

# BXjscls パッケージ (BXJS 文書クラス集) ソースコード説明書

八登崇之 (Takayuki YATO; aka. “ZR”)

v2.9b [2024/01/22]







---

この文書はソースコード説明書です。一般の文書作成者向けの解説については、ユーザーマニュアル `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 . . . . .	80
8.2	章・節 . . . . .	86
8.3	リスト環境 . . . . .	98
8.4	パラメータの設定 . . . . .	105
8.5	フロート . . . . .	107
8.6	キャプション . . . . .	108
<b>9</b>	<b>フォントコマンド</b>	<b>109</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>127</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>132</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>138</b>
<b>付録 B</b>	<b>和文ドライバ：minimal </b>	<b>139</b>
B.1	準備 . . . . .	139
B.2	(u)pTeX 用の設定 . . . . .	142
B.3	pdfTeX 用の処理 . . . . .	147
B.4	X <sub>g</sub> TeX 用の処理 . . . . .	147
B.5	後処理 (エンジン共通) . . . . .	148
<b>付録 C</b>	<b>和文ドライバ：standard </b>	<b>151</b>
C.1	準備 . . . . .	151
C.2	和文ドライバパラメタ . . . . .	152
C.3	共通処理 (1) . . . . .	152
C.4	pTeX 用設定 . . . . .	158
C.5	pdfTeX 用設定：CJK + bxcjkjatype . . . . .	163
C.6	X <sub>g</sub> TeX 用設定：xeCJK + zxjatype . . . . .	165
C.7	LuaTeX 用設定：LuaTeX-ja . . . . .	167
C.8	共通処理 (2) . . . . .	171
<b>付録 D</b>	<b>和文ドライバ：modern </b>	<b>172</b>
D.1	フォント設定 . . . . .	172
D.2	fixltx2e 読込 . . . . .	172
D.3	和文カテゴリコード . . . . .	173
D.4	完了 . . . . .	173
<b>付録 E</b>	<b>和文ドライバ：pandoc </b>	<b>173</b>

E.1	準備	173
E.2	和文ドライバパラメタ	174
E.3	dupload システム	175
E.4	lang 変数	176
E.5	geometry 変数	180
E.6	CJKmainfont 変数	180
E.7	Option clash 対策	180
E.8	レイアウト上書き禁止	180
E.9	paragraph のマーク	182
E.10	全角空白文字	182
E.11	hyperref 対策	183
E.12	Pandoc 要素に対する和文用の補正	183
E.13	ifPDFTeX スイッチ	184
E.14	完了	185
<b>付録 F</b>	<b>補助パッケージ一覧</b>	<b>186</b>
<b>付録 G</b>	<b>補助パッケージ：bxjscompat</b>	<b>186</b>
G.1	準備	186
G.2	8bit 欧文 TeX	187
G.3	X <sub>g</sub> TeX	187
G.4	LuaTeX	188
G.5	完了	189
<b>付録 H</b>	<b>補助パッケージ：bxjscjcat</b>	<b>190</b>
H.1	準備	190
H.2	和文カテゴリコードの設定	191
H.3	ギリシャ・キリル文字の扱い	191
H.4	初期設定	199
H.5	完了	199
<b>付録 I</b>	<b>補助パッケージ：bxjspandoc</b>	<b>199</b>
I.1	準備	199
I.2	パッケージオプション	200
I.3	パッケージ読込の阻止	200
I.4	fixltx2e パッケージ	200
I.5	cmap パッケージ	201
I.6	microtype パッケージ	201
I.7	Unicode 文字変換対策	201
I.8	PandoLa モジュール	202
I.9	完了	203

# 1 はじめに

---

この文書は「BXJS ドキュメントクラス」の DocStrip 形式のソースである。BXJS ドキュメントクラス（以降では「BXJS クラス」と略称する）は奥村晴彦氏および日本語  $\text{T}_{\text{E}}\text{X}$  開発コミュニティによる「 $\text{p}_{\text{L}}\text{T}_{\text{E}}\text{X}_{2_{\epsilon}}$  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  $\text{L}_{\text{A}}\text{T}_{\text{E}}\text{X}3$  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  $\text{T}_{\text{E}}\text{X}$  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てきました。アスキーのものが最近では modified BSD ライセンスになっていますので、私のももそれに準じて modified BSD とすることにします。

[2016-07-13] 日本語  $\text{T}_{\text{E}}\text{X}$  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  $\text{u}_{\text{P}}\text{L}_{\text{A}}\text{T}_{\text{E}}\text{X}$  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@needsp@tch [2016-08-22] 従来 jsclasses では、pL<sup>A</sup>T<sub>E</sub>X や L<sup>A</sup>T<sub>E</sub>X 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 pL<sup>A</sup>T<sub>E</sub>X 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 pL<sup>A</sup>T<sub>E</sub>X 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@needsp@tch
19 \jsc@needsp@tchfalse
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>Ǝ</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>Ǝ</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>Ǝ</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 \@tempwatrue
```

```

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「 $\varepsilon$ -p<sub>T</sub>E<sub>X</sub>」から「内部漢字コードが非 Unicode の  $\varepsilon$ -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 p $\TeX$ -ng であるか。

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

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

※ X $\TeX$  と Lua $\TeX$  は  $\epsilon$ - $\TeX$  拡張をもつ版のみがあり、NTT  $\mathcal{J}\TeX$  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\epsilonTeXversion{\jsWitheTeXtrue}
```

`\ifjsInPdfMode` [スイッチ] pdf $\TeX$ ・Lua $\TeX$  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 X $\TeX$  または Lua $\TeX$  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

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

```

112 \ifjswitheTeX \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 \ifjswitheTeX
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⟩}`

T<sub>E</sub>X 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に置き換える。

```

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`  $\epsilon$ - $\text{\TeX}$  拡張が有効なら通常の整数レジスタ、無効なら擬似整数レジスタを用いる。

```

\bxjs@advance@count 158 \ifjswitheTeX
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

```

`\jsSetQHLlength` `\jsSetQHLlength\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\jsSetQHLlength#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\cename bxjs@unit@\bxjs@tmpa\expandafter
184             \endcename\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、 $\text{L}^{\text{A}}\text{T}_{\text{E}}\text{X} 2_{\epsilon}$  の `b5paper` は  $250\text{mm} \times 176\text{mm}$  ですが、 $\text{pL}^{\text{A}}\text{T}_{\text{E}}\text{X} 2_{\epsilon}$  の `b5paper` は  $257\text{mm} \times 182\text{mm}$  になっています。ここでは  $\text{pL}^{\text{A}}\text{T}_{\text{E}}\text{X} 2_{\epsilon}$  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ため、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` の定義が変だったのでご迷惑をおかけしましたが, 標準的なドキュメントクラスと同様にポイント数から 10 を引いたものに直しました。

[2003-03-22] 14Q オプションを追加しました。

[2003-04-18] 12Q オプションを追加しました。

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

[2016-07-24] オプティカルサイズを調整するために NFSS 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>ε</sub>TeX 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>ε</sub>TeX 2<sub>ε</sub> 本体で宣言されています。

取りあえず、p<sub>ε</sub>TeX 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 L<sup>A</sup>T<sub>E</sub>X 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.)` L<sup>A</sup>T<sub>E</sub>X 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] L<sup>A</sup>T<sub>E</sub>X 2<sub>ε</sub> 2021-11-15 (l<sup>A</sup>T<sub>E</sub>X 2021/10/14 v1.2j) で `\@currentcounter` が追加されましたので、追従します。

```
341 \def\eqnarray{%
342   \stepcounter{equation}%
```

```

343 \def\@currentlabel{\p@equation\theequation}%
344 \def\@currentcounter{equation}%
345 \global\@eqnswtrue
346 \m@th
347 \global\@eqcnt\z@
348 \tabskip\@centering
349 \let\\\@eqnocr
350 $$\everycr{}\halign to\displaywidth\bgroup
351   \hskip\@centering$\displaystyle\tabskip\z@skip{##}$\@eqnse1
352   &\global\@eqcnt\@ne \hfil$\displaystyle{##}$\hfil
353   &\global\@eqcnt\tw@ $\displaystyle{##}$\hfil\tabskip\@centering
354   &\global\@eqcnt\thr@@ \hb@xt@\z@\bgroup\hss##\egroup
355   \tabskip\z@skip
356 \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] L<sup>A</sup>T<sub>E</sub>X 2<sub>ε</sub> 2021-11-15 (l<sup>A</sup>T<sub>E</sub>X 2021/10/14 v1.2j) で\@currentcounter  
が追加されましたので、追従します。

```

357 \DeclareOption{leqno}{\input{leqno.clo}}
358 \DeclareOption{fleqn}{\input{fleqn.clo}}%
359 % fleqn 用の eqnarray 環境の再定義
360 \def\eqnarray{%
361   \stepcounter{equation}%
362   \def\@currentlabel{\p@equation\theequation}%
363   \def\@currentcounter{equation}%
364   \global\@eqnswtrue\m@th
365   \global\@eqcnt\z@
366   \tabskip\mathindent
367   \let\=\@eqnocr
368   \setlength\abovedisplayskip{\topsep}%
369   \ifvmode
370     \addtolength\abovedisplayskip{\partopsep}%
371   \fi
372   \addtolength\abovedisplayskip{\parskip}%
373   \setlength\belowdisplayskip{\abovedisplayskip}%
374   \setlength\belowdisplayshortskip{\abovedisplayskip}%
375   \setlength\abovedisplayshortskip{\abovedisplayskip}%
376   $$\everycr{}\halign to\linewidth% $$
377   \bgroup
378   \hskip\@centering$\displaystyle\tabskip\z@skip{##}$\@eqnse1
379   &\global\@eqcnt\@ne \hfil$\displaystyle{##}$\hfil
380   &\global\@eqcnt\tw@
381     $\displaystyle{##}$\hfil \tabskip\@centering
382   &\global\@eqcnt\thr@@ \hb@xt@\z@\bgroup\hss##\egroup
383   \tabskip\z@skip\cr
384   }}

```

■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。

```
385 % \DeclareOption{openbib}{%
386 %   \AtEndOfPackage{%
387 %     \renewcommand\@openbib@code{%
388 %       \advance\leftmargin\bibindent
389 %       \itemindent -\bibindent
390 %       \listparindent \itemindent
391 %       \parsep \z@}%
392 %     \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：

```
393 %\let\bxjs@enablejfam\@undefined

enablejfam オプションの処理。

394 \def\bxjs@kv@enablejfam@true{\let\bxjs@enablejfam=t}
395 \def\bxjs@kv@enablejfam@false{\let\bxjs@enablejfam=f}
396 \def\bxjs@kv@enablejfam@default{\let\bxjs@enablejfam\@undefined}
397 \define@key{bxjs}{enablejfam}[true]{%
398   \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。

399 \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。

```
400 \newif\ifjsDraft
401 \DeclareOption{draft}{\jsDrafttrue \overfullrule=5pt }
402 \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, dviptdms などでも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。

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

---

```
407 \newif\if@english
408 \@englishfalse
409 \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。

---

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

### ■複合設定オプション

**TODO:3x** `\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。

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

---

複合設定オプションとは、「エンジンやドライバや和文ドライバの設定を含む、複数の設定を一度に行うオプション」のことである。ある特定の設定を短く書く必要性が高いと判断される場合に用意される。

`pandoc` オプションは、Pandoc で 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。

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

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

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

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

```
455 %\let\bxjs@engine@given\undefined
```

`\bxjs@engine@opt` 明示されたエンジンのオプション名。

```
456 %\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。

```
457 \DeclareOption{autodetect-engine}{%
458   \let\bxjs@engine@given=*}
```

```

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

```

484 %\let\bxjs@driver@given\@undefined
485 \let\bxjs@driver@@dvimode=0
486 \let\bxjs@driver@@dvipdfmx=1
487 \let\bxjs@driver@@pdfmode=2
488 \let\bxjs@driver@@xetex=3
489 \let\bxjs@driver@@dvips=4
490 \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。

```

491 %\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かがあるか」の点で異なる。

```

492 \DeclareOption{dvips}{%
493   \def\bxjs@driver@opt{dvips}%
494   \let\bxjs@driver@given\bxjs@driver@@dvips}
495 \DeclareOption{dviout}{%
496   \def\bxjs@driver@opt{dviout}%
497   \let\bxjs@driver@given\bxjs@driver@@dvimode}
498 \DeclareOption{xdvi}{%
499   \def\bxjs@driver@opt{xdvi}%
500   \let\bxjs@driver@given\bxjs@driver@@dvimode}

```



```

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

dvipdfmx-if-dvi は 2.0 版より非推奨となった。
519 \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 520 \@onlypreamble\bxjs@depre@opt
521 \def\bxjs@depre@opt#1#2{%
522   \ClassWarningNoLine\bxjs@clsname
523     {The old option '#1' is DEPRECATED\MessageBreak
524     and may be abolished in future!\MessageBreak
525     You should instead write:\MessageBreak
526     \space\space #2}}
527 \@onlypreamble\bxjs@depre@opt@do
528 \def\bxjs@depre@opt@do#1#2{%
529   \bxjs@depre@opt{#1}{#2}%
530   \setkeys{bxjs}{#2}}

```

`\ifbxjs@bigcode` [スイッチ]  $\text{upTeX}$  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。

```

531 \newif\ifbxjs@bigcode \bxjs@bigcodetrue

nbigcode / bigcode オプションの定義。
532 \DeclareOption{nbigcode}{%
533   \bxjs@bigcodefalse}
534 \DeclareOption{bigcode}{%
535   \bxjs@bigcodetrue}

```

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

```
536 \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。

```
537 \DeclareOption{nooldfontcommands}{%
```

```
538 \bxjs@oldfontcommandsfalse}
```

```
539 \DeclareOption{oldfontcommands}{%
```

```
540 \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。

```
541 \ifbxjs@brace@safe
```

```
542 \@onlypreamble\bxjs@register@badopt
```

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

```
544 \expandafter\@onlypreamble\csname bxjs@badopt/#1\endcsname
```

```
545 \@namedef{bxjs@badopt/#1}}
```

```
546 \g@addto@macro\bxjs@begin@document@hook{%
```

```
547 \@for\bxjs@tmpa:=\@unusedoptionlist\do{%
```

```
548 \@nameuse{bxjs@badopt/\bxjs@tmpa}}}
```

```
549 \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。

```
550 \@onlypreamble\bxjs@invalid@opt
```

```
551 \ifbxjs@brace@safe
```

```
552 \def\bxjs@invalid@opt#1#2{%
```

```
553 \bxjs@register@badopt{#1}{\ClassError\bxjs@clsname{#2}\@ehc}}
```

```
554 \else
```

```
555 \def\bxjs@invalid@opt#1#2{%
```

```
556 \DeclareOption{#1}{\ClassWarningNoLine\bxjs@clsname{#2}}
```

```
557 \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。

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

```
559 This class does not support 'winjis' option}
```

```
560 \bxjs@invalid@opt{mingoth}{%
```

```
561 This class does not support 'mingoth' option}
```

```
562 \bxjs@invalid@opt{jis}{%
```

```
563 This class does not support 'jis' option}
```

```
564 \if j\jsEngine\else
```

```

565 \bxjs@invalid@opt{tombo}{%
566 Option 'tombo' can be used only on (u)pLaTeX}
567 \bxjs@invalid@opt{tombow}{%
568 Option 'tombow' can be used only on (u)pLaTeX}
569 \bxjs@invalid@opt{mentuke}{%
570 Option 'mentuke' can be used only on (u)pLaTeX}
571 \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。

```

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

```

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

```

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

```

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

```

590 \def\bxjs@cls@setkeys#1#2{%
591 \let\bxjs@save@KV@errx\KV@errx
592 \def\KV@errx##1{\OptionNotUsed}%
593 \setkeys{#1}{#2}%
594 \let\KV@errx\bxjs@save@KV@errx}
595 \ifbxjs@brace@safe\else
596 \let\bxjs@cls@setkeys\bxjs@safe@setkeys
597 \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。

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

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

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

`\jsScale` [実数値マクロ] 和文スケール値。

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

`\bxjs@base@opt` 明示された base オプションの値。

```
625 %\let\bxjs@base@opt\undefined
```

base オプションの処理。

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

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

```
630 %\let\bxjs@jbase@opt\@undefined
```

`jbase` オプションの処理。

```
631 \define@key{bxjs}{jbase}{\edef\bxjs@jbase@opt{#1}}
```

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

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

```
633 %\let\bxjs@scale@opt\@undefined
```

`scale` オプションの処理。

```
634 \define@key{bxjs}{scale}{%
```

```
635 \edef\bxjs@scale@opt{#1}%
```

```
636 \let\jsScale\bxjs@scale@opt}
```

```
637 \define@key{bxjs}{jafontscale}{\setkeys{bxjs}{scale=#1}}
```

`noscale` オプションの処理。

**TODO:3.0** `noscale` は廃止の予定。

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

`\bxjs@param@mag` `mag` オプションの値。

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

`mag` オプションの処理。

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

`paper` オプションの処理。

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

`\bxjs@jadriver` 和文ドライバの名前。

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

`\bxjs@jadriver@opt` 明示された和文ドライバの名前。

```
643 %\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)。

```
644 \define@key{bxjs}{jadriver}{%
```

```
645 \bxjs@depre@opt{jadriver}{ja=#1}\edef\bxjs@jadriver@opt{#1}}
```

```
646 \define@key{bxjs}{ja}{\relax}{%
```

```
647 \ifx\relax#1\else\edef\bxjs@jadriver@opt{#1}\fi}
```

`\jsJaFont` 和文フォント設定の名前。

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

`jafont` オプションの処理。

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

`\jsJaParam` 和文ドライバパラメタの文字列。

```
650 \let\jsJaParam\@empty
```

`japaram` オプションの処理。

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

```
653 \define@key{bxjs}{pandoc}[]{%
654 \ExecuteOptions{pandoc}%
655 \edef\jsJaParam{\bxjs@catopt\jsJaParam{#1}}
656 \define@key{bxjs}{pandoc+}[]{%
657 \ExecuteOptions{pandoc+}%
658 \edef\jsJaParam{\bxjs@catopt\jsJaParam{#1}}}
```

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

```
659 \let\bxjs@magstyle@@mag=m
660 \let\bxjs@magstyle@@real=r
661 \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。

```
662 \let\bxjs@magstyle@@usemag\bxjs@magstyle@@mag
663 \let\bxjs@magstyle@@nomag\bxjs@magstyle@@real
664 \bxjs@cslet{bxjs@magstyle@@nomag*}\bxjs@magstyle@@xreal
```

`\bxjs@magstyle@@default` は既定の値を表す。

```
665 \let\bxjs@magstyle@@default\bxjs@magstyle@@usemag
666 \ifx l\jsEngine \ifnum\luatexversion>86
667 \let\bxjs@magstyle@@default\bxjs@magstyle@@xreal
668 \fi\fi
669 \ifjsWithpTeXng
670 \let\bxjs@magstyle@@default\bxjs@magstyle@@xreal
671 \fi
672 \let\bxjs@magstyle\bxjs@magstyle@@default
```

`magstyle` オプションの処理。

```
673 \define@key{bxjs}{magstyle}{%
674 \bxjs@csletcs{bxjs@magstyle}{bxjs@magstyle@@#1}%
675 \ifx\bxjs@magstyle\relax
676 \bxjs@error@keyval{magstyle}{#1}%
677 \let\bxjs@magstyle\bxjs@magstyle@@default
678 \fi}
```

`\bxjs@geometry` `geometry` オプションの指定値。

```
679 \let\bxjs@geometry@@class=c
680 \let\bxjs@geometry@@user=u
681 \bxjs@declare@enum@option{geometry}{geometry}{class}
```

`\ifbxjs@fancyhdr` [スイッチ] fancyhdr の指定値。fancyhdr パッケージに対する調整を行うか。

```
682 \bxjs@declare@bool@option{fancyhdr}{fancyhdr}{true}
```

`\ifbxjs@dvi@opt` [スイッチ] dvi オプションが指定されたか。

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

DVI モードのドライバとドライバ種別との対応。

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

dvi オプションの処理。

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

`\bxjs@driver@given` を未定義にしていることに注意。

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

```
699 \newif\ifbxjs@layout@buggyhmargin
```

`\ifbxjs@force@chapterabstract` [スイッチ] abstract 環境を chapterabstract にするか。

※ bxjsbook では常に真。bxjsreport では layout が v1 の場合に真になる。

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

layout オプションの処理。

```
702 \@namedef{bxjs@kv@layout@v1}{%
703 %<book>\bxjs@layout@buggyhmargintrue
704 %<report>\bxjs@force@chapterabstracttrue
705 }
706 \@namedef{bxjs@kv@layout@v2}{%
707 %<book>\bxjs@layout@buggyhmarginfalse
708 %<report>\bxjs@force@chapterabstractfalse
709 }
710 \define@key{bxjs}{layout}{%
711   \bxjs@set@keyval{layout}{#1}{}}
```

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

```

712 %\let\bxjs@textwidth@limit@opt\@undefined
713 \define@key{bxjs}{textwidth-limit}{%
714   \bxjs@depre@opt{textwidth-limit}{textwidth=#1zw}%
715   \edef\bxjs@textwidth@limit@opt{#1}}

```

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

```

716 %\let\bxjs@textwidth@opt\@undefined
717 \define@key{bxjs}{textwidth}{\edef\bxjs@textwidth@opt{#1}}
718 \define@key{bxjs}{line_length}{\setkeys{bxjs}{textwidth=#1}}

```

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

```

719 %\let\bxjs@number@of@lines@opt\@undefined
720 \define@key{bxjs}{number-of-lines}{\edef\bxjs@number@of@lines@opt{#1}}
721 \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。

```

722 %\let\bxjs@paragraph@mark\@undefined
723 \define@key{bxjs}{paragraph-mark}{%
724   \edef\bxjs@paragraph@mark{#1}}

```

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

```

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

```

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

```

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

```

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

```

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

```

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

```

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

```

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

```

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

```

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

```

739 \bxjs@declare@bool@option{use-zw}{usezw}{true}

```



```
740 \DeclareOption{noz}{\bxjs@depre@opt@do{noz}{use-zw=false}}
741 \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。

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

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

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

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

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

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

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

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

```
762 %<article>\ExecuteOptions{a4paper,oneside,onecolumn,notitlepage,final}
763 %<book>\ExecuteOptions{a4paper,twoside,onecolumn,titlepage,openright,final}
764 %<report>\ExecuteOptions{a4paper,oneside,onecolumn,titlepage,openany,final}
765 %<slide>\ExecuteOptions{36pt,a4paper,landscape,oneside,onecolumn,titlepage,final}
766 \ProcessOptions\relax
767 \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。

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

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

```
792 \@expandtwoargs\@removeelement
```

```

793 {papersize}\@classoptionslist\@classoptionslist
794 \@expandtwoargs\@removeelement
795 {10pt}\@classoptionslist\@classoptionslist
796 \@expandtwoargs\@removeelement
797 {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 pL<sup>A</sup>T<sub>E</sub>X か upL<sup>A</sup>T<sub>E</sub>X 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] pL<sup>A</sup>T<sub>E</sub>X/ upL<sup>A</sup>T<sub>E</sub>X を自動判別するオプション autodetect-engine を新設しました。upL<sup>A</sup>T<sub>E</sub>X 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。

```

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

```

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

```

811 \ifx *\bxjs@engine@given
812 \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)pL<sup>A</sup>T<sub>E</sub>X 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。

```

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

```

```

817 \fi\fi
818 \fi
819 \ifx\bxjs@engine@given\@undefined\else
820 \ifx\bxjs@engine@given\bxjs@tmpb\else
821 \ClassError\bxjs@clsname
822 {Option '\bxjs@engine@opt' used on wrong engine}\@ehc
823 \fi
824 \fi


```

エンジンが pT<sub>E</sub>X-ng の場合、グローバルオプションに `uplatex` を追加する。

```

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

```

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

```

DVI 出力のエンジンである場合の追加処理。

```

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

```

```
858 \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)。

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

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

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

```
886 \ifx\bxjs@driver@given\bxjs@driver@@none
887   \bxjs@papersizefalse
888 \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。

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

標準の和文ドライバの名前の定数。

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

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

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

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

```
923 \ifx\bxjs@magstyle@@default\bxjs@magstyle@@mag\else
```

```

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

```

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

```

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

```

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

```

949 \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。

```

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

```
955 \g@addto@macro\@classoptionslist{,nosetpagesize}
```

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

```
956 \ifbxjs@oldfontcommands
```

```
957 \AtEndOfClass{\allowoldfontcommands}
```

```
958 \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 pL<sup>A</sup>T<sub>ε</sub>X 2<sub>ε</sub> はトンボ出力幅を両側に 1 インチとっていますので、dvips 使用時に

```
-0 -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] L<sup>A</sup>T<sub>ε</sub>X 2<sub>ε</sub> 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 L<sup>A</sup>T<sub>ε</sub>X 2<sub>ε</sub> 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。

---

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

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

```
961 \begingroup\expandafter\expandafter\expandafter\endgroup
```

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

```
963 \setlength{\stockwidth}{\paperwidth}
```

```
964 \setlength{\stockheight}{\paperheight}
```

```
965 \advance \stockwidth 2in
```

```
966 \advance \stockheight 2in
```

```
967 \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。

```
968 %<slide>\def\n@baseline{13}%
969 %<!slide>\ifdim\bxjs@param@basefontsize<10pt \def\n@baseline{15}%
970 %<!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。

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

---

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

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

- `geometry` 側でオプション `truedimen` を指定してください。
- `geometry` 側でオプション `mag` は使えません。

---

設定すべき `\mag` 値を (基底サイズ)/(10 pt) × 1000 と算出。BXJS クラスでは、`\mag` を直接指定したい場合は、`geometry` 側ではなくクラスのオプションで行うものとする。

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

```

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

```

---

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

---

※ 2.9 版において \p@? 表記を廃止。

```

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

```

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

use-zw が真の時は \zw を \jsZw と同義にする。

```

1006 \newdimen\jsZw
1007 \jsZw=10\jsc@mpt \jsZw=\jsScale\jsZw
1008 \ifbxjs@usezw
1009 \providecommand*\zw{\jsZw}
1010 \fi

```

\zwspace 全角幅の水平空き。

```

1011 \def\zwspace{\hskip\jsZw\relax}

```

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

```

1012 \ifjsc@mag@xreal
1013 \RequirePackage{type1cm}
1014 \let\jsc@invscale\bxjs@invscale

```

---

```

1015 \ifbxjs@TUenc
1016 \expandafter\let\csname TU/lmr/m/n/10\endcsname\relax
1017 \else
1018 \expandafter\let\csname OT1/cmr/m/n/10\endcsname\relax
1019 \fi
1020 \expandafter\let\csname OMX/cmex/m/n/10\endcsname\relax
1021 \let\jsc@get@external@font\get@external@font
1022 \def\get@external@font{%
1023 \jsc@preadjust@extract@font
1024 \jsc@get@external@font}
1025 \def\jsc@fstrunc#1{%

```

```

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

```
\jsc@bigskip 1060 %\def\jsc@medskip{\vspace\jsc@medskipamount}
```

```
1061 %\def\jsc@bigskip{\vspace\jsc@bigskipamount}
```

```
\jsc@smallskipamount
```

```
\jsc@medskipamount 1062 \newskip\jsc@smallskipamount
```

```
\jsc@bigskipamount 1063 \jsc@smallskipamount=3\jsc@mpt plus 1\jsc@mpt minus 1\jsc@mpt
```

```
1064 %\newskip\jsc@medskipamount
```

```
1065 %\jsc@medskipamount =6\jsc@mpt plus 2\jsc@mpt minus 2\jsc@mpt
```

```
1066 %\newskip\jsc@bigskipamount
```

```
1067 %\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] トンボオプションが指定されているとき「だけ」`\stockwidth`, `\stockheight` が定義されています。

■**pagesize スペシャルの出力** [2003-05-17] `dvipdfm(x)` の `pagesize` スペシャルを出力します。

[2004-08-08] 今の `dvipdfmx` は `dvips` 用スペシャルを理解するようなので外しました。

```
1068 % \ifpapersize
1069 %   \setlength{\@tempdima}{\paperwidth}
1070 %   \setlength{\@tempdimb}{\paperheight}
1071 %   \iftombow
1072 %     \advance \@tempdima 2truein
1073 %     \advance \@tempdimb 2truein
1074 %   \fi
1075 %   \AtBeginDvi{\special{pdf: pagesize width \the\@tempdima\space height \the\@tempdimb}}
1076 % \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。

---

```
1077 %</class>
1078 %<*\jsclasses>
1079 %<*\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,  $\text{T}_{\text{E}}\text{X}$  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,  $\text{pT}_{\text{E}}\text{X}$  (アスキーが日本語化した  $\text{T}_{\text{E}}\text{X}$ ) 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。

```

1080 %</class>
1081 %<*minijs>
1082 %% min/goth -> jis/jisg (for pLaTeX only)
1083 \ifnum\jis"2121="3000 \else
1084 \for\@tempa:=5,6,7,8,9,10,10.95,12,14.4,17.28,20.74,24.88\do{%
1085   \expandafter\let\csname JY1/mc/m/n/\@tempa\endcsname\relax
1086   \expandafter\let\csname JY1/gt/m/n/\@tempa\endcsname\relax
1087   \expandafter\let\csname JT1/mc/m/n/\@tempa\endcsname\relax
1088   \expandafter\let\csname JT1/gt/m/n/\@tempa\endcsname\relax
1089 }
1090 \def\Cjascale{0.924690}
1091 \DeclareFontShape{JY1}{mc}{m}{n}{<-> s * [0.961] jis}{}
1092 \DeclareFontShape{JY1}{gt}{m}{n}{<-> s * [0.961] jisg}{}
1093 \DeclareFontShape{JT1}{mc}{m}{n}{<-> s * [0.961] tmin10}{}
1094 \DeclareFontShape{JT1}{gt}{m}{n}{<-> s * [0.961] tgoth10}{}
1095 \fi
1096 %</minijs>
1097 %<*class>
1098 %<!*jspf>
1099 \def\Cjascale{0.924690}
1100 \ifmingoth
1101   \DeclareFontShape{\jsc@JYn}{mc}{m}{n}{<-> s * [0.961] \jsc@pfx@ min10}{}
1102   \DeclareFontShape{\jsc@JYn}{gt}{m}{n}{<-> s * [0.961] \jsc@pfx@ goth10}{}
1103   \DeclareFontShape{\jsc@JTn}{mc}{m}{n}{<-> s * [0.961] \jsc@pfx@ tmin10}{}
1104   \DeclareFontShape{\jsc@JTn}{gt}{m}{n}{<-> s * [0.961] \jsc@pfx@ tgoth10}{}
1105 \else
1106   \ifjisfont
1107     \DeclareFontShape{\jsc@JYn}{mc}{m}{n}{<-> s * [0.961] \jsc@pfx@ jis}{}
1108     \DeclareFontShape{\jsc@JYn}{gt}{m}{n}{<-> s * [0.961] \jsc@pfx@ jisg}{}
1109     \DeclareFontShape{\jsc@JTn}{mc}{m}{n}{<-> s * [0.961] \jsc@pfx@ tmin10}{}
1110     \DeclareFontShape{\jsc@JTn}{gt}{m}{n}{<-> s * [0.961] \jsc@pfx@ tgoth10}{}
1111   \else
1112     \if@jsc@uplatex
1113       \DeclareFontShape{JY2}{mc}{m}{n}{<-> s * [0.924690] upjisr-h}{}
1114       \DeclareFontShape{JY2}{gt}{m}{n}{<-> s * [0.924690] upjisg-h}{}
1115       \DeclareFontShape{JT2}{mc}{m}{n}{<-> s * [0.924690] upjisr-v}{}
1116       \DeclareFontShape{JT2}{gt}{m}{n}{<-> s * [0.924690] upjisg-v}{}
1117     \else
1118       \DeclareFontShape{\jsc@JYn}{mc}{m}{n}{<-> s * [0.961] \jsc@pfx@ jis}{}
1119       \DeclareFontShape{\jsc@JYn}{gt}{m}{n}{<-> s * [0.961] \jsc@pfx@ jisg}{}

```

```

1120     \DeclareFontShape{\jsc@JTn}{mc}{m}{n}{<-> s * [0.961] \jsc@pfx@ tmin10}{-}
1121     \DeclareFontShape{\jsc@JTn}{gt}{m}{n}{<-> s * [0.961] \jsc@pfx@ tgoth10}{-}
1122     \fi
1123     \fi
1124 \fi
1125 %</!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。

