi}
3963 \def\bxjs@jaspace@@zenkaku{\hskip 1\jsZw\relax}
3964 \def\bxjs@jaspace@@nibu{\hskip .5\jsZw\relax}
3965 \def\bxjs@jaspace@@shibu{\hskip .25\jsZw\relax}
```

終わり。

```
3966 \fi
```

■和文ドライバ読み フックを実行する。

```
3967 \bxjs@pre@jadriver@hook
```

和文ドライバのファイルを読み込む。

```
3968 \input{bxjsja-\bxjs@jadriver.def}
```

おしまい。

```
3969 %</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` `XYTeX`
  - `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  $\epsilon$ -`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おらず、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。

```
3970 %<*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-jā 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

```
3971 %<*minimal>
```

```
3972 %% 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 jIEX と Omega 系。

```
3973 \let\bxjs@tmpa\relax
3974 \ifx J\jsEngine \def\bxjs@tmpa{NTT-jTeX}\fi
3975 \ifx O\jsEngine \def\bxjs@tmpa{Omega}\fi
3976 \ifx\bxjs@tmpa\relax \expandafter\@gobble
3977 \else
3978   \ClassError\bxjs@clsname
3979   {The engine in use (\bxjs@tmpa) is not supported}
```

```

3980    {It's a fatal error. I'll quit right now.}
3981    \expandafter\@firstofone
3982 \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`。

```

3983 \def\DeclareJaTextFontCommand#1#2{%
3984   \DeclareRobustCommand#1[1]{%
3985     \relax
3986     \ifmmode \expandafter\nfss@text \fi
3987     {#2##1}}%
3988 }

```

`\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。

```

3989 \def\DeclareJaMathFontCommand#1#2{%
3990   \DeclareRobustCommand#1[1]{%
3991     \relax
3992     \ifmmode\else \non@alpherr{#1\space}\fi
3993     \nfss@text{\fontfamily\familydefault
3994               \fontseries{m}\fontshape{n}\selectfont\relax
3995               #2##1}%
3996   }%
3997 }

```

`\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。

```

3998 \long\def\bxjs@@CSsfdefault{\sfdefault}%
3999 \onlypreamble\bxjs@if@sf@default
4000 \def\bxjs@if@sf@default#1{%
4001   \ifx\familydefault\bxjs@@CSsfdefault#1\fi
4002   \g@addto@macro\bxjs@begin@document@hook{%
4003     \ifx\familydefault\bxjs@@CSsfdefault#1\fi}%
4004 }

```

`\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。

```

4005 \@tempdima\p@ \bxjs@invscale\@tempdima\jsScale
4006 \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。

```

4007 \def\jsLetHeadChar#1#2{%
4008   \begingroup
4009     \escapechar=`\ %
4010     \let\bxjs@tmpa={% brace-match-hack
4011       \bxjs@let@hchar@exp#2}%
4012   \endgroup
4013   \let#1\bxjs@g@tmpa}
4014 \def\bxjs@let@hchar@exp{%
4015   \futurelet\@let@token\bxjs@let@hchar@exp@a}
4016 \def\bxjs@let@hchar@exp@a{%
4017   \bxjs@cond@ifcat\noexpand\@let@token\bgroup\fi{% 波括弧
4018     \bxjs@let@hchar@out\let\relax
4019   }{\bxjs@cond@ifcat\noexpand\@let@token\@sptoken\fi{% 空白
4020     \bxjs@let@hchar@out\let\space%
4021   }{\bxjs@cond@if\noexpand\@let@token\@backslashchar\fi{% バックスラッシュ
4022     \bxjs@let@hchar@out\let\@backslashchar
4023   }{\bxjs@let@hchar@exp@b}}}}
4024 \def\bxjs@let@hchar@exp@b#1{%
4025   \expandafter\bxjs@let@hchar@exp@c\string#1?\@nil#1}
4026 \def\bxjs@let@hchar@exp@c#1#2\@nil{%
4027   %\message{<#1#2>}%
4028   \bxjs@cond@if#1\@backslashchar\fi{% 制御綴
4029     \bxjs@cond\expandafter\ifx\noexpand\@let@token\@let@token\fi{%
4030       \bxjs@let@hchar@out\let\relax
4031     }{%else
4032       \expandafter\bxjs@let@hchar@exp
4033     }%
4034   }{%else
4035     \bxjs@let@hchar@chr#1%
4036   }}
4037 \def\bxjs@let@hchar@chr#1{%
4038   \bxjs@let@hchar@out\def{{#1}}}
4039 \def\bxjs@let@hchar@out#1#2{%
4040   \global#1\bxjs@g@tmpa#2\relax
4041   \toks@\bgroup}% skip to right brace

```

UTF-8 のバイト列を扱うコード。

```

4042 \chardef\bxjs@let@hchar@csta=128
4043 \chardef\bxjs@let@hchar@cstb=192
4044 \chardef\bxjs@let@hchar@cstc=224
4045 \chardef\bxjs@let@hchar@cstd=240
4046 \chardef\bxjs@let@hchar@cste=248
4047 \let\bxjs@let@hchar@chr@ue@a\bxjs@let@hchar@chr
4048 \def\bxjs@let@hchar@chr@ue#1{%
4049   \@tempcnta=`#1\relax
4050   %\message{\the\@tempcnta}%
4051   \bxjs@cond@ifnum\@tempcnta<\bxjs@let@hchar@csta\fi{%
4052     \bxjs@let@hchar@chr@ue@a#1%
4053   }{\bxjs@cond@ifnum\@tempcnta<\bxjs@let@hchar@cstb\fi{%

```

```

4054 \bxjs@let@hchar@out\let\relax
4055 }{\bxjs@cond@ifnum\@tempcnta<\bxjs@let@hchar@cstc\fi{%
4056 \bxjs@let@hchar@chr@ue@b
4057 }{\bxjs@cond@ifnum\@tempcnta<\bxjs@let@hchar@cstd\fi{%
4058 \bxjs@let@hchar@chr@ue@c
4059 }{\bxjs@cond@ifnum\@tempcnta<\bxjs@let@hchar@cste\fi{%
4060 \bxjs@let@hchar@chr@ue@d
4061 }{%else
4062 \bxjs@let@hchar@out\let\relax
4063 }}}}
4064 \def\bxjs@let@hchar@chr@ue@a#1{%
4065 \bxjs@let@hchar@out\def{{#1}}}
4066 \def\bxjs@let@hchar@chr@ue@b#1#2{%
4067 \bxjs@let@hchar@out\def{{#1#2}}}
4068 \def\bxjs@let@hchar@chr@ue@c#1#2#3{%
4069 \bxjs@let@hchar@out\def{{#1#2#3}}}
4070 \def\bxjs@let@hchar@chr@ue@d#1#2#3#4{%
4071 \bxjs@let@hchar@out\def{{#1#2#3#4}}}

```

## B.2 (u)pTeX 用の設定

```
4072 \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。

```

4073 \def\bxjs@let@hchar@chr@pp#1#2{%
4074 \expandafter\bxjs@let@hchar@chr@pp@a\meaning#2\relax#1#2}
4075 \def\bxjs@let@hchar@chr@pp@a#1#2\relax#3#4{%
4076 %\message{(\meaning#3:\meaning#4)}%
4077 \bxjs@cond@if#1k\fi{%
4078 \bxjs@let@hchar@out\def{{#4}}}%
4079 }{%else
4080 \bxjs@let@hchar@chr@ue#3#4%
4081 }}
4082 \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。

```

4083 \edef\jsc@JYn{\ifjsWithupTeX JY2\else JY1\fi}
4084 \edef\jsc@JTn{\ifjsWithupTeX JT2\else JT1\fi}
4085 \edef\jsc@pfx0{\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。

```
4086 \@onlypreamble\bxjs@declarefontshape
4087 \ifjsWithupTeX
4088 \def\bxjs@declarefontshape{%
4089 \DeclareFontShape{JY2}{mc}{m}{n}{<->s*[\bxjs@scale]upjpnrm-h}{}%
4090 \DeclareFontShape{JY2}{gt}{m}{n}{<->s*[\bxjs@scale]upjpngt-h}{}%
4091 \DeclareFontShape{JT2}{mc}{m}{n}{<->s*[\bxjs@scale]upjpnrm-v}{}%
4092 \DeclareFontShape{JT2}{gt}{m}{n}{<->s*[\bxjs@scale]upjpngt-v}{}%
4093 }
4094 \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。

```
4095 \else
4096 \def\bxjs@declarefontshape{%
4097 \DeclareFontShape{JY1}{mc}{m}{n}{<->s*[\bxjs@scale]jis}{}%
4098 \DeclareFontShape{JY1}{gt}{m}{n}{<->s*[\bxjs@scale]jisg}{}%
4099 \DeclareFontShape{JT1}{mc}{m}{n}{<->s*[\bxjs@scale]tmin10}{}%
4100 \DeclareFontShape{JT1}{gt}{m}{n}{<->s*[\bxjs@scale]tgoth10}{}%
4101 }
4102 \def\bxjs@sizereference{jis}
4103 \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。

```
4104 \def\bxjs@next#1/#2/#3/#4/#5\relax{%
4105   \def\bxjs@tmpb{#5}}
4106 \ifjsWithpTeXng \def\bxjs@tmpb{10}%
4107 \else
4108 \expandafter\expandafter\expandafter\bxjs@next
4109 \expandafter\string\the\jfont\relax
4110 \fi
4111 \@for\bxjs@tmpa:={\jsc@JYn/mc/m/n,\jsc@JYn/gt/m/n,%
4112   \jsc@JTn/mc/m/n,\jsc@JTn/gt/m/n}\do
4113   {\expandafter\let\csname\bxjs@tmpa/10\endcsname=\@undefined
4114   \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 `1zw` の大きさが指定されたサイズではなく既にスケール（この値を  $f$  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。

```
4115 \begingroup
4116 % 参照用フォント (\bxjs@sizereference) の全角空白の幅を取得
4117 \font\bxjs@tmpa=\bxjs@sizereference\space at 10pt
```



```

4118 \setbox\z@\hbox{\bxjs@tmpa\char\jis"2121\relax}
4119 % 幅が丁度 10pt なら補正は不要
4120 \ifdim\wd\z@=10pt
4121 \global\let\bxjs@scale\jsScale
4122 \else
4123 % (10*s)/(10*f) として計算、\bxjs@invscale は BXJS で定義
4124 \edef\bxjs@tmpa{\strip@pt\wd\z@}
4125 \@tempdima=10pt \@tempdima=\jsScale\@tempdima
4126 \bxjs@invscale\@tempdima\bxjs@tmpa
4127 \xdef\bxjs@scale{\strip@pt\@tempdima}
4128 \fi
4129 \endgroup
4130 %\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。

```
4131 \bxjs@declarefontshape
```

フォント代替の明示的定義。

```

4132 \DeclareFontShape{\jsc@JYn}{mc}{m}{it}{<->ssub*mc/m/n}{}
4133 \DeclareFontShape{\jsc@JYn}{mc}{m}{sl}{<->ssub*mc/m/n}{}
4134 \DeclareFontShape{\jsc@JYn}{mc}{m}{sc}{<->ssub*mc/m/n}{}
4135 \DeclareFontShape{\jsc@JYn}{gt}{m}{it}{<->ssub*gt/m/n}{}
4136 \DeclareFontShape{\jsc@JYn}{gt}{m}{sl}{<->ssub*gt/m/n}{}
4137 \DeclareFontShape{\jsc@JYn}{mc}{bx}{it}{<->ssub*gt/m/n}{}
4138 \DeclareFontShape{\jsc@JYn}{mc}{bx}{sl}{<->ssub*gt/m/n}{}
4139 \DeclareFontShape{\jsc@JYn}{gt}{bx}{it}{<->ssub*gt/m/n}{}
4140 \DeclareFontShape{\jsc@JYn}{gt}{bx}{sl}{<->ssub*gt/m/n}{}
4141 \DeclareFontShape{\jsc@JYn}{mc}{b}{n}{<->ssub*mc/bx/n}{}
4142 \DeclareFontShape{\jsc@JYn}{mc}{b}{it}{<->ssub*mc/bx/n}{}
4143 \DeclareFontShape{\jsc@JYn}{mc}{b}{sl}{<->ssub*mc/bx/n}{}
4144 \DeclareFontShape{\jsc@JYn}{gt}{b}{n}{<->ssub*gt/bx/n}{}
4145 \DeclareFontShape{\jsc@JYn}{gt}{b}{it}{<->ssub*gt/bx/n}{}
4146 \DeclareFontShape{\jsc@JYn}{gt}{b}{sl}{<->ssub*gt/bx/n}{}
4147 \DeclareFontShape{\jsc@JTn}{mc}{m}{it}{<->ssub*mc/m/n}{}
4148 \DeclareFontShape{\jsc@JTn}{mc}{m}{sl}{<->ssub*mc/m/n}{}
4149 \DeclareFontShape{\jsc@JTn}{mc}{m}{sc}{<->ssub*mc/m/n}{}
4150 \DeclareFontShape{\jsc@JTn}{gt}{m}{it}{<->ssub*gt/m/n}{}
4151 \DeclareFontShape{\jsc@JTn}{gt}{m}{sl}{<->ssub*gt/m/n}{}
4152 \DeclareFontShape{\jsc@JTn}{mc}{bx}{it}{<->ssub*gt/m/n}{}
4153 \DeclareFontShape{\jsc@JTn}{mc}{bx}{sl}{<->ssub*gt/m/n}{}
4154 \DeclareFontShape{\jsc@JTn}{gt}{bx}{it}{<->ssub*gt/m/n}{}
4155 \DeclareFontShape{\jsc@JTn}{gt}{bx}{sl}{<->ssub*gt/m/n}{}
4156 \DeclareFontShape{\jsc@JTn}{mc}{b}{n}{<->ssub*mc/bx/n}{}
4157 \DeclareFontShape{\jsc@JTn}{mc}{b}{it}{<->ssub*mc/bx/n}{}
4158 \DeclareFontShape{\jsc@JTn}{mc}{b}{sl}{<->ssub*mc/bx/n}{}
4159 \DeclareFontShape{\jsc@JTn}{gt}{b}{n}{<->ssub*gt/bx/n}{}
4160 \DeclareFontShape{\jsc@JTn}{gt}{b}{it}{<->ssub*gt/bx/n}{}
4161 \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。

```

4162 \@ifl@t@r\fmtversion{2020/10/01}
4163     {\jsc@needspace@tchfalse}{\jsc@needspace@tchtrue}
4164 \ifjsc@needspace@tch      % --- for 2020-02-02 or older BEGIN
4165 \ifx\@rmfamilyhook\@undefined % old
4166 \DeclareRobustCommand\rmfamily
4167     {\not@math@alphabet\rmfamily\mathrm
4168      \romanfamily\rmdefault\kanjifamily\mcdefault\selectfont}
4169 \DeclareRobustCommand\sffamily
4170     {\not@math@alphabet\sffamily\mathsf
4171      \romanfamily\sfdefault\kanjifamily\gtdefault\selectfont}
4172 \DeclareRobustCommand\ttfamily
4173     {\not@math@alphabet\ttfamily\mathtt
4174      \romanfamily\ttdefault\kanjifamily\gtdefault\selectfont}
4175 \g@addto@macro\bxjs@begin@document@hook{%
4176     \ifx\mweights@init\@undefined\else % mweights.sty is loaded
4177     % my definitions above should have been overwritten, recover it!
4178     % \selectfont is executed twice but I don't care about speed...
4179     \expandafter\g@addto@macro\csname rmfamily \endcsname
4180         {\kanjifamily\mcdefault\selectfont}%
4181     \expandafter\g@addto@macro\csname sffamily \endcsname
4182         {\kanjifamily\gtdefault\selectfont}%
4183     \expandafter\g@addto@macro\csname ttfamily \endcsname
4184         {\kanjifamily\gtdefault\selectfont}%
4185     \fi}
4186 \else % 2020-02-02
4187 \g@addto@macro\@rmfamilyhook
4188     {\prepare@family@series@update@kanji{mc}\mcdefault}
4189 \g@addto@macro\@sffamilyhook
4190     {\prepare@family@series@update@kanji{gt}\gtdefault}
4191 \g@addto@macro\@ttfamilyhook
4192     {\prepare@family@series@update@kanji{gt}\gtdefault}
4193 \fi
4194 \else % --- for 2020-02-02 or older END & for 2020-10-01 BEGIN
4195 \AddToHook{rmfamily}%
4196     {\prepare@family@series@update@kanji{mc}\mcdefault}
4197 \AddToHook{sffamily}%
4198     {\prepare@family@series@update@kanji{gt}\gtdefault}
4199 \AddToHook{ttfamily}%
4200     {\prepare@family@series@update@kanji{gt}\gtdefault}
4201 \fi % --- for 2020-10-01 END
4202 \ifx\DeclareFixJFMCJKTextFontCommand\@undefined
4203 \DeclareJaTextFontCommand{\textmc}{\mcfamily}
4204 \DeclareJaTextFontCommand{\textgt}{\gtfamily}
4205 \fi
4206 \bxjs@if@sf@default{%