```

1126 %<*jspf>
1127 \def\Cjascale{0.903375}
1128 \ifmingoth
1129 \DeclareFontShape{\jsc@JYn}{mc}{m}{n}{<-> s * [0.93885] \jsc@pfx@ min10}{-}
1130 \DeclareFontShape{\jsc@JYn}{gt}{m}{n}{<-> s * [0.93885] \jsc@pfx@ goth10}{-}
1131 \DeclareFontShape{\jsc@JTn}{mc}{m}{n}{<-> s * [0.93885] \jsc@pfx@ tmin10}{-}
1132 \DeclareFontShape{\jsc@JTn}{gt}{m}{n}{<-> s * [0.93885] \jsc@pfx@ tgoth10}{-}
1133 \else
1134 \ifjisfont
1135 \DeclareFontShape{\jsc@JYn}{mc}{m}{n}{<-> s * [0.93885] \jsc@pfx@ jis}{-}
1136 \DeclareFontShape{\jsc@JYn}{gt}{m}{n}{<-> s * [0.93885] \jsc@pfx@ jisg}{-}
1137 \DeclareFontShape{\jsc@JTn}{mc}{m}{n}{<-> s * [0.93885] \jsc@pfx@ tmin10}{-}
1138 \DeclareFontShape{\jsc@JTn}{gt}{m}{n}{<-> s * [0.93885] \jsc@pfx@ tgoth10}{-}
1139 \else
1140 \if@jsc@uplatex
1141 \DeclareFontShape{JY2}{mc}{m}{n}{<-> s * [0.903375] upjisr-h}{-}
1142 \DeclareFontShape{JY2}{gt}{m}{n}{<-> s * [0.903375] upjisg-h}{-}
1143 \DeclareFontShape{JT2}{mc}{m}{n}{<-> s * [0.903375] upjisr-v}{-}
1144 \DeclareFontShape{JT2}{gt}{m}{n}{<-> s * [0.903375] upjisg-v}{-}
1145 \else
1146 \DeclareFontShape{\jsc@JYn}{mc}{m}{n}{<-> s * [0.93885] \jsc@pfx@ jis}{-}
1147 \DeclareFontShape{\jsc@JYn}{gt}{m}{n}{<-> s * [0.93885] \jsc@pfx@ jisg}{-}
1148 \DeclareFontShape{\jsc@JTn}{mc}{m}{n}{<-> s * [0.93885] \jsc@pfx@ tmin10}{-}
1149 \DeclareFontShape{\jsc@JTn}{gt}{m}{n}{<-> s * [0.93885] \jsc@pfx@ tgoth10}{-}
1150 \fi
1151 \fi
1152 \fi
1153 %</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ふ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。

```
1154 % \DeclareFontShape{\jsc@JYn}{mc}{bx}{n}{<->ssub*gt/m/n}{ } % in \jsc@JYnmc
1155 % \DeclareFontShape{\jsc@JYn}{gt}{bx}{n}{<->ssub*gt/m/n}{ } % in \jsc@JYngt
1156 \DeclareFontShape{\jsc@JYn}{mc}{m}{it}{<->ssub*mc/m/n}{ }
1157 \DeclareFontShape{\jsc@JYn}{mc}{m}{sl}{<->ssub*mc/m/n}{ }
1158 \DeclareFontShape{\jsc@JYn}{mc}{m}{sc}{<->ssub*mc/m/n}{ }
1159 \DeclareFontShape{\jsc@JYn}{gt}{m}{it}{<->ssub*gt/m/n}{ }
1160 \DeclareFontShape{\jsc@JYn}{gt}{m}{sl}{<->ssub*gt/m/n}{ }
1161 \DeclareFontShape{\jsc@JYn}{mc}{bx}{it}{<->ssub*gt/m/n}{ }
1162 \DeclareFontShape{\jsc@JYn}{mc}{bx}{sl}{<->ssub*gt/m/n}{ }
1163 % \DeclareFontShape{\jsc@JTn}{mc}{bx}{n}{<->ssub*gt/m/n}{ } % in \jsc@JTnmc
1164 % \DeclareFontShape{\jsc@JTn}{gt}{bx}{n}{<->ssub*gt/m/n}{ } % in \jsc@JTngt
1165 \DeclareFontShape{\jsc@JTn}{mc}{m}{it}{<->ssub*mc/m/n}{ }
1166 \DeclareFontShape{\jsc@JTn}{mc}{m}{sl}{<->ssub*mc/m/n}{ }
1167 \DeclareFontShape{\jsc@JTn}{mc}{m}{sc}{<->ssub*mc/m/n}{ }
1168 \DeclareFontShape{\jsc@JTn}{gt}{m}{it}{<->ssub*gt/m/n}{ }
1169 \DeclareFontShape{\jsc@JTn}{gt}{m}{sl}{<->ssub*gt/m/n}{ }
1170 \DeclareFontShape{\jsc@JTn}{mc}{bx}{it}{<->ssub*gt/m/n}{ }
1171 \DeclareFontShape{\jsc@JTn}{mc}{bx}{sl}{<->ssub*gt/m/n}{ }
```

[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。

```
1172 %</class>
1173 %<*class|minijs>
1174 %% ad-hoc "relation font"
1175 \@ifl@t@r\fmtversion{2020/10/01}
1176   {\jsc@needspace@tchfalse}{\jsc@needspace@tchtrue}
1177 \ifjsc@needspace@tch      % --- for 2020-02-02 or older BEGIN
1178 \ifx\@rmfamilyhook\@undefined % old
1179 \DeclareRobustCommand\rmfamily
1180   {\not@math@alphabet\rmfamily\mathrm
1181     \romanfamily\rmdefault\kanjifamily\mcdefault\selectfont}
1182 \DeclareRobustCommand\sffamily
1183   {\not@math@alphabet\sffamily\mathsf
1184     \romanfamily\sfdefault\kanjifamily\gtdefault\selectfont}
1185 \DeclareRobustCommand\ttfamily
1186   {\not@math@alphabet\ttfamily\mathtt
1187     \romanfamily\ttdefault\kanjifamily\gtdefault\selectfont}
1188 \AtBeginDocument{%
1189   \ifx\mweights@init\@undefined\else % mweights.sty is loaded
1190     % my definitions above should have been overwritten, recover it!
1191     % \selectfont is executed twice but I don't care about speed...
1192     \expandafter\g@addto@macro\csname rmfamily \endcsname
1193     {\kanjifamily\mcdefault\selectfont}%
```

```

1194 \expandafter\g@addto@macro\csname sffamily \endcsname
1195   {\kanjifamily\gtdefault\selectfont}%
1196 \expandafter\g@addto@macro\csname ttfamily \endcsname
1197   {\kanjifamily\gtdefault\selectfont}%
1198 \fi}
1199 \else % 2020-02-02
1200 \g@addto@macro\@rmfamilyhook
1201   {\prepare@family@series@update@kanji{mc}\mcdefault}
1202 \g@addto@macro\@sffamilyhook
1203   {\prepare@family@series@update@kanji{gt}\gtdefault}
1204 \g@addto@macro\@ttfamilyhook
1205   {\prepare@family@series@update@kanji{gt}\gtdefault}
1206 \fi
1207 \else % --- for 2020-02-02 or older END & for 2020-10-01 BEGIN
1208 \AddToHook{rmfamily}%
1209   {\prepare@family@series@update@kanji{mc}\mcdefault}
1210 \AddToHook{sffamily}%
1211   {\prepare@family@series@update@kanji{gt}\gtdefault}
1212 \AddToHook{ttfamily}%
1213   {\prepare@family@series@update@kanji{gt}\gtdefault}
1214 \fi % --- for 2020-10-01 END
1215 %</class|minijs>
1216 %<*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。

```

1217 \ifx\DeclareFixJFMCJKTextFontCommand\@undefined
1218 \DeclareRobustCommand\textmc[1]{%
1219   \relax\ifmmode \expandafter\nfss@text \fi{\mcfamily #1}}
1220 \DeclareRobustCommand\textgt[1]{%
1221   \relax\ifmmode \expandafter\nfss@text \fi{\gtfamily #1}}
1222 \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。

```
1223 \def\reDeclareMathAlphabet#1#2#3{%
1224   \edef\@tempa{\expandafter@gobble\string#2}%
1225   \edef\@tempb{\expandafter@gobble\string#3}%
1226   \edef\@tempc{\string @\expandafter@gobbletwo\string#2}%
1227   \ifx\@tempc\@tempa%
1228     \edef\@tempa{\expandafter@gobbletwo\string#2}%
1229     \edef\@tempb{\expandafter@gobbletwo\string#3}%
1230   \fi
1231   \begingroup
1232     \let\protect\noexpand
1233     \def\@tempaa{\relax}%
1234     \expandafter\ifx\csname RDMAorg@\@tempa\endcsname\relax
1235       \edef\@tempaa{\expandafter\def\expandafter\noexpand%
1236         \csname RDMAorg@\@tempa\endcsname{%
1237           \expandafter\noexpand\csname\@tempa\endcsname}}%
1238     \fi
1239     \def\@tempbb{\relax}%
1240     \expandafter\ifx\csname RDMAorg@\@tempb\endcsname\relax
1241       \edef\@tempbb{\expandafter\def\expandafter\noexpand%
1242         \csname RDMAorg@\@tempb\endcsname{%
1243           \expandafter\noexpand\csname\@tempb\endcsname}}%
1244     \fi
1245     \edef\@tempc{\@tempaa\@tempbb}%
1246     \expandafter\endgroup\@tempc%
1247     \edef#1{\noexpand\protect\expandafter\noexpand\csname%
1248       \expandafter@gobble\string#1\space\space\endcsname}%
1249     \expandafter\edef\csname\expandafter@gobble\string#1\space\space\endcsname%
1250       {\noexpand\DualLang@mathalph@bet%
1251         {\expandafter\noexpand\csname RDMAorg@\@tempa\endcsname}%
1252         {\expandafter\noexpand\csname RDMAorg@\@tempb\endcsname}%
1253       }%
1254   }
1255 \onlypreamble\reDeclareMathAlphabet
1256 \def\DualLang@mathalph@bet#1#2{%
1257   \relax\ifmmode
1258     \ifx\math@bgroup\bgroup%      2e normal style (\mathrm{...})
1259       \bgroup\let\DualLang@Mfontsw\DLMfontsw@standard
1260     \else
1261       \ifx\math@bgroup\relax%    2e two letter style (\rm->\mathrm)
1262         \let\DualLang@Mfontsw\DLMfontsw@oldstyle
1263       \else
1264         \ifx\math@bgroup\@empty%  2.09 oldfont style ({\mathrm ...})
```

```

1265     \let\DualLang@Mfontsw\DLMfontsw@oldfont
1266     \else%                panic! assume 2e normal style
1267     \bgroup\let\DualLang@Mfontsw\DLMfontsw@standard
1268     \fi
1269     \fi
1270     \fi
1271 \else
1272     \let\DualLang@Mfontsw@firstoftwo
1273     \fi
1274 \DualLang@Mfontsw{#1}{#2}%
1275 }
1276 \def\DLMfontsw@standard#1#2#3{#1{#2{#3}}\egroup}
1277 \def\DLMfontsw@oldstyle#1#2{#1\relax\@fontswitch\relax{#2}}
1278 \def\DLMfontsw@oldfont#1#2{#1\relax#2\relax}
1279 \if@enablejfam
1280 \DeclareSymbolFont{mincho}{\jsc@JYn}{mc}{m}{n}
1281 \DeclareSymbolFontAlphabet{\mathmc}{mincho}
1282 \SetSymbolFont{mincho}{bold}{\jsc@JYn}{gt}{m}{n}
1283 \jfam\symmincho
1284 \DeclareMathAlphabet{\mathgt}{\jsc@JYn}{gt}{m}{n}
1285 \AtBeginDocument{%
1286     \reDeclareMathAlphabet{\mathrm}{\@mathrm}{\@mathmc}
1287     \reDeclareMathAlphabet{\mathbf}{\@mathbf}{\@mathgt}}
1288 \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。

```
1289 % \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。

```

1290 \prebreakpenalty\jis"2147=10000    % 5000  '
1291 \postbreakpenalty\jis"2148=10000   % 5000  "
1292 \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。

```
1293 \inhibitxspcode`!=1
```

```
1294 \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

```
1295 % \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。

1296 \xspcode`+=3

1297 \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。

1298 \xspcode`^^80=3

1299 \xspcode`^^81=3

1300 \xspcode`^^82=3

1301 \xspcode`^^83=3

1302 \xspcode`^^84=3

1303 \xspcode`^^85=3

1304 \xspcode`^^86=3

1305 \xspcode`^^87=3

1306 \xspcode`^^88=3

1307 \xspcode`^^89=3

1308 \xspcode`^^8a=3

1309 \xspcode`^^8b=3

1310 \xspcode`^^8c=3

1311 \xspcode`^^8d=3

1312 \xspcode`^^8e=3

1313 \xspcode`^^8f=3

1314 \xspcode`^^90=3

1315 \xspcode`^^91=3

1316 \xspcode`^^92=3

1317 \xspcode`^^93=3

1318 \xspcode`^^94=3

1319 \xspcode`^^95=3

1320 \xspcode`^^96=3

1321 \xspcode`^^97=3

1322 \xspcode`^^98=3

1323 \xspcode`^^99=3

1324 \xspcode`^^9a=3

1325 \xspcode`^^9b=3

1326 \xspcode`^^9c=3

1327 \xspcode`^^9d=3

1328 \xspcode`^^9e=3

1329 \xspcode`^^9f=3

1330 \xspcode`^^a0=3

1331 \xspcode`^^a1=3

1332 \xspcode`^^a2=3

1333 \xspcode`^^a3=3

1334 \xspcode`^^a4=3

1335 \xspcode`^^a5=3

1336 \xspcode`^^a6=3  
1337 \xspcode`^^a7=3  
1338 \xspcode`^^a8=3  
1339 \xspcode`^^a9=3  
1340 \xspcode`^^aa=3  
1341 \xspcode`^^ab=3  
1342 \xspcode`^^ac=3  
1343 \xspcode`^^ad=3  
1344 \xspcode`^^ae=3  
1345 \xspcode`^^af=3  
1346 \xspcode`^^b0=3  
1347 \xspcode`^^b1=3  
1348 \xspcode`^^b2=3  
1349 \xspcode`^^b3=3  
1350 \xspcode`^^b4=3  
1351 \xspcode`^^b5=3  
1352 \xspcode`^^b6=3  
1353 \xspcode`^^b7=3  
1354 \xspcode`^^b8=3  
1355 \xspcode`^^b9=3  
1356 \xspcode`^^ba=3  
1357 \xspcode`^^bb=3  
1358 \xspcode`^^bc=3  
1359 \xspcode`^^bd=3  
1360 \xspcode`^^be=3  
1361 \xspcode`^^bf=3  
1362 \xspcode`^^c0=3  
1363 \xspcode`^^c1=3  
1364 \xspcode`^^c2=3  
1365 \xspcode`^^c3=3  
1366 \xspcode`^^c4=3  
1367 \xspcode`^^c5=3  
1368 \xspcode`^^c6=3  
1369 \xspcode`^^c7=3  
1370 \xspcode`^^c8=3  
1371 \xspcode`^^c9=3  
1372 \xspcode`^^ca=3  
1373 \xspcode`^^cb=3  
1374 \xspcode`^^cc=3  
1375 \xspcode`^^cd=3  
1376 \xspcode`^^ce=3  
1377 \xspcode`^^cf=3  
1378 \xspcode`^^d0=3  
1379 \xspcode`^^d1=3  
1380 \xspcode`^^d2=3  
1381 \xspcode`^^d3=3  
1382 \xspcode`^^d4=3  
1383 \xspcode`^^d5=3  
1384 \xspcode`^^d6=3

```

1385 \xspcode^^^d7=3
1386 \xspcode^^^d8=3
1387 \xspcode^^^d9=3
1388 \xspcode^^^da=3
1389 \xspcode^^^db=3
1390 \xspcode^^^dc=3
1391 \xspcode^^^dd=3
1392 \xspcode^^^de=3
1393 \xspcode^^^df=3
1394 \xspcode^^^e0=3
1395 \xspcode^^^e1=3
1396 \xspcode^^^e2=3
1397 \xspcode^^^e3=3
1398 \xspcode^^^e4=3
1399 \xspcode^^^e5=3
1400 \xspcode^^^e6=3
1401 \xspcode^^^e7=3
1402 \xspcode^^^e8=3
1403 \xspcode^^^e9=3
1404 \xspcode^^^ea=3
1405 \xspcode^^^eb=3
1406 \xspcode^^^ec=3
1407 \xspcode^^^ed=3
1408 \xspcode^^^ee=3
1409 \xspcode^^^ef=3
1410 \xspcode^^^f0=3
1411 \xspcode^^^f1=3
1412 \xspcode^^^f2=3
1413 \xspcode^^^f3=3
1414 \xspcode^^^f4=3
1415 \xspcode^^^f5=3
1416 \xspcode^^^f6=3
1417 \xspcode^^^f7=3
1418 \xspcode^^^f8=3
1419 \xspcode^^^f9=3
1420 \xspcode^^^fa=3
1421 \xspcode^^^fb=3
1422 \xspcode^^^fc=3
1423 \xspcode^^^fd=3
1424 \xspcode^^^fe=3
1425 \xspcode^^^ff=3
1426 %</class>
1427 %</jsclasses>
1428 %<*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 `sfcodes` 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。

---

```
1429 \chardef\bxjs@periodchar=\  
1430 \bxjs@robust@def\bxjs@SE{%  
1431 \ifnum\spacefactor<\@m \spacefactor\@m  
1432 \else \spacefactor\sfcodes\bxjs@periodchar  
1433 \fi}  
1434 \ifbxjs@fix@at@cmd  
1435 \def\@{\bxjs@SE{}}  
1436 \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>\@viiipt</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を着目して、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。

```
1437 %\let\bxjs@prev@set@fontsize\@undefined
1438 \@onlypreamble\bxjs@patch@set@fontsize
1439 \def\bxjs@patch@set@fontsize{%
1440   \ifx\bxjs@prev@set@fontsize\set@fontsize\else
1441     \def\bxjs@tmpa{\def\set@fontsize####1####2####3}%
1442   \expandafter\bxjs@tmpa\expandafter{%
1443     \set@fontsize{##1}{##2}{##3}%
1444   % 末尾にコードを追加
1445   \expandafter\def\expandafter\size@update\expandafter{%
1446     \size@update
1447     \jsFontSizeChanged}%
1448 }
1449   \let\bxjs@prev@set@fontsize\set@fontsize
1450   \fi}
```

この場とパッケージ末尾で作動させる。

```
1451 \bxjs@patch@set@fontsize
1452 \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。

```
1453 \newcommand*\jsFontSizeChanged{%
1454   \jsZw=\f@size\p@
1455   \jsZw=\jsScale \jsZw
1456   \ifdim\parindent>\z@
1457     \if@english \parindent=1em
1458     \else       \parindent=1\jsZw
1459   \fi
1460   \fi\relax
1461   \jsResetDimen}
```

`\jsResetDimen` ユーザ定義用のフック。

```
1462 \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。

```
1463 \ifjsc@mag
1464   \let\jsc@setfontsize\@setfontsize
1465 \else
1466   \def\jsc@setfontsize#1#2#3{%
1467     \@setfontsize#1{#2\jsc@empt}{#3\jsc@empt}}
1468 % microtype 対策
1469 \ifjswitheTeX\if j\jsEngine\else
1470   \def\jsc@setfontsize#1#2#3{%
1471     \edef\bxjs@sfs@next{%
1472       \unexpanded{\@setfontsize#1}%
1473       {\the\dimexpr#2\jsc@empt\relax}{\the\dimexpr#3\jsc@empt\relax}%
1474     }\bxjs@sfs@next}
1475 \fi\fi
1476 \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。

---

```
1477 \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。

---

```
1478 \newif\ifnarrowbaselines
```

```
1479 \if@english
```



```

1480 \narrowbaselinestruе
1481 \fi
1482 \def\narrowbaselines{%
1483 \narrowbaselinestruе
1484 \skip0=\abovedisplayskip
1485 \skip2=\abovedisplaysshortskip
1486 \skip4=\belowdisplayskip
1487 \skip6=\belowdisplaysshortskip
1488 % 一時的に警告を無効化する
1489 \let\bxjs@save@nomath\@nomath
1490 \let\@nomath\@gobble
1491 \@currsize\selectfont
1492 \let\@nomath\bxjs@save@nomath
1493 \abovedisplayskip=\skip0
1494 \abovedisplaysshortskip=\skip2
1495 \belowdisplayskip=\skip4
1496 \belowdisplaysshortskip=\skip6\relax}
1497 \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。

```

1498 \def\bxjs@if@narrowbaselines{%
1499 \ifnarrowbaselines\expandafter\@firstoftwo
1500 \else \expandafter\@secondoftwo
1501 \fi
1502 }

```

---

\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 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。

---

```

1503 \renewcommand{\normalsize}{%
1504 \bxjs@if@narrowbaselines{%
1505 \jsc@setfontsize\normalsize\@xpt\@xiipt
1506 }{%else

```

```

1507 \jsc@setfontsize\normalsize\@xpt{\n@baseline}%
1508 }%

```

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。

```

1509 \abovedisplayskip 11\jsc@mpt \@plus3\jsc@mpt \@minus4\jsc@mpt
1510 \abovedisplayshortskip \z@ \@plus3\jsc@mpt
1511 \belowdisplayskip 9\jsc@mpt \@plus3\jsc@mpt \@minus4\jsc@mpt
1512 \belowdisplayshortskip \belowdisplayskip

```

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

```

1513 \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。

```

1514 %</class>
1515 %<*class|minijs>
1516 %% initialize
1517 \normalsize
1518 %</class|minijs>
1519 %<*class>

```

\Cht 基準となる長さの設定をします。 p<sub>L</sub>A<sub>T</sub>E<sub>X</sub> 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 p<sub>T</sub>E<sub>X</sub> 系以外では未定義なのでここで定義する。

```

1520 \ifx\Cht\@undefined \newdimen\Cht \fi
1521 \ifx\Cdp\@undefined \newdimen\Cdp \fi
1522 \ifx\Cwd\@undefined \newdimen\Cwd \fi
1523 \ifx\Cvs\@undefined \newdimen\Cvs \fi
1524 \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。

```

1525 \setlength\Cht{0.88\jsZw}
1526 \setlength\Cdp{0.12\jsZw}
1527 \setlength\Cwd{1\jsZw}
1528 \setlength\Cvs{\baselineskip}
1529 \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。

```

1530 \newcommand{\small}{%
1531   \bxjs@if@narrowbaselines{%
1532     <!kiyou>   \jsc@setfontsize\small\@ixpt{11}%
1533     <kiyou>    \jsc@setfontsize\small{8.8888}{11}%
1534   }{%else
1535     <!kiyou>   \jsc@setfontsize\small\@ixpt{13}%
1536     <kiyou>    \jsc@setfontsize\small{8.8888}{13.2418}%
1537   }%
1538   \abovedisplayskip 9\jsc@mpt \@plus3\jsc@mpt \@minus4\jsc@mpt
1539   \abovedisplayshortskip \z@ \@plus3\jsc@mpt
1540   \belowdisplayskip \abovedisplayskip
1541   \belowdisplayshortskip \belowdisplayskip
1542   \def\@listi{\leftmargin\leftmargini
1543             \topsep \z@
1544             \parsep \z@
1545             \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。

```

1546 \newcommand{\footnotesize}{%
1547   \bxjs@if@narrowbaselines{%
1548     <!kiyou>   \jsc@setfontsize\footnotesize\@viiipt{9.5}%
1549     <kiyou>    \jsc@setfontsize\footnotesize{8.8888}{11}%
1550   }{%else
1551     <!kiyou>   \jsc@setfontsize\footnotesize\@viiipt{11}%
1552     <kiyou>    \jsc@setfontsize\footnotesize{8.8888}{13.2418}%
1553   }%
1554   \abovedisplayskip 6\jsc@mpt \@plus2\jsc@mpt \@minus3\jsc@mpt
1555   \abovedisplayshortskip \z@ \@plus2\jsc@mpt
1556   \belowdisplayskip \abovedisplayskip
1557   \belowdisplayshortskip \belowdisplayskip
1558   \def\@listi{\leftmargin\leftmargini
1559             \topsep \z@
1560             \parsep \z@
1561             \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  
`\tiny` 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` 行が揃うようになります。

`\LARGE` [2004-11-03] `\HUGE` を追加。

```

\huge 1562 \newcommand{\scriptsize}{\jsc@setfontsize\scriptsize\@viiipt\@viiipt}
1563 \newcommand{\tiny}{\jsc@setfontsize\tiny\@vpt\@vpt}
\Huge 1564 \if@twocolumn
\HUGE 1565 <!kiyou> \newcommand{\large}{\jsc@setfontsize\large\@xipt{\n@baseline}}

```

```

1566 %<kiyou> \newcommand{\large}{\jsc@setfontsize\large{11.111}{\n@baseline}}
1567 \else
1568 %<!kiyou> \newcommand{\large}{\jsc@setfontsize\large\@xiipt{17}}
1569 %<kiyou> \newcommand{\large}{\jsc@setfontsize\large{11.111}{17}}
1570 \fi
1571 %<!kiyou>\newcommand{\Large}{\jsc@setfontsize\Large\@xivpt{21}}
1572 %<kiyou>\newcommand{\Large}{\jsc@setfontsize\Large{12.222}{21}}
1573 \newcommand{\LARGE}{\jsc@setfontsize\LARGE\@xviipt{25}}
1574 \newcommand{\huge}{\jsc@setfontsize\huge\@xxpt{28}}
1575 \newcommand{\Huge}{\jsc@setfontsize\Huge\@xxvpt{33}}
1576 \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。

```

1577 \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。『`pLATEX 2ε` 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。

```

1578 % \newcommand{\headfont}{\bfseries}
1579 \newcommand{\headfont}{\sffamily}
1580 % \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。

```

1581 %<!kiyou>\setlength\columnsep{2\Cwd}
1582 %<kiyou>\setlength\columnsep{28truebp}
1583 \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

```

```

1584 \setlength\lineskip{1\jsc@empt}
1585 \setlength\normallineskip{1\jsc@empt}
1586 \setlength\lineskiplimit{1\jsc@empt}
1587 \setlength\normallineskiplimit{1\jsc@empt}

```

`\baselinestretch` 実際に行送りが `\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。

```
1588 \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。

```

1589 \setlength\parskip{\z@}
1590 \if@slide
1591 \setlength\parindent{0\p@}
1592 \else
1593 \setlength\parindent{1\Cwd}
1594 \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 `\@medpenalty` になっています。ここはオリジナル通りです。

```

\@highpenalty 1595 \@lowpenalty 51
1596 \@medpenalty 151
1597 \@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。

```
1598 % \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。

```
1599 % \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。

```

1600 \@onlypreamble\bxjs@bd@pre@geometry@hook
1601 \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。

```
1602 \ifjsc@mag
1603 \mag=\bxjs@param@mag
1604 \fi
1605 \setlength{\topskip}{10\jsc@empt}
```

`\jsSetQHLLength` のための和文単位の定義。

```
1606 \def\bxjs@unit@trueQ{0.25trueem}\let\bxjs@unit@trueH\bxjs@unit@trueQ
1607 \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。

```
1608 \@tempswafalse
1609 \def\bxjs@tmpdo{\@ifnextchar\bgroup\bxjs@tmpdo@a\remove@to@nnil}
1610 \def\bxjs@tmpdo@a#1{\edef\bxjs@tmpa{#1}%
1611 \@ifnextchar\bgroup\bxjs@tmpdo@b\remove@to@nnil}
1612 \def\bxjs@tmpdo@b#1{\edef\bxjs@tmpa{\bxjs@tmpa,#1}%
1613 \@ifnextchar\@nnil\bxjs@tmpdo@c\remove@to@nnil}
1614 \def\bxjs@tmpdo@c\@nnil{\@tempswatru
1615 \edef\bxjs@param@paper{papersize={\bxjs@tmpa}}}
1616 \expandafter\bxjs@tmpdo\bxjs@param@paper\@nnil
```

`W,H` の形式について。

```
1617 \if@tempswa\else
1618 \def\bxjs@tmpa{\@nil,\@nil}
1619 \def\bxjs@tmpdo#1,#2,#3\@nnil{%
1620 \def\bxjs@tmpb{#3}\ifx\bxjs@tmpa\bxjs@tmpb
1621 \@tempswatru\edef\bxjs@param@paper{papersize={#1,#2}}\fi}
1622 \expandafter\bxjs@tmpdo\bxjs@param@paper,\@nil,\@nil\@nnil
1623 \fi
```

`W*H` の形式について。

```
1624 \if@tempswa\else
1625 \def\bxjs@tmpa{\@nil*\@nil}
1626 \def\bxjs@tmpdo#1*#2*#3\@nnil{%
1627 \def\bxjs@tmpb{#3}\ifx\bxjs@tmpa\bxjs@tmpb
1628 \@tempswatru\edef\bxjs@param@paper{papersize={#1,#2}}\fi}
1629 \expandafter\bxjs@tmpdo\bxjs@param@paper*\@nil*\@nil\@nnil
1630 \fi
```

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

```
1631 \edef\bxjs@layout@paper{%
1632 \ifjsc@mag truedimen,\fi
1633 \if@landscape landscape,\fi
1634 \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。

```
1635 %<*article|report>
1636 \def\bxjs@layout@base{%
```

```

1637 headheight=\topskip,footskip=0.03367\paperheight,%
1638 headsep=\footskip-\topskip,includeheadfoot,%
1639 }
1640 \edef\bxjs@layout{\bxjs@layout@base
1641 hscale=0.76,hmarginratio=1:1,%
1642 vscale=0.83,vmarginratio=1:1,%
1643 }
1644 %</article|report>
1645 %<*book>
1646 \def\bxjs@layout@base{%
1647 headheight=\topskip,headsep=6\jsc@mmm,nofoot,includeheadfoot,%
1648 }
1649 \ifbxjs@layout@buggyhmargin %---
1650 % アレ
1651 \edef\bxjs@layout{\bxjs@layout@base
1652 hmargin=36\jsc@mmm,hmarginratio=1:1,%
1653 vscale=0.83,vmarginratio=1:1,%
1654 }
1655 \else %---
1656 % 非アレ
1657 \edef\bxjs@layout{\bxjs@layout@base
1658 hmargin=18\jsc@mmm,%
1659 vscale=0.83,vmarginratio=1:1,%
1660 }
1661 \fi %---
1662 %</book>
1663 %<*slide>
1664 \def\bxjs@layout@base{%
1665 noheadfoot,%
1666 }
1667 \edef\bxjs@layout{\bxjs@layout@base
1668 hscale=0.9,hmarginratio=1:1,%
1669 vscale=0.944,vmarginratio=1:1,%
1670 }
1671 %</slide>

textwidth オプションの設定を反映する。
1672 %<!*book>
1673 \ifx\bxjs@textwidth@opt\undefined\else
1674 \jsSetQHLength\tempdima{\bxjs@textwidth@opt}
1675 \edef\bxjs@layout{\bxjs@layout width=\the\tempdima,}
1676 \fi
1677 %</!book>
1678 \ifx\bxjs@number@of@lines@opt\undefined\else
1679 \bxjs@gset@tempcnta{\bxjs@number@of@lines@opt}
1680 \edef\bxjs@layout{\bxjs@layout lines=\the\tempcnta,}
1681 \fi

```

\fullwidth [寸法レジスタ] ヘッダ・フッタ領域の横幅。

1682 \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。

```
1683 %<*book>
1684 \newcommand\jsTextWidthLimit{40}
1685 \@tempdima=\jsTextWidthLimit\Cwd
1686 \ifx\bxjs@textwidth@limit@opt\undefined\else
1687   \bxjs@gset@tempcnta{\bxjs@textwidth@limit@opt}
1688   \@tempdima=\@tempcnta\Cwd
1689 \fi
1690 \ifx\bxjs@textwidth@opt\undefined\else
1691   \jsSetQHLength\@tempdima{\bxjs@textwidth@opt}
1692 \fi
1693 \edef\bxjs@textwidth@limit{\the\@tempdima}
1694 \ifdim\@tempdima=\jsTextWidthLimit\Cwd\else
1695   \bxjs@invscale\@tempdima{\strip@pt\Cwd}
1696   \long\edef\jsTextWidthLimit{\strip@pt\@tempdima}
1697 \fi
1698 %</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。

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

\bxjs@postproc@layout geometry の後処理。

```
1701 \def\bxjs@postproc@layout{%
geometry のドライバを再設定する。
1702   \ifx\bxjs@geometry@driver\relax\else
1703     \let\Gm@driver\bxjs@geometry@driver
1704   \fi
\ht\strutbox の値を元に戻す。
1705   \ht\strutbox=\bxjs@save@ht@strutbox\relax
\textwidth の値を補正する。
1706   \ifbxjs@whole@zw@lines
1707     \@tempdimb=\textwidth
1708     \if@twocolumn \@tempdima=2\Cwd \else \@tempdima=1\Cwd \fi
1709     \advance\textwidth.005pt\relax
1710     \divide\textwidth\@tempdima \multiply\textwidth\@tempdima
1711     \advance\@tempdimb-\textwidth
1712     \advance\oddsidemargin 0.5\@tempdimb
1713     \advance\evensidemargin 0.5\@tempdimb
```



```
1714 \fi
1715 \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。

```
1716 %<*book>
1717 \@tempdima=\bxjs@textwidth@limit\relax
1718 \ifbxjs@whole@zw@lines
1719   \advance\@tempdima.005pt\relax
1720   \divide\@tempdima\Cwd \multiply\@tempdima\Cwd
1721 \fi
1722 \ifdim\textwidth>\@tempdima
1723   \textwidth=\@tempdima
1724   \addtolength\evensidemargin{\fullwidth-\textwidth}
1725 \fi
1726 %</book>
```

\textheight 関連の調整。

```
1727 \@tempdimb=\textheight
1728 \advance\textheight-\topskip
1729 \advance\textheight.005pt\relax
1730 \divide\textheight\baselineskip \multiply\textheight\baselineskip
1731 \advance\textheight\topskip
1732 \advance\@tempdimb-\textheight
1733 \advance\topmargin0.5\@tempdimb
```

\headheight 関連の調整。

```
1734 \@tempdima=\topskip
1735 \advance\headheight\@tempdima
1736 \advance\topmargin-\@tempdima
```

marginpar 関連の調整。

```
1737 \setlength\marginparsep{\columnsep}
1738 \setlength\marginparpush{\baselineskip}
1739 \setlength\marginparwidth{\paperwidth-\oddsidemargin-1truein%
1740   -\textwidth-10\jsc@mmm-\marginparsep}
1741 \ifbxjs@whole@zw@lines
1742   \divide\marginparwidth\Cwd \multiply\marginparwidth\Cwd
1743 \fi
```

連動する変数。

```
1744 \maxdepth=.5\topskip
1745 \stockwidth=\paperwidth
1746 \stockheight=\paperheight
1747 }
```

\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。

```
1748 \edef\jsGeometryOptions{%
1749   \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ヤムニヤ。

```
1750 \def\bxjs@geometry@name{geometry}
1751 \ifbxjs@old@hook@system
1752   \let\bxjs@apply@bd@pre@geometry@hook\AtBeginDocument
1753 \else
1754   \def\bxjs@apply@bd@pre@geometry@hook{%
1755     \AddToHook{begindocument}[\bxjs@geometry@name]}
1756 \fi
```

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

```
1757 \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。

```
1758 \ifbxjs@papersize
1759   \ifx\bxjs@driver@given\bxjs@driver@@dvipdfmx
1760     \PassOptionsToPackage{dvipdfm}{geometry}
1761   \else\ifx\bxjs@driver@given\bxjs@driver@@dvimode
1762     \PassOptionsToPackage{dvipdfm}{geometry}
1763   \fi\fi
1764   \let\bxPapersizeSpecialDone=t
1765 \else
1766   \PassOptionsToPackage{driver=none}{geometry}
1767 \fi
```

ここで geometry を読み込む。

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

```
1768 \bxjs@apply@bd@pre@geometry@hook{%
1769   \bxjs@bd@pre@geometry@hook
1770   \@nameuse{ImposeOldLuaTeXBehavior}}
1771 \bxjs@preproc@layout
1772 \edef\bxjs@next{%
1773   \noexpand\RequirePackage[\bxjs@layout@paper,\bxjs@layout]{geometry}%
1774 } \bxjs@next
1775 \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。

```
1776 \let\bxjs@geometry@driver\Gm@driver
1777 \bxjs@postproc@layout
```

geometry のドライバ自動判別に対する前処理。

```
1778 \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。

```
1779 \ifpackagelater{geometry}{2010/02/12}{-}{%else
1780 \PackageError\bxjs@clsname
1781 {Your 'geometry' package is too old (< v5.0)}%
1782 {\@ehc}%
1783 \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。

```
1784 \ifjsWithpTeXng
1785 \ifx\Gm@driver\@empty
1786 \def\Gm@driver{pdftex}%
1787 \fi
1788 \fi}
```

`\setpagelayout` ページレイアウト設定のためのユーザ命令。

```
1789 \def\setpagelayout{%
1790 \bxjs@ifplus{\bxjs@setpagelayout@a\tw@}{%else
1791 \ifstar{\bxjs@setpagelayout@a@ne}{\bxjs@setpagelayout@a\z@}}
1792 \def\bxjs@setpagelayout@a#1#2{%
1793 \ifcase#1% modify
1794 \def\bxjs@next{\ifjsc@mag truedimen,\fi #2}%
1795 \or% reset(*)
1796 \def\bxjs@next{reset,\bxjs@layout@paper,#2}%
1797 \or% semireset(+)
1798 \def\bxjs@next{reset,\bxjs@layout@paper,\bxjs@layout@base,#2}%
1799 \fi
1800 \bxjs@preproc@layout
1801 \edef\bxjs@next{%
1802 \noexpand\geometry{\bxjs@next}%
1803 }\bxjs@next
1804 \bxjs@postproc@layout}
```

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

```
1805 \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。

```
1806 \g@addto@macro\bxjs@begin@document@hook{%
1807 \ifdim\textwidth=.5\maxdimen
1808 \ClassError\bxjs@clsname
1809 {Page layout is not properly set}%
1810 {\@ehd}%
1811 \fi}
1812 \def\jsUseMinimalPageLayout{%
1813 \setlength{\textwidth}{6.5in}%
```

```

1814 \setlength{\textheight}{8in}}
    \setpagelayout はとりあえず無効にしておく。
1815 \let\bxjs@geometry@driver\relax
1816 \def\setpagelayout{%
1817   \bxjs@ifplus{\bxjs@pagelayout@a}{%else
1818     \@ifstar{\bxjs@pagelayout@a}{\bxjs@pagelayout@a}}
1819 \def\bxjs@pagelayout@a#1{%
1820   \ClassError\bxjs@clsname
1821   {Command '\string\setpagelayout' is not supported,\MessageBreak
1822     because 'geometry' value is not 'class'}\@eha}
1823 %
1824 \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。

---

```

1825 %<*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。

```

1826 \setlength\topskip{1.38zw}%% from 10\jsc@mpt (2016-08-17)
1827 \if@slide
1828   \setlength\headheight{0\jsc@mpt}
1829 \else
1830   \setlength\headheight{20\jsc@mpt}%% from 2\topskip (2016-08-17); from \topskip (2003-
    06-26)
1831 \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) となっていたのですが、ここでは A4 判のときちょうど 1cm となるように、`\paperheight` の 0.03367 倍 (最小 `\baselineskip`) としました。書籍については、フッタは使わないことにして、ゼロにしました。

```

1832 %<*article|kiyou>
1833 \if@slide
1834   \setlength\footskip{0pt}
1835 \else

```

```

1836 \setlength\footskip{0.03367\paperheight}
1837 \ifdim\footskip<\baselineskip
1838   \setlength\footskip{\baselineskip}
1839 \fi
1840 \fi
1841 %</article|kiyou>
1842 %<jspf>\setlength\footskip{9\jsc@mmm}
1843 %<*book>
1844 \if@report
1845   \setlength\footskip{0.03367\paperheight}
1846   \ifdim\footskip<\baselineskip
1847     \setlength\footskip{\baselineskip}
1848   \fi
1849 \else
1850   \setlength\footskip{0pt}
1851 \fi
1852 %</book>
1853 %<*report>
1854 \setlength\footskip{0.03367\paperheight}
1855 \ifdim\footskip<\baselineskip
1856   \setlength\footskip{\baselineskip}
1857 \fi
1858 %</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)。

```

1859 %<*article>
1860 \if@slide
1861   \setlength\headsep{0\jsc@mpt}
1862   \addtolength\headsep{-\topskip}%% added (2016-10-08)
1863   \addtolength\headsep{10\jsc@mpt}%% added (2016-10-08)
1864 \else
1865   \setlength\headsep{\footskip}
1866   \addtolength\headsep{-\topskip}
1867 \fi
1868 %</article>
1869 %<*book>
1870 \if@report
1871   \setlength\headsep{\footskip}
1872   \addtolength\headsep{-\topskip}
1873 \else
1874   \setlength\headsep{6\jsc@mmm}
1875   \addtolength\headsep{-\topskip}%% added (2016-10-08)
1876   \addtolength\headsep{10\jsc@mpt}%% added (2016-10-08)
1877 \fi

```

```

1878 %</book>
1879 %<*report>
1880 \setlength\headsep{\footskip}
1881 \addtolength\headsep{-\topskip}
1882 %</report>
1883 %<*jspf>
1884 \setlength\headsep{9\jsc@mmm}
1885 \addtolength\headsep{-\topskip}
1886 %</jspf>
1887 %<*kiyou>
1888 \setlength\headheight{0\jsc@mpt}
1889 \setlength\headsep{0\jsc@mpt}
1890 \addtolength\headsep{-\topskip}%% added (2016-10-08)
1891 \addtolength\headsep{10\jsc@mpt}%% added (2016-10-08)
1892 %</kiyou>

```

`\maxdepth` `\maxdepth` は本文最下行の最大の深さで、plain  $\TeX$  や  $\LaTeX$  2.09 では 4pt に固定でした。 $\LaTeX$ 2e 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。

```

1893 \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。

```

1894 \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。

```

1895 %<*article>
1896 \if@slide
1897 \setlength\fullwidth{0.9\paperwidth}
1898 \else
1899 \setlength\fullwidth{0.76\paperwidth}
1900 \fi
1901 \if@twocolumn \@tempdima=2zw \else \@tempdima=1zw \fi
1902 \divide\fullwidth\@tempdima \multiply\fullwidth\@tempdima
1903 \setlength\textwidth{\fullwidth}

```

```

1904 %</article>
1905 %<*book>
1906 \if@report
1907 \setlength\fullwidth{0.76\paperwidth}
1908 \else
1909 \setlength\fullwidth{\paperwidth}
1910 \addtolength\fullwidth{-36\jsc@mmm}
1911 \fi
1912 \if@twocolumn \@tempdima=2zw \else \@tempdima=1zw \fi
1913 \divide\fullwidth\@tempdima \multiply\fullwidth\@tempdima
1914 \setlength\textwidth{\fullwidth}
1915 \if@report \else
1916 \if@twocolumn \else
1917 \ifdim \fullwidth>40zw
1918 \setlength\textwidth{40zw}
1919 \fi
1920 \fi
1921 \fi
1922 %</book>
1923 %<*report>
1924 \setlength\fullwidth{0.76\paperwidth}
1925 \if@twocolumn \@tempdima=2zw \else \@tempdima=1zw \fi
1926 \divide\fullwidth\@tempdima \multiply\fullwidth\@tempdima
1927 \setlength\textwidth{\fullwidth}
1928 %</report>
1929 %<*jspf>
1930 \setlength\fullwidth{50zw}
1931 \addtolength\fullwidth{8\jsc@mmm}
1932 \setlength\textwidth{\fullwidth}
1933 %</jspf>
1934 %<*kiyou>
1935 \setlength\fullwidth{48zw}
1936 \addtolength\fullwidth{\columnsep}
1937 \setlength\textwidth{\fullwidth}
1938 %</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)。

```
1939 %<*article|book|report>
1940 \if@slide
1941 \setlength{\textheight}{0.95\paperheight}
1942 \else
1943 \setlength{\textheight}{0.83\paperheight}
1944 \fi
1945 \addtolength{\textheight}{-10\jsc@empt}%% from -\topskip (2016-10-08); from -
    \headheight (2003-06-26)
1946 \addtolength{\textheight}{-\headsep}
1947 \addtolength{\textheight}{-\footskip}
1948 \addtolength{\textheight}{-\topskip}
1949 \divide\textheight\baselineskip
1950 \multiply\textheight\baselineskip
1951 %</article|book|report>
1952 %<jspf>\setlength{\textheight}{51\baselineskip}
1953 %<kiyou>\setlength{\textheight}{47\baselineskip}
1954 \addtolength{\textheight}{\topskip}
1955 \addtolength{\textheight}{0.1\jsc@empt}
1956 %<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 L<sup>A</sup>T<sub>E</sub>X 2<sub>ε</sub> 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。

```
1957 \def\flushbottom{%
1958   \def\@textbottom{\vskip \z@ \@plus.1\jsc@empt}%
1959   \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。

```
1960 \setlength\marginparsep{\columnsep}
1961 \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。T<sub>E</sub>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 pL<sup>A</sup>T<sub>E</sub>X 2<sub>ε</sub> (`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。

```
1962 \setlength{\oddsidemargin}{\paperwidth}
1963 \addtolength{\oddsidemargin}{-\fullwidth}
1964 \setlength{\oddsidemargin}{.5\oddsidemargin}
1965 \iftombow
```



```

1966 \addtolength{\oddsidemargin}{-1in}
1967 \else
1968 \addtolength{\oddsidemargin}{-\inv@mag in}
1969 \fi
1970 \setlength{\evensidemargin}{\oddsidemargin}
1971 \if@mparswitch
1972 \addtolength{\evensidemargin}{\fullwidth}
1973 \addtolength{\evensidemargin}{-\textwidth}
1974 \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。

```

1975 \setlength\marginparwidth{\paperwidth}
1976 \addtolength\marginparwidth{-\oddsidemargin}
1977 \addtolength\marginparwidth{-\inv@mag in}
1978 \addtolength\marginparwidth{-\textwidth}
1979 \addtolength\marginparwidth{-10\jsc@mmm}
1980 \addtolength\marginparwidth{-\marginparsep}
1981 \@tempdima=1zw
1982 \divide\marginparwidth\@tempdima
1983 \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)。

```

1984 \setlength\topmargin{\paperheight}
1985 \addtolength\topmargin{-\textheight}
1986 \if@slide
1987 \addtolength\topmargin{-\headheight}
1988 \else
1989 \addtolength\topmargin{-10\jsc@empt}%% from -\topskip (2016-10-08); from -
    \headheight (2003-06-26)
1990 \fi
1991 \addtolength\topmargin{-\headsep}
1992 \addtolength\topmargin{-\footskip}
1993 \setlength\topmargin{0.5\topmargin}
1994 %<kiyou>\setlength\topmargin{81truebp}
1995 \iftombow
1996 \addtolength\topmargin{-1in}
1997 \else
1998 \addtolength\topmargin{-\inv@mag in}
1999 \fi
2000 %</jscsses>

```

## ■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用ことにする。

---

```
2001 \footnotesep=11\jsc@mp \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。

```
2002 \setlength{\skip\footins}{16\jsc@mp \@plus 5\jsc@mp \@minus 2\jsc@mp}
```

**■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。

```
2003 \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。

```
2004 \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。

```
2005 \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。

```
2006 \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。  
 2007 `\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。  
 2008 `\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。  
 2009 `\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。  
 2010 `\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。  
 2011 `\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。  
 2012 `\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。  
 2013 `\setlength\floatsep {12\jsc@empt \@plus 2\jsc@empt \@minus 2\jsc@empt}`  
 2014 `\setlength\textfloatsep{20\jsc@empt \@plus 2\jsc@empt \@minus 4\jsc@empt}`  
 2015 `\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` 2016 `\setlength\dblfloatsep {12\jsc@empt \@plus 2\jsc@empt \@minus 2\jsc@empt}`  
 2017 `\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` 2018 `\setlength\@fptop{0\jsc@empt \@plus 1fil}`  
 2019 `\setlength\@fpsep{8\jsc@empt \@plus 2fil}`  
 2020 `\setlength\@fpbot{0\jsc@empt \@plus 1fil}`

`\@dblftop` 段抜きフロートについての値です。  
`\@dblfpsep` 2021 `\setlength\@dblftop{0\jsc@empt \@plus 1fil}`  
 2022 `\setlength\@dblfpsep{8\jsc@empt \@plus 2fil}`  
`\@dblfpbot` 2023 `\setlength\@dblfpbot{0\jsc@empt \@plus 1fil}`

## 6 改ページ（日本語 TeX 開発コミュニティ版のみ）

`\pltx@cleartorightpage` [2017-02-24] コミュニティ版 pLaTeX の標準クラス 2017/02/15 に合わせて、同じ命令を追  
`\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。

```
2024 %\def\pltx@cleartorightpage{\clearpage\if@twoside
2025 % \ifodd\c@page
2026 % \iftdir
2027 % \hbox{}\thispagestyle{empty}\newpage
2028 % \if@twocolumn\hbox{}\newpage\fi
2029 % \fi
2030 % \else
2031 % \ifydir
2032 % \hbox{}\thispagestyle{empty}\newpage
2033 % \if@twocolumn\hbox{}\newpage\fi
2034 % \fi
2035 % \fi\fi}
2036 %\def\pltx@cleartoleftpage{\clearpage\if@twoside
2037 % \ifodd\c@page
2038 % \ifydir
2039 % \hbox{}\thispagestyle{empty}\newpage
2040 % \if@twocolumn\hbox{}\newpage\fi
2041 % \fi
2042 % \else
2043 % \iftdir
2044 % \hbox{}\thispagestyle{empty}\newpage
2045 % \if@twocolumn\hbox{}\newpage\fi
2046 % \fi
2047 % \fi\fi}
2048 \def\pltx@cleartooddpage{\clearpage\if@twoside
2049 \ifodd\c@page\else
2050 \hbox{}\thispagestyle{empty}\newpage
2051 \if@twocolumn\hbox{}\newpage\fi
2052 \fi\fi}
2053 \def\pltx@cleartoevenpage{\clearpage\if@twoside
2054 \ifodd\c@page
2055 \hbox{}\thispagestyle{empty}\newpage
2056 \if@twocolumn\hbox{}\newpage\fi
2057 \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。

```
2058 \let\pltx@cleartorightpage\pltx@cleartooddpage
2059 \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。
2060 \ifdim\vsiz=\z@
2061 \begingroup
2062 \toks@\expandafter{\clearpage}
2063 \xdef\clearpage{\noexpand\ifbxjs@after@preamble\the\toks@\noexpand\fi}
2064 \endgroup
2065 \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。

```
2066 %<*book|report>
2067 \if@openleft
2068 \let\cleardoublepage\pltx@cleartoleftpage
2069 \else\if@openright
2070 \let\cleardoublepage\pltx@cleartorightpage
2071 \fi\fi
2072 %</book|report>
```

## 7 ページスタイル

ページスタイルとして、 $\text{\LaTeX}$  2 $\epsilon$  (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 $\epsilon$  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。

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

柱を設定する命令は、右の柱が左の柱の下位にある場合は十分まともに動作します。たと

えば左マークを `\chapter`, 右マークを `\section` で変更する場合がこれにあたります。しかし, 同一ページに複数の `\markboth` があると, おかしな結果になることがあります。

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

`\ps@empty` `empty` ページスタイルの定義です。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。

```
2073 % \def\ps@empty{%
2074 %   \let\@mkboth\@gobbletwo
2075 %   \let\@oddhead\@empty
2076 %   \let\@oddfoot\@empty
2077 %   \let\@evenhead\@empty
2078 %   \let\@evenfoot\@empty}
```

`\ps@plainhead` `plainhead` はシンプルなヘッダだけのページスタイルです。

`\ps@plainfoot` `plainfoot` はシンプルなフッタだけのページスタイルです。

`\ps@plain` `plain` は `book` では `plainhead`, それ以外では `plainfoot` になります。

```
2079 \def\ps@plainfoot{%
2080   \let\@mkboth\@gobbletwo
2081   \let\@oddhead\@empty
2082   \def\@oddfoot{\normalfont\hfil\thepage\hfil}%
2083   \let\@evenhead\@empty
2084   \let\@evenfoot\@oddfoot}
2085 \def\ps@plainhead{%
2086   \let\@mkboth\@gobbletwo
2087   \let\@oddfoot\@empty
2088   \let\@evenfoot\@empty
2089   \def\@evenhead{%
2090     \if@mparswitch \hss \fi
2091     \hbox to \fullwidth{\textbf{\thepage}\hfil}%
2092     \if@mparswitch\else \hss \fi}%
2093   \def\@oddhead{%
2094     \hbox to \fullwidth{\hfil\textbf{\thepage}}\hss}}
2095 %<book>\let\ps@plain\ps@plainhead
2096 %<!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。

```
2097 %<*article|slide>
2098 \if@twoside
2099   \def\ps@headings{%
2100     \let\@oddfoot\@empty
2101     \let\@evenfoot\@empty
2102     \def\@evenhead{\if@mparswitch \hss \fi
2103       \underline{\hbox to \fullwidth{\textbf{\thepage}\hfil\leftmark}}}%
2104     \if@mparswitch\else \hss \fi}%

```

```

2105 \def\@oddhead{%
2106 \underline{%
2107 \hbox to \fullwidth{\rightmark}\hfil\textbf{\thepage}}\hss}%
2108 \let\@mkboth\markboth
2109 \def\sectionmark##1{\markboth{%
2110 \ifnum \c@secnumdepth >\z@ \bxjs@label@sect{section}\hskip1\jsw\fi
2111 ##1}}}%
2112 \def\subsectionmark##1{\markright{%
2113 \ifnum \c@secnumdepth >\@ne \bxjs@label@sect{subsection}\hskip1\jsw\fi
2114 ##1}}}%
2115 }
2116 \else % if not twoside
2117 \def\ps@headings{%
2118 \let\@oddfoot\@empty
2119 \def\@oddhead{%
2120 \underline{%
2121 \hbox to \fullwidth{\rightmark}\hfil\textbf{\thepage}}\hss}%
2122 \let\@mkboth\markboth
2123 \def\sectionmark##1{\markright{%
2124 \ifnum \c@secnumdepth >\z@ \bxjs@label@sect{section}\hskip1\jsw\fi
2125 ##1}}}%
2126 \fi
2127 %</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）。

```

2128 %<*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。

```

2129 \def\bxjs@maybe@autoxspacing{%
2130 \ifx\autoxspacing\undefined\else \autoxspacing \fi}

```

---

```

2131 \newif\if@omit@number
2132 \def\ps@headings{%
2133 \let\@oddfoot\@empty
2134 \let\@evenfoot\@empty
2135 \def\@evenhead{%
2136 \if@mparswitch \hss \fi
2137 \underline{\hbox to \fullwidth{\bxjs@maybe@autoxspacing
2138 \textbf{\thepage}\hfil\leftmark}}}%
2139 \if@mparswitch\else \hss \fi}%
2140 \def\@oddhead{\underline{\hbox to \fullwidth{\bxjs@maybe@autoxspacing
2141 {\if@twoside\rightmark\else\leftmark\fi}\hfil\textbf{\thepage}}}\hss}%
2142 \let\@mkboth\markboth
2143 \def\chaptermark##1{\markboth{%

```

```

2144 \ifnum \c@secnumdepth >\m@ne
2145 \if@mainmatter
2146 \if@omit@number\else
2147 \@chapapp\thechapter\@chappos\hskip1\jsZw
2148 \fi
2149 \fi
2150 \fi
2151 ##1}{}}%
2152 \def\sectionmark##1{\markright{%
2153 \ifnum \c@secnumdepth >\z@ \bxjs@label@sect{section}\hskip1\jsZw\fi
2154 ##1}}}%
2155 %</book|report>

```

最後は学会誌の場合です。

```

2156 %<*jspf>
2157 \def\ps@headings{%
2158 \def\@oddfont{\normalfont\hfil\thepage\hfil}
2159 \def\@evenfont{\normalfont\hfil\thepage\hfil}
2160 \def\@oddhead{\normalfont\hfil \@title \hfil}
2161 \def\@evenhead{\normalfont\hfil プラズマ・核融合学会誌\hfil}}
2162 %</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。

```

2163 \def\ps@myheadings{%
2164 \let\@oddfont\@empty\let\@evenfont\@empty
2165 \def\@evenhead{%
2166 \if@mparswitch \hss \fi%
2167 \hbox to \fullwidth{\thepage\hfil\leftmark}%
2168 \if@mparswitch\else \hss \fi}%
2169 \def\@oddhead{%
2170 \hbox to \fullwidth{\rightmark\hfil\thepage}\hss}%
2171 \let\@mkboth\@gobbletwo
2172 %<book|report> \let\chaptermark\@gobble
2173 \let\sectionmark\@gobble
2174 %<!book&!report> \let\subsectionmark\@gobble
2175 }

```

## 8 文書のマークアップ

### 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。

```

\date 2176 % \newcommand*{\title}[1]{\gdef\@title{#1}}
2177 % \newcommand*{\author}[1]{\gdef\@author{#1}}

```



```
2178 % \newcommand*\date}[1]{\gdef\@date{#1}}
2179 % \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。

```
2180 \newcommand*\jsSubtitle}[1]{\gdef\bxjs@subtitle{#1}}
2181 %\let\bxjs@subtitle\undefined

\title にフックを入れる。
2182 \renewcommand*\title}[1]{\bxjs@decl@subtitle\gdef\@title{#1}}
2183 \g@addto@macro\bxjs@begin@document@hook{\bxjs@decl@subtitle}
2184 \def\bxjs@decl@subtitle{%
2185   \global\let\bxjs@decl@subtitle\relax
2186   \ifx\subtitle\undefined
2187     \global\let\subtitle\jsSubtitle
2188   \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。

```
2189 \def\bxjs@annihilate@subtitle{%
2190   \ifx\subtitle\jsSubtitle \global\let\subtitle\relax \fi
2191   \global\let\jsSubtitle\relax}
```

---

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

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

```
2199 \def\plainifnotempty{%
2200   \ifx \@oddhead \@empty
```

```

2201 \ifx \@oddfoot \@empty
2202 \else
2203 \thispagestyle{plainfoot}%
2204 \fi
2205 \else
2206 \thispagestyle{plainhead}%
2207 \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。

```

2208 %<*article|book|report|slide>
2209 \if@titlepage
2210 \newcommand{\maketitle}{%
2211 \begin{titlepage}%
2212 \let\footnotesize\small
2213 \let\footnoterule\relax
2214 \let\footnote\thanks
2215 \null\vfil
2216 \if@slide
2217 {\footnotesize \@date}%
2218 \begin{center}
2219 \mbox{} \\\[1\jscZw]
2220 \large
2221 {\maybeblue\hrule height0\jsc@mpt depth2\jsc@mpt\relax}\par
2222 \jsc@smallskip
2223 \@title
2224 \ifx\bxjs@subtitle\@undefined\else
2225 \par\vskip\z@
2226 {\small \bxjs@subtitle\par}
2227 \fi
2228 \jsc@smallskip
2229 {\maybeblue\hrule height0\jsc@mpt depth2\jsc@mpt\relax}\par
2230 \vfill
2231 {\small \@author}%
2232 \end{center}
2233 \else
2234 \vskip 60\jsc@mpt
2235 \begin{center}%
2236 {\LARGE \@title \par}%
2237 \ifx\bxjs@subtitle\@undefined\else
2238 \vskip5\jsc@mpt
2239 {\normalsize \bxjs@subtitle\par}
2240 \fi
2241 \vskip 3em%
2242 {\large

```

```

2243         \lineskip .75em
2244         \begin{tabular}[t]{c}%
2245             \@author
2246         \end{tabular}\par}%
2247         \vskip 1.5em
2248         {\large \@date \par}%
2249     \end{center}%
2250     \fi
2251     \par
2252     \@thanks\vfil\null
2253 \end{titlepage}%
2254 \setcounter{footnote}{0}%
2255 \global\let\thanks\relax
2256 \global\let\maketitle\relax
2257 \global\let\@thanks\@empty
2258 \global\let\@author\@empty
2259 \global\let\@date\@empty
2260 \global\let\@title\@empty
2261 \global\let\title\relax
2262 \global\let\author\relax
2263 \global\let\date\relax
2264 \global\let\and\relax
2265 \bxjs@annihilate@subtitle
2266 }%
2267 \else
2268 \newcommand{\maketitle}{\par
2269 \begingroup
2270     \renewcommand\thefootnote{\@fnsymbol\c@footnote}%
2271     \def\@makefnmark{\rlap{\@textsuperscript{\normalfont\@thefnmark}}}%
2272     \long\def\@makefntext##1{\advance\leftskip 3\jsZw
2273         \parindent 1\jsZw\noindent
2274         \llap{\@textsuperscript{\normalfont\@thefnmark}\hskip0.3\jsZw}##1}%
2275     \if@twocolumn
2276         \ifnum \col@number=\@ne
2277             \@maketitle
2278         \else
2279             \twocolumn[\@maketitle]%
2280         \fi
2281     \else
2282         \newpage
2283         \global\@topnum\z@ % Prevents figures from going at top of page.
2284         \@maketitle
2285     \fi
2286     \plainifnotempty
2287     \@thanks
2288 \endgroup
2289 \setcounter{footnote}{0}%
2290 \global\let\thanks\relax
2291 \global\let\maketitle\relax

```

```

2292 \global\let\@thanks\@empty
2293 \global\let\@author\@empty
2294 \global\let\@date\@empty
2295 \global\let\@title\@empty
2296 \global\let\title\relax
2297 \global\let\author\relax
2298 \global\let\date\relax
2299 \global\let\and\relax
2300 \bxjs@annihilate@subtitle
2301 }

```

\@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。

```

2302 \def\@maketitle{%
2303 \newpage\null
2304 \vskip 2em
2305 \begin{center}%
2306 \let\footnote\thanks
2307 {\LARGE \@title \par}%
2308 \ifx\bxjs@subtitle\@undefined\else
2309 \vskip3\jsc@mpt
2310 {\normalsize \bxjs@subtitle\par}
2311 \fi
2312 \vskip 1.5em
2313 {\large
2314 \lineskip .5em
2315 \begin{tabular}[t]{c}%
2316 \@author
2317 \end{tabular}\par}%
2318 \vskip 1em
2319 {\large \@date}%
2320 \end{center}%
2321 \par\vskip 1.5em
2322 %<article|slide> \ifvoid\@abstractbox\else\centerline{\box\@abstractbox}\vskip1.5em\fi
2323 }
2324 \fi
2325 %</article|book|report|slide>
2326 %<*jspf>
2327 \newcommand{\maketitle}{\par
2328 \begingroup
2329 \renewcommand\thefootnote{\@fnsymbol\c@footnote}%
2330 \def\@makefnmark{\rlap{\@textsuperscript{\normalfont\thefnmark}}}%
2331 \long\def\@makefntext##1{\advance\leftskip 3\jsZw
2332 \parindent 1\jsZw\noindent
2333 \llap{\@textsuperscript{\normalfont\thefnmark}\hskip0.3\jsZw}##1}%
2334 \twocolumn[\@maketitle]%
2335 \plainifnotempty
2336 \@thanks
2337 \endgroup
2338 \setcounter{footnote}{0}%

```

```

2339 \global\let\thanks\relax
2340 \global\let\maketitle\relax
2341 \global\let\@thanks\@empty
2342 \global\let\@author\@empty
2343 \global\let\@date\@empty
2344 % \global\let\@title\@empty % \@title は柱に使う
2345 \global\let\title\relax
2346 \global\let\author\relax
2347 \global\let\date\relax
2348 \global\let\and\relax
2349 \ifx\authors@mail\@undefined\else{%
2350     \def\@makefntext{\advance\leftskip 3\jsZw \parindent -3\jsZw}%
2351     \footnotetext[0]{\itshape\authors@mail}%
2352 } \fi
2353 \global\let\authors@mail\@undefined}
2354 \def\@maketitle{%
2355     \newpage\null
2356     \vskip 6em % used to be 2em
2357     \begin{center}
2358         \let\footnote\thanks
2359         \ifx\@title\@undefined\else{\LARGE\headfont\@title\par} \fi
2360         \lineskip .5em
2361         \ifx\@author\@undefined\else
2362             \vskip 1em
2363             \begin{tabular}[t]{c}%
2364                 \@author
2365             \end{tabular}\par
2366         \fi
2367         \ifx\@etitle\@undefined\else
2368             \vskip 1em
2369             {\large \@etitle \par}%
2370         \fi
2371         \ifx\@eauthor\@undefined\else
2372             \vskip 1em
2373             \begin{tabular}[t]{c}%
2374                 \@eauthor
2375             \end{tabular}\par
2376         \fi
2377         \vskip 1em
2378         \@date
2379     \end{center}
2380     \vskip 1.5em
2381     \centerline{\box\@abstractbox}
2382     \ifx\@keywords\@undefined\else
2383         \vskip 1.5em
2384         \centerline{\parbox{157\jsc@mmm}{\textsf{Keywords:}}\ \small\@keywords}}
2385     \fi
2386     \vskip 1.5em}
2387 %</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。

```
2388 \def\bxjs@label@sect#1{%
2389   \ifundefined{label#1}{\@nameuse{the#1}}{\@nameuse{label#1}}
2390 \def\@secntformat#1{\bxjs@label@sect{#1}\quad}
```

`\@secapp` 節番号の接頭辞。

`\@secpos` 節番号の接尾辞。

```
2391 \ifnum\bxjs@label@section=\bxjs@label@section@@compat\else
2392 \def\@secapp{\presectionname}
2393 \def\@secpos{\postsectionname}
2394 \fi
```

`\labelsection` 節番号の出力書式。

```
2395 \ifnum\bxjs@label@section=\bxjs@label@section@@modern
2396 \def\labelsection{\@secapp\thesection\@secpos}
2397 \fi
```

---

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

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

それぞれの引数の意味は次の通りです。

**名** ユーザレベルコマンドの名前です (例: section)。

**レベル** 見出しの深さを示す数値です (chapter=1, section=2, ...)。この数値が `secnumdepth` 以下のとき見出し番号を出力します。

**字下げ** 見出しの字下げ量です。

**前アキ** この値の絶対値が見出し上側の空きです。負の場合は、見出し直後の段落をインデントしません。

**後アキ** 正の場合は、見出しの下の空きです。負の場合は、絶対値が見出しの右の空きです (見出しと同じ行から本文を始めます)。

**スタイル** 見出しの文字スタイルの設定です。

\* この \* 印がないと、見出し番号を付け、見出し番号のカウンタに 1 を加算します。

**別見出し** 目次や柱に出力する見出しです。

見出し 見出しです。

見出しの命令は通常 `\@startsection` とその最初の 6 個の引数として定義されます。

次は `\@startsection` の定義です。情報処理学会論文誌スタイルファイル (`ipsjcommon.sty`) を参考にさせていただきましたが、完全に行送り `\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。