```

```
4207 \renewcommand\kanjifamilydefault{\gtdefault}}
```

念のため。

```
4208 \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）。

```
4209 \def\bxjs@parse@qh#1{\let\bxjs@tmpb\relax}
```

```
4210 \let\bxjs@parse@qh@a\@undefined
```

```
4211 \let\bxjs@parse@qh@b\@undefined
```

## ■パラメタの設定

```
4212 \prebreakpenalty\jis"2147=10000
```

```
4213 \postbreakpenalty\jis"2148=10000
```

```
4214 \prebreakpenalty\jis"2149=10000
```

```
4215 \inhibitxspcode`!=1
```

```
4216 \inhibitxspcode`-=2
```

```
4217 \xspcode`+=3
```

```
4218 \xspcode`\%=3
```

"80～"FF の範囲の `\xspcode` を 3 に変更。

```
4219 \@tempcnta="80 \@whilenum\@tempcnta<"100 \do{%
```

```
4220 \xspcode\@tempcnta=3\advance\@tempcnta\@ne}
```

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

```
4221 \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。

```
4222 \begingroup
```

```
4223 \catcode`\!=0
```

`\bxjs@ptex@dir` 現在の組方向：t=縦、y=横、?=その他。

```
4224 \gdef\bxjs@ptex@dir{%
```

```
4225 !iftkdir t%
```

```
4226 !else!ifydir y%
```

```
4227 !else ?%
```

```
4228 !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。

```
4229 % 古い \@makefnmark の定義
```

```
4230 \long\def\bxjs@tmpa{\hbox{%
```

```
4231 !ifydir \@textsuperscript{\normalfont\@thefnmark}}%
```

```
4232 !else\hbox{\yoko\@textsuperscript{\normalfont\@thefnmark}}!fi}}
```

```

4233 \ifx\@makefnmark\bxjs@tmpa
4234 \long\gdef\@makefnmark{%
4235   !ifdir \hbox{}\hbox{\@textsuperscript{\normalfont\@thefnmark}}\hbox{}%
4236   !else\hbox{\yoko\@textsuperscript{\normalfont\@thefnmark}}!fi}
4237 \fi

```

エスケープ文字の変更はここまで。

```

4238 \endgroup

```

■minijs パッケージのブロック やっておく。

```

4239 \@namedef{ver@minijs.sty}{

```

### B.3 pdfTeX 用の処理

```

4240 \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。

```

4241 \let\bxjs@let@hchar@chr\bxjs@let@hchar@chr@ue

```

ムニャムニャ。

```

4242 \@onlypreamble\bxjs@cjk@loaded
4243 \def\bxjs@cjk@loaded{%
4244   \def\@footnotemark{%
4245     \leavevmode
4246     \ifhmode
4247       \edef\@x@sf{\the\spacefactor}%
4248       \ifdim\lastkern>\z@\ifdim\lastkern<5sp\relax
4249         \unkern\unkern
4250         \ifdim\lastskip>\z@ \unskip \fi
4251       \fi\fi
4252       \nobreak
4253     \fi
4254     \@makefnmark
4255     \ifhmode \spacefactor\@x@sf \fi
4256     \relax}%
4257 \let\bxjs@cjk@loaded\relax
4258 }
4259 \g@addto@macro\bxjs@begin@document@hook{%
4260   \@ifpackageloaded{CJK}{%
4261     \bxjs@cjk@loaded
4262   }{}%
4263 }

```

### B.4 X<sub>Y</sub>TeX 用の処理

```

4264 \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。

```

4265 \def\bxjs@let@hchar@chr#1{%
4266   \@tempcnta`#1\relax \divide\@tempcnta"800\relax

```

```

4267 \bxjs@cond\ifnum\@tempcnta=27 \fi{%
4268   \bxjs@let@hchar@chr@xe
4269 }{\bxjs@let@hchar@out\def{{#1}}}}
4270 \def\bxjs@let@hchar@chr@xe#1{%
4271   \lccode`0=`#1\relax
4272   \lowercase{\bxjs@let@hchar@out\def{{0}}}}

```

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

```

4273 \@onlypreamble\bxjs@do@precisetext
4274 \ifx\XeTeXgenerateactualtext\@undefined\else
4275   \def\bxjs@do@precisetext{%
4276     \XeTeXgenerateactualtext=\@ne}
4277 \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。

```

4278 \@onlypreamble\bxjs@do@simplejasetup
4279 \def\bxjs@do@simplejasetup{%
4280   \@namedef{bxjs@zeroglue/0.0pt}{T}%
4281   \ifnum\XeTeXinterchartokenstate>\z@
4282   \else\expandafter\ifx\csname bxjs@zeroglue/\the\XeTeXlinebreakskip\endcsname\relax\else
4283     \jsSimpleJaSetup
4284     \ClassInfo\bxjs@clsname
4285     {'\string\jsSimpleJaSetup' is applied\@gobble}%
4286   \fi\fi}

```

`\jsSimpleJaSetup` 日本語出力用の超簡易的な設定。

```

4287 \newcommand*{\jsSimpleJaSetup}{%
4288   \XeTeXlinebreaklocale "ja"\relax
4289   \XeTeXlinebreakskip=0pt plus 1pt minus 0.1pt\relax
4290   \XeTeXlinebreakpenalty=0\relax}

```

## B.5 後処理（エンジン共通）

```

4291 \fi\fi\fi

```

`simplejasetup` オプションの処理。

```

4292 \ifx\bxjs@do@simplejasetup\@undefined\else
4293   \g@addto@macro\bxjs@begin@document@hook{%
4294     \ifbxjs@simplejasetup
4295       \bxjs@do@simplejasetup
4296     \fi}
4297 \fi

```

`precisetext` オプションの処理。

```

4298 \ifbxjs@precisetext
4299   \ifx\bxjs@do@precisetext\@undefined
4300     \ClassWarning\bxjs@clsname
4301     {The current engine does not support the\MessageBreak

```

```

4302      'precise-text' option\@gobble}
4303  \else
4304    \bxjs@do@precisetext
4305  \fi
4306 \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るために `\everyparhook` を空にする。

```

4307 \g@addto@macro\bxjs@begin@document@hook{%
4308   \ifx\jsInhibitGlueAtParTop\@empty
4309     \def\bxjs@tmpa{\jsInhibitGlueAtParTop}%
4310     \ifx\everyparhook\bxjs@tmpa
4311       \let\everyparhook\@empty
4312     \fi
4313 \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。

```

4314 \ifnum\bxjs@everyparhook=\bxjs@everyparhook@@modern

```

まず `\everypar` を“乗っ取る” 処理を行う。

```

4315 \let\bxjs@everypar\everypar
4316 \newtoks\everypar
4317 \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。

```

4318 \bxjs@everypar{\the\expandafter\everypar\everyparhook}%
4319 \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。

```

4320 \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。

```

4321 \@onlypreamble\bxjs@adjust@fancyhdr
4322 \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。

```

4323 \def\bxjs@tmpa{\fancyplain{}{\sl\rightmark}\strut}%
4324 \def\bxjs@tmpb{\fancyplain{}{\rightmark}\strut}%
4325 \ifx\f@ncyselh\bxjs@tmpa \global\let\f@ncyselh\bxjs@tmpb \fi

```

```

4326 \ifx\@ncyerh\bxjs@tmpa \global\let\@ncyerh\bxjs@tmpb \fi
4327 \ifx\@ncyolh\bxjs@tmpa \global\let\@ncyolh\bxjs@tmpb \fi
4328 \ifx\@ncyorh\bxjs@tmpa \global\let\@ncyorh\bxjs@tmpb \fi
4329 \def\bxjs@tmpa{\fancyplain{}{\sl\leftmark}\strut}%
4330 \def\bxjs@tmpb{\fancyplain{}{\leftmark}\strut}%
4331 \ifx\@ncyelh\bxjs@tmpa \global\let\@ncyelh\bxjs@tmpb \fi
4332 \ifx\@ncyerh\bxjs@tmpa \global\let\@ncyerh\bxjs@tmpb \fi
4333 \ifx\@ncyolh\bxjs@tmpa \global\let\@ncyolh\bxjs@tmpb \fi
4334 \ifx\@ncyorh\bxjs@tmpa \global\let\@ncyorh\bxjs@tmpb \fi
4335 \def\bxjs@tmpa{\rm\thepage\strut}%
4336 \def\bxjs@tmpb{\thepage\strut}%
4337 \ifx\@ncyecf\bxjs@tmpa \global\let\@ncyecf\bxjs@tmpb \fi
4338 \ifx\@ncyocf\bxjs@tmpa \global\let\@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。

```

4339 \ifx\fullwidth\@undefined\else \ifdim\textwidth<\fullwidth
4340   \setlength{\@tempdima}{\fullwidth-\textwidth}%
4341   \edef\bxjs@tmpa{\noexpand\fancyhfoffset[EL,OR]{\the\@tempdima}%
4342   }\bxjs@tmpa
4343 \fi\fi
4344 \PackageInfo\bxjs{clsname
4345   {Patch to fancyhdr is applied\@gobble}}

```

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

```

4346 \def\bxjs@pagestyle@hook{%
4347   \@ifpackageloaded{fancyhdr}{%
4348     \bxjs@adjust@fancyhdr
4349     \global\let\bxjs@adjust@fancyhdr\relax
4350   }{}}

```

\pagestyle にフックを入れ込む。

```

4351 \let\bxjs@org@pagestyle\pagestyle
4352 \def\pagestyle{%
4353   \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。

```

4354 \g@addto@macro\bxjs@begin@document@hook{%
4355   \bxjs@pagestyle@hook
4356   \global\let\bxjs@pagestyle@hook\relax}

```

終わり。

```
4357 \fi
```

以上で終わり。

```
4358 %</minimal>
```

## 付録 C 和文ドライバ：standard 🐼

標準のドライバ。

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

### C.1 準備

```
4359 %<*standard>
4360 %% 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。

```
4361 \input{bxjsja-minimal.def}
```

`simplejasetup` は `standard` では無効になる。

```
4362 \bxjs@simplejasetupfalse
```

#### ■環境検査

**TODO:3.0** 以下で 3.0 版でのバージョン要件の予定について述べておく。

`standard` 和文ドライバの処理系バージョン要件（minimal からの差分）は以下の通りである。

- `upTeX`： 0.29 版 [2010/01] 以上
- `LuaTeX`： 0.85 版 [2015/11] 以上
- `XƎTeX`： 0.9999 版 [2013/03] 以上

加えて、以下の要件を定める。

- `pTeX` 系以外のエンジンでは `ε-TeX` 拡張を必須とする。  
※ `bxckjatype` 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。

```
4363 \ifjsWithTeX
4364 \IfFileExists{etoolbox.sty}{%
```



```

4365 \RequirePackage{etoolbox}[2007/10/08]% v1.3
4366 }{}
4367 \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 のファミリーは bxjsStd とする。

\ifbxjs@jp@jismmiv 2004JIS 字形を優先させるか。

```

4368 \newif\ifbxjs@jp@jismmiv

    jis2004 オプションの処理。
4369 \bxjs@cslet{bxjs@kv@jis2004@true}\bxjs@jp@jismmivtrue
4370 \bxjs@cslet{bxjs@kv@jis2004@false}\bxjs@jp@jismmivfalse
4371 \define@key{bxjsStd}{jis2004}[true]{%
4372 \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。

```

4373 \newif\ifbxjs@jp@units

    units オプションの処理。
4374 \let\bxjs@kv@units@true\bxjs@jp@unitstrue
4375 \let\bxjs@kv@units@false\bxjs@jp@unitsfalse
4376 \define@key{bxjsStd}{units}[true]{%
4377 \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。

```

4378 \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。
4379 \define@key{bxjsStd}{font}{%
4380 \edef\bxjs@jp@font{\bxjs@catopt\bxjs@jp@font{#1}}}

```

\ifbxjs@jp@strong@cmd \strong 命令を補填するか。

```

4381 \newif\ifbxjs@jp@strong@cmd \bxjs@jp@strong@cmdtrue

    strong-cmd オプションの処理。
4382 \let\bxjs@kv@strongcmd@true\bxjs@jp@strong@cmdtrue
4383 \let\bxjs@kv@strongcmd@false\bxjs@jp@strong@cmdfalse
4384 \define@key{bxjs}{strong-cmd}[true]{\bxjs@set@keyval{strongcmd}{#1}{}}

```

実際の japaram の値を適用する。

```

4385 \def\bxjs@next#1{\bxjs@safe@setkeys{bxjsStd}{#1}}
4386 \expandafter\bxjs@next\expandafter{\jsJaParam}

```

## C.3 共通処理 (1)

```
4387 \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。

```
4388 \@onlypreamble\bxjs@apply@mmiv
4389 \def\bxjs@apply@mmiv{%
4390   \g@addto@macro\@classoptionslist{,jis2004}
4391   \% \@ifpackagewith 判定への対策
4392   \PassOptionsToPackage{jis2004}{otf}
4393   \global\let\bxjs@apply@mmiv\relax}
4394 \ifbxjs@jp@jismmiv \bxjs@apply@mmiv \fi
```

■**和文用単位をサポート** エンジンが (u)pTeX の場合は units を無効にする。

```
4395 \if j\jsEngine
4396   \bxjs@jp@unitsfalse
4397 \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。

```
4398 \ifbxjs@jp@units
4399   \IfFileExists{bxcalc.sty}{%
4400     \RequirePackage{bxcalc}[2018/01/28]%v1.0a
4401     \ifx\usepTeXunits\@undefined
4402       \PackageWarningNoLine\bxjs@clsname
4403         {Cannot support pTeX units (zw etc.), since\MessageBreak
4404           the package 'bxcalc' is too old}%
4405       \bxjs@jp@unitsfalse
4406     \else \usepTeXunits
4407     \fi
4408   }{%else
4409     \PackageWarningNoLine\bxjs@clsname
4410       {Cannot support pTeX units (zw etc.), since\MessageBreak
4411         the package 'bxcalc' is unavailable}%
4412     \bxjs@jp@unitsfalse
4413   }
4414 \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。

```
4415 \ifbxjs@jp@units
4416 \def\bxjs@parse@qh#1{\let\bxjs@tmpb\relax}
4417 \let\bxjs@parse@qh@a\@undefined
4418 \let\bxjs@parse@qh@b\@undefined
4419 \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。

```

4420 \ifbxjs@jp@units
4421   \def\bxjs@let@lenexpr#1#2{%
4422     \edef#1{#2}%
4423     \expandafter\CUXParseExpr\expandafter#1\expandafter{#1}}
4424 \else
4425   \def\bxjs@let@lenexpr{\edef}
4426 \fi

```

## ■\strong 命令の補填

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

```

4427 \ifbxjs@jp@strong@cmd\jsAtEndOfClass{%
4428   \ifx\strong\@undefined\ifx\strongenv\@undefined
4429     \newcommand*{\strongenv}{\jsStrongText}%
4430     \DeclareTextFontCommand{\strong}{\jsStrongText}%
4431     \newcommand*{\strongfontdeclare}{\jsStrongDeclare}%
4432   \fi\fi
4433 }\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。

```

4434 \@onlypreamble\bxjs@adjust@jafont
4435 \def\bxjs@adjust@jafont#1{%
4436   \ifx\jsJaFont\bxjs@@auto
4437     \bxjs@get@kanjiEmbed
4438     \ifx\bxjs@jaEmbed\relax
4439       \let\bxjs@tmpa\@empty
4440     \else
4441       \let\bxjs@tmpa\bxjs@jaEmbed
4442       \ifx\bxjs@jaVariant\bxjs@@hziv
4443         \bxjs@apply@mmiv
4444       \fi
4445     \fi
4446   \else
4447     \let\bxjs@tmpa\jsJaFont
4448   \fi
4449   \if f#1\ifx\bxjs@tmpa\bxjs@@noEmbed
4450     \ClassWarningNoLine\bxjs@clsname
4451     {Option 'jafont=noEmbed' is ignored, because it is\MessageBreak
4452     not available on the current situation}%

```

```

4453 \let\bxjs@tmpa\@empty
4454 \fi\fi
4455 }
4456 \def\bxjs@@auto{auto}
4457 \def\bxjs@@noEmbed{noEmbed}
4458 \def\bxjs@@hziv{-04}

```

\bxjs@jaEmbed 現在の updmap の jaEmbed・jaVariant パラメタの値。 \bxjs@get@kanjiEmbed により実  
 \bxjs@jaVariant 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。