```
2398 \def\@startsection#1#2#3#4#5#6{%
2399   \ifnoskipsec \leavevmode \fi
2400   \par
2401 % 見出し上の空きを \@tempskipa にセットする
2402   \@tempskipa #4\relax
2403 % \@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
2404   \if@english \@afterindentfalse \else \@afterindenttrue \fi
2405 % 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
2406   \ifdim \@tempskipa <\z@
2407     \@tempskipa -\@tempskipa \@afterindentfalse
2408   \fi
2409   \if@nobreak
2410 %   \everypar{\everyparhook}% これは間違い
2411     \everypar{}%
2412   \else
2413     \addpenalty\@secpenalty
2414 % 次の行は削除
2415 %   \addvspace\@tempskipa
2416 % 次の \noindent まで追加
2417     \ifdim \@tempskipa >\z@
2418       \if@slide\else
2419         \null
2420         \vspace*{-\baselineskip}%
2421       \fi
2422       \vskip\@tempskipa
2423     \fi
2424   \fi
2425   \noindent
2426 % 追加終わり
2427   \@ifstar
2428     {\@ssect{#3}{#4}{#5}{#6}}%
2429     {\@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。

```
2430 \ifnum\bxjs@everyparhook=\bxjs@everyparhook@@compat
2431   \let\bxjs@if@ceph\@firstofone
```

```
2432 \else \let\bxjs@if@ceph@gobble
2433 \fi
```

---

```
2434 \def\@sect#1#2#3#4#5#6[#7]#8{%
2435   \ifnum #2>\c@secnumdepth
2436     \let\@svsec\@empty
2437   \else
2438     \refstepcounter{#1}%
2439     \protected@edef\@svsec{\@secntformat{#1}\relax}%
2440   \fi
2441 % 見出し後の空きを \@tempskipa にセット
2442   \@tempskipa #5\relax
2443 % 条件判断の順序を入れ替えました
2444   \ifdim \@tempskipa<\z@
2445     \def\@svsechd{%
2446       #6{\hskip #3\relax
2447         \@svsec #8}%
2448       \csname #1mark\endcsname{#7}%
2449       \addcontentsline{toc}{#1}{%
2450         \ifnum #2>\c@secnumdepth \else
2451           \protect\numberline{\bxjs@label@sect{#1}}%
2452         \fi
2453         #7}}% 目次にフルネームを載せるなら #8
2454   \else
2455     \begingroup
2456     \interlinepenalty \@M % 下から移動
2457     #6{%
2458       \@hangfrom{\hskip #3\relax\@svsec}%
2459     % \interlinepenalty \@M % 上に移動
2460     #8\@par}%
2461   \endgroup
2462   \csname #1mark\endcsname{#7}%
2463   \addcontentsline{toc}{#1}{%
2464     \ifnum #2>\c@secnumdepth \else
2465       \protect\numberline{\bxjs@label@sect{#1}}%
2466     \fi
2467     #7}% 目次にフルネームを載せるならここは #8
2468   \fi
2469   \@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。

```
2470 \def\@xsect#1{%
2471 % 見出しの後ろの空きを \@tempskipa にセット
2472   \@tempskipa #1\relax
2473 % 条件判断の順序を変えました
```



```

2474 \ifdim \@tempskipa<\z@
2475   \nobreakfalse
2476   \global\@noskipsectrue
2477   \everypar{%
2478     \if@noskipsec
2479       \global\@noskipsecfalse
2480       {\setbox\z@\lastbox}%
2481       \clubpenalty\@M
2482       \begingroup \@svsechd \endgroup
2483       \unskip
2484       \@tempskipa #1\relax
2485       \hskip -\@tempskipa
2486     \else
2487       \clubpenalty \@clubpenalty
2488       \everypar\expandafter{\bxjs@if@ceph\everyparhook}%

```

**TODO:** ↑ ナニコレ？

```

2489   \fi\bxjs@if@ceph\everyparhook}%
2490 \else
2491   \par \nobreak
2492   \vskip \@tempskipa
2493   \@afterheading
2494 \fi
2495 \if@slide
2496   {\vskip\if@twocolumn-5\jsc@mpt\else-6\jsc@mpt\fi
2497   \maybeblue\hrule height0\jsc@mpt depth1\jsc@mpt
2498   \vskip\if@twocolumn 4\jsc@mpt\else 7\jsc@mpt\fi\relax}%
2499 \fi
2500 \par % 2000-12-18
2501 \ignorespaces}
2502 \def\@ssect#1#2#3#4#5{%
2503   \@tempskipa #3\relax
2504   \ifdim \@tempskipa<\z@
2505     \def\@svsechd{#4{\hskip #1\relax #5}}%
2506   \else
2507     \begingroup
2508       #4{%
2509         \@hangfrom{\hskip #1}%
2510         \interlinepenalty \@M #5\@@par}%
2511     \endgroup
2512   \fi
2513   \@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 2514 \newcommand*\chaptermark[1]{ }
\subsubsectionmark 2515 % \newcommand*{\sectionmark}[1]{ }
\paragraphmark
\subparagraphmark

```

```

2516 % \newcommand*{\subsectionmark}[1]{}
2517 % \newcommand*{\subsubsectionmark}[1]{}
2518 % \newcommand*{\paragraphmark}[1]{}
2519 % \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。

```

2520 %<!book&!report>\setcounter{secnumdepth}{3}
2521 %<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 2522 \newcounter{part}
                2523 %<book|report>\newcounter{chapter}
\c@subsubsection 2524 %<book|report>\newcounter{section}[chapter]
                \c@paragraph 2525 %<!book&!report>\newcounter{section}
\c@subparagraph 2526 \newcounter{subsection}[section]
                2527 \newcounter{subsubsection}[subsection]
                2528 \newcounter{paragraph}[subsubsection]
                2529 \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。

```

2530 \renewcommand{\thepart}{\@Roman\c@part}
2531 %<*!book&!report>
2532 \ifnum\bxjs@label@section=\bxjs@label@section@compat
2533 \renewcommand{\thesection}{\presectionname\@arabic\c@section\postsectionname}
2534 \renewcommand{\thesubsection}{\@arabic\c@section.\@arabic\c@subsection}
2535 \else
2536 \renewcommand{\thesection}{\@arabic\c@section}
2537 \renewcommand{\thesubsection}{\thesection.\@arabic\c@subsection}
2538 \fi
2539 %</!book&!report>
2540 %<*book|report>
2541 \renewcommand{\thechapter}{\@arabic\c@chapter}
2542 \renewcommand{\thesection}{\thechapter.\@arabic\c@section}
2543 \renewcommand{\thesubsection}{\thesection.\@arabic\c@subsection}

```

```

2544 %</book|report>
2545 \renewcommand{\thesubsubsection}{%
2546   \thesubsection.\@arabic\c@subsubsection}
2547 \renewcommand{\theparagraph}{%
2548   \thesubsubsection.\@arabic\c@paragraph}
2549 \renewcommand{\thesubparagraph}{%
2550   \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。

```

2551 %<book|report>\newcommand{\@chapapp}{\prechaptername}
2552 %<book|report>\newcommand{\@chappos}{\postchaptername}

```

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

`\frontmatter` ページ番号をローマ数字にし、章番号を付けないようにします。

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

```

2553 %<*book|report>
2554 \newcommand\frontmatter{%
2555   \pltx@cleartooddpage
2556   \@mainmatterfalse
2557   \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。

```

2558 \newcommand\mainmatter{%
2559   \pltx@cleartooddpage
2560   \@mainmattertrue
2561   \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。

```

2562 \newcommand\backmatter{%
2563   \if@openleft
2564     \cleardoublepage
2565   \else\if@openright
2566     \cleardoublepage
2567   \else
2568     \clearpage
2569   \fi\fi
2570   \@mainmatterfalse}
2571 %</book|report>

```

## ■部

`\part` 新しい部を始めます。

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

```
\secdef{星なし}{星あり}
```

**星なし** \* のない形の定義です。

**星あり** \* のある形の定義です。

`\secdef` は次のようにして使います。

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

まず `book` と `report` のクラス以外です。

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

`book` および `report` クラスの場合は、少し複雑です。

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

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

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

ます。

```
2599 %<*!book&!report>
2600 \def\@part[#1]#2{%
2601   \ifnum \c@secnumdepth >\m@ne
2602     \refstepcounter{part}%
2603     \addcontentsline{toc}{part}{%
2604       \prepartname\thepart\postpartname\hspace{1\jsZw}#1}%
2605   \else
2606     \addcontentsline{toc}{part}{#1}%
2607   \fi
2608   \markboth{}{}%
2609   {\parindent\z@
2610     \raggedright
2611     \interlinepenalty \@M
2612     \normalfont
2613     \ifnum \c@secnumdepth >\m@ne
2614       \Large\headfont\prepartname\thepart\postpartname
2615       \par\nobreak
2616     \fi
2617     \huge \headfont #2%
2618     \markboth{}{}\par}%
2619   \nobreak
2620   \vskip 3ex
2621   \@afterheading}
2622 %</!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。

```
2623 %<*book|report>
2624 \def\@part[#1]#2{%
2625   \ifnum \c@secnumdepth >-2\relax
2626     \refstepcounter{part}%
2627     \addcontentsline{toc}{part}{%
2628       \prepartname\thepart\postpartname\hspace{1\jsZw}#1}%
2629   \else
2630     \addcontentsline{toc}{part}{#1}%
2631   \fi
2632   \markboth{}{}%
2633   {\centering
2634     \interlinepenalty \@M
2635     \normalfont
2636     \ifnum \c@secnumdepth >-2\relax
2637       \huge\headfont \prepartname\thepart\postpartname
2638       \par\vskip20\jsc@empt
2639     \fi
2640     \Huge \headfont #2\par}%
2641   \@endpart}
2642 %</book|report>
```

\@spart 番号を付けない部です。

```

2643 %<!*book&!report>
2644 \def\@spart#1{%
2645     \parindent \z@ \raggedright
2646     \interlinepenalty \@M
2647     \normalfont
2648     \huge \headfont #1\par}%
2649 \nobreak
2650 \vskip 3ex
2651 \@afterheading}
2652 %</!book&!report>
2653 %<*book|report>
2654 \def\@spart#1{%
2655     \centering
2656     \interlinepenalty \@M
2657     \normalfont
2658     \Huge \headfont #1\par}%
2659 \@endpart}
2660 %</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。

```

2661 %<*book|report>
2662 \def\@endpart{\vfil\newpage
2663     \if@twoside
2664     \if@openleft %% added (2017/02/24)
2665     \null\thispagestyle{empty}\newpage
2666     \else\if@openright %% added (2016/12/13)
2667     \null\thispagestyle{empty}\newpage
2668     \fi\fi %% added (2016/12/13, 2017/02/24)
2669     \fi
2670     \if@restonecol
2671     \twocolumn
2672     \fi}
2673 %</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。

```

2674 %<*book|report>
2675 \newcommand{\chapter}{%
2676     \if@openleft\cleardoublepage\else
2677     \if@openright\cleardoublepage\else\clearpage\fi\fi
2678     \plainifnotempty % 元: \thispagestyle{plain}

```

```

2679 \global\@topnum\z@
2680 \if@english \@afterindentfalse \else \@afterindenttrue \fi
2681 \secdef
2682   {\@omit@numberfalse\@chapter}%
2683   {\@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。

```

2684 \def\@chapter[#1]#2{%
2685   \ifnum \c@secnumdepth >\m@ne
2686     \if@mainmatter
2687       \refstepcounter{chapter}%
2688       \typeout{\@chapapp\thechapter\@chappos}%
2689       \addcontentsline{toc}{chapter}%
2690         {\protect\numberline
2691 %       %{\if@english\thechapter\else\@chapapp\thechapter\@chappos\fi}%
2692         {\@chapapp\thechapter\@chappos}%
2693         #1}%
2694     \else\addcontentsline{toc}{chapter}{#1}\fi
2695   \else
2696     \addcontentsline{toc}{chapter}{#1}%
2697   \fi
2698   \chaptermark{#1}%
2699   \addtocontents{lof}{\protect\addvspace{10\jsc@empt}}%
2700   \addtocontents{lot}{\protect\addvspace{10\jsc@empt}}%
2701   \if@twocolumn
2702     \@topnewpage[\@makechapterhead{#2}]%
2703   \else
2704     \@makechapterhead{#2}%
2705     \@afterheading
2706   \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。

```

2707 \def\@makechapterhead#1{%
2708   \vspace*{2\Cvs}% 欧文は 50pt
2709   {\parindent \z@ \raggedright \normalfont
2710     \ifnum \c@secnumdepth >\m@ne
2711       \if@mainmatter
2712         \huge\headfont \@chapapp\thechapter\@chappos
2713         \par\nobreak
2714         \vskip \Cvs % 欧文は 20pt
2715       \fi
2716     \fi
2717     \interlinepenalty\@M
2718     \Huge \headfont #1\par\nobreak
2719     \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。

```

2720 \def\@schapter#1{%

```

```

2721 \chaptermark{#1}%
2722 \if@twocolumn
2723   \@topnewpage[\@makeschapterhead{#1}]%
2724 \else
2725   \@makeschapterhead{#1}\@afterheading
2726 \fi}

```

`\@makeschapterhead` 番号なしの章見出しです。

```

2727 \def\@makeschapterhead#1{%
2728   \vspace*{2\Cvs}% 欧文は 50pt
2729   {\parindent \z@ \raggedright
2730     \normalfont
2731     \interlinepenalty\@M
2732     \Huge \headfont #1\par\nobreak
2733     \vskip 3\Cvs}} % 欧文は 40pt
2734 %</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。

```

2735 \if@twocolumn
2736   \newcommand{\section}{%
2737     %<jspf>\ifx\maketitle\relax\else\maketitle\fi
2738     \@startsection{section}{1}{\z@}%
2739     %<!kiyou> {0.6\Cvs}{0.4\Cvs}%
2740     %<kiyou>  {\Cvs}{0.5\Cvs}%
2741     %  {\normalfont\large\headfont\@secapp}}
2742     {\normalfont\large\headfont\raggedright}}
2743 \else
2744   \newcommand{\section}{%
2745     \if@slide\clearpage\fi
2746     \@startsection{section}{1}{\z@}%
2747     {\Cvs \@plus.5\Cdp \@minus.2\Cdp}% 前アキ
2748     {.5\Cvs \@plus.3\Cdp}% 後アキ
2749     %  {\normalfont\Large\headfont\@secapp}}
2750     {\normalfont\Large\headfont\raggedright}}
2751 \fi

```

`\subsection` 同上です。

```

2752 \if@twocolumn
2753   \newcommand{\subsection}{\@startsection{subsection}{2}{\z@}%
2754     {\z@}{\if@slide .4\Cvs \else \z@ \fi}%
2755     {\normalfont\normalsize\headfont}}
2756 \else
2757   \newcommand{\subsection}{\@startsection{subsection}{2}{\z@}%
2758     {\Cvs \@plus.5\Cdp \@minus.2\Cdp}% 前アキ

```



```

2759     {.5\Cvs \@plus.3\Cdp}% 後アキ
2760     {\normalfont\large\headfont}}
2761 \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)。

```

2762 \if@twocolumn
2763   \newcommand{\subsubsection}{\@startsection{subsubsection}{3}{\z@}%
2764     {\z@}{\if@slide .4\Cvs \else \z@ \fi}%
2765     {\normalfont\normalsize\headfont}}
2766 \else
2767   \newcommand{\subsubsection}{\@startsection{subsubsection}{3}{\z@}%
2768     {\Cvs \@plus.5\Cdp \@minus.2\Cdp}%
2769     {\if@slide .5\Cvs \@plus.3\Cdp \else \z@ \fi}%
2770     {\normalfont\normalsize\headfont}}
2771 \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。

```
2772 \let\jsJaChar\@empty
```

```

2773 \newcommand\jsParagraphMark{\relax\jsJaChar{■}}
2774 \let\bxjs@org@paragraph@mark\jsParagraphMark
2775 \ifx\bxjs@paragraph@mark\@empty
2776   \let\jsParagraphMark\@empty
2777 \else\ifx\bxjs@paragraph@mark\@undefined\else
2778   \long\edef\jsParagraphMark{\noexpand\jsJaChar{\bxjs@paragraph@mark}}
2779 \fi\fi
2780 \if@twocolumn
2781   \newcommand{\paragraph}{\@startsection{paragraph}{4}{\z@}%
2782     {\z@}{\if@slide .4\Cvs \else -1\jsZw\fi}% 改行せず 1\jsZw のアキ
2783 %<jspf>     {\normalfont\normalsize\headfont}}
2784 %<!\jspf>   {\normalfont\normalsize\headfont\jsParagraphMark}}
2785 \else
2786   \newcommand{\paragraph}{\@startsection{paragraph}{4}{\z@}%

```

```

2787     {0.5\Cvs \@plus.5\Cdp \@minus.2\Cdp}%
2788     {\if@slide .5\Cvs \@plus.3\Cdp \else -1\jsZw\fi}% 改行せず 1\jsZw のアキ
2789 %<jspf>     {\normalfont\normalsize\headfont}}
2790 %<!jspf>     {\normalfont\normalsize\headfont\jsParagraphMark}}
2791 \fi

```

`\subparagraph` 見出しの後ろで改行されません。

```

2792 \if@twocolumn
2793   \newcommand{\subparagraph}{\@startsection{subparagraph}{5}{\z@}%
2794     {\z@}{\if@slide .4\Cvs \@plus.3\Cdp \else -1\jsZw\fi}%
2795     {\normalfont\normalsize\headfont}}
2796 \else
2797   \newcommand{\subparagraph}{\@startsection{subparagraph}{5}{\z@}%
2798     {\z@}{\if@slide .5\Cvs \@plus.3\Cdp \else -1\jsZw\fi}%
2799     {\normalfont\normalsize\headfont}}
2800 \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 `\@listk` です ( $k = i, ii, iii, iv$ )。 `\@listk` は `\leftmargin` を `\leftmargink` 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。

```

2801 \if@slide
2802   \setlength\leftmargini{1\jsZw}
2803 \else
2804   \if@twocolumn
2805     \setlength\leftmargini{2\jsZw}
2806   \else
2807     \setlength\leftmargini{3\jsZw}
2808   \fi
2809 \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 2810 \if@slide
\leftmarginv 2811   \setlength\leftmarginii {1\jsZw}
\leftmarginvi 2812   \setlength\leftmarginiii{1\jsZw}
2813   \setlength\leftmarginiv  {1\jsZw}
2814   \setlength\leftmarginv  {1\jsZw}
2815   \setlength\leftmarginvi {1\jsZw}
2816 \else
2817   \setlength\leftmarginii {2\jsZw}
2818   \setlength\leftmarginiii{2\jsZw}

```

```

2819 \setlength\leftmarginiv {2\jsZw}
2820 \setlength\leftmarginv {1\jsZw}
2821 \setlength\leftmarginvi {1\jsZw}
2822 \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。

```

2823 \setlength \labelsep {0.5\jsZw} %.5em
2824 \setlength \labelwidth{\leftmargini}
2825 \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。

```

2826 \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 2827 \@beginparpenalty -\@lowpenalty
\@itempenalty 2828 \@endparpenalty -\@lowpenalty
2829 \@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  $\pm_{0.1}^{0.2} \text{\baselineskip}$  を思い切って外しました。

```

2830 \def\@listi{\leftmargin\leftmargini
2831 \parsep \z@
2832 \topsep 0.5\baselineskip
2833 \itemsep \z@ \relax}
2834 \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)。

```

2835 \@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 2836 \def\@listii{\leftmargin\leftmarginii
\@listiv 2837 \labelwidth\leftmarginii \advance\labelwidth-\labelsep
2838 \topsep \z@
\@listv 2839 \parsep \z@
\@listvi 2840 \itemsep\parsep}
2841 \def\@listiii{\leftmargin\leftmarginiii
2842 \labelwidth\leftmarginiii \advance\labelwidth-\labelsep
2843 \topsep \z@
2844 \parsep \z@

```

```

2845 \itemsep\parsep}
2846 \def\@listiv {\leftmargin\leftmarginiv
2847             \labelwidth\leftmarginiv
2848             \advance\labelwidth-\labelsep}
2849 \def\@listv {\leftmargin\leftmarginv
2850             \labelwidth\leftmarginv
2851             \advance\labelwidth-\labelsep}
2852 \def\@listvi {\leftmargin\leftmarginvi
2853             \labelwidth\leftmarginvi
2854             \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。

```

2855 \renewcommand{\theenumi}{\@arabic\c@enumi}
2856 \renewcommand{\theenumii}{\@alph\c@enumii}
2857 \renewcommand{\theenumiii}{\@roman\c@enumiii}
2858 \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。

---

```

2859 \newcommand*{\jsInJaParen}[1]{%
2860   \mbox{\jsInhibitGlue (#1) \jsInhibitGlue}}
2861 \newcommand{\labelenumi}{\theenumi.}
2862 \newcommand{\labelenumii}{\jsInJaParen{\theenumii}}
2863 \newcommand{\labelenumiii}{\theenumiii.}
2864 \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 2865 \renewcommand{\p@enumii}{\theenumi}
2866 \renewcommand{\p@enumiii}{\theenumi\jsInhibitGlue (\theenumii )}
2867 \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`  
`\labelitemiii`  
`\labelitemiv`

```

2868 \newcommand\labelitemi{\textbullet}
2869 \newcommand\labelitemii{\normalfont\bfseries \textendash}
2870 \newcommand\labelitemiii{\textasteriskcentered}
2871 \newcommand\labelitemiv{\textperiodcentered}

```

## ■description 環境

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

```

2872 \newenvironment{description}{%
2873   \list{}{%
2874     \labelwidth=\leftmargin
2875     \labelsep=1\jsZw
2876     \advance \labelwidth by -\labelsep
2877     \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。

```

2878 \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。

```

2879 %<*book|report>
2880 \newenvironment{chapterabstract}{%
2881   \begin{list}{}{%
2882     \listparindent=1\jsZw
2883     \itemindent=\listparindent
2884     \rightmargin=0pt
2885     \leftmargin=5\jsZw}\item[]}{\end{list}\vspace{\baselineskip}}
2886 %</book|report>

```

“普通の” abstract 環境の定義。

```
2887 %<*article|report|slide>
2888 \newbox\@abstractbox
2889 \if@titlepage
2890 \newenvironment{abstract}{%
2891 \titlepage
2892 \null\vfil
2893 \@beginparpenalty\@lowpenalty
2894 \begin{center}%
2895 \headfont \abstractname
2896 \@endparpenalty\@M
2897 \end{center}%
2898 \par}%
2899 {\par\vfil\null\endtitlepage}
2900 \else
2901 \newenvironment{abstract}{%
2902 \if@twocolumn
2903 \ifx\maketitle\relax
2904 \section*{\abstractname}%
2905 \else
2906 \global\setbox\@abstractbox\hbox\bgroup
2907 \begin{minipage}[b]{\textwidth}
2908 \small\parindent1\jsZw
2909 \begin{center}%
2910 {\headfont \abstractname\vspace{-.5em}\vspace{\z@}}%
2911 \end{center}%
2912 \list{}{%
2913 \listparindent\parindent
2914 \itemindent \listparindent
2915 \rightmargin \leftmargin}%
2916 \item\relax
2917 \fi
2918 \else
2919 \small
2920 \begin{center}%
2921 {\headfont \abstractname\vspace{-.5em}\vspace{\z@}}%
2922 \end{center}%
2923 \list{}{%
2924 \listparindent\parindent
2925 \itemindent \listparindent
2926 \rightmargin \leftmargin}%
2927 \item\relax
2928 \fi}{\if@twocolumn
2929 \ifx\maketitle\relax
2930 \else
2931 \endlist\end{minipage}\egroup
2932 \fi
```

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。

```

2933     \else
2934     \endlist
2935     \fi}
2936 \fi
2937 %</article|report|slide>
2938 %<*jspf>
2939 \newbox\@abstractbox
2940 \newenvironment{abstract}{%
2941   \global\setbox\@abstractbox\hbox\bgroup
2942   \begin{minipage}[b]{157\jsc@mmm}{\sffamily Abstract}\par
2943     \small
2944     \if@english \parindent6\jsc@mmm \else \parindent1\jsZw \fi}%
2945   {\end{minipage}\egroup}
2946 %</jspf>

```

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

```

2947 %<*book|report>
2948 \ifbxjs@force@chapterabstract
2949   \let\abstract\chapterabstract
2950   \let\endabstract\endchapterabstract
2951 \fi
2952 %</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。

```

2953 %<*jspf>
2954 %\newbox\@keywordsbox
2955 %\newenvironment{keywords}{%
2956 %   \global\setbox\@keywordsbox\hbox\bgroup
2957 %   \begin{minipage}[b]{1570\jsc@mmm}{\sffamily Keywords:}\par
2958 %     \small\parindent0\jsZw}%
2959 %   {\end{minipage}\egroup}
2960 %</jspf>

```

## ■verse 環境

`verse` (*env.*) 詩のための `verse` 環境です。

```

2961 \newenvironment{verse}{%
2962   \let \\\=\@centercr
2963   \list{}{%
2964     \itemsep \z@
2965     \itemindent -2\jsZw % 元: -1.5em
2966     \listparindent\itemindent
2967     \rightmargin \z@
2968     \advance\leftmargin 2\jsZw}% 元: 1.5em

```

```
2969 \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。

```
2970 \newenvironment{quotation}{%
2971 \list{}{%
2972 \listparindent\parindent
2973 \itemindent\listparindent
2974 \rightmargin \z0}%
2975 \item\relax}{\endlist}
```

### ■quote 環境

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

```
2976 \newenvironment{quote}%
2977 {\list{}{\rightmargin\z0}\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。

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

```

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

[2017-02-24] コミュニティ版 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、従来の挙動は何も変わっていません。また、`book` 以外の場合のページ番号のリセットもコミュニティ版 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。

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



```

2987     \@restonecolfalse\newpage
2988     \fi
2989     \thispagestyle{empty}%
2990     \ifodd\c@page\setcounter{page}\@ne\else\setcounter{page}\z@\fi %% 2017-
      02-24
2991   }%
2992   {\if@restonecol\twocolumn \else \newpage \fi
2993     \if@twoside\else
2994       \setcounter{page}\@ne
2995     \fi}

```

## ■付録

`\appendix` 本文と付録を分離するコマンドです。

```

2996 %<!*book&!report>
2997 \newcommand{\appendix}{\par
2998   \setcounter{section}{0}%
2999   \setcounter{subsection}{0}%
3000   \ifnum\bxjs@label@section=\bxjs@label@section@@compat
3001     \gdef\presectionname{\appendixname}%
3002     \gdef\postsectionname{}%
3003   % \gdef\thesection{\@Alph\c@section}% [2003-03-02]
3004     \gdef\thesection{\presectionname\@Alph\c@section\postsectionname}%
3005     \gdef\thesubsection{\@Alph\c@section.\@arabic\c@subsection}%
3006   \else
3007     \gdef\@secapp{\appendixname}%
3008     \gdef\@secpos{}%
3009     \gdef\thesection{\@Alph\c@section}%
3010     \fi}
3011 %</!*book&!report>
3012 %<*book|report>
3013 \newcommand{\appendix}{\par
3014   \setcounter{chapter}{0}%
3015   \setcounter{section}{0}%
3016   \gdef\@chapapp{\appendixname}%
3017   \gdef\@chappos{}%
3018   \gdef\thechapter{\@Alph\c@chapter}}
3019 %</book|report>

```

## 8.4 パラメータの設定

### ■array と tabular 環境

`\arraycolsep` array 環境の列間には `\arraycolsep` の 2 倍の幅の空きが入ります。

```
3020 \setlength\arraycolsep{5\jsc@mpt}
```

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

```
3021 \setlength\tabcolsep{6\jsc@mpt}
```

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

```
3022 \setlength\arrayrulewidth{.4\p@}
```

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

```
3023 \setlength\doublerulesep{2\p@}
```

### ■`tabbing` 環境

`\tabbingsep` `\'` コマンドで入るアキです。

```
3024 \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。

```
3025 \skip\@mpfootins = \skip\footins
```

### ■`framebox` 環境

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

`\fboxrule` `\fbox`, `\framebox` の罫線の幅です。

```
3026 \setlength\fboxsep{3\jsc@mp@}
```

```
3027 \setlength\fboxrule{.4\p@}
```

### ■`equation` と `eqnarray` 環境

`\theequation` 数式番号を出力するコマンドです。

```
3028 %<!book&!report>\renewcommand \theequation {\@arabic\c@equation}
```

```
3029 %<*book|report>
```

```
3030 \@addtoreset{equation}{chapter}
```

```
3031 \renewcommand\theequation
```

```
3032 {\ifnum \c@chapter>z@ \thechapter.\fi \@arabic\c@equation}
```

```
3033 %</book|report>
```

`\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。

```
3034 % \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。

```
3035 % \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。

```
3036 % \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。

```
3037 %<!*book&!report>
3038 \newcounter{figure}
3039 \renewcommand \thefigure {\@arabic\c@figure}
3040 %</!*book&!report>
3041 %<*book|report>
3042 \newcounter{figure}[chapter]
3043 \renewcommand \thefigure
3044     {\ifnum \c@chapter>\z@ \thechapter.\fi \@arabic\c@figure}
3045 %</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 3046 \def\fps@figure{tbp}
3047 \def\ftype@figure{1}
\fnum@figure 3048 \def\ext@figure{lof}
3049 \def\fnum@figure{\figurename\nobreak\thefigure}
```

`figure (env.)` \* 形式は段抜きのフロートです。

```
figure* (env.) 3050 \newenvironment{figure}%
3051             {\@float{figure}}%
3052             {\end@float}
3053 \newenvironment{figure*}%
3054             {\@dblfloat{figure}}%
3055             {\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。

```

3056 %<!*book&!report>
3057 \newcounter{table}
3058 \renewcommand\thetable{\@arabic\c@table}
3059 %</!*book&!report>
3060 %<*book|report>
3061 \newcounter{table}[chapter]
3062 \renewcommand \thetable
3063     {\ifnum \c@chapter>\z@ \thechapter.\fi \@arabic\c@table}
3064 %</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  
`\ftype@table` した。

```

\ext@table 3065 \def\fps@table{tbp}
3066 \def\ftype@table{2}
\fnun@table 3067 \def\ext@table{lot}
3068 \def\fnun@table{\tablename\nobreak\thetable}

```

`table` (*env.*) \* は段抜きのフロートです。

```

table* (env.) 3069 \newenvironment{table}%
3070     {\@float{table}}%
3071     {\end@float}
3072 \newenvironment{table*}%
3073     {\@dblfloat{table}}%
3074     {\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  
`\belowcaptionskip` ていたもので、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。

```

3075 \newlength\abovecaptionskip
3076 \newlength\belowcaptionskip
3077 \setlength\abovecaptionskip{5\jsc@empt} % 元: 10\p@
3078 \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。

```

3079 %<!*jspf>
3080 % \long\def\@makecaption#1#2{\small
3081 %     \advance\leftskip10\jsc@mmm

```

```

3082 % \advance\rightskip10\jsc@mmm
3083 % \vskip\abovecaptionskip
3084 % \sbox\@tempboxa{#1\hskip1\jsZw\relax #2}%
3085 % \ifdim \wd\@tempboxa >\hsize
3086 % #1\hskip1\jsZw\relax #2\par
3087 % \else
3088 % \global \@minipagefalse
3089 % \hb@xt@\hsize{\hfil\box\@tempboxa\hfil}%
3090 % \fi
3091 % \vskip\belowcaptionskip}}
3092 \long\def\@makecaption#1#2{ {\small
3093 \advance\leftskip .0628\linewidth
3094 \advance\rightskip .0628\linewidth
3095 \vskip\abovecaptionskip
3096 \sbox\@tempboxa{#1\zwspace#2}%
3097 \ifdim \wd\@tempboxa <\hsize \centering \fi
3098 #1\zwspace#2\par
3099 \vskip\belowcaptionskip}}
3100 %</!jspf>
3101 %<*jspf>
3102 \long\def\@makecaption#1#2{%
3103 \vskip\abovecaptionskip
3104 \sbox\@tempboxa{\small\sffamily #1\quad #2}%
3105 \ifdim \wd\@tempboxa >\hsize
3106 { \small\sffamily
3107 \list{#1}{%
3108 \renewcommand{\makelabel}[1]{##1\hfil}
3109 \itemsep \z@
3110 \itemindent \z@
3111 \labelsep \z@
3112 \labelwidth 11\jsc@mmm
3113 \listparindent\z@
3114 \leftmargin 11\jsc@mmm}\item\relax #2\endlist}
3115 \else
3116 \global \@minipagefalse
3117 \hb@xt@\hsize{\hfil\box\@tempboxa\hfil}%
3118 \fi
3119 \vskip\belowcaptionskip}
3120 %</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`

`\if@jsc@warnoldfontcmd` は BXJS クラスでは不使用。

`\if@jsc@warnoldfontcmdexception` は `\allow/disallowoldfontcommands` の状態を表す。

---

```
3121 \newif\if@jsc@warnoldfontcmd
3122 \@jsc@warnoldfontcmdtrue
3123 \newif\if@jsc@warnoldfontcmdexception
3124 \@jsc@warnoldfontcmdexceptionfalse
```