```

4459 \let\bxjs@jaEmbed\relax
4460 \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。

```

4461 \@onlypreamble\bxjs@get@kanjiEmbed
4462 \def\bxjs@get@kanjiEmbed{%
4463   \begingroup\setbox\z@=\hbox{%
4464     \global\let\bxjs@tmpdo\@empty
4465     \def\bxjs@next##1##2##3{%
4466       \def##1####1##3 ####2\@nil####3\@nnil{%
4467         \ifx$####1$\gdef##2{####2}\fi}%
4468       \g@addto@macro\bxjs@tmpdo{%
4469         \expandafter##1\bxjs@tmpa\@nil##3 \@nil\@nnil}}}%
4470   \bxjs@next\bxjs@tmpdo@a\bxjs@g@tmpa{kanjiEmbed}%
4471   \bxjs@next\bxjs@tmpdo@b\bxjs@g@tmpa{jaEmbed}%
4472   \bxjs@next\bxjs@tmpdo@c\bxjs@g@tmpb{kanjiVariant}%
4473   \bxjs@next\bxjs@tmpdo@d\bxjs@g@tmpb{jaVariant}%
4474   %
4475   \global\let\bxjs@g@tmpa\relax
4476   \global\let\bxjs@g@tmpb\relax
4477   \endlinechar\m@ne
4478   \let\do\@makeother\dospecials
4479   \catcode32=10 \catcode12=10 %form-feed
4480   \let\bxjs@tmpa\@empty
4481   \openin\@inputcheck="|kpsewhich updmap.cfg"\relax
4482   \ifeof\@inputcheck\else
4483     \read\@inputcheck to\bxjs@tmpa
4484     \closein\@inputcheck
4485     \fi
4486     \ifx\bxjs@tmpa\@empty\else
4487       \openin\@inputcheck="\bxjs@tmpa"\relax
4488       \@tempwattrue
4489       \loop\if@tempswa
4490         \read\@inputcheck to\bxjs@tmpa
4491         \bxjs@tmpdo
4492         \ifeof\@inputcheck \@tempwafalse \fi
4493       \repeat
4494     \fi
4495   }\endgroup

```

```

4496 \let\bxjs@jaEmbed\bxjs@g@tmpa
4497 \let\bxjs@jaVariant\bxjs@g@tmpb
4498 }

```

`\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。

```

4499 \@onlypreamble\bxjs@resolve@jafont@paren
4500 \def\bxjs@resolve@jafont@paren#1{%
4501   \def\bxjs@tmpb{\let#1}%
4502   \expandafter\bxjs@resolve@jafont@paren@a#1\@nil()\@nil\@nnil#1}
4503 \@onlypreamble\bxjs@resolve@jafont@paren@a
4504 \def\bxjs@resolve@jafont@paren@a#1(#2)#3\@nil#4\@nnil#5{%
4505   \ifx\relax#4\relax \bxjs@tmpb#5%
4506   \else
4507     \edef\bxjs@tmpa{#1\bxjs@jafont@paren{#2}#3}%
4508     \bxjs@tmpb\bxjs@tmpa
4509   \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。

```

4510 \newcommand*\jachar[1]{%
4511   \begingroup
\jsLetHeadChar で先頭の “文字” を拾ってそれを \bxjs@jachar に渡す。
4512   \jsLetHeadChar\bxjs@tmpa{#1}%
4513   \ifx\bxjs@tmpa\relax
4514     \ClassWarningNoLine\bxjs@clsname
4515       {Illegal argument given to \string\jachar}%
4516   \else
4517     \expandafter\bxjs@jachar\expandafter{\bxjs@tmpa}%
4518   \fi
4519   \endgroup}

```

`\jsJaChar` を `\jachar` と等価にする。

```

4520 \let\jsJaChar\jachar

```

下請けの `\bxjs@jachar` の実装はエンジンにより異なる。

```

4521 \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。

```

4522 \PassOptionsToPackage{setpagesize=false}{hyperref}

```

`\bxjs@fix@hyperref@unicode` `hyperref` の `unicode` オプションの値を固定する。

```

4523 \@onlypreamble\bxjs@fix@hyperref@unicode
4524 \def\bxjs@fix@hyperref@unicode#1{%
4525   \PassOptionsToPackage{bxjs/hook=#1}{hyperref}%

```

```

4526 \namedef{KV@Hyp@bxjs/hook}##1{%
4527   \KV@Hyp@unicode{##1}%
4528   \def\KV@Hyp@unicode####1{%
4529     \expandafter\ifx\csname if##1\expandafter\endcsname
4530     \csname if####1\endcsname\else
4531     \ClassWarningNoLine{bxjs@clsname
4532       {Blcoked hyperref option 'unicode=####1'}%
4533     \fi
4534   }%
4535 }%
4536 }

```

`\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。

```

4537 \@onlypreamble\jsCheckHyperrefUnicode
4538 \let\jsCheckHyperrefUnicode\empty
4539 \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。

```

4540 \@onlypreamble\bxjs@check@hyperref@unicode
4541 \def\bxjs@check@hyperref@unicode#1{%
4542   \g@addto@macro\jsCheckHyperrefUnicode{%
4543     \@tempswafalse
4544     \begingroup
4545       \expandafter\ifx\csname ifHy@unicode\endcsname\relax
4546       \aftergroup\@tempwattrue \fi
4547       \expandafter\ifx\csname ifHy@unicode\expandafter\endcsname
4548       \csname if#1\endcsname
4549       \aftergroup\@tempwattrue \fi
4550     \endgroup
4551     \if@tempswa\else
4552       \ClassError{bxjs@clsname
4553         {The value of hyperref 'unicode' key is not suitable\MessageBreak
4554           for the present engine (must be #1)}}%
4555       {\@ehc}%
4556     \fi}}

```

`\bxjs@urgent@special` DVI のなるべく早い位置に special を出力する。

```

4557 \@onlypreamble\bxjs@urgent@special
4558 \@onlypreamble\bxjs@urgent@special@a

```

L<sup>A</sup>T<sub>E</sub>X 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。

```

4559 \ifbxjs@old@hook@system
4560 \def\bxjs@urgent@special#1{%
4561   \AtBeginDvi{\special{#1}}%
4562   \g@addto@macro\bxjs@begin@document@hook{%
4563     \@ifpackageloaded{atbegshi}{%
4564       \begingroup

```

```

4565      \toks\z@{\special{#1}}%
4566      \toks\tw@\expandafter{\AtBegShi@HookFirst}%
4567      \xdef\AtBegShi@HookFirst{\the\toks@\the\toks\tw@}%
4568      \endgroup
4569    }{}%
4570  }%
4571 }

```

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。

```

4572 \else
4573   \def\bxjs@urgent@special#1{%
4574     \bxjs@urgent@special@a
4575     \AddToHook{shipout/firstpage}[pxjahyper/enc]{\special{#1}}
4576   \def\bxjs@urgent@special@a{%
4577     \DeclareHookRule{shipout/firstpage}{pxjahyper/enc}{<}{hyperref}%
4578     \global\let\bxjs@urgent@special@a\relax}
4579 \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。

```

4580 \@onlypreamble\bxjs@patch@RDMA@for@bm
4581 \def\bxjs@patch@RDMA@for@bm{%
  実際に改変するのは \DualLang@mathalph@bet である。
4582   \let\bxjs@org@DualLang@mathalph@bet\DualLang@mathalph@bet
4583   \def\DualLang@mathalph@bet{%

```

`\bm` の引数の中（ここでは `\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。

```

4584     \ifx\bm@\@firstofone \expandafter\@firstoftwo
4585     \else \expandafter\bxjs@org@DualLang@mathalph@bet
4586     \fi}%

```

4587 }

## C.4 pTeX 用設定

4588 \if j\jsEngine

### ■共通命令の実装

```
4589 \newcommand*\setkanjiskip{\jsSetKanjiSkip}
4590 \newcommand*\getkanjiskip{\jsGetKanjiSkip}
4591 \def\jsApplyKanjiSkip#1{%
4592   \kanjiskip=#1\relax}
4593 \newcommand*\setxkanjiskip{\jsSetXKanjiSkip}
4594 \newcommand*\getxkanjiskip{\jsGetXKanjiSkip}
4595 \def\jsApplyXKanjiSkip#1{%
4596   \xkanjiskip=#1\relax}
```

\jaJaChar のサブマクロ。

```
4597 \def\bxjs@jachar#1{%
4598   \bxjs@jachar@a#1...\@nil}
4599 \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それをそのまま出力する。

```
4600   \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し、そのスカラー値を \@tempcnta に代入する。

```
4601   \else\ifx.#3%
4602     \@tempcnta`#1 \multiply\@tempcnta64
4603     \advance\@tempcnta`#2 \advance\@tempcnta-"3080
4604     \bxjs@jachar@b
4605   \else\ifx.#4%
4606     \@tempcnta`#1 \multiply\@tempcnta64
4607     \advance\@tempcnta`#2 \multiply\@tempcnta64
4608     \advance\@tempcnta`#3 \advance\@tempcnta-"E2080
4609     \bxjs@jachar@b
4610   \else
4611     \@tempcnta`#1 \multiply\@tempcnta64
4612     \advance\@tempcnta`#2 \multiply\@tempcnta64
4613     \advance\@tempcnta`#3 \multiply\@tempcnta64
4614     \advance\@tempcnta`#4 \advance\@tempcnta-"3C82080
4615     \bxjs@jachar@b
4616   \fi\fi\fi}
```

符号値が \@tempcnta の和文文字を出力する処理。

```
4617 \ifjsWithupTeX
4618   \def\bxjs@jachar@b{\kchar\@tempcnta}
4619 \else
4620   \def\bxjs@jachar@b{%
4621     \ifx\bxUInt\@undefined\else
4622       \bxUInt{\@tempcnta}%
```



```
4623 \fi}
4624 \fi
```

和欧文間空白の命令 `\jathinspace` の実装。

```
4625 \ifbxjs@jaspace@cmd
4626 \def\jathinspace{\hskip\xkanjiskip}
4627 \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。

```
4628 \ifbxjs@jp@jismmiv
4629 \PassOptionsToPackage{prefer2004jis}{pxchfon}
4630 \PassOptionsToPackage{prefer2004jis}{pxbabel}
4631 \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` が指定されている場合は、その値を `pxchfon` のオプションに追加する。

```
4632 \let\bxjs@jafont@paren\@firstofone
4633 \let\bxjs@tmpa\jsJaFont
4634 \ifx\bxjs@tmpa\bxjs@@auto
4635 \let\bxjs@tmpa\@empty
4636 \else\ifx\bxjs@tmpa\bxjs@@noEmbed
4637 \def\bxjs@tmpa{noembed}
4638 \fi\fi
4639 \bxjs@resolve@jafont@paren\bxjs@tmpa
4640 \edef\bxjs@tmpa{\bxjs@catopt\bxjs@tmpa\bxjs@jp@font}
4641 \ifx\bxjs@tmpa\@empty\else
4642 \edef\bxjs@next{%
4643 \noexpand\RequirePackage[\bxjs@tmpa]{pxchfon}[2010/05/12]% v0.5
4644 }\bxjs@next
4645 \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……。)

```
4646 \begingroup
4647 \global\let\bxjs@g@tmpa\relax
4648 \catcode`\|=0 \catcode`\|=12
4649 |def|bxjs@tmpdo#1|@nil{%
4650 |bxjs@tmpdo@a#1|@nil\RequirePackage|@nnil}%
4651 |def|bxjs@tmpdo@a#1\RequirePackage#2|@nnil{%
4652 |ifx$#1$|bxjs@tmpdo@b#2|@nil keyval|@nnil |fi}%
```

```

4653 |catcode`\|=0 \catcode`\|=12
4654 \def\bxjs@tmpdo@b#1keyval#2\@nnil{%
4655   \ifx$#2$\else
4656     \xdef\bxjs@g@tmpa{%
4657       \noexpand\PassOptionsToPackage{scale=\jsScale}{otf}}%
4658   \fi}
4659 \@firstofone{%
4660   \catcode10=12 \endlinechar\m@ne
4661   \let\do\@makeoother \dospecials \catcode32=10
4662   \openin\@inputcheck=otf.sty\relax
4663   \@tempswattrue
4664   \loop\if@tempswa
4665     \ifeof\@inputcheck \@tempswafalse \fi
4666     \if@tempswa
4667       \read\@inputcheck to\bxjs@next
4668       \expandafter\bxjs@tmpdo\bxjs@next\@nil
4669     \fi
4670   \repeat
4671   \closein\@inputcheck
4672 \endgroup}
4673 \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]：upL<sup>A</sup>T<sub>E</sub>X 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]：pL<sup>A</sup>T<sub>E</sub>X 上の unicode 対応を試験的サポート。
- 1.3 版 [2023-03-01]：pL<sup>A</sup>T<sub>E</sub>X 上の unicode 対応を正式サポート。

```

4674 \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）「unicode 対応」が行われた場合は検証は無効化される。

```

4675 \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。

```

4676 \bxjs@if@package@atleast{plautopatch}{2020/05/25}{% v0.9g
4677   \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、upL<sup>A</sup>T<sub>E</sub>X 上の unicode には対応していることに注意。

```

4678 \let\bxjs@avail@hy@unicode=f

```

```

4679 \if \ifx t\bxjs@plautopatch@new T%
4680     \else\ifbxjs@old@hook@system F\else T\fi\fi T%
4681     \IfFileExists{pxjahyper-uni.def}{\let\bxjs@avail@hy@unicode=t}{%}
4682 \fi
4683 \if t\bxjs@avail@hy@unicode
4684     \ifjsWithupTeX

```

必要条件が満たされていて、かつ up $\text{\LaTeX}$  である場合の処理。もしファイルフックが利用可能ならば、hyperref が読み込まれた場合にその直後に pxjahyper-uni.def が読まれるようにする。

※そうでないなら、前提条件より pxjahyper が読み込まれるはずなので何もしなくてよい。

```

4685     \ifbxjs@old@hook@system\else
4686         \AddToHook{\bxjs@CGHN{package/hyperref/after}}{%
4687             \input{pxjahyper-uni.def}}
4688     \fi
4689 \else

```

必要条件が満たされていて、かつ p $\text{\LaTeX}$  である場合の処理。pxjahyper が「p $\text{\LaTeX}$  上の unicode 対応をもつほど新しい版（1.3 版以降）」であるかを判定する方法はない。しかし、新しい  $\text{\LaTeX}$  システムで unicode を無効にするのは避けたいので、 $\text{\LaTeX}$  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「 $\text{\LaTeX}$  がカーネルが新しい」かを判定する。

```

4690     \let\bxjs@avail@hy@unicode=f
4691     \ifx t\bxjs@plautopatch@new
4692         \bxjs@if@format@at@least{2023/06/01}{\let\bxjs@avail@hy@unicode=t}{%}
4693     \fi
4694 \fi
4695 \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。

```

4696 \if f\bxjs@avail@hy@unicode
4697     \PassOptionsToPackage{unicode=false}{hyperref}
4698 \fi
4699 \fi

```

tounicode special 命令を出力する。

```

4700 \if \ifx\bxjs@driver@given\bxjs@driver@@dvipdfmx T%
4701     \else\ifjsWithpTeXng T\else F\fi\fi T%
4702 \IfFileExists{pxjahyper-enc.sty}{\@tempwattrue}{\@tempwafalse}
4703 \if@tempwa
4704     \RequirePackage{pxjahyper-enc}[2020/10/05]%v0.6
4705     \ifbxjs@bigcode\else \suppressbigcode \fi
4706 \else
4707     \ifnum\jis"2121="A1A1 %euc
4708         \bxjs@urgent@special{pdf:tounicode EUC-UCS2}
4709     \else\ifnum\jis"2121="8140 %sjis
4710         \bxjs@urgent@special{pdf:tounicode 90ms-RKSJ-UCS2}

```

```

4711 \else\ifnum\jis"2121="3000 %uptex
4712 \ifbxjs@bigcode
4713 \bxjs@urgent@special{pdf:tounicode UTF8-UTF16}
4714 \PassOptionsToPackage{bigcode}{pxjahyper}
4715 \else
4716 \bxjs@urgent@special{pdf:tounicode UTF8-UCS2}
4717 \PassOptionsToPackage{nobigcode}{pxjahyper}
4718 \fi
4719 \fi\fi\fi
4720 \let\bxToUnicodeSpecialDone=t
4721 \fi
4722 \fi

```

■和文数式ファミリー 和文数式ファミリー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。

```

4723 \ifx f\bxjs@enablejfam\else
4724 \@enablejfamtrue
4725 \fi

```

実際に和文用の数式ファミリーの設定を行う。

```

4726 \if@enablejfam
4727 \DeclareSymbolFont{mincho}{\jsc@JYn}{mc}{m}{n}
4728 \DeclareSymbolFontAlphabet{\mathmc}{mincho}
4729 \SetSymbolFont{mincho}{bold}{\jsc@JYn}{gt}{m}{n}
4730 \jfam\symmincho
4731 \DeclareMathAlphabet{\mathgt}{\jsc@JYn}{gt}{m}{n}
4732 \g@addto@macro\bxjs@begin@document@hook{%
4733 \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。

```

4734 \@ifpackageloaded{bm}{\bxjs@patch@RDMA@for@bm}{}%
4735 \reDeclareMathAlphabet{\mathrm}{\@mathrm}{\@mathmc}%
4736 \reDeclareMathAlphabet{\mathbf}{\@mathbf}{\@mathgt}%
4737 \reDeclareMathAlphabet{\mathsf}{\@mathsf}{\@mathgt}%
4738 \fi}
4739 \fi

```

## C.5 pdfTeX 用設定：CJK + bxcjkatype

```

4740 \else\if \if p\jsEngine T\else\if n\jsEngine T\else F\fi\fi T

```

■`bxcjkatype` パッケージの読込 `\jsJaFont` が指定されている場合は、その値を `bxcjkatype` 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 `bxcjkatype` の側で行われる。

※ Pandoc モードでは `autotilde` を指定しない。

```

4741 \bxjs@adjust@jafont{f}
4742 \let\bxjs@jafont@paren\@firstofone

```

```

4743 \bxjs@resolve@jafont@paren\bxjs@tmpa
4744 \edef\bxjs@tmpa{\bxjs@catopt\bxjs@tmpa\bxjs@jp@font}
4745 \edef\bxjs@tmpa{\bxjs@catopt\bxjs@tmpa{whole}}
4746 \ifx\bxjs@jadriver\bxjs@@pandoc\else
4747   \edef\bxjs@tmpa{\bxjs@catopt\bxjs@tmpa{autotilde}}
4748 \fi
4749 \edef\bxjs@next{%
4750   \noexpand\RequirePackage[\bxjs@tmpa]{bxcjkatype}[2013/10/15]% v0.2c
4751 } \bxjs@next
4752 \bxjs@cjk@loaded

```

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

```

4753 \ifbxjs@hyperref@enc
4754   \PassOptionsToPackage{unicode}{hyperref}
4755 \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。

※ bxcjkatype 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。

```

4756 \ifx\bxcjkatypeHyperrefPatchDone\@undefined
4757 \begingroup
4758   \CJK@input{UTF8.bdg}
4759 \endgroup
4760 \g@addto@macro\pdfstringdefPreHook{%
4761   \@nameuse{CJK@UTF8Binding}%
4762 }
4763 \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。

```

4764 \ifx\bxcjkatypeHyperrefPatchDone\@undefined
4765 \g@addto@macro\pdfstringdefPreHook{%
4766   \ifx~\bxjs@@CJKtilde
4767     \let\bxjs@org@LetUnexpandableSpace\HyPsd@LetUnexpandableSpace
4768     \let\HyPsd@LetUnexpandableSpace\bxjs@LetUnexpandableSpace
4769     \let~\@empty
4770   \fi
4771 }
4772 \def\bxjs@@CJKtilde{\CJKecglue\ignorespaces}
4773 \def\bxjs@@tildecmd{~}
4774 \def\bxjs@LetUnexpandableSpace#1{%
4775   \def\bxjs@tmpa{#1}\ifx\bxjs@tmpa\bxjs@@tildecmd\else
4776     \bxjs@org@LetUnexpandableSpace#1%
4777   \fi}
4778 \fi

```

■共通命令の実装

```

4779 \newskip\jsKanjiSkip
4780 \newskip\jsXKanjiSkip
4781 \ifx\CJKecglue\@undefined
4782   \def\CJKtilde{\CJK@global\def~{\CJKecglue\ignorespaces}}
4783 \fi
4784 \newcommand*{\setkanjiskip}{\jsSetKanjiSkip}
4785 \newcommand*{\getkanjiskip}{\jsGetKanjiSkip}
4786 \newcommand*{\autospace}{\jsEnableKanjiSkip}
4787 \newcommand*{\noautospace}{\jsDisableKanjiSkip}
4788 \protected\def\bxjs@CJKglue{\hskip\jsKanjiSkip}
4789 \def\jsApplyKanjiSkip#1{%
4790   \jsKanjiSkip=#1\relax
4791   \let\CJKglue\bxjs@CJKglue}
4792 \newcommand*{\setxkanjiskip}{\jsSetXKanjiSkip}
4793 \newcommand*{\getxkanjiskip}{\jsGetXKanjiSkip}
4794 \newcommand*{\autoxspace}{\jsEnableXKanjiSkip}
4795 \newcommand*{\noautoxspace}{\jsDisableXKanjiSkip}
4796 \protected\def\bxjs@CJKecglue{\hskip\jsXKanjiSkip}
4797 \def\jsApplyXKanjiSkip#1{%
4798   \jsXKanjiSkip=#1\relax
4799   \let\CJKecglue\bxjs@CJKecglue}

```

\jachar のサブマクロの実装。

```

4800 \def\bxjs@jachar#1{%
4801   \CJKforced{#1}}

```

和欧文間空白の命令 \jathinspace の実装。

```

4802 \ifbxjs@jaspace@cmd
4803   \protected\def\jathinspace{\CJKecglue}
4804 \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。

```

4805 \ifx t\bxjs@enablejfam
4806   \ClassWarningNoLine\bxjs@clsname
4807     {You cannot use 'enablejfam=true', since the\MessageBreak
4808       CJK package does not support Japanese math}
4809 \fi

```

## C.6 Xe<sub>La</sub>TeX 用設定：xeCJK + zxjatype

```

4810 \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。

```

4811 \RequirePackage{zxjatype}
4812 \PassOptionsToPackage{no-math}{fontspec}%!
4813 \PassOptionsToPackage{xetex}{graphicx}%!
4814 \PassOptionsToPackage{xetex}{graphics}%!
4815 \ifx\zxJaFamilyName\@undefined

```

```

4816 \ClassError\bxjs@clsname
4817 {xeCJK or zxjatype is too old}\@ehc
4818 \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。

```

4819 \bxjs@adjust@jafont{f}
4820 \let\bxjs@jafont@paren\@gobble
4821 \bxjs@resolve@jafont@paren\bxjs@tmpa
4822 \edef\bxjs@tmpa{\bxjs@catopt\bxjs@tmpa\bxjs@jp@font}
4823 \ifx\bxjs@tmpa\@empty
4824 \setCJKmainfont[BoldFont=HaranoAjiGothic-Medium.otf]{HaranoAjiMincho-
Regular.otf}
4825 \setCJKsansfont[BoldFont=HaranoAjiGothic-Medium.otf]{HaranoAjiGothic-
Medium.otf}
4826 \else
4827 \edef\bxjs@next{%
4828 \noexpand\RequirePackage[\bxjs@tmpa]{zxjafont}[2013/01/28]% v0.2a
4829 }\bxjs@next
4830 \fi

```

■hyperref 対策 unicode オプションの指定に関する話。

X<sub>Ǝ</sub>TeX 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、X<sub>Ǝ</sub>TeX 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、「X<sub>Ǝ</sub>TeX 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。

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

**TODO:3.0** バージョン要件を見直して暫定措置を解除する。

```

4831 \ifx\strcmp\undefined\else % 未定義なら条件を満たさない
4832 \ifnum\strcmp{\the\XeTeXversion\XeTeXrevision}{0.99992}>\m@ne
4833 \ifbxjs@hyperref@enc
4834 \PassOptionsToPackage{unicode}{hyperref}
4835 \fi
4836 \fi
4837 \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。

```
4838 \AtEndOfClass{%
4839 \def\@gnewline #1{%
4840   \ifvmode \@nolnerr
4841   \else
4842     \unskip \reserved@e {\reserved@f#1}\nobreak \hfil \break \null
4843     \nobreak \hskip-1sp\hskip1sp\relax
4844     \ignorespaces
4845   \fi}
4846 }
```

## ■共通命令の実装

```
4847 \newskip\jsKanjiSkip
4848 \newskip\jsXKanjiSkip
4849 \ifx\CJKecglue\@undefined
4850   \def\CJKtilde{\CJK@global\def~{\CJKecglue\ignorespaces}}
4851 \fi
4852 \newcommand*{\setkanjiskip}{\jsSetKanjiSkip}
4853 \newcommand*{\getkanjiskip}{\jsGetKanjiSkip}
4854 \newcommand*{\autospaceing}{\jsEnableKanjiSkip}
4855 \newcommand*{\noautospaceing}{\jsDisableKanjiSkip}
4856 \protected\def\bxjs@CJKglue{\hskip\jsKanjiSkip}
4857 \def\jsApplyKanjiSkip#1{%
4858   \jsKanjiSkip=#1\relax
4859   \xeCJKsetup{CJKglue={\bxjs@CJKglue}}}
4860 \newcommand*{\setxkanjiskip}{\jsSetXKanjiSkip}
4861 \newcommand*{\getxkanjiskip}{\jsGetXKanjiSkip}
4862 \newcommand*{\autoxspaceing}{\jsEnableXKanjiSkip}
4863 \newcommand*{\noautoxspaceing}{\jsDisableXKanjiSkip}
4864 \protected\def\bxjs@CJKecglue{\hskip\jsXKanjiSkip}
4865 \def\jsApplyXKanjiSkip#1{%
4866   \jsXKanjiSkip=#1\relax
4867   \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。

```
4868 \ifx\mcfamily\@undefined
4869   \protected\def\mcfamily{\CJKfamily{\CJKrmdefault}}
4870   \protected\def\gtfamily{\CJKfamily{\CJKsfdefault}}
4871 \fi
```

\jachar のサブマクロの実装。

```
4872 \def\bxjs@jachar#1{%
4873   \xeCJKDeclareCharClass{CJK}{`#1}\relax
4874   #1}
```



`\jathinspace` の実装。

```
4875 \ifbxjs@jaspace@cmd
4876   \protected\def\jathinspace{\CJKecglue}
4877 \fi
```

■和文数式ファミリー 和文数式ファミリー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。

```
4878 \ifx t\bxjs@enablejfam
4879   \@enablejfamtrue
4880 \fi
```

実際に和文用の数式ファミリーの設定を行う。

※ FIXME: 要検討。

```
4881 \if@enablejfam
4882   \xeCJKsetup{CJKmath=true}
4883 \fi
```

## C.7 LuaTeX 用設定: LuaTeX-ja

```
4884 \else\if 1\jsEngine
```

■LuaTeX-ja パッケージの読み込み `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。

```
4885 \let\zw\@undefined
4886 \RequirePackage{luatexja}
4887 \edef\bxjs@next{%
4888   \noexpand\RequirePackage[scale=\jsScale]{luatexja-fontspec}[2015/08/26]%
4889 }\bxjs@next
```

`\set@fontsize` へのパッチ適用を再度行う。

```
4890 \bxjs@patch@set@fontsize
```

フォント代替の明示的定義。

```
4891 \DeclareFontShape{JY3}{mc}{m}{it}{<->ssub*mc/m/n}{}
4892 \DeclareFontShape{JY3}{mc}{m}{sl}{<->ssub*mc/m/n}{}
4893 \DeclareFontShape{JY3}{mc}{m}{sc}{<->ssub*mc/m/n}{}
4894 \DeclareFontShape{JY3}{gt}{m}{it}{<->ssub*gt/m/n}{}
4895 \DeclareFontShape{JY3}{gt}{m}{sl}{<->ssub*gt/m/n}{}
4896 \DeclareFontShape{JY3}{mc}{bx}{it}{<->ssub*gt/m/n}{}
4897 \DeclareFontShape{JY3}{mc}{bx}{sl}{<->ssub*gt/m/n}{}
4898 \DeclareFontShape{JY3}{gt}{bx}{it}{<->ssub*gt/m/n}{}
4899 \DeclareFontShape{JY3}{gt}{bx}{sl}{<->ssub*gt/m/n}{}

```

```

4900 \DeclareFontShape{JY3}{mc}{b}{n}{<->ssub*mc/bx/n}{}
4901 \DeclareFontShape{JY3}{mc}{b}{it}{<->ssub*mc/bx/n}{}
4902 \DeclareFontShape{JY3}{mc}{b}{sl}{<->ssub*mc/bx/n}{}
4903 \DeclareFontShape{JY3}{gt}{b}{n}{<->ssub*gt/bx/n}{}
4904 \DeclareFontShape{JY3}{gt}{b}{it}{<->ssub*gt/bx/n}{}
4905 \DeclareFontShape{JY3}{gt}{b}{sl}{<->ssub*gt/bx/n}{}
4906 \DeclareFontShape{JT3}{mc}{m}{it}{<->ssub*mc/m/n}{}
4907 \DeclareFontShape{JT3}{mc}{m}{sl}{<->ssub*mc/m/n}{}
4908 \DeclareFontShape{JT3}{mc}{m}{sc}{<->ssub*mc/m/n}{}
4909 \DeclareFontShape{JT3}{gt}{m}{it}{<->ssub*gt/m/n}{}
4910 \DeclareFontShape{JT3}{gt}{m}{sl}{<->ssub*gt/m/n}{}
4911 \DeclareFontShape{JT3}{mc}{bx}{it}{<->ssub*gt/m/n}{}
4912 \DeclareFontShape{JT3}{mc}{bx}{sl}{<->ssub*gt/m/n}{}
4913 \DeclareFontShape{JT3}{gt}{bx}{it}{<->ssub*gt/m/n}{}
4914 \DeclareFontShape{JT3}{gt}{bx}{sl}{<->ssub*gt/m/n}{}
4915 \DeclareFontShape{JT3}{mc}{b}{n}{<->ssub*mc/bx/n}{}
4916 \DeclareFontShape{JT3}{mc}{b}{it}{<->ssub*mc/bx/n}{}
4917 \DeclareFontShape{JT3}{mc}{b}{sl}{<->ssub*mc/bx/n}{}
4918 \DeclareFontShape{JT3}{gt}{b}{n}{<->ssub*gt/bx/n}{}
4919 \DeclareFontShape{JT3}{gt}{b}{it}{<->ssub*gt/bx/n}{}
4920 \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。

```

4921 \bxjs@adjust@jafont{t}
4922 \ifx\bxjs@tmpa\bxjs@noEmbed
4923   \def\bxjs@tmpa{noembed}
4924 \fi
4925 \let\bxjs@jafont@paren@gobble
4926 \bxjs@resolve@jafont@paren\bxjs@tmpa
4927 \edef\bxjs@tmpa{\bxjs@catopt\bxjs@tmpa\bxjs@jp@font}
4928 \ifx\bxjs@tmpa@empty
4929   \defaultjfontfeatures{ Kerning=Off }
4930   \setmainjfont[BoldFont=HaranoAjiGothic-Medium.otf,JFM=ujis]{HaranoAjiMincho-
     Regular.otf}
4931   \setsansjfont[BoldFont=HaranoAjiGothic-Medium.otf,JFM=ujis]{HaranoAjiGothic-
     Medium.otf}
4932 \else
4933   \edef\bxjs@next{%
4934     \noexpand\RequirePackage[\bxjs@tmpa]{luatexja-preset}%
4935   }\bxjs@next
4936 \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。

```

4937 \@ifpackagelater{luatexja}{2016/03/31}{}{\%else

```

```

4938 \DeclareRobustCommand\rmfamily
4939 {\not@math@alphabet\rmfamily\mathrm
4940 \romanfamily\rmdefault\kanjifamily\mcdefault\selectfont}
4941 \DeclareRobustCommand\sffamily
4942 {\not@math@alphabet\sffamily\mathsf
4943 \romanfamily\sfdefault\kanjifamily\gtdefault\selectfont}
4944 \DeclareRobustCommand\ttfamily
4945 {\not@math@alphabet\ttfamily\mathtt
4946 \romanfamily\ttdefault\kanjifamily\gtdefault\selectfont}
4947 }
4948 \long\def\jttdefault{\gtdefault}
4949 \unless\ifx\@ltj@match@familytrue\@undefined
4950 \@ltj@match@familytrue
4951 \fi
4952 \g@addto@macro\bxjs@begin@document@hook{%
4953 \@ifpackageloaded{bm}{\bxjs@patch@RDMA@for@bm}{}%
4954 \reDeclareMathAlphabet{\mathrm}{\mathrm}{\mathmc}%
4955 \reDeclareMathAlphabet{\mathbf}{\mathbf}{\mathgt}%
4956 \reDeclareMathAlphabet{\mathsf}{\mathsf}{\mathgt}}%
4957 \bxjs@if@sf@default{%
4958 \renewcommand\kanjifamilydefault{\gtdefault}}

```

## ■和文パラメタの設定

```

4959 % 次の3つは既定値の通り
4960 %\ltjsetparameter{prebreakpenalty={`',10000}}
4961 %\ltjsetparameter{postbreakpenalty={`",10000}}
4962 %\ltjsetparameter{prebreakpenalty={`",10000}}
4963 \ltjsetparameter{jaxspmode={`!,1}}
4964 \ltjsetparameter{jaxspmode={`〒,2}}
4965 \ltjsetparameter{alxspmode={`+,3}}
4966 \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 Lua<sub>TEX</sub>-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備えて同等のものを用意する。

```

4967 \ifx\ltjfakeparbegin\@undefined
4968 \protected\def\ltjfakeparbegin{%
4969 \ifhmode
4970 \relax\directlua{%
4971 \luatexja.jfmglue.create_beginpar_node()}}
4972 \fi}
4973 \fi

```

`ltjs*` クラスの定義と同等になるようにパッチを当てる。

```

4974 \unless\ifnum\bxjs@everyparhook=\bxjs@everyparhook@@none
4975 \begingroup

```