`\jsc@DeclareOldFontCommand`

```
3125 \newcommand*{\jsc@DeclareOldFontCommand}[3]{%
3126   \g@addto@macro\bxjs@oldfontcmd@list{\do#1}%
3127   \DeclareOldFontCommand{#1}{%
3128     \bxjs@oldfontcmd{#1}#2%
3129   }{%
3130     \bxjs@oldfontcmd{#1}#3%
3131   }%
3132 }
3133 \DeclareRobustCommand*\jsc@warnoldfontcmd[1]{%
3134   \ClassInfo\bxjs@clsname
3135   {Old font command '\string#1' is used!!\MessageBreak
3136     The first occurrence is}%
3137 }
```

---

`\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。

```
3138 \DeclareRobustCommand*\allowoldfontcommands}{%
3139   \@jsc@warnoldfontcmdexceptiontrue}
3140 \DeclareRobustCommand*\disallowoldfontcommands}{%
3141   \@jsc@warnoldfontcmdexceptionfalse}

3142 \let\bxjs@oldfontcmd@list\@empty
3143 \def\bxjs@oldfontcmd#1{%
3144   \expandafter\bxjs@oldfontcmd@a\csname bxjs@ofc/\string#1\endcsname#1}
3145 \def\bxjs@oldfontcmd@a#1#2{%
3146   \if@jsc@warnoldfontcmdexception\else
3147     \global\@jsc@warnoldfontcmdfalse
3148     \ifx#1\relax
3149       \global\let#1=t%
3150       \jsc@warnoldfontcmd{#2}%
3151     \fi
3152   \fi}
```

```

3153 \def\bxjs@warnoldfontcmd@final{%
3154 % \par
3155 \global\let\bxjs@warnoldfontcmd@final\@empty
3156 \let\@tempa\@empty
3157 \def\do##1{%
3158   \ifundefined{bxjs@ofc/\string##1}{\%else
3159     \edef\@tempa{\@tempa \space\string##1}}}%
3160 \bxjs@oldfontcmd@list
3161 \ifx\@tempa\@empty\else
3162   \ClassWarningNoLine\bxjs@clsname
3163     {Some old font commands were used in text:\MessageBreak
3164     \space\@tempa\MessageBreak
3165     You should note, that since 1994 LaTeX2e provides a\MessageBreak
3166     new font selection scheme called NFSS2 with several\MessageBreak
3167     new, combinable font commands. The
3168     class provides\MessageBreak
3169     the old font commands only for compatibility}
3170 \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。

```

3171 \ifbxjs@old@hook@system
3172   \AtEndDocument{%
3173     \g@addto@macro\clearpage{\bxjs@warnoldfontcmd@final}}
3174 \else
3175   \AddToHook{enddocument/afterlastpage}{\bxjs@warnoldfontcmd@final}
3176 \fi

```

`\mc` フォントファミリーを変更します。

```

\gt 3177 \jsc@DeclareOldFontCommand{\mc}{\normalfont\mcfamily}{\mathmc}
\rm 3178 \jsc@DeclareOldFontCommand{\gt}{\normalfont\gtfamily}{\mathgt}
\sf 3179 \jsc@DeclareOldFontCommand{\rm}{\normalfont\rmfamily}{\mathrm}
\sf 3180 \jsc@DeclareOldFontCommand{\sf}{\normalfont\sffamily}{\mathsf}
\tt 3181 \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すコマンドは `\mdseries` です。

```

3182 \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  
`\sl` せん（警告メッセージを出力します）。通常のアップライト体に戻すコマンドは `\upshape`  
`\sc` です。

```

3183 \jsc@DeclareOldFontCommand{\it}{\normalfont\itshape}{\mathit}

```

```
3184 \jsc@DeclareOldFontCommand{\sl}{\normalfont\slshape}{\@nomath\sl}
3185 \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 3186 \DeclareRobustCommand*{\cal}{\@fontswitch\relax\mathcal}
3187 \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`  $\geq$  `\@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。



```

3188 \newcommand\@pnumwidth{1.55em}
3189 \newcommand\@tocrmarg{2.55em}
3190 \newcommand\@dotsep{4.5}
3191 %<!book&!report>\setcounter{tocdepth}{2}
3192 %<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)

```

3193 \newdimen\jsc@tocl@width
3194 \newcommand{\tableofcontents}{%
3195 %<*book|report>
3196 \settowidth\jsc@tocl@width{\headfont\prechaptername\postchaptername}%
3197 \settowidth\@tempdima{\headfont\appendixname}%
3198 \ifdim\jsc@tocl@width<\@tempdima \setlength\jsc@tocl@width{\@tempdima}\fi
3199 \ifdim\jsc@tocl@width<2\jsZw \divide\jsc@tocl@width by 2 \advance\jsc@tocl@width 1\jsZw\fi
3200 \if@twocolumn
3201 \@restonecoltrue\onecolumn
3202 \else
3203 \@restonecolfalse
3204 \fi
3205 \chapter*{\contentsname}%
3206 \@mkboth{\contentsname}{}%
3207 %</book|report>
3208 %<*!book&!report>
3209 \settowidth\jsc@tocl@width{\headfont\presectionname\postsectionname}%
3210 \settowidth\@tempdima{\headfont\appendixname}%
3211 \ifdim\jsc@tocl@width<\@tempdima\relax\setlength\jsc@tocl@width{\@tempdima}\fi
3212 \ifdim\jsc@tocl@width<2\jsZw \divide\jsc@tocl@width by 2 \advance\jsc@tocl@width 1\jsZw\fi
3213 \section*{\contentsname}%
3214 \@mkboth{\contentsname}{\contentsname}%
3215 %</!book&!report>
3216 \@starttoc{toc}%
3217 %<book|report> \if@restonecol\twocolumn\fi
3218 }

```

\l@part 部の目次です。

```

3219 \newcommand*{\l@part}[2]{%
3220 \ifnum \c@tocdepth >-2\relax
3221 %<!book&!report> \addpenalty\@secpenalty
3222 %<book|report> \addpenalty{-\@highpenalty}%
3223 \addvspace{2.25em \@plus\jsc@empt}%
3224 \begingroup
3225 \parindent \z@
3226 % \@pnumwidth should be \@tocrmarg
3227 % \rightskip \@pnumwidth
3228 \rightskip \@tocrmarg

```

```

3229     \parfillskip -\rightskip
3230     {\leavevmode
3231     \large \headfont
3232     \setlength\@lnumwidth{4\jsZw}%
3233     #1\hfil \hb@xt@\@pnumwidth{\hss #2}}\par
3234     \nobreak
3235 %<book|report>     \global\@nobreaktrue
3236 %<book|report>     \everypar{\global\@nobreakfalse\everypar{}}%
3237     \endgroup
3238     \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)

```

3239 %<*book|report>
3240 \newcommand*{\l@chapter}[2]{%
3241   \ifnum \c@tocdepth >\m@ne
3242     \addpenalty{-\@highpenalty}%
3243     \addvspace{1.0em \@plus\jsc@mpt}%
3244 %   \vskip 1.0em \@plus\p@ % book.cls では↑がこうなっている
3245     \begingroup
3246     \parindent\z@
3247 %   \rightskip\@pnumwidth
3248     \rightskip\@tocrmarg
3249     \parfillskip-\rightskip
3250     \leavevmode\headfont
3251 %   % \if@english\setlength\@lnumwidth{5.5em}\else\setlength\@lnumwidth{4.683\jsZw}\fi
3252     \setlength\@lnumwidth{\jsc@tocl@width}\advance\@lnumwidth 2.683\jsZw
3253     \advance\leftskip\@lnumwidth \hskip-\leftskip
3254     #1\nobreak\hfil\nobreak\hbox to\@pnumwidth{\hss#2}\par
3255     \penalty\@highpenalty
3256     \endgroup
3257     \fi}
3258 %</book|report>

```

\l@section 節の目次です。

```

3259 %<*!book&!report>
3260 \newcommand*{\l@section}[2]{%
3261   \ifnum \c@tocdepth >\z@
3262     \addpenalty{\@secpenalty}%
3263     \addvspace{1.0em \@plus\jsc@mpt}%
3264     \begingroup
3265     \parindent\z@
3266 %   \rightskip\@pnumwidth
3267     \rightskip\@tocrmarg
3268     \parfillskip-\rightskip
3269     \leavevmode\headfont
3270 %   % \setlength\@lnumwidth{4\jsZw}% 元 1.5em [2003-03-02]
3271     \setlength\@lnumwidth{\jsc@tocl@width}\advance\@lnumwidth 2\jsZw

```

```

3272     \advance\leftskip\@lnumwidth \hskip-\leftskip
3273     #1\nobreak\hfil\nobreak\hbox to\@pnumwidth{\hss#2}\par
3274     \endgroup
3275     \fi}
3276 %</!book&!report>

```

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

```

3277 %<book|report> % \newcommand*{\l@section}{\@dottedtocline{1}{1\jsZw}{3.683\jsZw}}

```

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

\l@subsection 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@subsubsection 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 3278 %<!*book&!report>
3279 % \newcommand*{\l@subsection} {\@dottedtocline{2}{1.5em}{2.3em}}
3280 % \newcommand*{\l@subsubsection}{\@dottedtocline{3}{3.8em}{3.2em}}
3281 % \newcommand*{\l@paragraph} {\@dottedtocline{4}{7.0em}{4.1em}}
3282 % \newcommand*{\l@subparagraph} {\@dottedtocline{5}{10em}{5em}}
3283 %
3284 % \newcommand*{\l@subsection} {\@dottedtocline{2}{1zw}{3zw}}
3285 % \newcommand*{\l@subsubsection}{\@dottedtocline{3}{2\jsZw}{3\jsZw}}
3286 % \newcommand*{\l@paragraph} {\@dottedtocline{4}{3\jsZw}{3\jsZw}}
3287 % \newcommand*{\l@subparagraph} {\@dottedtocline{5}{4\jsZw}{3\jsZw}}
3288 %
3289 \newcommand*{\l@subsection}{%
3290     \@tempdima\jsc@tocl@width \advance\@tempdima -1\jsZw
3291     \@dottedtocline{2}{\@tempdima}{3\jsZw}}
3292 \newcommand*{\l@subsubsection}{%
3293     \@tempdima\jsc@tocl@width \advance\@tempdima 0\jsZw
3294     \@dottedtocline{3}{\@tempdima}{4\jsZw}}
3295 \newcommand*{\l@paragraph}{%
3296     \@tempdima\jsc@tocl@width \advance\@tempdima 1\jsZw
3297     \@dottedtocline{4}{\@tempdima}{5\jsZw}}
3298 \newcommand*{\l@subparagraph}{%
3299     \@tempdima\jsc@tocl@width \advance\@tempdima 2\jsZw
3300     \@dottedtocline{5}{\@tempdima}{6\jsZw}}
3301 %</!book&!report>
3302 %<*book|report>
3303 % \newcommand*{\l@subsection} {\@dottedtocline{2}{3.8em}{3.2em}}
3304 % \newcommand*{\l@subsubsection}{\@dottedtocline{3}{7.0em}{4.1em}}
3305 % \newcommand*{\l@paragraph} {\@dottedtocline{4}{10em}{5em}}
3306 % \newcommand*{\l@subparagraph} {\@dottedtocline{5}{12em}{6em}}
3307 \newcommand*{\l@section}{%
3308     \@tempdima\jsc@tocl@width \advance\@tempdima -1\jsZw
3309     \@dottedtocline{1}{\@tempdima}{3.683\jsZw}}
3310 \newcommand*{\l@subsection}{%
3311     \@tempdima\jsc@tocl@width \advance\@tempdima 2.683\jsZw
3312     \@dottedtocline{2}{\@tempdima}{3.5\jsZw}}

```

```

3313 \newcommand*\l@subsubsection}{%
3314     \@tempdima\jsc@tocl@width \advance\@tempdima 6.183\jsZw
3315     \@dottedtocline{3}{\@tempdima}{4.5\jsZw}}
3316 \newcommand*\l@paragraph}{%
3317     \@tempdima\jsc@tocl@width \advance\@tempdima 10.683\jsZw
3318     \@dottedtocline{4}{\@tempdima}{5.5\jsZw}}
3319 \newcommand*\l@subparagraph}{%
3320     \@tempdima\jsc@tocl@width \advance\@tempdima 16.183\jsZw
3321     \@dottedtocline{5}{\@tempdima}{6.5\jsZw}}
3322 %</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。

```

3323 \newdimen\@lnumwidth
3324 \def\numberline#1{\hb@xt@\@lnumwidth{#1\hfil}\hspace{0pt}}

```

`\@dottedtocline` L<sup>A</sup>T<sub>E</sub>X 本体 (l<sup>t</sup>sect.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。

```

3325 \def\jsTocLine{\leaders\hbox{%
3326     $\m@th \mkern \@dotsep mu\hbox{.}\mkern \@dotsep mu$\}\hfill}
3327 \def\@dottedtocline#1#2#3#4#5{\ifnum #1>\c@tocdepth \else
3328     \vskip \z@ \@plus.2\jsc@mpt
3329     {\leftskip #2\relax \rightskip \@tocrmarg \parfillskip -\rightskip
3330     \parindent #2\relax\@afterindenttrue
3331     \interlinepenalty\@M
3332     \leavevmode
3333     \@lnumwidth #3\relax
3334     \advance\leftskip \@lnumwidth \null\nobreak\hskip -\leftskip
3335     {#4}\nobreak
3336     \jsTocLine \nobreak\hb@xt@\@pnumwidth{%
3337         \hfil\normalfont \normalcolor #5}\par}\fi}

```

## ■ 図目次と表目次

`\listoffigures` 図目次を出力します。

```

3338 \newcommand{\listoffigures}{%
3339 %<*book|report>
3340 \if@twocolumn\@restonecoltrue\onecolumn
3341 \else\@restonecolfalse\fi

```

```

3342 \chapter*{\listfigurename}%
3343 \@mkboth{\listfigurename}{}%
3344 %</book|report>
3345 %<!*book&!report>
3346 \section*{\listfigurename}%
3347 \@mkboth{\listfigurename}{\listfigurename}%
3348 %</!book&!report>
3349 \@starttoc{lof}%
3350 %<book|report> \if@restonecol\twocolumn\fi
3351 }

```

`\l@figure` 図目次の項目を出力します。

```
3352 \newcommand{\l@figure}{\@dottedtocline{1}{1\jsZw}{3.683\jsZw}}
```

`\listoftables` 表目次を出力します。

```

3353 \newcommand{\listoftables}{%
3354 %<*book|report>
3355 \if@twocolumn\@restonecoltrue\onecolumn
3356 \else\@restonecolfalse\fi
3357 \chapter*{\listtablename}%
3358 \@mkboth{\listtablename}{}%
3359 %</book|report>
3360 %<!*book&!report>
3361 \section*{\listtablename}%
3362 \@mkboth{\listtablename}{\listtablename}%
3363 %</!book&!report>
3364 \@starttoc{lot}%
3365 %<book|report> \if@restonecol\twocolumn\fi
3366 }

```

`\l@table` 表目次は図目次と同じです。

```
3367 \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。

```

3368 \newdimen\bibindent
3369 \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。

```

3370 \newenvironment{thebibliography}[1]{%
3371 \@jsc@warnoldfontcmdexceptiontrue
3372 \global\let\presectionname\relax
3373 \global\let\postsectionname\relax

```

```

3374 %<article|slide> \section*{\refname}\mkboth{\refname}{\refname}%
3375 %<*kiyou>
3376 \vspace{1.5\baselineskip}
3377 \subsubsection*{\refname}\mkboth{\refname}{\refname}%
3378 \vspace{0.5\baselineskip}
3379 %</kiyou>
3380 <book|report> \chapter*{\bibname}\mkboth{\bibname}{}%
3381 %<book|report> \addcontentsline{toc}{chapter}{\bibname}%
3382 \list{\biblabel{\@arabic\c@enumiv}}%
3383     {\settowidth\labelwidth{\biblabel{#1}}%
3384     \leftmargin\labelwidth
3385     \advance\leftmargin\labelsep
3386     \@openbib@code
3387     \usecounter{enumiv}%
3388     \let\p@enumiv\@empty
3389     \renewcommand\theenumiv{\@arabic\c@enumiv}}%
3390 %<kiyou> \small
3391 \sloppy
3392 \clubpenalty4000
3393 \@clubpenalty\clubpenalty
3394 \widowpenalty4000%
3395 \sfcode`.\@m}
3396 {\def\@noitemerr
3397   {\@latex@warning{Empty `thebibliography' environment}}%
3398 \endlist}

```

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

```
3399 \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。

```
3400 \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。

```
3401 % \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、必要に応じて生かしてください。かっこの前後に入るグルーを `\jsInhibitGlue` で取っていますので、オリジナル同様、 `Knuth-\cite{knu}]` のように半角空白で囲んでください。

```

3402 % \def\@citex[#1]#2{\leavevmode
3403 %   \let\@citea\@empty
3404 %   \@cite{\@for\@citeb:=#2\do
3405 %     {\@citea\def\@citea{, \inhibitglue\penalty\@m }%
3406 %     \edef\@citeb{\expandafter\@firstofone\@citeb\@empty}}%
3407 %   \if@filesw\immediate\write\@auxout{\string\citation{\@citeb}}\fi

```

```

3408 %      \@ifundefined{b@\citeb}{\mbox{\normalfont\bfseries ?}}%
3409 %      \G@refundefinedtrue
3410 %      \@latex@warning
3411 %      {Citation `@\citeb' on page \thepage \space undefined}}%
3412 %      {\@cite@ofmt{\csname b@\citeb\endcsname}}}{#1}}
3413 % \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。
3414 % \DeclareRobustCommand\cite{\unskip
3415 %   \@ifnextchar [{\@tempwatrue\@citex}{\@tempwafalse\@citex []}]
3416 % \def\@cite#1#2{$^{\hbox{\scriptsize#1\if@tempswa
3417 %   , \jsInhibitGlue\ #2\fi) }}$}

```

### 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くにマージンがずれる現象を直しました (Thanks: 藤村さん)。

```

3418 \newenvironment{theindex}{% 索引を 3 段組で出力する環境
3419   \if@twocolumn
3420     \onecolumn\@restonecolfalse
3421   \else
3422     \clearpage\@restonecoltrue
3423   \fi
3424   \columnseprule.4pt \columnsep 2\jsZw
3425   \ifx\multicols\undefined
3426 %<book|report>   \twocolumn[\@makeschapterhead{\indexname}]%
3427 %<book|report>   \addcontentsline{toc}{chapter}{\indexname}]%
3428 %<!book&!report> \def\presectionname{}\def\postsectionname{}%
3429 %<!book&!report> \twocolumn[\section*{\indexname}]%
3430   \else
3431     \ifdim\textwidth<\fullwidth
3432       \setlength{\evensidemargin}{\oddsidemargin}
3433       \setlength{\textwidth}{\fullwidth}
3434       \setlength{\linewidth}{\fullwidth}
3435 %<book|report>   \begin{multicols}{3}[\chapter*{\indexname}]%
3436 %<book|report>   \addcontentsline{toc}{chapter}{\indexname}]%
3437 %<!book&!report> \def\presectionname{}\def\postsectionname{}%
3438 %<!book&!report> \begin{multicols}{3}[\section*{\indexname}]%
3439   \else
3440 %<book|report>   \begin{multicols}{2}[\chapter*{\indexname}]%
3441 %<book|report>   \addcontentsline{toc}{chapter}{\indexname}]%
3442 %<!book&!report> \def\presectionname{}\def\postsectionname{}%
3443 %<!book&!report> \begin{multicols}{2}[\section*{\indexname}]%
3444   \fi
3445   \fi
3446 %<book|report>   \@mkboth{\indexname}{}%
3447 %<!book&!report> \mkboth{\indexname}{\indexname}%

```

```

3448 \plainifnotempty % \thispagestyle{plain}
3449 \parindent\z@
3450 \parskip\z@ \@plus .3\jsc@empt\relax
3451 \let\item\@idxitem
3452 \raggedright
3453 \footnotesize\narrowbaselines
3454 }{
3455 \ifx\multicols\@undefined
3456 \if@restonecol\onecolumn\fi
3457 \else
3458 \end{multicols}
3459 \fi
3460 \clearpage
3461 }

```

`\@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 3462 \newcommand{\@idxitem}{\par\hangindent 4\jsZw} % 元 40pt
\subsubitem 3463 \newcommand{\subitem}{\@idxitem \hspace*{2\jsZw}} % 元 20pt
3464 \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。

```

3465 \newcommand{\indexspace}{\par \vskip 10\jsc@empt \@plus5\jsc@empt \@minus3\jsc@empt\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`) などもいいでしょう。

```

3466 \newcommand\seename{\if@english see\else →\fi}
3467 \newcommand\alsoname{\if@english see also\else →\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。

---

```

3468 \ifx\pltx@foot@penalty\@undefined
3469 \let\footnotes@ve=\footnote
3470 \def\footnote{\jsInhibitGlue\footnotes@ve}
3471 \let\footnotemarks@ve=\footnotemark
3472 \def\footnotemark{\jsInhibitGlue\footnotemarks@ve}
3473 \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> の変更に従いました (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 `newpTEXtext` 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)。

```
3474 \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。

```
3475 % \def\thefootnote{\ifnum\c@footnote>\z@ 注\kern0.1\jsw\@arabic\c@footnote\fi}
```

`\footnoterule` 本文と脚注の間の罫線です。

```
3476 \renewcommand{\footnoterule}{%
3477   \kern-2.6\jsc@empt \kern-.4\p@
3478   \hrule width .4\columnwidth
3479   \kern 2.6\jsc@empt}
```

`\c@footnote` 脚注番号は章ごとにリセットされます。

```
3480 %<book|report>\@addtoreset{footnote}{chapter}
```

`\@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>L</sub>A<sub>T</sub>E<sub>X</sub> 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>L</sub>A<sub>T</sub>E<sub>X</sub> のバグ修正に従いました。

[2016-11-29] 古い p<sub>L</sub>A<sub>T</sub>E<sub>X</sub> 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<sub>A</sub>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、追従します。なお、L<sub>A</sub>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。

```

3481 \long\def\@footnotetext{%
3482   \insert\footins\bgroup
3483     \normalfont\footnotesize
3484     \interlinepenalty\interfootnotelinepenalty
3485     \splittopskip\footnotesepp
3486     \splitmaxdepth \dp\strutbox \floatingpenalty \@MM
3487     \hsize\columnwidth \@parboxrestore
3488     \def\@currentcounter{footnote}%
3489     \protected@edef\@currentlabel{%
3490       \csname p@footnote\endcsname\@thefnmark
3491     }%
3492     \color@begingroup
3493     \@makefnmark{%
3494       \rule\z@\footnotesepp\ignorespaces}%
3495     \futurelet\jsc@next\jsc@fo@t}
3496 \def\jsc@fo@t{\ifcat\bgroup\noexpand\jsc@next \let\jsc@next\jsc@fo@t
3497               \else \let\jsc@next\jsc@fo@t\fi \jsc@next}
3498 \def\jsc@fo@t{\bgroup\aftergroup\jsc@@foot\let\jsc@next}
3499 \def\jsc@fo@t#1{\#1\jsc@@foot}
3500 \def\jsc@@foot{\@finalstrut\strutbox\color@endgroup\egroup
3501   \ifx\pltx@foot@penalty\@undefined\else
3502     \ifhmode\null\fi
3503     \ifnum\pltx@foot@penalty=\z@\else
3504       \penalty\pltx@foot@penalty
3505       \pltx@foot@penalty\z@
3506     \fi
3507   \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。

```

3508 \newcommand\@makefnmark[1]{%
3509   \advance\leftskip 3\jsZw
3510   \parindent 1\jsZw
3511   \noindent
3512   \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。

```

3513 % \def\@xfootnotenext[#1]{%
3514 %   \begingroup

```

```

3515 %      \ifnum#1>\z@
3516 %          \csname c@\@mpfn\endcsname #1\relax
3517 %          \unrestored@protected@xdef\@thefnmark{\thempfn}%
3518 %      \else
3519 %          \unrestored@protected@xdef\@thefnmark{}%
3520 %      \fi
3521 %  \endgroup
3522 %  \@footnotetext}

```

---

ここまでのコードは JS クラスを踏襲する。

---

## 11 段落の頭へのグルー挿入禁止

段落頭のかぎかっこなどを見かけ 1 字半下げから全角 1 字下げに直します。

`\jsInhibitGlueAtParTop` 「段落頭の括弧の空き補正」の処理を `\jsInhibitGlueAtParTop` という命令にして、これを再定義可能にした。

```
3523 \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。

```

3524 \def\everyparhook{\jsInhibitGlueAtParTop}
3525 \ifnum\bxjs@everyparhook=\bxjs@everyparhook@@compat
3526 \g@addto@macro\bxjs@begin@document@hook{\everypar{\everyparhook}}
3527 \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` は実はテンポラリーではなく「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。

---

```

3528 \ifx j\jsEngine
3529 \def\@inhibitglue{%
3530   \futurelet\@let@token\@@inhibitglue}
3531 \begingroup

```

```

3532 \let\GDEF=\gdef
3533 \let\CATCODE=\catcode
3534 \let\ENDGROUP=\endgroup
3535 \CATCODE`k=12
3536 \CATCODE`a=12
3537 \CATCODE`n=12
3538 \CATCODE`j=12
3539 \CATCODE`i=12
3540 \CATCODE`c=12
3541 \CATCODE`h=12
3542 \CATCODE`r=12
3543 \CATCODE`t=12
3544 \CATCODE`e=12
3545 \GDEF\KANJI@CHARACTER{kanji character }
3546 \ENDGROUP
3547 \def\@inhibitglue{%
3548   \expandafter\expandafter\expandafter\jsc@inhibitglue\expandafter\meaning\expandafter\@let@
3549 \expandafter\def\expandafter\jsc@inhibitglue\expandafter#\expandafter1\KANJI@CHARACTER#2#3\j
3550   \def\jsc@ig@temp{#1}%
3551   \ifx\jsc@ig@temp\@empty
3552     \ifnum\the\inhibitxspcode`#2=2\relax
3553       \inhibitglue
3554     \fi
3555   \fi}
3556 \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。

---

```

3557 \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] `ltxlists.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。

```

3558 \def\@doendpe{%
3559   \@endpetrue
3560   \def\par{%
3561     \@restorepar\clubpenalty\@clubpenalty\everypar{\everyparhook}\par\@endpefalse}%
3562   \everypar{\setbox\z@\lastbox}\everypar{\everyparhook}\@endpefalse\everyparhook}}

```

[2017-08-31] `minipage` 環境にも対策します。

```

3563 \def\@setminipage{%
3564   \@minipagetrue
3565   \everypar{\@minipagefalse\everypar{\everyparhook}}%
3566 }

```

\item 命令の直後です。

```
3567 \def\@item[#1]{%
3568   \if@noperitem
3569     \@donoperitem
3570   \else
3571     \if@inlabel
3572       \indent \par
3573     \fi
3574     \ifhmode
3575       \unskip\unskip \par
3576     \fi
3577     \if@newlist
3578       \if@nobreak
3579         \@nbitem
3580       \else
3581         \addpenalty\@beginparpenalty
3582         \addvspace\@topsep
3583         \addvspace[-\parskip]%
3584       \fi
3585     \else
3586       \addpenalty\@itempenalty
3587       \addvspace\itemsep
3588     \fi
3589     \global\@inlabeltrue
3590   \fi
3591   \everypar{%
3592     \@minipagefalse
3593     \global\@newlistfalse
3594     \if@inlabel
3595       \global\@inlabelfalse
3596       {\setbox\z@\lastbox
3597        \ifvoid\z@
3598          \kern-\itemindent
3599        \fi}%
3600     \box\@labels
3601     \penalty\z@
3602   \fi
3603   \if@nobreak
3604     \@nobreakfalse
3605     \clubpenalty \@M
3606   \else
3607     \clubpenalty \@clubpenalty
3608     \everypar{\everyparhook}%
3609   \fi\everyparhook}%
3610 \if@noitemarg
3611   \@noitemargfalse
3612 \if@nbrlist
3613   \refstepcounter\@listctr
```

```

3614   \fi
3615   \fi
3616   \sbox\@tempboxa{\makelabel{#1}}%
3617   \global\setbox\@labels\hbox{%
3618     \unhbox\@labels
3619     \hskip \itemindent
3620     \hskip -\labelwidth
3621     \hskip -\labelsep
3622     \ifdim \wd\@tempboxa >\labelwidth
3623       \box\@tempboxa
3624     \else
3625       \hbox to\labelwidth {\unhbox\@tempboxa}%
3626   \fi
3627   \hskip \labelsep}%
3628   \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。

```

3629 \def\@afterheading{%
3630   \@nobreaktrue
3631   \everypar{%
3632     \if@nobreak
3633       \@nobreakfalse
3634       \clubpenalty \@M
3635       \if@afterindent \else
3636         {\setbox\z@\lastbox}%
3637       \fi
3638     \else
3639       \clubpenalty \@clubpenalty
3640       \everypar{\everyparhook}%
3641     \fi\everyparhook}}

```

---

「`\everyparhook` 用のパッチ処理」はここまで。

---

```
3642 \fi
```

`\@gnewline` についてはちょっと複雑な心境です。もともとの  $\text{p}\text{L}\text{A}\text{T}\text{E}\text{X} 2_{\epsilon}$  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。

---

```
3643 \ifnum\bxjs@everyparhook=\bxjs@everyparhook@@none\else
```

```

3644 \AtEndOfClass{%
3645 \def\@gnewline #1{%
3646   \ifvmode
3647     \@nolnerr
3648   \else
3649     \unskip \reserved@e {\reserved@f#1}\nobreak \hfil \break \null
3650     \jsInhibitGlue \ignorespaces
3651   \fi}
3652 }
3653 \fi

```

## 12 いろいろなロゴ

L<sup>A</sup>T<sub>E</sub>X 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。

---

```

3654 \if@jslogo
3655   \IfFileExists{jslogo.sty}{%
3656     \RequirePackage{jslogo}%
3657   }{%
3658     \ClassWarningNoLine\bxjs@clsname
3659     {The package 'jslogo' is not installed.\MessageBreak
3660      It is included in the recent release of\MessageBreak
3661      the 'jsclasses' bundle}
3662   }
3663 \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: しっぽ愛好家さん)。

[2016-11-19] 本家の `ltxclass.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。

```

3664 %\let\ltx@ifnextchar\@ifnextchar
3665 %\def\ProvidesFile#1{%
3666 % \beginingroup
3667 % \catcode\ 10 %
3668 % \ifnum \endlinechar<256 %
3669 % \ifnum \endlinechar>\m@ne
3670 % \catcode\endlinechar 10 %
3671 % \fi
3672 % \fi
3673 % \@makeother\/%
3674 % \@makeother\&%
3675 % \ltx@ifnextchar[{\@providesfile{#1}}{\@providesfile{#1}[]}]

```

## 14 初期設定

### ■いろいろな語

```

\prepartname
\postpartname 3676 \newcommand{\prepartname}{\if@english Part~\else 第\fi}
\prechaptername 3677 \newcommand{\postpartname}{\if@english\else 部\fi}
\postchaptername 3678 %<book|report>\newcommand{\prechaptername}{\if@english Chapter~\else 第\fi}
\postsectionname 3679 %<book|report>\newcommand{\postchaptername}{\if@english\else 章\fi}
\presectionname 3680 \newcommand{\presectionname}{}% 第
\postsectionname 3681 \newcommand{\postsectionname}{}% 節

\contentsname
\listfigurename 3682 \newcommand{\contentsname}{\if@english Contents\else 目次\fi}
\listtablename 3683 \newcommand{\listfigurename}{\if@english List of Figures\else 図目次\fi}
3684 \newcommand{\listtablename}{\if@english List of Tables\else 表目次\fi}

\refname
\bibName 3685 \newcommand{\refname}{\if@english References\else 参考文献\fi}
\indexname 3686 \newcommand{\bibname}{\if@english Bibliography\else 参考文献\fi}
3687 \newcommand{\indexname}{\if@english Index\else 索引\fi}

\figurename
\tablename 3688 %<!jspf>\newcommand{\figurename}{\if@english Fig.~\else 図\fi}
3689 %<jspf>\newcommand{\figurename}{Fig.~}
3690 %<!jspf>\newcommand{\tablename}{\if@english Table~\else 表\fi}
3691 %<jspf>\newcommand{\tablename}{Table~}

\appendixname
\abstractname 3692 % \newcommand{\appendixname}{\if@english Appendix~\else 付録\fi}
3693 \newcommand{\appendixname}{\if@english \else 付録\fi}
3694 %<!book>\newcommand{\abstractname}{\if@english Abstract\else 概要\fi}

```

■今日の日付  $\LaTeX$  で処理した日付を出力します。和暦にするには `\和暦` と書いてください。



---

環境変数 SOURCE\_DATE\_EPOCH / FORCE\_SOURCE\_DATE が設定されている場合は“今日”が過去・未来の日付になる可能性がある。BXJS クラスでは、和暦の扱いは bxwareki パッケージに任せる。

※ 2.0 版より、完全に bxwareki に任せる。

\西暦 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=\<sup>^^</sup>e8<sup>^^</sup>a5<sup>^^</sup>bf<sup>^^</sup>e6<sup>^^</sup>9a<sup>^^</sup>a6
- \和暦=\<sup>^^</sup>e5<sup>^^</sup>92<sup>^^</sup>8c<sup>^^</sup>e6<sup>^^</sup>9a<sup>^^</sup>a6

と扱われるため、\<sup>^^</sup>e8 と \<sup>^^</sup>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。