```

4976 \let\% \@percentchar \def\@#1{[[\detokenize{#1}]]}
4977 \@gobble\if\def\bxjs@tmpa{\@{\everypar{}\fi}}
4978 \ifnum\bxjs@everyparhook=\bxjs@everyparhook@@compat
4979   \@gobble\if\def\bxjs@tmpa{\@{\everypar{\everyparhook}\fi}}\fi
4980 \directlua{
4981   local function patchcmd(cs, code, from, to)
4982     tex.sprint(code:gsub(from:gsub("%W", "%\\%\\%0"), "%0"..to)
4983       :gsub("macro:", \@gdef..cs, 1):gsub("->", "{", 1)..")
4984   end
4985   patchcmd(\@xsect, [[\meaning\xsect]],
4986     \@{\hskip-\@tempskipa}, \@{\ltjfakeparbegin})
4987   patchcmd(\@item, [[\meaning\@item]],
4988     \bxjs@tmpa, \@{\ltjfakeparbegin})
4989 \endgroup
4990 \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。

```

4991 \ifbxjs@hyperref@enc
4992   \PassOptionsToPackage{unicode}{hyperref}
4993   \bxjs@check@hyperref@unicode{true}
4994 \fi

```

### ■共通命令の実装

```

4995 \newcommand*{\setkanjiskip}{\jsSetKanjiSkip}
4996 \newcommand*{\getkanjiskip}{\jsGetKanjiSkip}
4997 \newcommand*{\setxkanjiskip}{\jsSetXKanjiSkip}
4998 \newcommand*{\getxkanjiskip}{\jsGetXKanjiSkip}
4999 \protected\def\autospacing{%
5000   \ltjsetparameter{autospacing=true}}
5001 \protected\def\noautospacing{%
5002   \ltjsetparameter{autospacing=false}}
5003 \protected\def\autoxspacing{%
5004   \ltjsetparameter{autoxspacing=true}}
5005 \protected\def\noautoxspacing{%
5006   \ltjsetparameter{autoxspacing=false}}
5007 \def\jsApplyKanjiSkip#1{%
5008   \ltjsetparameter{kanjiskip={#1}}}
5009 \def\jsApplyXKanjiSkip#1{%
5010   \ltjsetparameter{xkanjiskip={#1}}}

```

\jachar のサブマクロの実装。

```

5011 \def\bxjs@jachar#1{%
5012   \ltjjachar`#1\relax}

```

\jathinspace の実装。

```

5013 \ifbxjs@jaspace@cmd
5014   \protected\def\jathinspace{%
5015     \hskip\ltjgetparameter{xkanjiskip}\relax}

```

```
5016 \fi
```

■和文数式ファミリー LuaTeX-jd では和文数式ファミリーは常に有効で、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。

```
5017 \ifx f\bxjs@enablejfam
5018 \ClassWarningNoLine\bxjs@clsname
5019 {You cannot use 'enablejfam=false', since the\MessageBreak
5020 LuaTeX-jd always provides Japanese math families}
5021 \fi
```

## C.8 共通処理 (2)

```
5022 \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 5023 \ifx\DeclareFixJFMCJKTextFontCommand\@undefined
5024 \DeclareJaTextFontCommand{\textmc}{\mcfamily}
5025 \DeclareJaTextFontCommand{\textgt}{\gtfamily}
5026 \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

\mathgt バックの定義を行う。

```
5027 \ifx\mathmc\@undefined
5028 \DeclareJaMathFontCommand{\mathmc}{\mcfamily}
5029 \DeclareJaMathFontCommand{\mathgt}{\gtfamily}
5030 \fi
```

以上で終わり。

```
5031 %</standard>
```

## 付録 D 和文ドライバ：modern

モダンな設定。

standard ドライバの設定を引き継ぐ。

```
5032 %<*modern>
5033 \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。

```
5034 \ifnum0\if x\jsEngine1\fi\if 1\jsEngine1\fi=\z0
5035 \def\encodingdefault{T1}%
5036 \input{t1enc.def}%
```

```
5037 \fontencoding\encodingdefault\selectfont
5038 \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。

```
5039 \ifnum0\if x\jsEngine1\fi\if 1\jsEngine1\fi=\z@
5040 \renewcommand{\rmdefault}{lmr}
5041 \renewcommand{\sfdefault}{lms}
5042 \renewcommand{\ttdefault}{lmtt}
5043 \fi
```

大型演算子用の数式フォントの設定。

※ `amsmath` パッケージと同等にする。

```
5044 \DeclareFontShape{OMX}{cmex}{m}{n}{%
5045   <-7.5>cmex7<7.5-8.5>cmex8%
5046   <8.5-9.5>cmex9<9.5->cmex10}{}%
5047 \expandafter\let\csname OMX/cmex/m/n/10\endcsname\relax
amsmath 読み込時に上書きされるのを防ぐ。
5048 \def\cmex@opt{10}
```

## D.2 fixltx2e 読み込

※ `fixltx2e` 廃止前の L<sup>A</sup>T<sub>E</sub>X カーネルの場合。

```
5049 \ifx\@IncludeInRelease\@undefined
5050 \RequirePackage{fixltx2e}
5051 \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。

```
5052 \RequirePackage{bxjscjkcat}
```

## D.4 完了

おしまい。

```
5053 %</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

```
5054 %<*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。

```
5055 \if x\jsEngine
5056   \PassOptionsToPackage{nospace}{xeCJK}
5057 \fi
```

standard ドライバの設定を引き継ぐ。

```
5058 \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  $\epsilon$ -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。

```
5059 \RequirePackage{bxjspandoc}
```

$\epsilon$ -TeX ではない場合に警告を出す。

```
5060 \ifjsWitheTeX\else
5061   \ClassWarningNoLine{bxjs@clsname}
5062   {!!!!!!! WARNING !!!!!!!\MessageBreak
5063     This engine does not support e-TeX extension!\MessageBreak
5064     Some feature might not work properly}
5065 \fi
```

`\ifbxjs@bxghost@available` [スイッチ] bxghost パッケージが利用できるか。

```
5066 \newif\ifbxjs@bxghost@available
5067 \ifjsWitheTeX
5068   \RequirePackage{pdftexcmds}[2009/09/22]% v0.5
5069   \IfFileExists{bxghost.sty}{%
5070     \bxjs@bxghost@availabletrue
5071     \@namedef{bxjs@bgbv/79E70A0991967E27981832C84DB5DF99}{1}%v0.2.0
5072     \ifx\pdf@filemdfivesum\undefined\else
5073       \expandafter\ifx\csgname bxjs@bgbv/\pdf@filemdfivesum{bxghost.sty}%
5074       \endcsname\relax\else \bxjs@bxghost@availablefalse \fi
5075   \fi
5076 }{}
5077 \fi
```

その他の依存パッケージを読み込む。

```

5078 \RequirePackage{iftex}[2013/04/04]% v0.2
5079 \ifjsWitheTeX
5080 \RequirePackage{etoolbox}[2010/08/21]% v2.0
5081 \RequirePackage{filehook}[2011/10/12]% v0.5d
5082 \fi

```

## E.2 和文ドライバパラメタ

keyval のファミリーは bxjsPan とする。

`\ifbxjs@jp@fix@strong` 重要要素を補正するか。

```

5083 \newif\ifbxjs@jp@fix@strong \bxjs@jp@fix@strongtrue

fix-strong オプションの処理。

5084 \let\bxjs@kv@fixstrong@true\bxjs@jp@fix@strongtrue
5085 \let\bxjs@kv@fixstrong@false\bxjs@jp@fix@strongfalse
5086 \define@key{bxjsPan}{fix-strong}[true]{%
5087 \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。

```

5088 \newif\ifbxjs@jp@fix@code \bxjs@jp@fix@codetrue

fix-code オプションの処理。

5089 \let\bxjs@kv@fixcode@true\bxjs@jp@fix@codetrue
5090 \let\bxjs@kv@fixcode@false\bxjs@jp@fix@codefalse
5091 \define@key{bxjsPan}{fix-code}[true]{%
5092 \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。

```

5093 \chardef\bxjs@jp@strong=0

strong オプションの処理。

5094 \def\bxjs@kv@strong@bold{\chardef\bxjs@jp@strong=0 }
5095 \def\bxjs@kv@strong@sans{\chardef\bxjs@jp@strong=1 }
5096 \def\bxjs@kv@strong@boldsans{\chardef\bxjs@jp@strong=2 }
5097 \define@key{bxjsPan}{strong}{%
5098 \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 5099 \newif\ifbxjs@jp@or@indent \bxjs@jp@or@indenttrue
\ifbxjs@jp@or@block@heading 5100 \newif\ifbxjs@jp@or@secnumdepth \bxjs@jp@or@secnumdepthtrue
5101 \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。

```

5102 \define@key{bxjsPan}{_plus}[]{}
5103 \bxjs@jp@or@indentfalse

```



```

5104 \bxjs@jp@or@secnumdepthfalse
5105 \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。

```

5106 \let\bxjs@kv@orindent@true\bxjs@jp@or@indenttrue
5107 \let\bxjs@kv@orindent@false\bxjs@jp@or@indentfalse
5108 \define@key{bxjsPan}{or-indent}[true]{%
5109 \bxjs@set@keyval{orindent}{#1}{}}
5110 \let\bxjs@kv@orsecnumdepth@true\bxjs@jp@or@secnumdepthtrue
5111 \let\bxjs@kv@orsecnumdepth@false\bxjs@jp@or@secnumdepthfalse
5112 \define@key{bxjsPan}{or-secnumdepth}[true]{%
5113 \bxjs@set@keyval{orsecnumdepth}{#1}{}}
5114 \let\bxjs@kv@orblockheading@true\bxjs@jp@or@block@headingtrue
5115 \let\bxjs@kv@orblockheading@false\bxjs@jp@or@block@headingfalse
5116 \define@key{bxjsPan}{or-block-heading}[true]{%
5117 \bxjs@set@keyval{orblockheading}{#1}{}}

```

実際の japaram の値を適用する。

```

5118 \def\bxjs@next#1{\bxjs@safe@setkeys{bxjsPan}{#1}}
5119 \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み込みが `\@filewithoptions` で指示されて、しかもそのファイルが読み込み済である場合に、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。

```

5120 \@onlypreamble\bxjs@set@dupload@proc
5121 \def\bxjs@set@dupload@proc#1{%
5122 \expandafter\bxjs@set@dupload@proc@a\csname bxjs@dlp/#1\endcsname}
5123 \@onlypreamble\bxjs@set@dupload@proc@a
5124 \def\bxjs@set@dupload@proc@a#1{%
5125 \@onlypreamble#1\def#1#1}
5126 \def\bxjs@unset@dupload@proc#1{%
5127 \bxjs@cslet{bxjs@dlp/#1}\@undefined}

```

`\@if@options \@if@options` の再定義。

```

5128 \@onlypreamble\bxjs@org@if@options
5129 \let\bxjs@org@if@options\@if@options
5130 \@onlypreamble\bxjs@org@reset@options
5131 \let\bxjs@org@reset@options\relax

```

```

5132 \def\@if@options#1#2#3{%
5133   \let\bxjs@next\@secondoftwo
5134   \def\bxjs@tmpa{#1}\def\bxjs@tmpb{\@currentx}%
5135   \ifx\bxjs@tmpa\bxjs@tmpb
5136     \expandafter\ifx\csname bxjs@dlp/#2.#1\endcsname\relax\else
5137     \let\bxjs@next\@firstoftwo \fi
5138   \fi
5139   \bxjs@next\bxjs@do@dupload@proc\bxjs@org@if@options{#1}{#2}{#3}}
5140 \g@addto@macro\bxjs@begin@document@hook{%
5141   \let\@if@options\bxjs@org@if@options}
5142 \@onlypreamble\bxjs@do@dupload@proc
5143 \def\bxjs@do@dupload@proc#1#2#3{%
5144   \ifx\bxjs@org@reset@options\relax
5145     \let\bxjs@org@reset@options\@reset@options
5146   \fi
5147   \bxjs@csletcs{bxjs@next}{bxjs@dlp/#2.#1}%
5148   \def\@reset@options{%
5149     \let\@reset@options\bxjs@org@reset@options
5150     \@reset@options
5151     \bxjs@next{#3}}%
5152   \@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 L<sup>A</sup>T<sub>E</sub>X 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 X<sub>Y</sub>L<sup>A</sup>T<sub>E</sub>X か Lua<sup>A</sup>T<sub>E</sub>X 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。

```
5153 \let\bxjs@polyglossia@options\relax
```

エンジンが X<sub>Y</sub>L<sup>A</sup>T<sub>E</sub>X か Lua<sup>A</sup>T<sub>E</sub>X 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。

```
5154 \ifnum0\if x\jsEngine1\fi\if 1\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。

```
5155 \pandocSkipLoadPackage{polyglossia}
5156 \bxjs@set@dupload@proc{polyglossia.sty}{%
5157   \bxjs@unset@dupload@proc{polyglossia.sty}%
5158   \ClassWarning\bxjs@clsname
5159   {Package polyglossia is requested}%
5160   \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。

```
5161 \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。

```
5162   \ifboolexpr{test{\ifblank{##2}}or test{\ifstrequal{##2}{japanese}}}{%
5163     \ClassWarning\bxjs@clsname
5164     {Main language is 'japanese', thus fallback\MessageBreak
5165       definitions will be employed}%
5166     \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。

```
5167   }{%else
5168     \ClassWarning\bxjs@clsname
5169     {Main language is '##2',\MessageBreak
5170       thus polyglossia will be loaded}%
5171     \csundef{ver@polyglossia.sty}%
5172     \edef\bxjs@next{%
5173       \noexpand\RequirePackage[\bxjs@polyglossia@options]{polyglossia}[]%
5174     }\bxjs@next
5175     \setmainlanguage[##1]{##2}%
5176   }}
```

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。

```
5177 \g@addto@macro\bxjs@endpreamble@hook{%
5178   \ifx\bxjs@polyglossia@options\relax
5179     \csundef{ver@polyglossia.sty}%
5180   \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。

```
5181 \@onlypreamble\bxjs@pandoc@polyglossia@ja
5182 \def\bxjs@pandoc@polyglossia@ja{%
```

```

5183 \renewcommand*\setmainlanguage[2] [] {%
5184 \newcommand*\setotherlanguage[2] [] {%
5185   \ifblank{##2}{-}{%else
5186     \cslet{##2}\@empty \cslet{end##2}\@empty
5187     \cslet{text##2}\@firstofone}}%
5188 \newcommand*\setotherlanguages[2] [] {%
5189   \@for\bxjs@tmpa:={##2}\do{%
5190     \setotherlangauge{\bxjs@tmpa}}}%

```

Polyglossia の読込済マークは外れるようにしておく。

```

5191 \let\bxjs@polyglossia@options\relax}%

5192 \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。

※ X<sub>Y</sub>TEX 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。

```

5193 \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。

```

5194 \pandocSkipLoadPackage{babel}
5195 \bxjs@set@dupload@proc{babel.sty}{%
5196   \bxjs@unset@dupload@proc{babel.sty}%
5197   \ClassWarning\bxjs@clsname
5198     {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。

5199 \@tempswafalse \let\bxjs@babel@options\@empty
5200 \def\bxjs@tmpb{main=}%
5201 \@for\bxjs@tmpa:=#1\do{%
5202   \ifx\bxjs@tmpa\bxjs@tmpb \def\bxjs@tmpa{main=japanese}\fi
5203   \edef\bxjs@babel@options{\bxjs@babel@options,\bxjs@tmpa}}%
5204 \bxjs@cslet{ver@babel.sty}\@undefined
5205 \edef\bxjs@next{%
5206   \noexpand\RequirePackage[\bxjs@babel@options]{babel}\relax
5207 }\bxjs@next
5208 \RequirePackage{bxorigcapt}\relax}

```

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

```

5209 \g@addto@macro\bxjs@endpreamble@hook{%
5210   \ifx\bxjs@babel@options\relax
5211     \bxjs@cslet{ver@babel.sty}\@undefined
5212   \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。

```
5213 \ifjsWitheTeX
```

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

```
5214 \AtBeginOfFile{japanese.ldf}{\bxjs@begin@japanese@ldf@hook}
5215 \def\bxjs@begin@japanese@ldf@hook{%
5216   \let\bxjs@begin@japanese@ldf@hook\relax
5217   \let\bxjs@save@ProvidesLanguage\ProvidesLanguage
5218   \let\bxjs@save@LdfInit\LdfInit
5219   \def\ProvidesLanguage##1[##2]{\bxjs@do@japanese@ldf{##2}}%
5220   \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。

```
5221 \def\bxjs@do@japanese@ldf#1{\bxjs@do@japanese@ldf@a#1\@nil}
5222 \def\bxjs@do@japanese@ldf@a#1/#2/#3#4#5\@nil{%
5223   \let\LdfInit\bxjs@save@LdfInit
5224   \ClassInfo\bxjs@clsname
5225   {Release date of japanese.ldf is:\MessageBreak
5226     \@spaces #1/#2/#3#4\@gobble}%
5227   \ifnum#1#2#3#4<20201206 % v3.0
5228     \let\bxjs@japanese@ldf@skipped=t\csuse{endingput}%
5229   \fi}
5230 \AtEndOfFile{japanese.ldf}{\bxjs@end@japanese@ldf@hook}
5231 \def\bxjs@end@japanese@ldf@hook{%
5232   \let\bxjs@end@japanese@ldf@hook\relax
5233   \let\ProvidesLanguage\bxjs@save@ProvidesLanguage
5234   \let\LdfInit\bxjs@save@LdfInit
5235   \ifx t\bxjs@japanese@ldf@skipped
5236     \ClassWarningNoLine\bxjs@clsname
5237     {Loading japanese.ldf is skipped}%
5238   \fi}
```

ダミーの言語定義。

```
5238 \ifundef\l@japanese{\chardef\l@japanese\z0}{}%
5239 \let\datejapanese\@empty\let\captionjapanese\@empty
5240 \let\extrajapanese\@empty\let\noextrajapanese\@empty
5241 \main@language{japanese}%
5242 \fi}
5243 \g@addto@macro\bxjs@begin@document@hook{%
5244   \let\bxjs@begin@japanese@ldf@hook\relax
5245   \let\bxjs@end@japanese@ldf@hook\relax}
5246 \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。

```
5247 \bxjs@set@dupload@proc{geometry.sty}{%
5248   \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 \setmainfont の指定にまわす。

```
5249 \if l\jsEngine
5250   \pandocSkipLoadPackage{xeCJK}
5251   \providecommand*\setCJKmainfont{\setmainfont}
5252 \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。

```
5253 \if x\jsEngine
5254   \expandafter\g@addto@macro\csname opt@xeCJK.sty\endcsname{%
5255     ,space}
5256 \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。

```
5257 \ifjsWithTeX
5258 \onlypreamble\bxjs@info@or@ban
5259 \def\bxjs@info@or@ban#1{%
5260   \PackageInfo\bxjs@clsname
5261   {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。

```
5262 \unless\ifbxjs@jp@or@indent
5263   \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。

```
5264 \IfFileExists{parskip.sty}{%
5265   \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。

```
5266 }{%else
5267   \eappto\bxjs@endpreamble@hook{%
5268     \parindent=\the\parindent\relax
5269     \parskip=\the\parskip\relax}}
5270 \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。

```
5271 \ifbxjs@jp@or@secnumdepth\else
5272   \bxjs@info@or@ban{secnumdepth}
5273   \eappto\bxjs@endpreamble@hook{%
5274     \c@secnumdepth=\the\c@secnumdepth\relax}
5275 \fi
```

■**block-heading** について `\paragraph`、`\subparagraph` を別行見出しに変える処理を抑制する。

※ 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。

```
5276 \ifbxjs@jp@or@block@heading\else
5277   \let\bxjs@frozen@paragraph\paragraph
5278   \let\bxjs@frozen@subparagraph\subparagraph
5279   \bxjs@info@or@ban{block-heading}
5280   \appto\bxjs@endpreamble@hook{%
5281     \let\oldparagraph\@undefined
5282     \let\paragraph\bxjs@frozen@paragraph
5283     \let\subparagraph\bxjs@frozen@subparagraph}
5284 \fi
```

以上。

```
5285 \fi
```

## 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。

```
5286 \g@addto@macro\bxjs@begin@document@hook{%
5287   \@tempwafalse
```

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。

```
5288 \ifx\oldparagraph\undefined\else
5289   \@tempwattrue
5290 \fi
```

`\paragraph` が番号付きの場合は、マーク変更が必要である。

```
5291 \ifnum\c@secnumdepth>3
5292   \@tempwattrue
5293 \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。

```
5294 \if@tempwa\ifx\jsParagraphMark\bxjs@org@paragraph@mark
5295   \let\jsParagraphMark\@empty
5296 \fi\fi}
```

## E.10 全角空白文字

L<sup>A</sup>T<sub>E</sub>X 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)pL<sup>A</sup>T<sub>E</sub>X 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。

```
5297 \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。

```
5298 \ifnum0\if x\jsEngine1\fi\if 1\jsEngine1\fi>\z@
5299   \catcode"3000=\active
5300   \begingroup \catcode`\!=7
5301   \protected\gdef!!!!3000{\pandocZWSpace}
5302   \endgroup
5303 \else\ifx\DeclareUnicodeCharacter\undefined\else
5304   \DeclareUnicodeCharacter{3000}{\bxjs@zsp@char}
5305   \bxjs@protected\def\bxjs@zsp@char{\pandocZWSpace}
5306 \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。

```
5307 \if j\jsEngine
5308   \bxjs@fix@hyperref@unicode{false}
5309 \else
5310   \bxjs@fix@hyperref@unicode{true}
5311 \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 L<sup>A</sup>T<sub>E</sub>X 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)。

```
5312 \ifbxjs@jp@fix@strong\ifbxjs@jp@strong@cmd
5313   \let\orgtextbf\textbf
5314   \DeclareRobustCommand\pandocTextbf[1]{%
5315     \begingroup
5316       \let\textbf\orgtextbf
5317       \strong{#1}%
5318     \endgroup}%
5319   \g@addto@macro\bxjs@begin@document@hook{%
5320     \let\textbf\pandocTextbf}
5321 \fi\fi
```

`\strong` の書体を設定する。

```
5322 \jsAtEndOfClass{%
5323   \ifx\strongfontdeclare\@undefined\else
5324     \ifcase\bxjs@jp@strong
5325     \or \strongfontdeclare{\sffamily}%
5326     \or \strongfontdeclare{\sffamily\bfseries}%
5327   \fi
5328 \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 L<sup>A</sup>T<sub>E</sub>X 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。

```
5329 \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。

```
5330 \ifbxjs@bxghost@available
5331   \RequirePackage[verb]{bxghost}[2020/01/31]% v0.3.0
5332   \let\bxjs@eghostguarded\eghostguarded
5333 \else
5334   \chardef\bxjs@eghost@c=23
5335   \ifx j\jsEngine \xspcode\bxjs@eghost@c=3
5336   \else\ifx l\jsEngine \ltjsetParameter{alxspmode={\bxjs@eghost@c,3}}
5337   \else\ifx x\jsEngine %no-op
5338   \else \let\bxjs@eghost@c\@undefined
5339 \fi\fi\fi
5340 \ifx\bxjs@eghost@c\@undefined\else
5341   \font\bxjs@eghost@f=ec-lmr10 at 1.23456pt
5342   \def\bxjs@pan@eghost{\bgroup\bxjs@eghost@f\bxjs@eghost@c\egroup}
5343   \def\bxjs@eghostguarded#1{%
5344     \bxjs@pan@eghost\null#1\null\bxjs@pan@eghost}
5345 \fi
```

```

5346 \fi
5347 \ifx\bxjs@eghostguarded\@undefined\else
5348 \let\orgtexttt\texttt
5349 \DeclareRobustCommand\pandocTexttt[1]{%
5350 \ifmmode \nfss@text{\ttfamily #1}%
5351 \else
5352 \ifvmode \leavevmode \fi
5353 \bxjs@eghostguarded{\begingroup\ttfamily#1\endgroup}%
5354 \fi}
5355 \g@addto@macro\bxjs@begin@document@hook{%
5356 \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。

```

5357 \ifbxjs@bxghost@available\else
5358 \expandafter\def\expandafter\verb\expandafter{%
5359 \expandafter\bxjs@pan@eghost\verb}
5360 \g@addto@macro\verb@egroup{\bxjs@pan@eghost}
5361 \fi
5362 \fi
5363 \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測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。

```

5364 \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。

```

```

5365 \def\bxjs@check@frontier{%
5366 \expandafter\bxjs@check@frontier@a\@currnamestack\noindent...\@nil}
5367 \def\bxjs@check@frontier@a#1#2#3#4#5\@nil#6{%
5368 \ifx\noindent#4#6\fi}

```

```

\bxjs@unforge@ifPDFTeX \ifPDFTeX を偽（正しい値）にする。

```

```

5369 \onlypreamble\bxjs@unforge@ifPDFTeX
5370 \def\bxjs@unforge@ifPDFTeX{\global\bxjs@csletcs@ifPDFTeX}{iffalse}}

```

\bxjs@forge@ifPDFTeX \ifPDFTeX を真（偽装した値）にする。

```

5371 \onlypreamble\bxjs@forge@ifPDFTeX
5372 \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。

```

5373 \def\bxjs@unload@forge@ifPDFTeX{%
5374   \bxjs@unforge@ifPDFTeX
5375   \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。

```

5376 \jsAtEndOfClass{\bxjs@forge@ifPDFTeX}
5377 \ifjsWitheTeX
5378   \AtBeginOfEveryFile{\bxjs@check@frontier\bxjs@unforge@ifPDFTeX}%
5379   \AtEndOfEveryFile{\bxjs@check@frontier\bxjs@forge@ifPDFTeX}%
5380   \g@addto@macro\bxjs@endpreamble@hook{\bxjs@unload@forge@ifPDFTeX}
5381 \else
5382   \g@addto@macro\bxjs@begin@document@hook{\bxjs@unload@forge@ifPDFTeX}
5383 \fi
5384 \fi

```

## E.14 完了

おしまい。

```
5385 %</pandoc>
```

和文ドライバ実装はここまで。

```
5386 %</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。
- bxjscjkat : 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。

```
5387 %<*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

```
5388 %<*compat>
5389 \def\bxac@pkgname{bxjscompat}

\bxjx@engine エンジンの種別。

5390 \let\bxac@engine=n
5391 \def\bxac@do#1#2{%
5392   \edef\bxac@tmpa{\string#1}%
5393   \edef\bxac@tmpb{\meaning#1}%
5394   \ifx\bxac@tmpa\bxac@tmpb #2\fi}
5395 \bxac@do\kanjiskip{\let\bxac@engine=j}
5396 \bxac@do\XeTeXversion{\let\bxac@engine=x}
5397 \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。

```
5398 \ifx\jsAtEndOfClass\@undefined
5399   \let\bxac@delayed@if@bxjs\@firstofone
5400 \else \let\bxac@delayed@if@bxjs\jsAtEndOfClass
5401 \fi
```

\ImposeOldLuaTeXBehavior \ImposeOldLuaTeXBehavior は 0.85 版以降の Lua<sub>T</sub><sub>E</sub>X を一時的に pdf<sub>T</sub><sub>E</sub>X と互換である  
\RevokeOldLuaTeXBehavior ように見せかける。 \RevokeOldLuaTeXBehavior で元に戻すことができる。

※エンジンが Lua<sub>T</sub><sub>E</sub>X 以外の場合は何もしない。

```
5402 \newif\ifbxac@in@old@behavior
5403 \let\ImposeOldLuaTeXBehavior\relax
5404 \let\RevokeOldLuaTeXBehavior\relax
```

## G.2 8bit 欧文 T<sub>E</sub>X

```
5405 \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。

```
5406 \@tempcnta="80 \loop \ifnum \@tempcnta<"100
5407   \catcode \@tempcnta \active
5408   \advance \@tempcnta \@ne
5409 \repeat
```

以上。

```
5410 \fi
```

## G.3 X<sub>Ǝ</sub>T<sub>E</sub>X

```
5411 \ifx x\bxac@engine
```

■文字クラスの設定 X<sub>Ǝ</sub>T<sub>E</sub>X 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 L<sup>A</sup>T<sub>E</sub>X 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、X<sub>Ǝ</sub>T<sub>E</sub>X 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。

```
5412 \ifx\XeTeXcharclass\undefined\else
5413 \bxac@delayed@if@bxjs{%
5414   \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。

```
5415   \ifx\Xe@alloc@intercharclass\undefined\else
5416     \ifnum\Xe@alloc@intercharclass=\z@
5417       \PackageInfo\bxac@pkgname
5418       {Setting up interchar class for CJK...\@gobble}%
5419       \InputIfFileExists{load-unicode-xetex-classes.tex}{%
5420         \Xe@alloc@intercharclass=3
5421       }\%else
5422       \PackageWarning\bxac@pkgname
5423       {Cannot find file 'load-unicode-xetex-classes.tex'%
5424         \@gobble}%
5425     }%
5426   \fi\fi
```

フォーマット組込んだ時代の設定は不完全なところがあるので補正する。

```
5427   \ifnum\XeTeXcharclass"3041=\z@
5428     \PackageInfo\bxac@pkgname
5429     {Adjusting interchar class for CJK...\@gobble}%
5430     \@for\bxac@tmpb:={%
5431       3041,3043,3045,3047,3049,3063,3083,3085,3087,308E,%
5432       3095,3096,30A1,30A3,30A5,30A7,30A9,30C3,30E3,30E5,%
5433       30E7,30EE,30F5,30F6,30FC,31F0,31F1,31F2,31F3,31F4,%
5434       31F5,31F6,31F7,31F8,31F9,31FA,31FB,31FC,31FD,31FE,%
5435       31FF%
5436     }\do{\XeTeXcharclass"\bxac@tmpb=\@ne}%
5437   \fi
5438 }%
5439 }
5440 \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。

```
5441 \chardef\bxac@tmpb=11
5442 \def\bxac@do#1#2{%
5443   \@tempcnta=#1\relax
5444   \unless\ifnum\catcode\@tempcnta=\bxac@tmpb
5445     \chardef\bxac@tmpa=#2\relax
5446     \@whilenum{\@tempcnta<\bxac@tmpa}\do{%
5447       \catcode\@tempcnta\bxac@tmpb \advance\@tempcnta\@ne}%
5448   \fi}
```

```
5449 \bxac@do{"4E00"}{"9FCD}
```

以上。

```
5450 \fi
```

## G.4 LuaTeX

```
5451 \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。

```
5452 %\unless\ifnum\luatexversion<80 \ifnum\luatexversion<85
```

```
5453 % \chardef\pdftexversion=200
```

```
5454 % \def\pdftexrevision{0}
```

```
5455 % \let\pdftexbanner\luatexbanner
```

```
5456 %\fi\fi
```

\ImposeOldLuaTeXBehavior 0.85 版以降であるかを検査する。

```
\RevokeOldLuaTeXBehavior 5457 \begingroup\expandafter\expandafter\expandafter\endgroup
```

```
5458 \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。

```
5459 \def\bxac@ob@list{%
```

```
5460 \do{\let}\pdfoutput{\outputmode}%
```

```
5461 \do{\let}\pdfpagewidth{\pagewidth}%
```

```
5462 \do{\let}\pdfpageheight{\pageheight}%
```

```
5463 \do{\protected\edef}\pdfhorigin{\pdfvariable horigin}}%
```

```
5464 \do{\protected\edef}\pdfvorigin{\pdfvariable vorigin}}%
```

```
5465 \def\bxac@ob@do#1#2{\begingroup
```

```
5466 \expandafter\bxac@ob@do@a\csname bxac@\string#2\endcsname{#1}#2}
```

```
5467 \def\bxac@ob@do@a#1#2#3#4{\endgroup
```

```
5468 \ifbxac@in@old@behavior \let#1#3\relax #2#3#4\relax
```

```
5469 \else \let#3#1\relax \let#1@\undefined
```

```
5470 \fi}
```

```
5471 \protected\def\ImposeOldLuaTeXBehavior{%
```

```
5472 \unless\ifbxac@in@old@behavior
```

```
5473 \bxac@in@old@behaviortrue
```

```
5474 \let\do\bxac@ob@do \bxac@ob@list
```

```
5475 \fi}
```

```
5476 \protected\def\RevokeOldLuaTeXBehavior{%
```

```
5477 \ifbxac@in@old@behavior
```

```
5478 \bxac@in@old@behaviorfalse
```

```
5479 \let\do\bxac@ob@do \bxac@ob@list
```

```
5480 \fi}
```

```
5481 \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。

```
5482 \directlua{
```

```
5483 local function range(cs, ce, cc, ff)
```

```

5484         if ff or not tex.getcatcode(cs) == cc then
5485             local setcc = tex.setcatcode
5486             for c = cs, ce do setcc(c, cc) end
5487         end
5488     end
5489     range(0x3400, 0x4DB5, 11, false)
5490     \ifnum\luatexversion>64
5491     range(0x4DB5, 0x4DBF, 11, true)
5492     range(0x4E00, 0x9FCC, 11, false)
5493     range(0x9FCD, 0x9FFF, 11, true)
5494     range(0xAC00, 0xD7A3, 11, false)
5495     range(0x20000, 0x2A6D6, 11, false)
5496     range(0x2A6D7, 0x2A6FF, 11, true)
5497     range(0x2A700, 0x2B734, 11, false)
5498     range(0x2B735, 0x2B73F, 11, true)
5499     range(0x2B740, 0x2B81D, 11, false)
5500     range(0x2B81E, 0x2B81F, 11, true)
5501     range(0x2B820, 0x2CEA1, 11, false)
5502     range(0x2CEA2, 0x2FFFD, 11, true)
5503     \fi
5504 }

```

以上。