```
3695 \onlypreamble\bxjs@decl@Seireki@cmds
3696 \@tempwafalse
3697 \if p\jsEngine \@tempwattrue \fi
3698 \if n\jsEngine \@tempwattrue \fi
3699 \bxjs@cond\if@tempwa\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。

```
3700 \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 \<sup>^^</sup>e8 の定義文であることに注意。

```
3701 \def\bxjs@decl@Seireki@cmds{%
3702   \def\西暦{\jsSeirekitrue}%
3703   \def\和暦{\jsSeirekifalse\bxjs@wareki@used}}
```

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

```
\Wareki 3704 \def\Seireki{\jsSeirekitrue}
3705 \def\Wareki{\jsSeirekifalse\bxjs@wareki@used}

3706 \def\bxjs@if@use@seireki{\bxjs@cond\ifjsSeireki\fi}
3707 \def\bxjs@iaif\{noexpand~}
3708 }{%
```

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。

```
3709 \newif\if 西暦 \西暦 true
3710 \def\bxjs@decl@Seireki@cmds{%
3711   \def\西暦{\西暦 true}%
3712   \def\和暦{\西暦 false\bxjs@wareki@used}}
3713 \def\Seireki{\西暦 true}
3714 \def\Wareki{\西暦 false\bxjs@wareki@used}
```

```

3715 \def\bxjs@if@use@seireki{\bxjs@cond\if 西暦\fi}
3716 \let\bxjs@iai\@empty
3717 }
3718 \bxjs@decl@Seireki@cmds

```

`\ifbxjs@bxwareki@avail` `bxwareki` パッケージが利用できるか。

※ 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。

```

3719 \newif\ifbxjs@bxwareki@avail
3720 \IfFileExists{bxwareki.sty}{%
3721   \if \if n\jsEngine \ifjsWitheTeX T\else F\fi\else T\fi T%
3722   \RequirePackage{bxwareki}[2018/04/08]%v0.2
3723   \bxjs@bxwareki@availtrue
3724 \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。

```

3725 \ifbxjs@bxwareki@avail \let\bxjs@wareki@used\@empty
3726 \else
3727   \bxjs@robust@def\bxjs@wareki@used{%
3728     \global\let\bxjs@wareki@used\@empty
3729     \ClassWarning\bxjs@clsname
3730     {Wareki mode is not supported, since\MessageBreak
3731      'bxwareki' is unavailable, reported}}
3732   \g@addto@macro\bxjs@begin@document@hook{%
3733     \let\bxjs@wareki@used\@empty}
3734 \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。

```

3735 \ifbxjs@bxwareki@avail
3736   \let\jayear\warekiyear
3737   \def\bxjs@tmpa{H}\ifx\bxjs@tmpa\warekigengoinitial
3738     \newcount\heisei \heisei=\value{warekiyear}
3739   \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。

```

3740 \else
3741   \edef\jayear{\the\year \bxjs@iai}
3742 \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。

```

3743 %<*jsclasses>
3744 \newif\if 西暦 \西暦 true

```

```

3745 \def\西曆{\西曆 true}
3746 \def\和曆{\西曆 false}
3747 \newcount\heisei \heisei\year \advance\heisei-1988\relax
3748 \def\pltx@today@year@#1{%
3749   \ifnum\numexpr\year-#1=1 元\else
3750     \ifnum1=\iftdir\ifmdir0\else1\fi\else0\fi
3751       \kansuji\numexpr\year-#1\relax
3752     \else
3753       \number\numexpr\year-#1\relax\nobreak
3754     \fi
3755   \fi 年
3756 }
3757 \def\pltx@today@year{%
3758   \ifnum\numexpr\year*10000+\month*100+\day<19890108
3759     昭和\pltx@today@year@{1925}%
3760   \else\ifnum\numexpr\year*10000+\month*100+\day<20190501
3761     平成\pltx@today@year@{1988}%
3762   \else
3763     令和\pltx@today@year@{2018}%
3764   \fi\fi}
3765 %</jsclasses>
3766 \begingroup
3767 \let\bxjs@next\relax
3768 \ifbxjs@bxwareki@avail \ifx\warekigengo@\empty\else
3769   \def\bxjs@next{\warekitoday}
3770   \bxjs@test@engine\unexpanded{%
3771     \def\bxjs@next{\unexpanded\expandafter{\warekitoday}}}}
3772 \fi\fi
3773 \def\!#1#2#3{\noexpand#1\noexpand#2\noexpand#3}
3774 \ifx\bxjs@iaai@\empty \let\!\@empty \fi
3775 \xdef\bxjs@today{%
3776   \if@english
3777     \ifcase\month\or
3778       January\or February\or March\or April\or May\or June\or
3779       July\or August\or September\or October\or November\or December\fi
3780     \space\number\day, \number\year
3781   \else
3782     \ifx\bxjs@next\relax \expandafter\@firstoftwo
3783     \else \noexpand\bxjs@if@use@seireki
3784     \fi {%
3785       \number\year\bxjs@iaai\!年%
3786       \bxjs@iaai\number\month\bxjs@iaai\!月%
3787       \bxjs@iaai\number\day\bxjs@iaai\!日%
3788     }{\bxjs@next}%
3789   \fi}
3790 \endgroup
3791 \let\today\bxjs@today

```

texjporg 版の日本語用 Babel 定義ファイル (japanese.1df) が読み込まれた場合に影響を受けないようにする。

```
3792 \g@addto@macro\bxjs@begin@document@hook{%
3793   \ifx\bb1@jpn@maybekansuji\@undefined\else
3794     \bxjs@decl@Seireki@cmds
3795     \g@addto@macro\datejapanese{%
3796       \let\today\bxjs@today}%
3797   \fi}
```

---

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

```
3798 \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。

```
3799 %<slide>\pagestyle{empty}%
3800 %<article|report>\pagestyle{plain}%
3801 %<book>\pagestyle{headings}%
3802 \pagenumbering{arabic}
3803 \if@twocolumn
3804   \twocolumn
3805   \sloppy
3806   \flushbottom
3807 \else
3808   \onecolumn
3809   \raggedbottom
3810 \fi
3811 %<*slide>
3812 \renewcommand\familydefault{\sfdefault}
3813 \raggedright
3814 %</slide>
```

## 15 実験的コード

---

この節は JS クラスの話で、BXJS クラスには当てはまらない。

---

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

以上です。

## 16 BXJS 独自の追加処理

■**\strong 命令の補填** fontspec で提供される `\strong` 命令と `strongenv` 環境を全てのエンジンで使えるようにする。

※この実装は特にエンジンや和文処理パッケージに依存しないはずであるが、現状では standard 和文ドライバでの提供となっていて、そこで有効化のオプションが定義されている。ここでは `\js~` の名前で定義することにする。

`\jsStrongText` 強調用の宣言型命令。

```
3815 \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。

```
3816 \chardef\bxjs@strong@level=0
3817 \DeclareRobustCommand*\jsStrongDeclare[1]{%
3818   \bxjs@set@array@from@clist{\bxjs@strong}{#1}%
3819   \chardef\bxjs@strong@level\z@}
3820 \jsStrongDeclare{\bfseries}
3821 \def\bxjs@strong@text{%
3822   \bxjs@csletcs{\bxjs@tmpa}{\bxjs@strong/\the\bxjs@strong@level}%
3823   \ifx\bxjs@tmpa\relax
3824     \bxjs@advance@qc\bxjs@strong@level\m@ne \bxjs@strong@text
3825   \else \bxjs@advance@qc\bxjs@strong@level\@ne \bxjs@tmpa
3826   \fi}
3827 % \end{macro}
3828 %
3829 % \paragraph{共通命令の実装}
3830 %
3831 % |\jQ| 等の「単位」系の共通命令を実装する。
3832 %
3833 % \begin{macro}{\bxjs@const@unit}
3834 % 固定値の単位として使える制御綴を定義する。
3835 %
3836 % {\eTeX}拡張が使える場合は、
3837 % 「|\dimexpr|外部寸法表記|\relax|」の形式
3838 % (これは内部値なので単位として使える)
3839 % に展開されるマクロとして定義する。
3840 %   \begin{macrocode}
3841 \@onlypreamble\bxjs@const@unit
3842 \@onlypreamble\bxjs@const@unit@
3843 \ifjsWithTeX
3844   \def\bxjs@const@unit#1#2#3{%
3845     \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ー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)。

```
3846 \else
3847 \let\bxjs@Ct\fontdimen \font\bxjs@Ut=cmtex9 at 0.98245pt
3848 \bxjs@Ct8\bxjs@Ut=8sp \bxjs@Ct16\bxjs@Ut=\z@
3849 \def\bxjs@const@unit#1#2{%
3850 \expandafter\bxjs@const@unit@a\csname bxjs@#2\endcsname#1}
3851 \def\bxjs@const@unit@a#1#2#3{%
3852 \chardef#1\bxjs@Ct8\bxjs@Ut \bxjs@advance@qc#1\@ne \bxjs@Ct8\bxjs@Ut#1sp
3853 \bxjs@Ct#1\bxjs@Ut=#3\relax \def#2{\bxjs@Ct#1\bxjs@Ut}}
3854 \fi
```

\jQ \jQ と \jH はともに 0.25 mm に等しい。

```
\jH 3855 \bxjs@const@unit\jQ{jQ}{0.25mm}
3856 \let\jH\jQ
```

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

```
\trueH 3857 \ifjsc@mag
3858 \@tempdimb=\jsBaseFontSize\relax
3859 \edef\bxjs@tmpa{\strip@pt\@tempdimb}%
3860 \@tempdima=2.5mm
3861 \bxjs@invscale\@tempdima\bxjs@tmpa
3862 \bxjs@const@unit\trueQ{trueQ}{\@tempdima}
3863 \@tempdima=10pt
3864 \bxjs@invscale\@tempdima\bxjs@tmpa
3865 \bxjs@const@unit\bxjs@truept{truept}{\@tempdima}
3866 \else \let\trueQ\jQ \let\bxjs@truept\p@
3867 \fi
3868 \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。

```
3869 \@tempdima\trueQ \bxjs@invscale\@tempdima\jsScale
3870 \bxjs@const@unit\ascQ{ascQ}{\@tempdima}
3871 \@tempdima\bxjs@truept \bxjs@invscale\@tempdima\jsScale
3872 \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。

```
3873 \DeclareRobustCommand*\jsJaFontSize[2]{%
3874 \begingroup
3875 \bxjs@jafontsize@a{#1}%
3876 \@tempdimb\jsInverseScale\@tempdima
3877 \bxjs@jafontsize@a{#2}%
3878 \xdef\bxjs@g@tmpa{%
3879 \noexpand\fontsize{\the\@tempdimb}{\the\@tempdima}}%
3880 \endgroup\bxjs@g@tmpa}
```

```

3881 \def\bxjs@jafontsize@a#1{%
3882 \bxjs@parse@qh{#1}%
3883 \ifx\bxjs@tmpb\relax \def\bxjs@tmpb{#1}\fi
3884 \@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。

```
3885 \def\bxjs@let@lenexpr{\edef}
```

`\bxjs@kanjiskip` 和文間空白の量を表すテキスト。

```
3886 \def\bxjs@kanjiskip{0pt}
```

`\jsSetKanjiSkip` 和文間空白の量を設定する。

※`\setkanjiskip` の実体。

```

3887 \DeclareRobustCommand*\jsSetKanjiSkip[1]{%
3888 \bxjs@let@lenexpr\bxjs@kanjiskip{#1}%
3889 \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。

```

3890 \newcommand*\jsGetKanjiSkip{%
3891 \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。

```
3892 \newif\ifbxjs@kanjiskip@enabled \bxjs@kanjiskip@enabledtrue
```

`\jsEnableKanjiSkip` 和文間空白の挿入を有効／無効にする。

`\jsDisableKanjiSkip` ※ pTeX 系のエンジンや `luatexja` のパッケージを使用する場合はそれ自体がもつ制御機能を利用するため、これらの命令は使わない。

```

3893 \bxjs@robust@def\jsEnableKanjiSkip{%
3894 \bxjs@kanjiskip@enabledtrue
3895 \bxjs@reset@kanjiskip}
3896 \bxjs@robust@def\jsDisableKanjiSkip{%
3897 \bxjs@kanjiskip@enabledfalse
3898 \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。

```

3899 \bxjs@robust@def\bxjs@reset@kanjiskip{%
3900 \ifbxjs@kanjiskip@enabled
3901 \setlength{\@tempkipa}{\bxjs@kanjiskip}%
3902 \else \@tempkipa\z@
3903 \fi
3904 \jsApplyKanjiSkip\@tempkipa}

```

`\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。

```
3905 \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 3906 \def\bxjs@xkanjiskip{0pt}
\jsGetXKanjiSkip 3907 \DeclareRobustCommand*\jsSetXKanjiSkip[1]{%
\ifbxjs@xkanjiskip@enabled 3908 \bxjs@let@lenexpr\bxjs@xkanjiskip{#1}%
3909 \bxjs@reset@xkanjiskip}
\jsEnableXKanjiSkip 3910 \newcommand*\jsGetXKanjiSkip{%
\jsDisableXKanjiSkip 3911 \bxjs@xkanjiskip}
3912 \newif\ifbxjs@xkanjiskip@enabled \bxjs@xkanjiskip@enabledtrue
\bxjs@reset@xkanjiskip 3913 \bxjs@robust@def\jsEnableXKanjiSkip{%
\jsApplyXKanjiSkip 3914 \bxjs@xkanjiskip@enabledtrue
3915 \bxjs@reset@xkanjiskip}
3916 \bxjs@robust@def\jsDisableXKanjiSkip{%
3917 \bxjs@xkanjiskip@enabledfalse
3918 \bxjs@reset@xkanjiskip}
3919 \bxjs@robust@def\bxjs@reset@xkanjiskip{%
3920 \ifbxjs@xkanjiskip@enabled
3921 \setlength{\@tempskipa}{\bxjs@xkanjiskip}%
3922 \else \@tempskipa\z@
3923 \fi
3924 \jsApplyXKanjiSkip\@tempskipa}
3925 \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。

```

3926 \g@addto@macro\jsResetDimen{%
3927 \bxjs@reset@kanjiskip
3928 \bxjs@reset@xkanjiskip}

```

和文・和欧文間空白の初期値。

```

3929 \AtEndOfPackage{%
3930 \jsSetKanjiSkip{0pt plus.1\jsZw minus.01\jsZw}%
3931 \ifx\jsDocClass\jsSlide \jsSetXKanjiSkip{0.1em}%
3932 \else \jsSetXKanjiSkip{0.25em plus 0.15em minus 0.06em}%
3933 \fi
3934 }

```

## ■和文空白命令

```

3935 \ifbxjs@jaspace@cmd

```

\jaenspace 半角幅の水平空き。

```

3936 \def\jaenspace{\hskip.5\jsZw\relax}

```

\jathinspace 和欧文間空白を入れるユーザ命令。

```

3937 \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。

```

3938 \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）。

```
3939 \bxjs@protected\def\bxjs@choice@jathinspace{%
3940   \relax\ifmmode \mskip\medmuskip
3941   \else \jathinspace\ignorespaces
3942   \fi}
3943 \jsAtEndOfClass{%
3944   \ifjsWithTeX \let\>\bxjs@choice@jathinspace
3945   \else \def\>{\protect\bxjs@choice@jathinspace}%
3946   \fi}
```

`\jaspace jlreq` クラスと互換の命令。

```
3947 \DeclareRobustCommand*\jaspace}[1]{%
3948   \expandafter\ifx\csname bxjs@jaspace@@#1\endcsname\relax
3949   \ClassError\bxjs@clsname
3950   {Unknown jaspac: #1}{\@eha}%
3951   \else
3952   \csname bxjs@jaspace@@#1\endcsname
3953   \fi}
3954 \def\bxjs@jaspace@@zenkaku{\hskip 1\jsZw\relax}
3955 \def\bxjs@jaspace@@nibu{\hskip .5\jsZw\relax}
3956 \def\bxjs@jaspace@@shibu{\hskip .25\jsZw\relax}
```

終わり。

```
3957 \fi
```

■和文ドライバ読込 フックを実行する。

```
3958 \bxjs@pre@jadriver@hook
```

和文ドライバのファイルを読み込む。

```
3959 \input{bxjsja-\bxjs@jadriver.def}
```

おしまい。

```
3960 %</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 pdf $\TeX$  (DVI モードも含む)
  - l Lua $\TeX$  (//)
  - x X $\LaTeX$
  - j p $\TeX$  または up $\TeX$
  - 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 up $\TeX$  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 (pdf $\TeX$ ・Lua $\TeX$  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ータを表す文字列。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ー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」はこの設定と辻褃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。

```
3961 %<*drv>
```

## 付録 B 和文ドライバ：minimal

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

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

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

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

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

### B.1 準備

```
3962 %<*minimal>
```

```
3963 %% 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 jI<sub>ε</sub>X と Omega 系。

```
3964 \let\bxjs@tmpa\relax
```

```
3965 \ifx J\jsEngine \def\bxjs@tmpa{NTT-jTeX}\fi
```

```
3966 \ifx O\jsEngine \def\bxjs@tmpa{Omega}\fi
```

```
3967 \ifx\bxjs@tmpa\relax \expandafter\@gobble
```

```
3968 \else
```

```
3969 \ClassError\bxjs@clsname
```

```
3970 {The engine in use (\bxjs@tmpa) is not supported}
```

```

3971   {It's a fatal error. I'll quit right now.}
3972   \expandafter\@firstofone
3973 \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`。

```

3974 \def\DeclareJaTextFontCommand#1#2{%
3975   \DeclareRobustCommand#1[1]{%
3976     \relax
3977     \ifmmode \expandafter\nfss@text \fi
3978     {#2##1}}%
3979 }

```

`\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。

```

3980 \def\DeclareJaMathFontCommand#1#2{%
3981   \DeclareRobustCommand#1[1]{%
3982     \relax
3983     \ifmmode\else \non@alpherr{#1\space}\fi
3984     \nfss@text{\fontfamily\familydefault
3985               \fontseries{m}\fontshape{n}\selectfont\relax
3986               #2##1}%
3987   }%
3988 }

```

`\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。

```

3989 \long\def\bxjs@@CSsfdefault{\sfdefault}%
3990 \@onlypreamble\bxjs@if@sf@default
3991 \def\bxjs@if@sf@default#1{%
3992   \ifx\familydefault\bxjs@@CSsfdefault#1\fi
3993   \g@addto@macro\bxjs@begin@document@hook{%
3994     \ifx\familydefault\bxjs@@CSsfdefault#1\fi}%
3995 }

```

`\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。

```

3996 \@tempdima\p@ \bxjs@invscale\@tempdima\jsScale
3997 \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。

```

3998 \def\jsLetHeadChar#1#2{%
3999   \begingroup
4000     \escapechar=`\ %
4001     \let\bxjs@tmpa={% brace-match-hack
4002     \bxjs@let@hchar@exp#2}%
4003   \endgroup
4004   \let#1\bxjs@g@tmpa}
4005 \def\bxjs@let@hchar@exp{%
4006   \futurelet\@let@token\bxjs@let@hchar@exp@a}
4007 \def\bxjs@let@hchar@exp@a{%
4008   \bxjs@cond@ifcat\noexpand\@let@token\bgroup\fi{% 波括弧
4009     \bxjs@let@hchar@out\let\relax
4010   }{\bxjs@cond@ifcat\noexpand\@let@token\@sptoken\fi{% 空白
4011     \bxjs@let@hchar@out\let\space%
4012   }{\bxjs@cond@if\noexpand\@let@token\@backslashchar\fi{% バックスラッシュ
4013     \bxjs@let@hchar@out\let\@backslashchar
4014   }{\bxjs@let@hchar@exp@b}}}}
4015 \def\bxjs@let@hchar@exp@b#1{%
4016   \expandafter\bxjs@let@hchar@exp@c\string#1?\@nil#1}
4017 \def\bxjs@let@hchar@exp@c#1#2\@nil{%
4018   %\message{<#1#2>}%
4019   \bxjs@cond@if#1\@backslashchar\fi{% 制御綴
4020     \bxjs@cond\expandafter\ifx\noexpand\@let@token\@let@token\fi{%
4021       \bxjs@let@hchar@out\let\relax
4022     }{%else
4023       \expandafter\bxjs@let@hchar@exp
4024     }%
4025   }{%else
4026     \bxjs@let@hchar@chr#1%
4027   }}
4028 \def\bxjs@let@hchar@chr#1{%
4029   \bxjs@let@hchar@out\def{{#1}}}
4030 \def\bxjs@let@hchar@out#1#2{%
4031   \global#1\bxjs@g@tmpa#2\relax
4032   \toks@\bgroup}% skip to right brace

```

UTF-8 のバイト列を扱うコード。

```

4033 \chardef\bxjs@let@hchar@csta=128
4034 \chardef\bxjs@let@hchar@cstb=192
4035 \chardef\bxjs@let@hchar@cstc=224
4036 \chardef\bxjs@let@hchar@cstd=240
4037 \chardef\bxjs@let@hchar@cste=248
4038 \let\bxjs@let@hchar@chr@ue@a\bxjs@let@hchar@chr
4039 \def\bxjs@let@hchar@chr@ue#1{%
4040   \@tempcnta=`#1\relax
4041   %\message{\the\@tempcnta}%
4042   \bxjs@cond\ifnum\@tempcnta<\bxjs@let@hchar@csta\fi{%
4043     \bxjs@let@hchar@chr@ue@a#1%
4044   }{\bxjs@cond\ifnum\@tempcnta<\bxjs@let@hchar@cstb\fi{%

```

```

4045 \bxjs@let@hchar@out\let\relax
4046 }{\bxjs@cond@ifnum\@tempcnta<\bxjs@let@hchar@cstc\fi{%
4047 \bxjs@let@hchar@chr@ue@b
4048 }{\bxjs@cond@ifnum\@tempcnta<\bxjs@let@hchar@cstd\fi{%
4049 \bxjs@let@hchar@chr@ue@c
4050 }{\bxjs@cond@ifnum\@tempcnta<\bxjs@let@hchar@cste\fi{%
4051 \bxjs@let@hchar@chr@ue@d
4052 }{%else
4053 \bxjs@let@hchar@out\let\relax
4054 }}}}
4055 \def\bxjs@let@hchar@chr@ue@a#1{%
4056 \bxjs@let@hchar@out\def{#{#1}}
4057 \def\bxjs@let@hchar@chr@ue@b#1#2{%
4058 \bxjs@let@hchar@out\def{#{#1#2}}
4059 \def\bxjs@let@hchar@chr@ue@c#1#2#3{%
4060 \bxjs@let@hchar@out\def{#{#1#2#3}}
4061 \def\bxjs@let@hchar@chr@ue@d#1#2#3#4{%
4062 \bxjs@let@hchar@out\def{#{#1#2#3#4}}

```

## B.2 (u)pTeX 用の設定

```
4063 \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。

```

4064 \def\bxjs@let@hchar@chr@pp#1#2{%
4065 \expandafter\bxjs@let@hchar@chr@pp@a\meaning#2\relax#1#2}
4066 \def\bxjs@let@hchar@chr@pp@a#1#2\relax#3#4{%
4067 %\message{(\meaning#3:\meaning#4)}%
4068 \bxjs@cond@if#1k\fi{%
4069 \bxjs@let@hchar@out\def{#{#4}}%
4070 }{%else
4071 \bxjs@let@hchar@chr@ue#3#4%
4072 }}
4073 \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。

```

4074 \edef\jsc@JYn{\ifjsWithupTeX JY2\else JY1\fi}
4075 \edef\jsc@JTn{\ifjsWithupTeX JT2\else JT1\fi}
4076 \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。

```
4077 \@onlypreamble\bxjs@declarefontshape
4078 \ifjsWithupTeX
4079 \def\bxjs@declarefontshape{%
4080 \DeclareFontShape{JY2}{mc}{m}{n}{<->s*[\bxjs@scale]upjpnrm-h}{}%
4081 \DeclareFontShape{JY2}{gt}{m}{n}{<->s*[\bxjs@scale]upjpngt-h}{}%
4082 \DeclareFontShape{JT2}{mc}{m}{n}{<->s*[\bxjs@scale]upjpnrm-v}{}%
4083 \DeclareFontShape{JT2}{gt}{m}{n}{<->s*[\bxjs@scale]upjpngt-v}{}%
4084 }
4085 \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。

```
4086 \else
4087 \def\bxjs@declarefontshape{%
4088 \DeclareFontShape{JY1}{mc}{m}{n}{<->s*[\bxjs@scale]jis}{}%
4089 \DeclareFontShape{JY1}{gt}{m}{n}{<->s*[\bxjs@scale]jisg}{}%
4090 \DeclareFontShape{JT1}{mc}{m}{n}{<->s*[\bxjs@scale]tmin10}{}%
4091 \DeclareFontShape{JT1}{gt}{m}{n}{<->s*[\bxjs@scale]tgoth10}{}%
4092 }
4093 \def\bxjs@sizereference{jis}
4094 \fi
```

既で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。

```
4095 \def\bxjs@next#1/#2/#3/#4/#5\relax{%
4096   \def\bxjs@tmpb{#5}}
4097 \ifjsWithpTeXng \def\bxjs@tmpb{10}%
4098 \else
4099 \expandafter\expandafter\expandafter\bxjs@next
4100 \expandafter\string\the\jfont\relax
4101 \fi
4102 \@for\bxjs@tmpa:={\jsc@JYn/mc/m/n,\jsc@JYn/gt/m/n,%
4103   \jsc@JTn/mc/m/n,\jsc@JTn/gt/m/n}\do
4104   {\expandafter\let\csname\bxjs@tmpa/10\endcsname=\@undefined
4105   \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。

```
4106 \begingroup
4107 % 参照用フォント (\bxjs@sizereference) の全角空白の幅を取得
4108 \font\bxjs@tmpa=\bxjs@sizereference\space at 10pt
```

```

4109 \setbox\z@\hbox{\bxjs@tmpa\char\jis"2121\relax}
4110 % 幅が丁度 10pt なら補正は不要
4111 \ifdim\wd\z@=10pt
4112   \global\let\bxjs@scale\jsScale
4113 \else
4114 % (10*s)/(10*f) として計算、\bxjs@invscale は BXJS で定義
4115   \edef\bxjs@tmpa{\strip@pt\wd\z@}
4116   \@tempdima=10pt \@tempdima=\jsScale\@tempdima
4117   \bxjs@invscale\@tempdima\bxjs@tmpa
4118   \xdef\bxjs@scale{\strip@pt\@tempdima}
4119 \fi
4120 \endgroup
4121 %\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。

```
4122 \bxjs@declarefontshape
```

フォント代替の明示的定義。

```

4123 \DeclareFontShape{\jsc@JYn}{mc}{m}{it}{<->ssub*mc/m/n}{}
4124 \DeclareFontShape{\jsc@JYn}{mc}{m}{sl}{<->ssub*mc/m/n}{}
4125 \DeclareFontShape{\jsc@JYn}{mc}{m}{sc}{<->ssub*mc/m/n}{}
4126 \DeclareFontShape{\jsc@JYn}{gt}{m}{it}{<->ssub*gt/m/n}{}
4127 \DeclareFontShape{\jsc@JYn}{gt}{m}{sl}{<->ssub*gt/m/n}{}
4128 \DeclareFontShape{\jsc@JYn}{mc}{bx}{it}{<->ssub*gt/m/n}{}
4129 \DeclareFontShape{\jsc@JYn}{mc}{bx}{sl}{<->ssub*gt/m/n}{}
4130 \DeclareFontShape{\jsc@JYn}{gt}{bx}{it}{<->ssub*gt/m/n}{}
4131 \DeclareFontShape{\jsc@JYn}{gt}{bx}{sl}{<->ssub*gt/m/n}{}
4132 \DeclareFontShape{\jsc@JYn}{mc}{b}{n}{<->ssub*mc/bx/n}{}
4133 \DeclareFontShape{\jsc@JYn}{mc}{b}{it}{<->ssub*mc/bx/n}{}
4134 \DeclareFontShape{\jsc@JYn}{mc}{b}{sl}{<->ssub*mc/bx/n}{}
4135 \DeclareFontShape{\jsc@JYn}{gt}{b}{n}{<->ssub*gt/bx/n}{}
4136 \DeclareFontShape{\jsc@JYn}{gt}{b}{it}{<->ssub*gt/bx/n}{}
4137 \DeclareFontShape{\jsc@JYn}{gt}{b}{sl}{<->ssub*gt/bx/n}{}
4138 \DeclareFontShape{\jsc@JTn}{mc}{m}{it}{<->ssub*mc/m/n}{}
4139 \DeclareFontShape{\jsc@JTn}{mc}{m}{sl}{<->ssub*mc/m/n}{}
4140 \DeclareFontShape{\jsc@JTn}{mc}{m}{sc}{<->ssub*mc/m/n}{}
4141 \DeclareFontShape{\jsc@JTn}{gt}{m}{it}{<->ssub*gt/m/n}{}
4142 \DeclareFontShape{\jsc@JTn}{gt}{m}{sl}{<->ssub*gt/m/n}{}
4143 \DeclareFontShape{\jsc@JTn}{mc}{bx}{it}{<->ssub*gt/m/n}{}
4144 \DeclareFontShape{\jsc@JTn}{mc}{bx}{sl}{<->ssub*gt/m/n}{}
4145 \DeclareFontShape{\jsc@JTn}{gt}{bx}{it}{<->ssub*gt/m/n}{}
4146 \DeclareFontShape{\jsc@JTn}{gt}{bx}{sl}{<->ssub*gt/m/n}{}
4147 \DeclareFontShape{\jsc@JTn}{mc}{b}{n}{<->ssub*mc/bx/n}{}
4148 \DeclareFontShape{\jsc@JTn}{mc}{b}{it}{<->ssub*mc/bx/n}{}
4149 \DeclareFontShape{\jsc@JTn}{mc}{b}{sl}{<->ssub*mc/bx/n}{}
4150 \DeclareFontShape{\jsc@JTn}{gt}{b}{n}{<->ssub*gt/bx/n}{}
4151 \DeclareFontShape{\jsc@JTn}{gt}{b}{it}{<->ssub*gt/bx/n}{}
4152 \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。

```
4153 \@ifl@t@r\fmtversion{2020/10/01}
4154   {\jsc@needspace@tchfalse}{\jsc@needspace@tchtrue}
4155 \ifjsc@needspace@tch      % --- for 2020-02-02 or older BEGIN
4156 \ifx\@rmfamilyhook\@undefined % old
4157 \DeclareRobustCommand\rmfamily
4158   {\not@math@alphabet\rmfamily\mathrm
4159    \romanfamily\rmdefault\kanjifamily\mcdefault\selectfont}
4160 \DeclareRobustCommand\sffamily
4161   {\not@math@alphabet\sffamily\mathsf
4162    \romanfamily\sfdefault\kanjifamily\gtdefault\selectfont}
4163 \DeclareRobustCommand\ttfamily
4164   {\not@math@alphabet\ttfamily\mathtt
4165    \romanfamily\ttdefault\kanjifamily\gtdefault\selectfont}
4166 \g@addto@macro\bxjs@begin@document@hook{%
4167   \ifx@mweights@init\@undefined\else % mweights.sty is loaded
4168     % my definitions above should have been overwritten, recover it!
4169     % \selectfont is executed twice but I don't care about speed...
4170     \expandafter\g@addto@macro\csname rmfamily \endcsname
4171       {\kanjifamily\mcdefault\selectfont}%
4172     \expandafter\g@addto@macro\csname sffamily \endcsname
4173       {\kanjifamily\gtdefault\selectfont}%
4174     \expandafter\g@addto@macro\csname ttfamily \endcsname
4175       {\kanjifamily\gtdefault\selectfont}%
4176   \fi}
4177 \else % 2020-02-02
4178 \g@addto@macro\@rmfamilyhook
4179   {\prepare@family@series@update@kanji{mc}\mcdefault}
4180 \g@addto@macro\@sffamilyhook
4181   {\prepare@family@series@update@kanji{gt}\gtdefault}
4182 \g@addto@macro\@ttfamilyhook
4183   {\prepare@family@series@update@kanji{gt}\gtdefault}
4184 \fi
4185 \else % --- for 2020-02-02 or older END & for 2020-10-01 BEGIN
4186 \AddToHook{rmfamily}%
4187   {\prepare@family@series@update@kanji{mc}\mcdefault}
4188 \AddToHook{sffamily}%
4189   {\prepare@family@series@update@kanji{gt}\gtdefault}
4190 \AddToHook{ttfamily}%
4191   {\prepare@family@series@update@kanji{gt}\gtdefault}
4192 \fi % --- for 2020-10-01 END
4193 \ifx\DeclareFixJFMCJKTextFontCommand\@undefined
4194 \DeclareJaTextFontCommand{\textmc}{\mcfamily}
4195 \DeclareJaTextFontCommand{\textgt}{\gtfamily}
4196 \fi
4197 \bxjs@if@sf@default{%
```

```
4198 \renewcommand\kanjifamilydefault{\gtdefault}}
```