```

5505 \fi

```

## G.5 完了

おしまい。

```

5506 %</compat>

```

## 付録 H 補助パッケージ：bxjscjkat 🍱

modern ドライバ用の和文カテゴリを適用する。

### H.1 準備

```

5507 %<*cjkcat>
5508 \def\bxjx@pkgname{bxjscjkat}
5509 \newcount\bxjx@cna
5510 \@onlypreamble\bxjx@tmpdo
5511 \@onlypreamble\bxjx@tmpdo@a
5512 \@onlypreamble\bxjx@tmpdo@b

```

\bxjx@engine エンジンの種別。

```

5513 \let\bxjx@engine=n
5514 \def\bxjx@tmpdo#1#2{%
5515     \edef\bxjx@tmpa{\string#1}%
5516     \edef\bxjx@tmpb{\meaning#1}%

```

```

5517 \ifx\bxjx@tmpa\bxjx@tmpb #2\fi}
5518 \bxjx@tmpdo\kanjiskip{\let\bxjx@engine=j}
5519 \bxjx@tmpdo\enablecjktoken{%
5520 \ifx\ucs\@undefined\else \ifnum\ucs"3000="3000
5521 \let\bxjx@engine=u\fi\fi}
5522 \bxjx@tmpdo\XeTeXversion{\let\bxjx@engine=x}
5523 \bxjx@tmpdo\pdfTeXversion{\let\bxjx@engine=p}
5524 \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。

```

5525 \def\bxjx@tmpdo#1#2{%
5526 \if#1\bxjx@engine
5527 \@ifpackageloaded{#2}{\fi}%else
5528 \PackageError\bxjx@pkgname
5529 {Package '#2' must be loaded}%
5530 {Package loading is aborted.\MessageBreak\@ehc}%
5531 \endinput}
5532 \fi}
5533 \bxjx@tmpdo{p}{bxcjkatype}
5534 \bxjx@tmpdo{x}{xeCJK}
5535 \bxjx@tmpdo{l}{luatexja}

```

古い L<sup>A</sup>T<sub>E</sub>X の場合、\TextOrMath は fixltx2e パッケージで提供される。

```

5536 \ifx\TextOrMath\@undefined
5537 \RequirePackage{fixltx2e}
5538 \fi

```

## H.2 和文カテゴリコードの設定

upL<sup>A</sup>T<sub>E</sub>X の場合、和文カテゴリコードの設定を LuaL<sup>A</sup>T<sub>E</sub>X-j<sub>a</sub> と（ほぼ）等価なものに変更  
 する。

※ LuaL<sup>A</sup>T<sub>E</sub>X-j<sub>a</sub> との相違点：A830、A960、1B000。

```

5539 \if u\bxjx@engine
5540 \@for\bxjx@tmpa:={%
5541 0080,0100,0180,0250,02B0,0300,0500,0530,0590,0600,%
5542 0700,0750,0780,07C0,0800,0840,0860,08A0,0900,0980,%
5543 0A00,0A80,0B00,0B80,0C00,0C80,0D00,0D80,0E00,0E80,%
5544 0F00,1000,10A0,1200,1380,13A0,1400,1680,16A0,1700,%
5545 1720,1740,1760,1780,1800,18B0,1900,1950,1980,19E0,%
5546 1A00,1A20,1AB0,1B00,1B80,1BC0,1C00,1C50,1C80,1CC0,%
5547 1CD0,1D00,1D80,1DC0,1E00,1F00,2440,27C0,27F0,2800,%
5548 2A00,2C00,2C60,2C80,2D00,2D30,2D80,2DE0,2E00,4DC0,%
5549 A4D0,A500,A640,A6A0,A700,A720,A800,A830,A840,A880,%
5550 A8E0,A900,A930,A980,A9E0,AA00,AA60,AA80,AAE0,AB00,%
5551 AB30,AB70,ABC0,D800,DB80,DC00,E000,FB00,FB50,FE00,%
5552 FE70,FFF0,%
5553 10000,10080,10100,10140,10190,101D0,10280,102A0,%

```



```

5554 102E0,10300,10330,10350,10380,103A0,10400,10450,%
5555 10480,104B0,10500,10530,10600,10800,10840,10860,%
5556 10880,108E0,10900,10920,10980,109A0,10A00,10A60,%
5557 10A80,10AC0,10B00,10B40,10B60,10B80,10C00,10C80,%
5558 10E60,11000,11080,110D0,11100,11150,11180,111E0,%
5559 11200,11280,112B0,11300,11400,11480,11580,11600,%
5560 11660,11680,11700,118A0,11A00,11A50,11AC0,11C00,%
5561 11C70,11D00,12000,12400,12480,13000,14400,16800,%
5562 16A40,16AD0,16B00,16F00,1BC00,1BCA0,1D000,1D100,%
5563 1D200,1D300,1D360,1D400,1D800,1E000,1E800,1E900,%
5564 1EE00,1F000,1F030,1F0A0,1F300,1F600,1F650,1F680,%
5565 1F700,1F780,1F800,1F900,E0000,E0100,F0000,100000,%
5566 00C0%
5567 }\do{%
5568 \@tempcnta="\bxjx@tmpa\relax
5569 \@tempcntb\@tempcnta \advance\@tempcntb\m@ne
5570 \chardef\bxjx@tmpb\kcatcode\@tempcntb
5571 \kcatcode\@tempcnta=15 \kcatcode\@tempcntb\bxjx@tmpb}
5572 \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。

```

5573 \@onlypreamble\bxjx@grkcyr@list
5574 \def\bxjx@grkcyr@list{%
5575 \do{0391}{LGR}{\textAlpha}{A}%           % GR. C. L. ALPHA
5576 \do{0392}{LGR}{\textBeta}{B}%           % GR. C. L. BETA
5577 \do{0393}{LGR}{\textGamma}{\Gamma}%     % GR. C. L. GAMMA
5578 \do{0394}{LGR}{\textDelta}{\Delta}%     % GR. C. L. DELTA
5579 \do{0395}{LGR}{\textEpsilon}{E}%        % GR. C. L. EPSILON
5580 \do{0396}{LGR}{\textZeta}{Z}%           % GR. C. L. ZETA
5581 \do{0397}{LGR}{\textEta}{H}%            % GR. C. L. ETA
5582 \do{0398}{LGR}{\textTheta}{\Theta}%     % GR. C. L. THETA
5583 \do{0399}{LGR}{\textIota}{I}%           % GR. C. L. IOTA
5584 \do{039A}{LGR}{\textKappa}{K}%          % GR. C. L. KAPPA
5585 \do{039B}{LGR}{\textLambda}{\Lambda}%   % GR. C. L. LAMDA
5586 \do{039C}{LGR}{\textMu}{M}%            % GR. C. L. MU
5587 \do{039D}{LGR}{\textNu}{N}%            % GR. C. L. NU

```

5588 \do{039E}{LGR}{\textXi}{\Xi}%	% GR. C. L. XI
5589 \do{039F}{LGR}{\textOmicron}{O}%	% GR. C. L. OMICRON
5590 \do{03A0}{LGR}{\textPi}{\Pi}%	% GR. C. L. PI
5591 \do{03A1}{LGR}{\textRho}{\rho}%	% GR. C. L. RHO
5592 \do{03A3}{LGR}{\textSigma}{\Sigma}%	% GR. C. L. SIGMA
5593 \do{03A4}{LGR}{\textTau}{\tau}%	% GR. C. L. TAU
5594 \do{03A5}{LGR}{\textUpsilon}{\Upsilon}%	% GR. C. L. UPSILON
5595 \do{03A6}{LGR}{\textPhi}{\Phi}%	% GR. C. L. PHI
5596 \do{03A7}{LGR}{\textChi}{\chi}%	% GR. C. L. CHI
5597 \do{03A8}{LGR}{\textPsi}{\Psi}%	% GR. C. L. PSI
5598 \do{03A9}{LGR}{\textOmega}{\Omega}%	% GR. C. L. OMEGA
5599 \do{03B1}{LGR}{\textalpha}{\alpha}%	% GR. S. L. ALPHA
5600 \do{03B2}{LGR}{\textbeta}{\beta}%	% GR. S. L. BETA
5601 \do{03B3}{LGR}{\textgamma}{\gamma}%	% GR. S. L. GAMMA
5602 \do{03B4}{LGR}{\textdelta}{\delta}%	% GR. S. L. DELTA
5603 \do{03B5}{LGR}{\textepsilon}{\epsilon}%	% GR. S. L. EPSILON
5604 \do{03B6}{LGR}{\textzeta}{\zeta}%	% GR. S. L. ZETA
5605 \do{03B7}{LGR}{\texteta}{\eta}%	% GR. S. L. ETA
5606 \do{03B8}{LGR}{\texttheta}{\theta}%	% GR. S. L. THETA
5607 \do{03B9}{LGR}{\textiota}{\iota}%	% GR. S. L. IOTA
5608 \do{03BA}{LGR}{\textkappa}{\kappa}%	% GR. S. L. KAPPA
5609 \do{03BB}{LGR}{\textlambda}{\lambda}%	% GR. S. L. LAMDA
5610 \do{03BC}{LGR}{\textmu}{\mu}%	% GR. S. L. MU
5611 \do{03BD}{LGR}{\textnu}{\nu}%	% GR. S. L. NU
5612 \do{03BE}{LGR}{\textxi}{\xi}%	% GR. S. L. XI
5613 \do{03BF}{LGR}{\textomicron}{o}%	% GR. S. L. OMICRON
5614 \do{03C0}{LGR}{\textpi}{\pi}%	% GR. S. L. PI
5615 \do{03C1}{LGR}{\textrho}{\rho}%	% GR. S. L. RHO
5616 \do{03C2}{LGR}{\textvarsigma}{\varsigma}%	% GR. S. L. FINAL SIGMA
5617 \do{03C3}{LGR}{\textsigma}{\sigma}%	% GR. S. L. SIGMA
5618 \do{03C4}{LGR}{\texttau}{\tau}%	% GR. S. L. TAU
5619 \do{03C5}{LGR}{\textupsilon}{\upsilon}%	% GR. S. L. UPSILON
5620 \do{03C6}{LGR}{\textphi}{\phi}%	% GR. S. L. PHI
5621 \do{03C7}{LGR}{\textchi}{\chi}%	% GR. S. L. CHI
5622 \do{03C8}{LGR}{\textpsi}{\psi}%	% GR. S. L. PSI
5623 \do{03C9}{LGR}{\textomega}{\omega}%	% GR. S. L. OMEGA
5624 \do{0401}{T2A}{\CYRYO}{Y}%	% CY. C. L. IO
5625 \do{0410}{T2A}{\CYRA}{A}%	% CY. C. L. A
5626 \do{0411}{T2A}{\CYRB}{B}%	% CY. C. L. BE
5627 \do{0412}{T2A}{\CYRV}{V}%	% CY. C. L. VE
5628 \do{0413}{T2A}{\CYRG}{G}%	% CY. C. L. GHE
5629 \do{0414}{T2A}{\CYRD}{D}%	% CY. C. L. DE
5630 \do{0415}{T2A}{\CYRE}{E}%	% CY. C. L. IE
5631 \do{0416}{T2A}{\CYRZH}{ZH}%	% CY. C. L. ZHE
5632 \do{0417}{T2A}{\CYRZ}{Z}%	% CY. C. L. ZE
5633 \do{0418}{T2A}{\CYRI}{I}%	% CY. C. L. I
5634 \do{0419}{T2A}{\CYRISHRT}{SHRT}%	% CY. C. L. SHORT I
5635 \do{041A}{T2A}{\CYRK}{K}%	% CY. C. L. KA
5636 \do{041B}{T2A}{\CYRL}{L}%	% CY. C. L. EL

5637 \do{041C}{T2A}{\CYRM}{}%	% CY. C. L. EM
5638 \do{041D}{T2A}{\CYRN}{}%	% CY. C. L. EN
5639 \do{041E}{T2A}{\CYRO}{}%	% CY. C. L. O
5640 \do{041F}{T2A}{\CYRP}{}%	% CY. C. L. PE
5641 \do{0420}{T2A}{\CYRR}{}%	% CY. C. L. ER
5642 \do{0421}{T2A}{\CYRS}{}%	% CY. C. L. ES
5643 \do{0422}{T2A}{\CYRT}{}%	% CY. C. L. TE
5644 \do{0423}{T2A}{\CYRU}{}%	% CY. C. L. U
5645 \do{0424}{T2A}{\CYRF}{}%	% CY. C. L. EF
5646 \do{0425}{T2A}{\CYRH}{}%	% CY. C. L. HA
5647 \do{0426}{T2A}{\CYRC}{}%	% CY. C. L. TSE
5648 \do{0427}{T2A}{\CYRCH}{}%	% CY. C. L. CHE
5649 \do{0428}{T2A}{\CYRSH}{}%	% CY. C. L. SHA
5650 \do{0429}{T2A}{\CYRSHCH}{}%	% CY. C. L. SHCHA
5651 \do{042A}{T2A}{\CYRHRDSN}{}%	% CY. C. L. HARD SIGN
5652 \do{042B}{T2A}{\CYRERY}{}%	% CY. C. L. YERU
5653 \do{042C}{T2A}{\CYRSFTSN}{}%	% CY. C. L. SOFT SIGN
5654 \do{042D}{T2A}{\CYREREV}{}%	% CY. C. L. E
5655 \do{042E}{T2A}{\CYRYU}{}%	% CY. C. L. YU
5656 \do{042F}{T2A}{\CYRYA}{}%	% CY. C. L. YA
5657 \do{0430}{T2A}{\cyra}{}%	% CY. S. L. A
5658 \do{0431}{T2A}{\cyrb}{}%	% CY. S. L. BE
5659 \do{0432}{T2A}{\cyrv}{}%	% CY. S. L. VE
5660 \do{0433}{T2A}{\cyrg}{}%	% CY. S. L. GHE
5661 \do{0434}{T2A}{\cyrd}{}%	% CY. S. L. DE
5662 \do{0435}{T2A}{\cyre}{}%	% CY. S. L. IE
5663 \do{0436}{T2A}{\cyrrh}{}%	% CY. S. L. ZHE
5664 \do{0437}{T2A}{\cyrrz}{}%	% CY. S. L. ZE
5665 \do{0438}{T2A}{\cyri}{}%	% CY. S. L. I
5666 \do{0439}{T2A}{\cyrishrt}{}%	% CY. S. L. SHORT I
5667 \do{043A}{T2A}{\cyrk}{}%	% CY. S. L. KA
5668 \do{043B}{T2A}{\cyr1}{}%	% CY. S. L. EL
5669 \do{043C}{T2A}{\cyrm}{}%	% CY. S. L. EM
5670 \do{043D}{T2A}{\cyrn}{}%	% CY. S. L. EN
5671 \do{043E}{T2A}{\cyro}{}%	% CY. S. L. O
5672 \do{043F}{T2A}{\cyrp}{}%	% CY. S. L. PE
5673 \do{0440}{T2A}{\cyrr}{}%	% CY. S. L. ER
5674 \do{0441}{T2A}{\cyrsh}{}%	% CY. S. L. ES
5675 \do{0442}{T2A}{\cyrt}{}%	% CY. S. L. TE
5676 \do{0443}{T2A}{\cyru}{}%	% CY. S. L. U
5677 \do{0444}{T2A}{\cyrf}{}%	% CY. S. L. EF
5678 \do{0445}{T2A}{\cyrh}{}%	% CY. S. L. HA
5679 \do{0446}{T2A}{\cyrc}{}%	% CY. S. L. TSE
5680 \do{0447}{T2A}{\cyrch}{}%	% CY. S. L. CHE
5681 \do{0448}{T2A}{\cyrsh}{}%	% CY. S. L. SHA
5682 \do{0449}{T2A}{\cyrshch}{}%	% CY. S. L. SHCHA
5683 \do{044A}{T2A}{\cyrhrdsn}{}%	% CY. S. L. HARD SIGN
5684 \do{044B}{T2A}{\cyrery}{}%	% CY. S. L. YERU
5685 \do{044C}{T2A}{\cyrsftsn}{}%	% CY. S. L. SOFT SIGN

```

5686 \do{044D}{T2A}{\cyrrrev}{}%           % CY. S. L. E
5687 \do{044E}{T2A}{\cyryu}{}%             % CY. S. L. YU
5688 \do{044F}{T2A}{\cyrya}{}%             % CY. S. L. YA
5689 \do{0451}{T2A}{\cyryo}{}%             % CY. S. L. IO
5690 \do{00A7}{TS1}{\textsection}{\mathsection}% SECTION SYMBOL
5691 \do{00A8}{TS1}{\textasciidieresis}{}%   % DIAERESIS
5692 \do{00B0}{TS1}{\textdegree}{\mathdegree}% % DEGREE SIGN
5693 \do{00B1}{TS1}{\textpm}{\pm}%           % PLUS-MINUS SIGN
5694 \do{00B4}{TS1}{\textasciicute}{}%       % ACUTE ACCENT
5695 \do{00B6}{TS1}{\textparagraph}{\mathparagraph}% PILCROW SIGN
5696 \do{00D7}{TS1}{\texttimes}{\times}%     % MULTIPLICATION SIGN
5697 \do{00F7}{TS1}{\textdiv}{\div}%         % DIVISION SIGN
5698 }

```

`\mathdegree` 面倒なので補っておく。

```
5699 \providecommand*{\mathdegree}{\circ}
```

`\ifbxjx@gcc@cjkl` [スイッチ]「特定 CJK 曖昧文字」を和文扱いにするか。

```
5700 \newif\ifbxjx@gcc@cjkl
```

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

```
5701 \newcommand*\greekasCJK{%
```

```
5702   \bxjx@gcc@cjkltrue}
```

`\nogreekasCJK` [公開命令]「特定 CJK 曖昧文字」を欧文扱いにする。

```
5703 \newcommand*\nogreekasCJK{%
```

```
5704   \bxjx@gcc@cjklfalse}
```

`\bxjx@fake@grkl` `\bxjx@fake@grkl{〈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@grkl{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。

```

5705 \def\bxjx@tmpdo#1\relax{%
5706   \def\bxjx@fake@grkl##1##2{%
5707     \expandafter\bxjx@fake@grkl@a\meaning##2#1\@nil{##1}{##2}}%
5708   \def\bxjx@fake@grkl@a##1#1##2\@nil##3##4{%
5709     \ifx\##1\%
5710       \bxjx@canta##4\divide\bxjx@canta\@cclvi
5711       \multiply\bxjx@canta\@cclvi \advance\bxjx@canta`##3\relax
5712       \mathchar\bxjx@canta
5713     \else ##3\fi}
5714 } \expandafter\bxjx@tmpdo\string\mathchar\relax

```

## ■pdfTeX・upTeX の場合

```
5715 \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  $\text{\LaTeX}$  においても、必ず「`inputenc` が明示的に読み込まれた」状態になる。

```
5716 \ifpackageloaded{inputenc}{\else
5717   \RequirePackage[utf8]{inputenc}
5718 \def\bxjx@tmpa{utf8}
5719 \ifx\bxjx@tmpa\inputencdoingname
5720   \PackageWarningNoLine\bxjx@pkgname
5721     {Input encoding changed to utf8}%
5722   \inputencoding{utf8}%
5723 \fi
```

$\text{upTeX}$  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。

```
5724 \if u\bxjx@engine
5725 \kcatcode"0370=15
5726 \kcatcode"0400=15
5727 \kcatcode"0500=15
5728 \fi
```

各文字について `\DeclareUnicodeCharacter` を実行する。

```
5729 \def\bxjx@tmpdo#1{%
5730   \@tempcnta="#1\relax
5731   \expandafter\bxjx@tmpdo@a\csname bxjx@KC/\the\@tempcnta\endcsname{#1}}
5732 \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。

```
5733   \ifx\#5\%
5734     \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。

```
5735   \else\ifcat A\noexpand#5%
5736     \edef\bxjx@tmpa{\noexpand\bxjx@fake@grk{#5}%
5737       {\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。

```
5738   \else \def\bxjx@tmpa{#5}%
5739   \fi\fi
5740   \def\bxjx@tmpb{\bxjx@tmpdo@b{#1}{#2}{#3}{#4}}%
5741   \expandafter\bxjx@tmpb\expandafter{\bxjx@tmpa}}
```

以降はエンジン種別で分岐する。 $\text{upTeX}$  の場合。

```
5742 \if u\bxjx@engine
5743 \def\bxjx@tmpdo@b#1#2#3#4#5{%
```

引数 =  $\backslash\text{bxjx@KC}/\langle\text{符号値}\rangle\{\langle\text{符号値}\rangle\}\{\langle\text{fontenc}\rangle\}\{\langle\text{LICR}\rangle\}\{\langle\text{数式中の動作}\rangle\}$

当該の Unicode 文字の動作は「テキストでは  $\langle\text{LICR}\rangle$ 、数式では  $\langle\text{数式中の動作}\rangle$ 」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  $\backslash\text{bxjx@ja@or@not}$  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  $\backslash\text{bxjx@KC}/\dots$  を流用している。

```
5744 \kchardef#1=\@tempcnta
5745 \DeclareTextCommandDefault{#4}\backslash\text{bxjx@ja@or@not}{#1}{#3}{#4}}%
5746 \DeclareUnicodeCharacter{#2}\text{TextOrMath}{#4}{#5}}}
```

pdf $\text{\TeX}$  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  $\backslash\text{UTF}\{\langle\text{符号値}\rangle\}$  を使う（ $\backslash\text{UTF}$  は `bxckjatype` の命令）。 $\backslash\text{bxjx@KC}/\dots$  は使わないが定義済にする必要がある。

```
5747 \else\if p\backslash\text{bxjx@engine}
5748 \def\backslash\text{bxjx@tmpdo@b#1#2#3#4#5}{%
5749 \mathchardef#1=\@tempcnta
5750 \DeclareTextCommandDefault{#4}\backslash\text{bxjx@ja@or@not}\backslash\text{UTF}{#2}}{#3}{#4}}%
5751 \DeclareUnicodeCharacter{#2}\text{TextOrMath}{#4}{#5}}}
```

以上の処理を「特定 CJK 曖昧文字」の各々に適用する。

```
5753 \let\do\backslash\text{bxjx@tmpdo} \backslash\text{bxjx@grkcyr@list}
```

$\backslash\text{bxjx@DeclareUnicodeCharacter}$   $\backslash\text{bxjx@DeclareUnicodeCharacter}$  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。

```
5754 \@onlypreamble\backslash\text{bxjx@org@DeclareUnicodeCharacter}
5755 \let\backslash\text{bxjx@org@DeclareUnicodeCharacter}\backslash\text{DeclareUnicodeCharacter}
5756 \@onlypreamble\backslash\text{bxjx@DeclareUnicodeCharacter}
5757 \def\backslash\text{bxjx@DeclareUnicodeCharacter}#1#2{%
5758 \count@=#1\relax
5759 \expandafter\ifx\csname \backslash\text{bxjx@KC}/\the\count@\endcsname\relax
5760 \backslash\text{bxjx@org@DeclareUnicodeCharacter}{#1}{#2}%
5761 \else
5762 \wlog{\space\space skipped defining Unicode char U+#1}%
5763 \fi}
```

$\backslash\text{bxjx@ja@or@not}$   $\backslash\text{bxjx@ja@or@not}\{\langle\text{和文用定義}\rangle\}\{\langle\text{対象 fontenc}\rangle\}\{\langle\text{LICR}\rangle\}$  :  $\backslash\text{[no]greekasCJK}$  の状態に応じて和文または欧文で文字を出力する。

```
5764 \def\backslash\text{bxjx@ja@or@not}#1#2#3{%
```

$\backslash\text{greekasCJK}$  の場合は、無条件に  $\langle\text{和文用定義}\rangle$  を実行する。

```
5765 \ifbxjx@gcc@CJK #1%
```

$\backslash\text{nogreekasCJK}$  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。

```
5766 \else\expandafter\ifx\csname T@#2\endcsname\relax #1%
5767 \else \UseTextSymbol{#2}{#3}%
5768 \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。

```

5769 \begingroup
5770 \toks@\expandafter{\DeclareFontEncoding@{#1}{#2}{#3}}
5771 \xdef\next{\def\noexpand\DeclareFontEncoding@##1##2##3{%
5772   \noexpand\bxjx@swap@DUC@cmd
5773   \the\toks@
5774   \noexpand\bxjx@swap@DUC@cmd}}
5775 \endgroup\next
5776 \def\bxjx@swap@DUC@cmd{%
5777   \let\bxjx@tmpa\DeclareUnicodeCharacter
5778   \let\DeclareUnicodeCharacter\bxjx@DeclareUnicodeCharacter
5779   \let\bxjx@DeclareUnicodeCharacter\bxjx@tmpa
5780   \let\bxjx@tmpa\relax}

```

以上。

### ■ $\text{\XeTeX}$ ・ $\text{\LuaTeX}$ の場合

```

5781 \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。

```

5782 \def\bxjx@tmpdo#1{%
5783   \bxjx@cmta="#1\relax
5784   \begingroup
5785     \lccode`~=\bxjx@cmta
5786     \lowercase{\endgroup
5787       \bxjx@tmpdo@a{~}}{#1}}
5788 \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、 $\text{\upLaTeX}$  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。

```

5789   \ifx\\#5\\\let\bxjx@tmpa\relax
5790   \else\ifcat A\noexpand#5%
5791     \edef\bxjx@tmpa{\noexpand\bxjx@fake@grk{#5}%
5792       {\ifnum\uccode`#5=`#5\noexpand\Pi\else\noexpand\pi\fi}}%
5793   \else \def\bxjx@tmpa{#5}%
5794   \fi\fi
5795   \ifx\bxjx@tmpa\relax\else
5796     \mathcode\bxjx@cmta"8000 \let#1\bxjx@tmpa
5797   \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。

```

5798 \mathchardef\bxjx@tmpa="119
5799 \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。

$\text{\LuaTeX}$  の場合は、 $\text{\LuaTeX-j}$ a 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。

```
5800 \if l\bxjx@engine
5801   \protected\def\greekasCJK{%
5802     \bxjx@gcc@cjctrue
5803     \ltjsetparameter{jacharrange={+2, +8}}
5804   \protected\def\nogreekasCJK{%
5805     \bxjx@gcc@cjcfalse
5806     \ltjsetparameter{jacharrange={-2, -8}}
5807 \fi
```

X<sub>Ǝ</sub>TeX の場合、xeCJK は X<sub>Ǝ</sub>TeX 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。

```
5808 \if x\bxjx@engine
5809   \let\bxjx@gcc@cjkl@list\@empty
5810   \def\do#1#2#3#4{%
5811     \edef\bxjx@gcc@cjkl@list{\bxjx@gcc@cjkl@list
5812       \noexpand\XeTeXcharclass"#1\bxjx@canta}}
5813   \bxjx@grkcyrl@list
5814   \protected\def\greekasCJK{%
5815     \bxjx@gcc@cjctrue
5816     \bxjx@canta=\@ne \bxjx@gcc@cjkl@list}
5817   \protected\def\nogreekasCJK{%
5818     \bxjx@gcc@cjcfalse
5819     \bxjx@canta=\z@ \bxjx@gcc@cjkl@list}
5820 \fi
```

以上。

```
5821 \fi\fi
```

## H.4 初期設定

「特定 CJK 曖昧文字」を欧文扱いにする。

```
5822 \nogreekasCJK
```

## H.5 完了

おしまい。

```
5823 %</cjkcat>
```

## 付録 I 補助パッケージ：bxjspandoc 🐼

Pandoc の L<sup>A</sup>T<sub>E</sub>X 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 T<sub>E</sub>X 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。



## 1.1 準備

```
5824 %<*ancpandoc>
5825 %% このファイルは日本語文字を含みます.
5826 \def\bxjsp@pkgname{bxjspandoc}
```

`\bxjsp@engine` エンジンの種別。

```
5827 \let\bxjsp@engine=n
5828 \@onlypreamble\bxjsp@do
5829 \def\bxjsp@do#1#2{%
5830   \edef\bxjsp@tmpa{\string#1}%
5831   \edef\bxjsp@tmpb{\meaning#1}%
5832   \ifx\bxjsp@tmpa\bxjsp@tmpb #2\fi}
5833 \bxjsp@do\kanjiskip{\let\bxjsp@engine=j}
5834 \bxjsp@do\XeTeXversion{\let\bxjsp@engine=x}
5835 \bxjsp@do\pdftexversion{\let\bxjsp@engine=p}
5836 \bxjsp@do\luatexversion{\let\bxjsp@engine=l}
```

`\bxjsp@begin@document@hook` 文書本体開始時フック。

```
5837 \@onlypreamble\bxjsp@begin@document@hook
5838 \let\bxjsp@begin@document@hook\@empty
5839 \AtBeginDocument{\bxjsp@begin@document@hook}
```

`\ifbxjsp@babel@used` [スイッチ] Babel が読み込まれたか。

```
5840 \newif\ifbxjsp@babel@used
5841 \g@addto@macro\bxjsp@begin@document@hook{%
5842   \@ifpackageloaded{babel}{\bxjsp@babel@usedtrue}{}}
```

## 1.2 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 L<sup>A</sup>T<sub>E</sub>X 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。

```
5843 \newif\ifbxjsp@english
5844 \DeclareOption{english}{\bxjsp@englishttrue}
```

オプション定義はおしまい。

```
5845 \ProcessOptions*
```

## 1.3 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。

```
5846 \@onlypreamble\pandocSkipLoadFile
5847 \newcommand*\pandocSkipLoadFile[1]{%
```

```

5848 \expandafter\bxjsp@skip@load@file@a\csname ver@#1\endcsname{#1}}
5849 \def\bxjsp@skip@load@file@a#1#2{%
5850 \ifx#1\relax
5851 \def#1{2001/01/01}%
5852 \PackageInfo\bxjsp@pkgname
5853 {File '#2' marked as loaded\@gobble}%
5854 \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。

```

5855 \@onlypreamble\pandocSkipLoadPackage
5856 \newcommand*\pandocSkipLoadPackage[1]{%
5857 \pandocSkipLoadFile{#1.sty}}

```

## 1.4 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。

```

5858 \ifx\@IncludeInRelease\undefined\else
5859 \pandocSkipLoadPackage{fixltx2e}
5860 \fi

```

## 1.5 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。）

```

5861 \if j\bxjsp@engine
5862 \pandocSkipLoadPackage{cmap}
5863 \fi

```

## 1.6 microtype パッケージ

警告が多すぎなので消す。

```

5864 \if j\bxjsp@engine \else
5865 \PassOptionsToPackage{verbose=silent}{microtype}
5866 \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。

```

5867 \if j\bxjsp@engine
5868 \pandocSkipLoadPackage{microtype}
5869 \newcommand*\UseMicrotypeSet[2][]{ }

```

## I.7 Unicode 文字変換対策

Pandoc で  $\text{\LaTeX}$  形式に書き出す場合は、元データ中の一部の Unicode 文字を「 $\text{\LaTeX}$  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。

$\cdots \rightarrow \text{\ldots}$     ‘ $\rightarrow$ 、’ $\rightarrow$ ’    “ $\rightarrow$ 、” $\rightarrow$ ”

日本語  $\text{\LaTeX}$  では「 $\text{\LaTeX}$  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、「 $\cdots$ 」の置換を抑止する機能はないようである。そこで、「 $\text{\ldots}$ 」を『 $\cdots$ 』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  $\cdots$  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  $\cdots$  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  $\cdots$  に戻す動作に変更した。

```
5871 \DeclareRobustCommand{\pandocLdots}{%
5872   \let\bxjsp@do\bxjsp@ja@ellipsis
5873   \ifmmode \let\bxjsp@do\bxjsp@org@ldots
5874   \else\ifbxjsp@babel@used
5875     \expandafter\ifx\csname bxjsp@ld/\language\endcsname\relax
5876       \let\bxjsp@do\bxjsp@org@ldots \fi
5877   \fi\fi \bxjsp@do}
5878 \@namedef{bxjsp@ld/japanese}{1}
5879 \def\bxjsp@ja@ellipsis{...}
5880 \let\bxjsp@org@ldots\ldots
```

`\ldots` の実装を `\pandocLdots` に置き換える。

```
5881 \g@addto@macro\bxjsp@begin@document@hook{%
5882   \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。

```
5883 \long\def\bxjsp@tmpa{\ldots}%
5884 \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。

```
5885   \ifnum0\ifbxjsp@english\ifbxjsp@babel@used\else1\fi\fi=0
5886   \let\ldots\pandocLdots
5887 \fi
5888 \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  $\text{\XeTeX}$ ・ $\text{\LuaTeX}$  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。

```
5889 \chardef\bxjsp@cc@other=12
5890 \@onlypreamble\bxjsp@makeother@range
5891 \def\bxjsp@makeother@range#1#2{%
5892   \@tempcnta"#1\relax \@tempcntb"#2\relax
5893   \loop\ifnum\@tempcnta<\@tempcntb
5894     \catcode\@tempcnta\bxjsp@cc@other
5895     \advance\@tempcnta\@ne
5896   \repeat}
5897 \ifnum0\if x\bxjsp@engine1\fi\if 1\bxjsp@engine1\fi>0
5898   \catcode"1F23B=\bxjsp@cc@other
5899   \bxjsp@makeother@range{9FCD}{A000}
5900   \bxjsp@makeother@range{1B002}{1B170}
5901   \bxjsp@makeother@range{2B820}{2EBF0}
5902 \fi
```

## 1.8 PandoLa モジュール

インストール済であれば読み込む。

```
5903 \IfFileExists{bxpandola.sty}{%
5904   \RequirePackage{bxpandola}\relax
5905   \PackageInfo\bxjsp@pkgname
5906   {PandoLa module is loaded\@gobble}
5907 }{}
```

## 1.9 完了

おしまい。

```
5908 %</ancpandoc>
```

補助パッケージ実装はここまで。

```
5909 %</anc>
```