念のため。

```
4199 \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）。

```
4200 \def\bxjs@parse@qh#1{\let\bxjs@tmpb\relax}
```

```
4201 \let\bxjs@parse@qh@a\@undefined
```

```
4202 \let\bxjs@parse@qh@b\@undefined
```

### ■パラメタの設定

```
4203 \prebreakpenalty\jis"2147=10000
```

```
4204 \postbreakpenalty\jis"2148=10000
```

```
4205 \prebreakpenalty\jis"2149=10000
```

```
4206 \inhibitxspcode`!=1
```

```
4207 \inhibitxspcode`〒=2
```

```
4208 \xspcode`+=3
```

```
4209 \xspcode`%=3
```

"80~"FF の範囲の `\spcode` を 3 に変更。

```
4210 \@tempcnta="80 \@whilenum\@tempcnta<"100 \do{%
```

```
4211 \xspcode\@tempcnta=3\advance\@tempcnta\@ne}
```

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

```
4212 \let\jsInhibitGlueAtParTop\@inhibitglue
```

`\jsResetDimen` は空のままよい。

■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。

```
4213 \begingroup
```

```
4214 \catcode`\!=0
```

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

```
4215 \gdef\bxjs@ptex@dir{%
```

```
4216 !iftdir t%
```

```
4217 !else!ifydir y%
```

```
4218 !else ?%
```

```
4219 !fi!fi}
```

新版の pTeX で脚注番号の周囲の空気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。

```
4220 % 古い \@makefnmark の定義
```

```
4221 \long\def\bxjs@tmpa{\hbox{%
```

```
4222 !ifydir \@textsuperscript{\normalfont\@thefnmark}}%
```

```
4223 !else\hbox{\yoko\@textsuperscript{\normalfont\@thefnmark}}!fi}}
```

```

4224 \ifx\@makefnmark\bxjs@tmpa
4225 \long\gdef\@makefnmark{%
4226 !ifydir \hbox{ }\hbox{\@textsuperscript{\normalfont\@thefnmark}}\hbox{ }%
4227 !else\hbox{\yoko\@textsuperscript{\normalfont\@thefnmark}}!fi}
4228 \fi

```

エスケープ文字の変更はここまで。

```

4229 \endgroup

```

■minijs パッケージのブロック やっておく。

```

4230 \@namedef{ver@minijs.sty}{ }

```

### B.3 pdfTeX 用の処理

```

4231 \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。

```

4232 \let\bxjs@let@hchar@chr\bxjs@let@hchar@chr@ue

```

ムニャムニャ。

```

4233 \@onlypreamble\bxjs@cjk@loaded
4234 \def\bxjs@cjk@loaded{%
4235   \def\@footnotemark{%
4236     \leavevmode
4237     \ifhmode
4238       \edef\@x@sf{\the\spacefactor}%
4239       \ifdim\lastkern>\z@\ifdim\lastkern<5sp\relax
4240         \unkern\unkern
4241         \ifdim\lastskip>\z@ \unskip \fi
4242       \fi\fi
4243     \nobreak
4244   \fi
4245   \@makefnmark
4246   \ifhmode \spacefactor\@x@sf \fi
4247   \relax}%
4248 \let\bxjs@cjk@loaded\relax
4249 }
4250 \g@addto@macro\bxjs@begin@document@hook{%
4251   \@ifpackageloaded{CJK}{%
4252     \bxjs@cjk@loaded
4253   }{ }%
4254 }

```

### B.4 XeTeX 用の処理

```

4255 \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。

```

4256 \def\bxjs@let@hchar@chr#1{%
4257   \@tempcnta`#1\relax \divide\@tempcnta"800\relax

```

```

4258 \bxjs@cond@ifnum\@tempcnta=27 \fi{%
4259   \bxjs@let@hchar@chr@xe
4260 }{\bxjs@let@hchar@out\def{#{1}}}}
4261 \def\bxjs@let@hchar@chr@xe#1{%
4262   \lccode`0=`#1\relax
4263   \lowercase{\bxjs@let@hchar@out\def{0}}}}

```

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

```

4264 \@onlypreamble\bxjs@do@precisetext
4265 \ifx\XeTeXgenerateactualtext\@undefined\else
4266   \def\bxjs@do@precisetext{%
4267     \XeTeXgenerateactualtext=\@ne}
4268 \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。

```

4269 \@onlypreamble\bxjs@do@simplejasetup
4270 \def\bxjs@do@simplejasetup{%
4271   \@namedef{bxjs@zeroglue/0.0pt}{T}%
4272   \ifnum\XeTeXinterchartokenstate>\z@
4273   \else\expandafter\ifx\csname bxjs@zeroglue/\the\XeTeXlinebreakskip\endcsname\relax\else
4274     \jsSimpleJaSetup
4275     \ClassInfo\bxjs@clsname
4276     {'\string\jsSimpleJaSetup' is applied\@gobble}%
4277   \fi\fi}

```

`\jsSimpleJaSetup` 日本語出力用の超簡易的な設定。

```

4278 \newcommand*{\jsSimpleJaSetup}{%
4279   \XeTeXlinebreaklocale "ja"\relax
4280   \XeTeXlinebreakskip=0pt plus 1pt minus 0.1pt\relax
4281   \XeTeXlinebreakpenalty=0\relax}

```

## B.5 後処理 (エンジン共通)

```
4282 \fi\fi\fi
```

`simplejasetup` オプションの処理。

```

4283 \ifx\bxjs@do@simplejasetup\@undefined\else
4284   \g@addto@macro\bxjs@begin@document@hook{%
4285     \ifbxjs@simplejasetup
4286       \bxjs@do@simplejasetup
4287     \fi}
4288 \fi

```

`precisetext` オプションの処理。

```

4289 \ifbxjs@precisetext
4290   \ifx\bxjs@do@precisetext\@undefined
4291     \ClassWarning\bxjs@clsname
4292     {The current engine does not support the\MessageBreak

```

```

4293     'precise-text' option\@gobble}
4294   \else
4295     \bxjs@do@precisetext
4296   \fi
4297 \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。

```

4298 \g@addto@macro\bxjs@begin@document@hook{%
4299   \ifx\jsInhibitGlueAtParTop\@empty
4300     \def\bxjs@tmpa{\jsInhibitGlueAtParTop}%
4301     \ifx\everyparhook\bxjs@tmpa
4302       \let\everyparhook\@empty
4303     \fi
4304 \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。

```
4305 \ifnum\bxjs@everyparhook=\bxjs@everyparhook@@modern
```

まず `\everypar` を“乗っ取る” 処理を行う。

```

4306 \let\bxjs@everypar\everypar
4307 \newtoks\everypar
4308 \everypar\bxjs@everypar

```

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

```

4309 \bxjs@everypar{\the\expandafter\everypar\everyparhook}%
4310 \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。

```
4311 \ifbxjs@fancyhdr
```

`\bxjs@adjust@fancyhdr` `fancyhdr` の初期設定に関する変更の処理。 `fancyhdr` 読込完了と `\pagestyle{fancy}` 実行の間で実行されるべき。

```

4312 \@onlypreamble\bxjs@adjust@fancyhdr
4313 \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。

```

4314 \def\bxjs@tmpa{\fancyplain-{\sl\rightmark}\strut}%
4315 \def\bxjs@tmpb{\fancyplain-{\rightmark}\strut}%
4316 \ifx\f@ncyselh\bxjs@tmpa \global\let\f@ncyselh\bxjs@tmpb \fi

```

```

4317 \ifx\@ncyerh\bxjs@tmpa \global\let\@ncyerh\bxjs@tmpb \fi
4318 \ifx\@ncyolh\bxjs@tmpa \global\let\@ncyolh\bxjs@tmpb \fi
4319 \ifx\@ncyorh\bxjs@tmpa \global\let\@ncyorh\bxjs@tmpb \fi
4320 \def\bxjs@tmpa{\fancyplain{}{\s1\leftmark}\strut}%
4321 \def\bxjs@tmpb{\fancyplain{}{\leftmark}\strut}%
4322 \ifx\@ncyelh\bxjs@tmpa \global\let\@ncyelh\bxjs@tmpb \fi
4323 \ifx\@ncyerh\bxjs@tmpa \global\let\@ncyerh\bxjs@tmpb \fi
4324 \ifx\@ncyolh\bxjs@tmpa \global\let\@ncyolh\bxjs@tmpb \fi
4325 \ifx\@ncyorh\bxjs@tmpa \global\let\@ncyorh\bxjs@tmpb \fi
4326 \def\bxjs@tmpa{\rm\thepage\strut}%
4327 \def\bxjs@tmpb{\thepage\strut}%
4328 \ifx\@ncyecf\bxjs@tmpa \global\let\@ncyecf\bxjs@tmpb \fi
4329 \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。

```

4330 \ifx\fullwidth\undefined\else \ifdim\textwidth<\fullwidth
4331 \setlength{\@tempdima}{\fullwidth-\textwidth}%
4332 \edef\bxjs@tmpa{\noexpand\fancyhoffset [EL,OR]{\the\@tempdima}%
4333 } \bxjs@tmpa
4334 \fi\fi
4335 \PackageInfo\bxjs@clsname
4336 {Patch to fancyhdr is applied\@gobble}}

```

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

```

4337 \def\bxjs@pagestyle@hook{%
4338 \@ifpackageloaded{fancyhdr}{%
4339 \bxjs@adjust@fancyhdr
4340 \global\let\bxjs@adjust@fancyhdr\relax
4341 }{}}

```

\pagestyle にフックを入れ込む。

```

4342 \let\bxjs@org@pagestyle\pagestyle
4343 \def\pagestyle{%
4344 \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。

```

4345 \g@addto@macro\bxjs@begin@document@hook{%
4346 \bxjs@pagestyle@hook
4347 \global\let\bxjs@pagestyle@hook\relax}

```

終わり。

```
4348 \fi
```

以上で終わり。

```
4349 %</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

```
4350 %<*standard>
4351 %% 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。

```
4352 \input{bxjsja-minimal.def}
```

`simplejasetup` は `standard` では無効になる。

```
4353 \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
- `XƒTeX`： 0.9999 版 [2013/03] 以上

加えて、以下の要件を定める。

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

■パッケージ読込 利用可能な場合は `etoolbox` パッケージを読み込む。

※ 1.3 版は「`etoolbox` パッケージ」としての最古の版であるらしい。`\AtEndPreamble` はこの版で既に利用可能である。

```
4354 \ifjsWitheTeX
4355 \IfFileExists{etoolbox.sty}{%
```

```

4356 \RequirePackage{etoolbox}[2007/10/08]% v1.3
4357 }{}
4358 \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。

```

4359 \newif\ifbxjs@jp@jismmiv

jis2004 オプションの処理。
4360 \let\bxjs@kv@jis2004@true\bxjs@jp@jismmivtrue
4361 \let\bxjs@kv@jis2004@false\bxjs@jp@jismmivfalse
4362 \define@key{bxjsStd}{jis2004}[true]{%
4363 \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。

```

4364 \newif\ifbxjs@jp@units

units オプションの処理。
4365 \let\bxjs@kv@units@true\bxjs@jp@unitstrue
4366 \let\bxjs@kv@units@false\bxjs@jp@unitsfalse
4367 \define@key{bxjsStd}{units}[true]{%
4368 \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。

```

4369 \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。
4370 \define@key{bxjsStd}{font}{%
4371 \edef\bxjs@jp@font{\bxjs@catopt\bxjs@jp@font{#1}}}

```

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

```

4372 \newif\ifbxjs@jp@strong@cmd \bxjs@jp@strong@cmdtrue

strong-cmd オプションの処理。
4373 \let\bxjs@kv@strongcmd@true\bxjs@jp@strong@cmdtrue
4374 \let\bxjs@kv@strongcmd@false\bxjs@jp@strong@cmdfalse
4375 \define@key{bxjs}{strong-cmd}[true]{\bxjs@set@keyval{strongcmd}{#1}{}}

```

実際の japaram の値を適用する。

```

4376 \def\bxjs@next#1{\bxjs@safe@setkeys{bxjsStd}{#1}}
4377 \expandafter\bxjs@next\expandafter{\jsJaParam}

```

## C.3 共通処理 (1)



```
4378 \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。

```
4379 \@onlypreamble\bxjs@apply@mmiv
4380 \def\bxjs@apply@mmiv{%
4381   \g@addto@macro\@classoptionslist{,jis2004}
4382   \% \@ifpackagewith 判定への対策
4383   \PassOptionsToPackage{jis2004}{otf}
4384   \global\let\bxjs@apply@mmiv\relax}
4385 \ifbxjs@jp@jismmiv \bxjs@apply@mmiv \fi
```

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

```
4386 \if j\jsEngine
4387   \bxjs@jp@unitsfalse
4388 \fi
```

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

```
4389 \ifbxjs@jp@units
4390   \IfFileExists{bxcalc.sty}{%
4391     \RequirePackage{bxcalc}[2018/01/28]%v1.0a
4392     \ifx\usepTeXunits\@undefined
4393       \PackageWarningNoLine\bxjs@clsname
4394         {Cannot support pTeX units (zw etc.), since\MessageBreak
4395         the package 'bxcalc' is too old}%
4396       \bxjs@jp@unitsfalse
4397     \else \usepTeXunits
4398     \fi
4399   }{%else
4400     \PackageWarningNoLine\bxjs@clsname
4401       {Cannot support pTeX units (zw etc.), since\MessageBreak
4402       the package 'bxcalc' is unavailable}%
4403     \bxjs@jp@unitsfalse
4404   }
4405 \fi
```

bxcalc で和文用単位をサポートした場合は、\bxjs@parse@qh の処理は不要になるので無効化する。

```
4406 \ifbxjs@jp@units
4407 \def\bxjs@parse@qh#1{\let\bxjs@tmpb\relax}
4408 \let\bxjs@parse@qh@a\@undefined
4409 \let\bxjs@parse@qh@b\@undefined
4410 \fi
```

\bxjs@let@lenexpr \bxjs@let@lenexpr\CS{(長さ式)}: 長さ式に bxcalc の展開を適用した結果のトークン列を \CS に代入する。

```

4411 \ifbxjs@jp@units
4412 \def\bxjs@let@lenexpr#1#2{%
4413 \edef#1{#2}%
4414 \expandafter\CUXParseExpr\expandafter#1\expandafter{#1}}
4415 \else
4416 \def\bxjs@let@lenexpr{\edef}
4417 \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。

```

4418 \ifbxjs@jp@strong@cmd\jsAtEndOfClass{%
4419 \ifx\strong\undefined\ifx\strongenv\undefined
4420 \newcommand*\strongenv{\jsStrongText}%
4421 \DeclareTextFontCommand{\strong}{\jsStrongText}%
4422 \newcommand*\strongfontdeclare{\jsStrongDeclare}%
4423 \fi\fi
4424 }\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。

```

4425 \@onlypreamble\bxjs@adjust@jafont
4426 \def\bxjs@adjust@jafont#1{%
4427 \ifx\jsJaFont\bxjs@@auto
4428 \bxjs@get@kanjiEmbed
4429 \ifx\bxjs@jaEmbed\relax
4430 \let\bxjs@tmpa\@empty
4431 \else
4432 \let\bxjs@tmpa\bxjs@jaEmbed
4433 \ifx\bxjs@jaVariant\bxjs@@hziv
4434 \bxjs@apply@mmiv
4435 \fi
4436 \fi
4437 \else
4438 \let\bxjs@tmpa\jsJaFont
4439 \fi
4440 \if #1\ifx\bxjs@tmpa\bxjs@@noEmbed
4441 \ClassWarningNoLine\bxjs@clsname
4442 {Option 'jafont=noEmbed' is ignored, because it is\MessageBreak
4443 not available on the current situation}%

```

```

4444 \let\bxjs@tmpa\@empty
4445 \fi\fi
4446 }
4447 \def\bxjs@@auto{auto}
4448 \def\bxjs@noEmbed{noEmbed}
4449 \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。

```

4450 \let\bxjs@jaEmbed\relax
4451 \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。

```

4452 \@onlypreamble\bxjs@get@kanjiEmbed
4453 \def\bxjs@get@kanjiEmbed{%
4454 \begingroup\setbox\z@=\hbox{%
4455 \global\let\bxjs@tmpdo\@empty
4456 \def\bxjs@next##1##2##3{%
4457 \def##1###1##3 ###2\@nil###3\@nnil{%
4458 \ifx$###1$\gdef##2{###2}\fi}%
4459 \g@addto@macro\bxjs@tmpdo{%
4460 \expandafter##1\bxjs@tmpa\@nil##3 \@nil\@nnil}}%
4461 \bxjs@next\bxjs@tmpdo@a\bxjs@g@tmpa{kanjiEmbed}%
4462 \bxjs@next\bxjs@tmpdo@b\bxjs@g@tmpa{jaEmbed}%
4463 \bxjs@next\bxjs@tmpdo@c\bxjs@g@tmpb{kanjiVariant}%
4464 \bxjs@next\bxjs@tmpdo@d\bxjs@g@tmpb{jaVariant}%
4465 %
4466 \global\let\bxjs@g@tmpa\relax
4467 \global\let\bxjs@g@tmpb\relax
4468 \endlinechar\m@ne
4469 \let\do\@makeother\dospecials
4470 \catcode32=10 \catcode12=10 %form-feed
4471 \let\bxjs@tmpa\@empty
4472 \openin\@inputcheck="|kpsewhich updmap.cfg"\relax
4473 \ifeof\@inputcheck\else
4474 \read\@inputcheck to\bxjs@tmpa
4475 \closein\@inputcheck
4476 \fi
4477 \ifx\bxjs@tmpa\@empty\else
4478 \openin\@inputcheck="\bxjs@tmpa"\relax
4479 \@tempwattrue
4480 \loop\if@tempswa
4481 \read\@inputcheck to\bxjs@tmpa
4482 \bxjs@tmpdo
4483 \ifeof\@inputcheck \@tempwafalse \fi
4484 \repeat
4485 \fi
4486 }\endgroup

```

```

4487 \let\bxjs@jaEmbed\bxjs@g@tmpa
4488 \let\bxjs@jaVariant\bxjs@g@tmpb
4489 }

```

`\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。

```

4490 \@onlypreamble\bxjs@resolve@jafont@paren
4491 \def\bxjs@resolve@jafont@paren#1{%
4492   \def\bxjs@tmpb{\let#1}%
4493   \expandafter\bxjs@resolve@jafont@paren@a#1\@nil()\@nil\@nnil#1}
4494 \@onlypreamble\bxjs@resolve@jafont@paren@a
4495 \def\bxjs@resolve@jafont@paren@a#1(#2)#3\@nil#4\@nnil#5{%
4496   \ifx\relax#4\relax \bxjs@tmpb#5%
4497   \else
4498     \edef\bxjs@tmpa{#1\bxjs@jafont@paren{#2}#3}%
4499     \bxjs@tmpb\bxjs@tmpa
4500   \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。

```

4501 \newcommand*\jachar[1]{%
4502   \begingroup

```

`\jsLetHeadChar` で先頭の “文字” を拾ってそれを `\bxjs@jachar` に渡す。

```

4503   \jsLetHeadChar\bxjs@tmpa{#1}%
4504   \ifx\bxjs@tmpa\relax
4505     \ClassWarningNoLine\bxjs@clsname
4506     {Illegal argument given to \string\jachar}%
4507   \else
4508     \expandafter\bxjs@jachar\expandafter{\bxjs@tmpa}%
4509   \fi
4510 \endgroup}

```

`\jsJaChar` を `\jachar` と等価にする。

```

4511 \let\jsJaChar\jachar

```

下請けの `\bxjs@jachar` の実装はエンジンにより異なる。

```

4512 \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。

```

4513 \PassOptionsToPackage{setpagesize=false}{hyperref}

```

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

```

4514 \@onlypreamble\bxjs@fix@hyperref@unicode
4515 \def\bxjs@fix@hyperref@unicode#1{%
4516   \PassOptionsToPackage{bxjs/hook=#1}{hyperref}%

```

```

4517 \@namedef{KV@Hyp@bxjs/hook}##1{%
4518   \KV@Hyp@unicode{##1}%
4519   \def\KV@Hyp@unicode####1{%
4520     \expandafter\ifx\csname if##1\expandafter\endcsname
4521     \csname if####1\endcsname\else
4522     \ClassWarningNoLine\bxjs@clsname
4523     {Blcoked hyperref option 'unicode=####1'}%
4524     \fi
4525   }%
4526 }%
4527 }

```

`\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。

```

4528 \@onlypreamble\jsCheckHyperrefUnicode
4529 \let\jsCheckHyperrefUnicode\empty
4530 \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。

```

4531 \@onlypreamble\bxjs@check@hyperref@unicode
4532 \def\bxjs@check@hyperref@unicode#1{%
4533   \g@addto@macro\jsCheckHyperrefUnicode{%
4534     \@tempwafalse
4535     \begingroup
4536     \expandafter\ifx\csname ifHy@unicode\endcsname\relax
4537     \aftergroup\@tempwatrue \fi
4538     \expandafter\ifx\csname ifHy@unicode\expandafter\endcsname
4539     \csname if#1\endcsname
4540     \aftergroup\@tempwatrue \fi
4541   \endgroup
4542   \if@tempwa\else
4543     \ClassError\bxjs@clsname
4544     {The value of hyperref 'unicode' key is not suitable\MessageBreak
4545     for the present engine (must be #1)}%
4546     {\@ehc}%
4547   \fi}}

```

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

```

4548 \@onlypreamble\bxjs@urgent@special
4549 \@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。

```

4550 \ifbxjs@old@hook@system
4551 \def\bxjs@urgent@special#1{%
4552   \AtBeginDvi{\special{#1}}%
4553   \g@addto@macro\bxjs@begin@document@hook{%
4554     \ifpackageloaded{atbegshi}{%
4555     \begingroup

```

```

4556     \toks\z@{\special{#1}}%
4557     \toks\tw@\expandafter{\AtBegShi@HookFirst}%
4558     \xdef\AtBegShi@HookFirst{\the\toks@\the\toks\tw@}%
4559     \endgroup
4560   }{}%
4561 }%
4562 }

```

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。

```

4563 \else
4564   \def\bxjs@urgent@special#1{%
4565     \bxjs@urgent@special@a
4566     \AddToHook{shipout/firstpage}[pxjahyper/enc]{\special{#1}}
4567   \def\bxjs@urgent@special@a{%
4568     \DeclareHookRule{shipout/firstpage}{pxjahyper/enc}{<}{hyperref}%
4569     \global\let\bxjs@urgent@special@a\relax}
4570 \fi

```

## C.4 pTeX 用設定

```
4571 \if j\jsEngine
```

### ■ 共通命令の実装

```

4572 \newcommand*{\setkanjiskip}{\jsSetKanjiSkip}
4573 \newcommand*{\getkanjiskip}{\jsGetKanjiSkip}
4574 \def\jsApplyKanjiSkip#1{%
4575   \kanjiskip=#1\relax}
4576 \newcommand*{\setxkanjiskip}{\jsSetXKanjiSkip}
4577 \newcommand*{\getxkanjiskip}{\jsGetXKanjiSkip}
4578 \def\jsApplyXKanjiSkip#1{%
4579   \xkanjiskip=#1\relax}

```

\jaJaChar のサブマクロ。

```

4580 \def\bxjs@jachar#1{%
4581   \bxjs@jachar@a#1...\@nil}
4582 \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。

```
4583   \ifx.#2#1%
```

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

```

4584   \else\ifx.#3%
4585     \@tempcnta`#1 \multiply\@tempcnta64
4586     \advance\@tempcnta`#2 \advance\@tempcnta-"3080
4587     \bxjs@jachar@b
4588   \else\ifx.#4%

```

```

4589 \@tempcnta`#1 \multiply\@tempcnta64
4590 \advance\@tempcnta`#2 \multiply\@tempcnta64
4591 \advance\@tempcnta`#3 \advance\@tempcnta-"E2080
4592 \bxjs@jachar@b
4593 \else
4594 \@tempcnta`#1 \multiply\@tempcnta64
4595 \advance\@tempcnta`#2 \multiply\@tempcnta64
4596 \advance\@tempcnta`#3 \multiply\@tempcnta64
4597 \advance\@tempcnta`#4 \advance\@tempcnta-"3C82080
4598 \bxjs@jachar@b
4599 \fi\fi\fi}

```

符号値が \@tempcnta の和文文字を出力する処理。

```

4600 \ifjsWithupTeX
4601 \def\bxjs@jachar@b{\kchar\@tempcnta}
4602 \else
4603 \def\bxjs@jachar@b{%
4604 \ifx\bxUInt\@undefined\else
4605 \bxUInt{\@tempcnta}%
4606 \fi}
4607 \fi

```

和欧文間空白の命令 \jathinspace の実装。

```

4608 \ifbxjs@jaspace@cmd
4609 \def\jathinspace{\hskip\xkanjiskip}
4610 \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。

```

4611 \ifbxjs@jp@jismmiv
4612 \PassOptionsToPackage{prefer2004jis}{pxchfon}
4613 \PassOptionsToPackage{prefer2004jis}{pxbabel}
4614 \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。

```

4615 \let\bxjs@jafont@paren\@firstofone
4616 \let\bxjs@tmpa\jsJaFont
4617 \ifx\bxjs@tmpa\bxjs@@auto
4618 \let\bxjs@tmpa\@empty
4619 \else\ifx\bxjs@tmpa\bxjs@@noEmbed
4620 \def\bxjs@tmpa{noembed}
4621 \fi\fi
4622 \bxjs@resolve@jafont@paren\bxjs@tmpa
4623 \edef\bxjs@tmpa{\bxjs@catopt\bxjs@tmpa\bxjs@jp@font}
4624 \ifx\bxjs@tmpa\@empty\else

```

```

4625 \edef\bxjs@next{%
4626   \noexpand\RequirePackage[\bxjs@tmpa]{pxchfon}[2010/05/12]% v0.5
4627 } \bxjs@next
4628 \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……。)

```

4629 \begingroup
4630 \global\let\bxjs@g@tmpa\relax
4631 \catcode`\|=0 \catcode`\|=12
4632 |def|bxjs@tmpdo#1|@nil{%
4633   |bxjs@tmpdo@a#1|@nil\RequirePackage|@nnil}%
4634 |def|bxjs@tmpdo@a#1\RequirePackage#2|@nnil{%
4635   |ifx$#1$|bxjs@tmpdo@b#2|@nil keyval|@nnil |fi}%
4636 |catcode`\|=0 \catcode`\|=12
4637 \def\bxjs@tmpdo@b#1keyval#2\@nnil{%
4638   \ifx$#2$\else
4639     \xdef\bxjs@g@tmpa{%
4640       \noexpand\PassOptionsToPackage{scale=\jsScale}{otf}}%
4641   \fi}
4642 \@firstofone{%
4643   \catcode10=12 \endlinechar\m@ne
4644   \let\do\@makeother \dospecials \catcode32=10
4645   \openin\@inputcheck=otf.sty\relax
4646   \@tempwattrue
4647   \loop\if@tempwa
4648     \ifeof\@inputcheck \@tempwafalse \fi
4649     \if@tempwa
4650       \read\@inputcheck to\bxjs@next
4651       \expandafter\bxjs@tmpdo\bxjs@next\@nil
4652     \fi
4653   \repeat
4654   \closein\@inputcheck
4655 \endgroup}
4656 \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。



```
4657 \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。

```
4658 \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。

```
4659 \bxjs@if@package@at@least{plautopatch}{2020/05/25}{% v0.9g
```

```
4660 \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。

```
4661 \let\bxjs@avail@hy@unicode=f
```

```
4662 \if \ifx t\bxjs@plautopatch@new T%
```

```
4663 \else\ifbxjs@old@hook@system F\else T\fi\fi T%
```

```
4664 \IfFileExists{pxjahyper-uni.def}{\let\bxjs@avail@hy@unicode=t}{}
```

```
4665 \fi
```

```
4666 \if t\bxjs@avail@hy@unicode
```

```
4667 \ifjsWithupTeX
```

必要条件が満たされていて、かつ upL<sup>A</sup>T<sub>E</sub>X 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なくてよい。

```
4668 \ifbxjs@old@hook@system\else
```

```
4669 \AddToHook{\bxjs@CGHN{package/hyperref/after}}{%
```

```
4670 \input{pxjahyper-uni.def}}
```

```
4671 \fi
```

```
4672 \else
```

必要条件が満たされていて、かつ pL<sup>A</sup>T<sub>E</sub>X である場合の処理。pxjahyper が「pL<sup>A</sup>T<sub>E</sub>X 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 L<sup>A</sup>T<sub>E</sub>X システムで unicode を無効にするのは避けたいので、L<sup>A</sup>T<sub>E</sub>X 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「L<sup>A</sup>T<sub>E</sub>X がカーネルが新しい」かを判定する。

```
4673 \let\bxjs@avail@hy@unicode=f
```

```
4674 \ifx t\bxjs@plautopatch@new
```

```
4675 \bxjs@if@format@at@least{2023/06/01}{\let\bxjs@avail@hy@unicode=t}{}
```

```
4676 \fi
```

```
4677 \fi
```

```
4678 \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。

```
4679 \if f\bxjs@avail@hy@unicode
4680   \PassOptionsToPackage{unicode=false}{hyperref}
4681 \fi
4682 \fi
```

tounicode special 命令を出力する。

```
4683 \if \ifx\bxjs@driver@given\bxjs@driver@@dvipdfmx T%
4684   \else\ifjsWithpTeXng T\else F\fi\fi T%
4685   \IfFileExists{pxjahyper-enc.sty}{\@tempwatrue}{\@tempwafalse}
4686   \if@tempswa
4687     \RequirePackage{pxjahyper-enc}[2020/10/05]%v0.6
4688     \ifbxjs@bigcode\else \suppressbigcode \fi
4689   \else
4690     \ifnum\jis"2121="A1A1 %euc
4691       \bxjs@urgent@special{pdf:tounicode EUC-UCS2}
4692     \else\ifnum\jis"2121="8140 %sjis
4693       \bxjs@urgent@special{pdf:tounicode 90ms-RKSJ-UCS2}
4694     \else\ifnum\jis"2121="3000 %uptex
4695       \ifbxjs@bigcode
4696         \bxjs@urgent@special{pdf:tounicode UTF8-UTF16}
4697         \PassOptionsToPackage{bigcode}{pxjahyper}
4698       \else
4699         \bxjs@urgent@special{pdf:tounicode UTF8-UCS2}
4700         \PassOptionsToPackage{nobigcode}{pxjahyper}
4701       \fi
4702     \fi\fi\fi
4703     \let\bxToUnicodeSpecialDone=t
4704   \fi
4705 \fi
```

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

```
4706 \ifx f\bxjs@enablejfam\else
4707   \@enablejfamtrue
4708 \fi
```

実際に和文用の数式ファミリの設定を行う。

```
4709 \if@enablejfam
4710   \DeclareSymbolFont{mincho}{\jsc@JYn}{mc}{m}{n}
4711   \DeclareSymbolFontAlphabet{\mathmc}{mincho}
4712   \SetSymbolFont{mincho}{bold}{\jsc@JYn}{gt}{m}{n}
4713   \jfam\symmincho
4714   \DeclareMathAlphabet{\mathgt}{\jsc@JYn}{gt}{m}{n}
4715   \g@addto@macro\bxjs@begin@document@hook{%
4716     \ifx\reDeclareMathAlphabet\undefined\else
```

`\reDeclareMathAlphabet` を適用した数式英字フォント命令は L<sup>A</sup>T<sub>E</sub>X の通常の命令とは定

義文の形が異なる。このため `bm` パッケージを読み込んで `\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の暫定対応として、`bm` が読み込まれた場合は数式英字フォントの和文連動を無効にする。

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

```
4717 \ifpackageloaded{bm}{\fi}%else
4718 \reDeclareMathAlphabet{\mathrm}{\@mathrm}{\@mathmc}%
4719 \reDeclareMathAlphabet{\mathbf}{\@mathbf}{\@mathgt}%
4720 \reDeclareMathAlphabet{\mathsf}{\@mathsf}{\@mathgt}%
4721 }%
4722 \fi}
4723 \fi
```

## C.5 pdfTeX 用設定：CJK + bxcjkjatype

```
4724 \else\if \if p\jsEngine T\else\if n\jsEngine T\else F\fi\fi T
```

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

※ Pandoc モードでは `autotilde` を指定しない。

```
4725 \bxjs@adjust@jafont{f}
4726 \let\bxjs@jafont@paren\@firstofone
4727 \bxjs@resolve@jafont@paren\bxjs@tmpa
4728 \edef\bxjs@tmpa{\bxjs@catopt\bxjs@tmpa\bxjs@jp@font}
4729 \edef\bxjs@tmpa{\bxjs@catopt\bxjs@tmpa{whole}}
4730 \ifx\bxjs@jadriver\bxjs@@pandoc\else
4731 \edef\bxjs@tmpa{\bxjs@catopt\bxjs@tmpa{autotilde}}
4732 \fi
4733 \edef\bxjs@next{%
4734 \noexpand\RequirePackage[\bxjs@tmpa]{bxcjkjatype}[2013/10/15]% v0.2c
4735 }\bxjs@next
4736 \bxjs@cjk@loaded
```

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

```
4737 \ifbxjs@hyperref@enc
4738 \PassOptionsToPackage{unicode}{hyperref}
4739 \fi
```

`\hypersetup` 命令で (CJK\* 環境に入れなくても) 日本語文字を含む文書情報を設定できるようにするための細工。

※ `bxcjkjatype` を `whole` 付きで使っていることが前提。

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

```
4740 \ifx\bxcjkjatypeHyperrefPatchDone\@undefined
4741 \begingroup
```

```

4742 \CJK@input{UTF8.bdg}
4743 \endgroup
4744 \g@addto@macro\pdfstringdefPreHook{%
4745 \@nameuse{CJK@UTF8Binding}%
4746 }
4747 \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。

```

4748 \ifx\bxckjatypeHyperrefPatchDone\@undefined
4749 \g@addto@macro\pdfstringdefPreHook{%
4750 \ifx~\bxjs@CJKtilde
4751 \let\bxjs@org@LetUnexpandableSpace\HyPsd@LetUnexpandableSpace
4752 \let\HyPsd@LetUnexpandableSpace\bxjs@LetUnexpandableSpace
4753 \let~\@empty
4754 \fi
4755 }
4756 \def\bxjs@CJKtilde{\CJKecglue\ignorespaces}
4757 \def\bxjs@tildecmd{~}
4758 \def\bxjs@LetUnexpandableSpace#1{%
4759 \def\bxjs@tmpa{#1}\ifx\bxjs@tmpa\bxjs@tildecmd\else
4760 \bxjs@org@LetUnexpandableSpace#1%
4761 \fi}
4762 \fi

```

### ■ 共通命令の実装

```

4763 \newskip\jsKanjiSkip
4764 \newskip\jsXKanjiSkip
4765 \ifx\CJKecglue\@undefined
4766 \def\CJKtilde{\CJK@global\def~{\CJKecglue\ignorespaces}}
4767 \fi
4768 \newcommand*\setkanjiskip{\jsSetKanjiSkip}
4769 \newcommand*\getkanjiskip{\jsGetKanjiSkip}
4770 \newcommand*\autospacing{\jsEnableKanjiSkip}
4771 \newcommand*\noautospacing{\jsDisableKanjiSkip}
4772 \protected\def\bxjs@CJKglue{\hskip\jsKanjiSkip}
4773 \def\jsApplyKanjiSkip#1{%
4774 \jsKanjiSkip=#1\relax
4775 \let\CJKglue\bxjs@CJKglue}
4776 \newcommand*\setxkanjiskip{\jsSetXKanjiSkip}
4777 \newcommand*\getxkanjiskip{\jsGetXKanjiSkip}
4778 \newcommand*\autoxspacing{\jsEnableXKanjiSkip}
4779 \newcommand*\noautoxspacing{\jsDisableXKanjiSkip}
4780 \protected\def\bxjs@CJKecglue{\hskip\jsXKanjiSkip}
4781 \def\jsApplyXKanjiSkip#1{%
4782 \jsXKanjiSkip=#1\relax
4783 \let\CJKecglue\bxjs@CJKecglue}

```

\jachar のサブマクロの実装。

```

4784 \def\bxjs@jachar#1{%

```

```
4785 \CJKforced{#1}}
```

和欧文間空白の命令 `\jathinspace` の実装。

```
4786 \ifbxjs@jaspace@cmd
4787 \protected\def\jathinspace{\CJKkecglue}
4788 \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。

```
4789 \ifx t\bxjs@enablejfam
4790 \ClassWarningNoLine\bxjs@clsname
4791 {You cannot use 'enablejfam=true', since the\MessageBreak
4792 CJK package does not support Japanese math}
4793 \fi
```

## C.6 Xe<sub>3</sub>TeX 用設定：xeCJK + zxjatype

```
4794 \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。

```
4795 \RequirePackage{zxjatype}
4796 \PassOptionsToPackage{no-math}{fontspec}%!
4797 \PassOptionsToPackage{xetex}{graphicx}%!
4798 \PassOptionsToPackage{xetex}{graphics}%!
4799 \ifx\zxJaFamilyName\undefined
4800 \ClassError\bxjs@clsname
4801 {xeCJK or zxjatype is too old}\@ehc
4802 \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。

```
4803 \bxjs@adjust@jafont{f}
4804 \let\bxjs@jafont@paren@gobble
4805 \bxjs@resolve@jafont@paren\bxjs@tmpa
4806 \edef\bxjs@tmpa{\bxjs@catopt\bxjs@tmpa\bxjs@jp@font}
4807 \ifx\bxjs@tmpa\empty
4808 \setCJKmainfont[BoldFont=HaranoAjiGothic-Medium.otf]{HaranoAjiMincho-
Regular.otf}
4809 \setCJKsansfont[BoldFont=HaranoAjiGothic-Medium.otf]{HaranoAjiGothic-
Medium.otf}
4810 \else
4811 \edef\bxjs@next{%
4812 \noexpand\RequirePackage[\bxjs@tmpa]{zxjafont}[2013/01/28]% v0.2a
4813 }\bxjs@next
4814 \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。

※ TeX の小数の精度は十進で 4 桁までしか保証されないので、\stricmp 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、\stricmp 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。

```
4815 \ifx\stricmp\undefined\else % 未定義なら条件を満たさない
4816 \ifnum\stricmp{\the\XeTeXversion\XeTeXrevision}{0.99992}>\m@ne
4817 \ifbxjs@hyperref@enc
4818 \PassOptionsToPackage{unicode}{hyperref}
4819 \fi
4820 \fi
4821 \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。

```
4822 \AtEndOfClass{%
4823 \def\@gnewline #1{%
4824 \ifvmode \@nolnerr
4825 \else
4826 \unskip \reserved@e {\reserved@f#1}\nobreak \hfil \break \null
4827 \nobreak \hskip-1sp\hskip1sp\relax
4828 \ignorespaces
4829 \fi}
4830 }
```

### ■共通命令の実装

```
4831 \newskip\jsKanjiSkip
4832 \newskip\jsXKanjiSkip
4833 \ifx\CJKecglue\undefined
4834 \def\CJKtilde{\CJK@global\def~{\CJKecglue\ignorespaces}}
4835 \fi
4836 \newcommand*{\setkanjiskip}{\jsSetKanjiSkip}
4837 \newcommand*{\getkanjiskip}{\jsGetKanjiSkip}
```

```

4838 \newcommand*\autospacing{\jsEnableKanjiSkip}
4839 \newcommand*\noautospacing{\jsDisableKanjiSkip}
4840 \protected\def\bxjs@CJKglue{\hskip\jsKanjiSkip}
4841 \def\jsApplyKanjiSkip#1{%
4842   \jsKanjiSkip=#1\relax
4843   \xeCJKsetup{CJKglue={\bxjs@CJKglue}}}
4844 \newcommand*\setxkanjiskip{\jsSetXKanjiSkip}
4845 \newcommand*\getxkanjiskip{\jsGetXKanjiSkip}
4846 \newcommand*\autoxspacing{\jsEnableXKanjiSkip}
4847 \newcommand*\noautoxspacing{\jsDisableXKanjiSkip}
4848 \protected\def\bxjs@CJKeclue{\hskip\jsXKanjiSkip}
4849 \def\jsApplyXKanjiSkip#1{%
4850   \jsXKanjiSkip=#1\relax
4851   \xeCJKsetup{CJKeclue={\bxjs@CJKeclue}}}

```

`\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。

```

4852 \ifx\mcfamily\undefined
4853   \protected\def\mcfamily{\CJKfamily{\CJKrdefault}}
4854   \protected\def\gtfamily{\CJKfamily{\CJKsfdefault}}
4855 \fi

```

`\jachar` のサブマクロの実装。

```

4856 \def\bxjs@jachar#1{%
4857   \xeCJKDeclareCharClass{CJK}{`#1}\relax
4858   #1}

```

`\jathinspace` の実装。

```

4859 \ifbxjs@jaspace@cmd
4860   \protected\def\jathinspace{\CJKeclue}
4861 \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。

```

4862 \ifx t\bxjs@enablejfam
4863   \@enablejfamtrue
4864 \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。

```

4865 \if@enablejfam
4866   \xeCJKsetup{CJKmath=true}
4867 \fi

```

## C.7 Lua<sub>T</sub><sub>E</sub>X 用設定：Lua<sub>T</sub><sub>E</sub>X-ja

```

4868 \else\if l\jsEngine

```

■Lua<sub>T</sub><sub>E</sub>X-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。

```
4869 \let\zw\@undefined
4870 \RequirePackage{luatexja}
4871 \edef\bxjs@next{%
4872   \noexpand\RequirePackage[scale=\jsScale]{luatexja-fontspec}[2015/08/26]%
4873 }\bxjs@next
```

\set@fontsize へのパッチ適用を再度行う。

```
4874 \bxjs@patch@set@fontsize
```

フォント代替の明示的定義。

```
4875 \DeclareFontShape{JY3}{mc}{m}{it}{<->ssub*mc/m/n}{ }
4876 \DeclareFontShape{JY3}{mc}{m}{sl}{<->ssub*mc/m/n}{ }
4877 \DeclareFontShape{JY3}{mc}{m}{sc}{<->ssub*mc/m/n}{ }
4878 \DeclareFontShape{JY3}{gt}{m}{it}{<->ssub*gt/m/n}{ }
4879 \DeclareFontShape{JY3}{gt}{m}{sl}{<->ssub*gt/m/n}{ }
4880 \DeclareFontShape{JY3}{mc}{bx}{it}{<->ssub*gt/m/n}{ }
4881 \DeclareFontShape{JY3}{mc}{bx}{sl}{<->ssub*gt/m/n}{ }
4882 \DeclareFontShape{JY3}{gt}{bx}{it}{<->ssub*gt/m/n}{ }
4883 \DeclareFontShape{JY3}{gt}{bx}{sl}{<->ssub*gt/m/n}{ }
4884 \DeclareFontShape{JY3}{mc}{b}{n}{<->ssub*mc/bx/n}{ }
4885 \DeclareFontShape{JY3}{mc}{b}{it}{<->ssub*mc/bx/n}{ }
4886 \DeclareFontShape{JY3}{mc}{b}{sl}{<->ssub*mc/bx/n}{ }
4887 \DeclareFontShape{JY3}{gt}{b}{n}{<->ssub*gt/bx/n}{ }
4888 \DeclareFontShape{JY3}{gt}{b}{it}{<->ssub*gt/bx/n}{ }
4889 \DeclareFontShape{JY3}{gt}{b}{sl}{<->ssub*gt/bx/n}{ }
4890 \DeclareFontShape{JT3}{mc}{m}{it}{<->ssub*mc/m/n}{ }
4891 \DeclareFontShape{JT3}{mc}{m}{sl}{<->ssub*mc/m/n}{ }
4892 \DeclareFontShape{JT3}{mc}{m}{sc}{<->ssub*mc/m/n}{ }
4893 \DeclareFontShape{JT3}{gt}{m}{it}{<->ssub*gt/m/n}{ }
4894 \DeclareFontShape{JT3}{gt}{m}{sl}{<->ssub*gt/m/n}{ }
4895 \DeclareFontShape{JT3}{mc}{bx}{it}{<->ssub*gt/m/n}{ }
4896 \DeclareFontShape{JT3}{mc}{bx}{sl}{<->ssub*gt/m/n}{ }
4897 \DeclareFontShape{JT3}{gt}{bx}{it}{<->ssub*gt/m/n}{ }
4898 \DeclareFontShape{JT3}{gt}{bx}{sl}{<->ssub*gt/m/n}{ }
4899 \DeclareFontShape{JT3}{mc}{b}{n}{<->ssub*mc/bx/n}{ }
4900 \DeclareFontShape{JT3}{mc}{b}{it}{<->ssub*mc/bx/n}{ }
4901 \DeclareFontShape{JT3}{mc}{b}{sl}{<->ssub*mc/bx/n}{ }
4902 \DeclareFontShape{JT3}{gt}{b}{n}{<->ssub*gt/bx/n}{ }
4903 \DeclareFontShape{JT3}{gt}{b}{it}{<->ssub*gt/bx/n}{ }
4904 \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。

```
4905 \bxjs@adjust@jafont{t}
4906 \ifx\bxjs@tmpa\bxjs@@noEmbed
4907   \def\bxjs@tmpa{noembed}
4908 \fi
4909 \let\bxjs@jafont@paren@gobble
4910 \bxjs@resolve@jafont@paren\bxjs@tmpa
4911 \edef\bxjs@tmpa{\bxjs@catopt\bxjs@tmpa\bxjs@jp@font}
4912 \ifx\bxjs@tmpa@empty
4913   \defaultjfontfeatures{ Kerning=Off }
4914   \setmainjfont [BoldFont=HaranoAjiGothic-Medium.otf,JFM=ujis]{HaranoAjiMincho-
     Regular.otf}
4915   \setsansjfont [BoldFont=HaranoAjiGothic-Medium.otf,JFM=ujis]{HaranoAjiGothic-
     Medium.otf}
4916 \else
4917   \edef\bxjs@next{%
4918     \noexpand\RequirePackage[\bxjs@tmpa]{luatexja-preset}%
4919   }\bxjs@next
4920 \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。

```
4921 \@ifpackagelater{luatexja}{2016/03/31}{}{%else
4922 \DeclareRobustCommand\rmfamily
4923   {\not@math@alphabet\rmfamily\mathrm
4924   \romanfamily\rmdefault\kanjifamily\mcdefault\selectfont}
4925 \DeclareRobustCommand\sffamily
4926   {\not@math@alphabet\sffamily\mathsf
4927   \romanfamily\sfdefault\kanjifamily\gtdefault\selectfont}
4928 \DeclareRobustCommand\ttfamily
4929   {\not@math@alphabet\ttfamily\mathtt
4930   \romanfamily\ttdefault\kanjifamily\gtdefault\selectfont}
4931 }
4932 \long\def\jttdefault{\gtdefault}
4933 \unless\ifx\@ltj@match@familytrue\@undefined
4934   \@ltj@match@familytrue
4935 \fi
4936 \g@addto@macro\bxjs@begin@document@hook{%
4937   \reDeclareMathAlphabet{\mathrm}{\mathrm}{\mathmc}%
4938   \reDeclareMathAlphabet{\mathbf}{\mathbf}{\mathgt}%
4939   \reDeclareMathAlphabet{\mathsf}{\mathsf}{\mathgt}}%
4940 \bxjs@if@sf@default{%
4941   \renewcommand\kanjifamilydefault{\gtdefault}}
```

## ■和文パラメタの設定

4942 % 次の3つは既定値の通り

```
4943 %\ltjsetparameter{prebreakpenalty={` ,1000}}
```

```

4944 %\ltjsetparameter{postbreakpenalty={`“,10000}}
4945 %\ltjsetparameter{prebreakpenalty={`” ,10000}}
4946 \ltjsetparameter{jaxspmde={`!,1}}
4947 \ltjsetparameter{jaxspmde={`⎯,2}}
4948 \ltjsetparameter{alxspmde={`+,3}}
4949 \ltjsetparameter{alxspmde={`%,3}}

```

■段落頭でのグルー挿入禁止 基本的に現状の `ltjs*` クラスの処理に合わせる。

※`\jsInhibitGlueAtParTop` は使わない。

`\ltjfakeparbegin` 現在の LuaTeX-jā 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。

```

4950 \ifx\ltjfakeparbegin\@undefined
4951   \protected\def\ltjfakeparbegin{%
4952     \ifhmode
4953       \relax\directlua{%
4954         luatexja.jfmglue.create_beginpar_node()}
4955     \fi}
4956 \fi

```

`ltjs*` クラスの定義と同等になるようにパッチを当てる。

```

4957 \unless\ifnum\bxjs@everyparhook=\bxjs@everyparhook@@none
4958 \begingroup
4959   \let%\@percentchar \def\@#1{[[\detokenize{#1}]]}
4960   \@gobble\if\def\bxjs@tmpa{\@{\everypar{}}\fi}
4961   \ifnum\bxjs@everyparhook=\bxjs@everyparhook@@compat
4962     \@gobble\if\def\bxjs@tmpa{\@{\everypar{\everyparhook}}\fi}\fi
4963   \directlua{
4964     local function patchcmd(cs, code, from, to)
4965       tex.sprint(code:gsub(from:gsub("%W", "%\\%\\%0"), "%0"..to)
4966         :gsub("macro:", \@gdef..cs, 1):gsub("->", "{", 1).."}")
4967     end
4968     patchcmd(\@xsect, [[\meaning\xsect]],
4969       \@{\hskip-\@tempskipa}, \@{\ltjfakeparbegin})
4970     patchcmd(\@item, [[\meaning\@item]],
4971       \bxjs@tmpa, \@{\ltjfakeparbegin})
4972   \endgroup
4973 \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。

```

4974 \ifbxjs@hyperref@enc
4975   \PassOptionsToPackage{unicode}{hyperref}
4976   \bxjs@check@hyperref@unicode{true}
4977 \fi

```

■共通命令の実装

```

4978 \newcommand*{\setkanjiskip}{\jsSetKanjiSkip}

```

```

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

```

\jachar のサブマクロの実装。

```

4994 \def\bxjs@jachar#1{%
4995   \ltjjachar`#1\relax}

```

\jathinspace の実装。

```

4996 \ifbxjs@jaspacespace@cmd
4997 \protected\def\jathinspace{%
4998   \hskip\ltjgetparameter{xkanjiskip}\relax}
4999 \fi

```

■和文数式ファミリ Lua<sub>T</sub><sub>E</sub>X-ja では和文数式ファミリは常に有効で、既にこの時点で必要な設定は済んでいる。従って @enablejfam は常に真になる。

```

5000 \ifx f\bxjs@enablejfam
5001   \ClassWarningNoLine\bxjs@clsname
5002     {You cannot use 'enablejfam=false', since the\MessageBreak
5003     LuaTeX-ja always provides Japanese math families}
5004 \fi

```

## C.8 共通処理 (2)

```
5005 \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 5006 \ifx\DeclareFixJFMCJKTextFontCommand\@undefined
5007 \DeclareJaTextFontCommand{\textmc}{\mcfamily}
5008 \DeclareJaTextFontCommand{\textgt}{\gtfamily}
5009 \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。

```
5010 \ifx\mathmc\@undefined
```

```

5011 \DeclareJaMathFontCommand{\mathmc}{\mcfamily}
5012 \DeclareJaMathFontCommand{\mathgt}{\gtfamily}
5013 \fi

```

以上で終わり。

```
5014 %</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。

```

5015 %<*modern>
5016 \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。

```

5017 \ifnum0\if x\jsEngine1\fi\if 1\jsEngine1\fi=\z@
5018 \def\encodingdefault{T1}%
5019 \input{t1enc.def}%
5020 \fontencoding\encodingdefault\selectfont
5021 \fi

```

基本フォントを Latin Modern フォントファミリーに変更する。

※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。

```

5022 \ifnum0\if x\jsEngine1\fi\if 1\jsEngine1\fi=\z@
5023 \renewcommand{\rmdefault}{lmr}
5024 \renewcommand{\sfdefault}{lms}
5025 \renewcommand{\ttdefault}{lmtt}
5026 \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。

```

5027 \DeclareFontShape{OMX}{cmex}{m}{n}{%
5028 <-7.5>cmex7<7.5-8.5>cmex8%
5029 <8.5-9.5>cmex9<9.5->cmex10}{}%
5030 \expandafter\let\csname OMX/cmex/m/n/10\endcsname\relax

```

`amsmath` 読込時に上書きされるのを防ぐ。

```
5031 \def\cmex@opt{10}
```

### D.2 fixltx2e 読込

※ `fixltx2e` 廃止前の L<sup>A</sup>T<sub>E</sub>X カーネルの場合。

```
5032 \ifx\@IncludeInRelease\@undefined
5033 \RequirePackage{fixltx2e}
5034 \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。

```
5035 \RequirePackage{bxjscjkat}
```

### D.4 完了

おしまい。

```
5036 %</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

```
5037 %<*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。

```
5038 \if x\jsEngine
5039 \PassOptionsToPackage{nospace}{xeCJK}
5040 \fi
```

standard ドライバの設定を引き継ぐ。

```
5041 \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。

- p<sub>TEX</sub> 系も含めて全てのエンジン種別で  $\epsilon$ -<sub>TEX</sub> 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。

```
5042 \RequirePackage{bxjspandoc}
```

$\epsilon$ -<sub>TEX</sub> ではない場合に警告を出す。

```

5043 \ifjswitheTeX\else
5044 \ClassWarningNoLine{bxjs@clsname
5045 {!!!!!!! WARNING !!!!!!!\MessageBreak
5046 This engine does not support e-TeX extension!\MessageBreak
5047 Some feature might not work properly}
5048 \fi

```

`\ifbxjs@bxghost@available` [スイッチ] `bxghost` パッケージが利用できるか。

```

5049 \newif\ifbxjs@bxghost@available
5050 \ifjswitheTeX
5051 \RequirePackage{pdftexcmds}[2009/09/22]% v0.5
5052 \IfFileExists{bxghost.sty}{%
5053 \bxjs@bxghost@availabletrue
5054 \@namedef{bxjs@bgbv/79E70A0991967E27981832C84DB5DF99}{1}%v0.2.0
5055 \ifx\pdf@filemdfivesum\undefined\else
5056 \expandafter\ifx\cname bxjs@bgbv/\pdf@filemdfivesum{bxghost.sty}%
5057 \endcsname\relax\else \bxjs@bxghost@availablefalse \fi
5058 \fi
5059 }{}
5060 \fi

```

その他の依存パッケージを読み込む。

```

5061 \RequirePackage{iftex}[2013/04/04]% v0.2
5062 \ifjswitheTeX
5063 \RequirePackage{etoolbox}[2010/08/21]% v2.0
5064 \RequirePackage{filehook}[2011/10/12]% v0.5d
5065 \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。

```

5066 \newif\ifbxjs@jp@fix@strong \bxjs@jp@fix@strongtrue
5067 \ifjswitheTeX
5068 \ifx\fix@strong\undefined\fix@strongfalse\else
5069 \fix@strong オプションの処理。
5070 \let\bxjs@kv@fixstrong@true\bxjs@jp@fix@strongtrue
5071 \let\bxjs@kv@fixstrong@false\bxjs@jp@fix@strongfalse
5072 \define@key{bxjsPan}{fix-strong}[true]{%
5073 \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。

```

5074 \newif\ifbxjs@jp@fix@code \bxjs@jp@fix@codetrue
5075 \ifjswitheTeX
5076 \ifx\fix@code\undefined\fix@codefalse\else
5077 \fix@code オプションの処理。
5078 \let\bxjs@kv@fixcode@true\bxjs@jp@fix@codetrue
5079 \let\bxjs@kv@fixcode@false\bxjs@jp@fix@codefalse
5080 \define@key{bxjsPan}{fix-code}[true]{%
5081 \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。

```
5076 \chardef\bxjs@jp@strong=0
```

`strong` オプションの処理。

```
5077 \def\bxjs@kv@strong@bold{\chardef\bxjs@jp@strong=0 }
5078 \def\bxjs@kv@strong@sans{\chardef\bxjs@jp@strong=1 }
5079 \def\bxjs@kv@strong@boldsans{\chardef\bxjs@jp@strong=2 }
5080 \define@key{bxjsPan}{strong}{%
5081 \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 5082 \newif\ifbxjs@jp@or@indent \bxjs@jp@or@indenttrue
\ifbxjs@jp@or@block@heading 5083 \newif\ifbxjs@jp@or@secnumdepth \bxjs@jp@or@secnumdepthtrue
5084 \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。

```
5085 \define@key{bxjsPan}{_plus}[]{%
5086 \bxjs@jp@or@indentfalse
5087 \bxjs@jp@or@secnumdepthfalse
5088 \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。

```
5089 \let\bxjs@kv@orindent@true\bxjs@jp@or@indenttrue
5090 \let\bxjs@kv@orindent@false\bxjs@jp@or@indentfalse
5091 \define@key{bxjsPan}{or-indent}[true]{%
5092 \bxjs@set@keyval{orindent}{#1}{}}
5093 \let\bxjs@kv@orsecnumdepth@true\bxjs@jp@or@secnumdepthtrue
5094 \let\bxjs@kv@orsecnumdepth@false\bxjs@jp@or@secnumdepthfalse
5095 \define@key{bxjsPan}{or-secnumdepth}[true]{%
5096 \bxjs@set@keyval{orsecnumdepth}{#1}{}}
5097 \let\bxjs@kv@orblockheading@true\bxjs@jp@or@block@headingtrue
5098 \let\bxjs@kv@orblockheading@false\bxjs@jp@or@block@headingfalse
5099 \define@key{bxjsPan}{or-block-heading}[true]{%
5100 \bxjs@set@keyval{orblockheading}{#1}{}}
```

実際の `japaram` の値を適用する。

```
5101 \def\bxjs@next#1{\bxjs@safe@setkeys{bxjsPan}{#1}}
5102 \expandafter\bxjs@next\expandafter{\jsJaParam}
```

### E.3 dupload システム

**TODO:** 新しいカーネルで利用可能な機構での代替を検討する。カーネルへのパッチは排除したいので。

パッケージが重複して読み込まれたときに “option clash” の検査をスキップする。この時に何らかのコードを実行させることができる。

`\bxjs@set@dupload@proc \bxjs@set@dupload@proc{〈ファイル名〉}{〈定義本体〉}`： 指定の名前の特定のファイルの読込が `\@filewithoptions` で指示されて、しかもそのファイルが読込済である場合に、オプション重複検査をスキップして、代わりに〈定義本体〉のコードを実行する。このコード中で #1 は渡されたオプション列のテキストに置換される。

```

5103 \@onlypreamble\bxjs@set@dupload@proc
5104 \def\bxjs@set@dupload@proc#1{%
5105   \expandafter\bxjs@set@dupload@proc@a\csname bxjs@dlp/#1\endcsname}
5106 \@onlypreamble\bxjs@set@dupload@proc@a
5107 \def\bxjs@set@dupload@proc@a#1{%
5108   \@onlypreamble#1\def#1##1}
5109 \def\bxjs@unset@dupload@proc#1{%
5110   \bxjs@cslet{bxjs@dlp/#1}\@undefined}

```

`\@if@options \@if@options` の再定義。

```

5111 \@onlypreamble\bxjs@org@if@options
5112 \let\bxjs@org@if@options\@if@options
5113 \@onlypreamble\bxjs@org@reset@options
5114 \let\bxjs@org@reset@options\relax
5115 \def\@if@options#1#2#3{%
5116   \let\bxjs@next\@secondoftwo
5117   \def\bxjs@tmpa{#1}\def\bxjs@tmpb{\@currentx}%
5118   \ifx\bxjs@tmpa\bxjs@tmpb
5119     \expandafter\ifx\csname bxjs@dlp/#2.#1\endcsname\relax\else
5120     \let\bxjs@next\@firstoftwo \fi
5121   \fi
5122   \bxjs@next\bxjs@do@dupload@proc\bxjs@org@if@options{#1}{#2}{#3}}
5123 \g@addto@macro\bxjs@begin@document@hook{%
5124   \let\@if@options\bxjs@org@if@options}
5125 \@onlypreamble\bxjs@do@dupload@proc
5126 \def\bxjs@do@dupload@proc#1#2#3{%
5127   \ifx\bxjs@org@reset@options\relax
5128     \let\bxjs@org@reset@options\@reset@options
5129   \fi
5130   \bxjs@csletcs{bxjs@next}{bxjs@dlp/#2.#1}%
5131   \def\@reset@options{%
5132     \let\@reset@options\bxjs@org@reset@options
5133     \@reset@options
5134     \bxjs@next{#3}}%
5135   \@firstoftwo}

```

## E.4 lang 変数

`lang=ja` という言語指定が行われると、2.12 版より前の Pandoc はこれに対応していなかったため不完全な Babel や Polyglossia の設定を出力してしまっていた。現在では `lang=ja` 指定について正しく 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 `XYTeX` か `LuaTeX` の場合に利用されていた。

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

```
5136 \let\bxjs@polyglossia@options\relax
```

エンジンが `XYTeX` か `LuaTeX` の場合が対象になる。

※この場合 `etoolbox` が使用可能になっている。

```
5137 \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。

```
5138 \pandocSkipLoadPackage{polyglossia}
```

```
5139 \bxjs@set@dupload@proc{polyglossia.sty}{%
```

```
5140 \bxjs@unset@dupload@proc{polyglossia.sty}{%
```

```
5141 \ClassWarning\bxjs@clsname
```

```
5142 {Package polyglossia is requested}%
```

```
5143 \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。

```
5144 \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。

```
5145 \ifboolexpr{test{\ifblank{##2}}or test{\ifstrequal{##2}{japanese}}}{%
```

```
5146 \ClassWarning\bxjs@clsname
```

```
5147 {Main language is 'japanese', thus fallback\MessageBreak
```

```
5148 definitions will be employed}%
```

```
5149 \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。

```
5150 }{%else
```

```
5151 \ClassWarning\bxjs@clsname
```

```
5152 {Main language is '##2',\MessageBreak
```

```
5153 thus polyglossia will be loaded}%
```

```
5154 \csundef{ver@polyglossia.sty}%
```

```

5155     \edef\bxjs@next{%
5156         \noexpand\RequirePackage[\bxjs@polyglossia@options]{polyglossia} []%
5157     }\bxjs@next
5158     \setmainlanguage[##1]{##2}%
5159 }}}
```

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。

```

5160 \g@addto@macro\bxjs@endpreamble@hook{%
5161     \ifx\bxjs@polyglossia@options\relax
5162         \csundef{ver@polyglossia.sty}%
5163     \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。

```

5164 \@onlypreamble\bxjs@pandoc@polyglossia@ja
5165 \def\bxjs@pandoc@polyglossia@ja{%
5166     \renewcommand*\setmainlanguage[2] []{%
5167         \newcommand*\setotherlanguage[2] []{%
5168             \ifblank{##2}{-}{%else
5169                 \cslet{##2}\@empty \cslet{end##2}\@empty
5170                 \cslet{text##2}\@firstofone}}%
5171         \newcommand*\setotherlanguages[2] []{%
5172             \@for\bxjs@tmpa:={##2}\do{%
5173                 \setotherlanguage{\bxjs@tmpa}}%
5174         }
```

Polyglossia の読込済マークは外れるようにしておく。

```

5174     \let\bxjs@polyglossia@options\relax}%
5175 \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。

※ Xe<sub>La</sub>TeX では 2.15 版で、Lua<sub>La</sub>TeX 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。

```

5176 \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。

```

5177 \pandocSkipLoadPackage{babel}
5178 \bxjs@set@dupload@proc{babel.sty}{%
5179     \bxjs@unset@dupload@proc{babel.sty}%
5180     \ClassWarning\bxjs@clsname
5181     {Package babel is requested}%
5182 }
```

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。

```

5182     \@tempwafalse \let\bxjs@babel@options\@empty
```

```

5183 \def\bxjs@tmpb{main=}%
5184 \@for\bxjs@tmpa:=#1\do{%
5185   \ifx\bxjs@tmpa\bxjs@tmpb \def\bxjs@tmpa{main=japanese}\fi
5186   \edef\bxjs@babel@options{\bxjs@babel@options,\bxjs@tmpa}%
5187   \bxjs@cslet{ver@babel.sty}\@undefined
5188   \edef\bxjs@next{%
5189     \noexpand\RequirePackage[\bxjs@babel@options]{babel}\relax
5190   }\bxjs@next
5191   \RequirePackage{bxorigcapt}\relax}

```

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

```

5192 \g@addto@macro\bxjs@endpreamble@hook{%
5193   \ifx\bxjs@babel@options\relax
5194     \bxjs@cslet{ver@babel.sty}\@undefined
5195   \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。

```
5196 \ifjswitheTeX
```

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

```

5197 \AtBeginOfFile{japanese.ldf}{\bxjs@begin@japanese@ldf@hook}
5198 \def\bxjs@begin@japanese@ldf@hook{%
5199   \let\bxjs@begin@japanese@ldf@hook\relax
5200   \let\bxjs@save@ProvidesLanguage\ProvidesLanguage
5201   \let\bxjs@save@LdfInit\LdfInit
5202   \def\ProvidesLanguage##1[##2]{\bxjs@do@japanese@ldf{##2}}%
5203   \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。

```

5204 \def\bxjs@do@japanese@ldf#1{\bxjs@do@japanese@ldf@a#1\@nil}
5205 \def\bxjs@do@japanese@ldf@a#1/#2/#3#4#5\@nil{%
5206   \let\LdfInit\bxjs@save@LdfInit
5207   \ClassInfo\bxjs@clsname
5208   {Release date of japanese.ldf is:\MessageBreak
5209     \@spaces #1/#2/#3#4\@gobble}%
5210   \ifnum#1#2#3#4<20201206 % v3.0
5211     \let\bxjs@japanese@ldf@skipped=t\csuse{endinput}%
5212   \fi}
5213 \AtEndOfFile{japanese.ldf}{\bxjs@end@japanese@ldf@hook}
5214 \def\bxjs@end@japanese@ldf@hook{%
5215   \let\bxjs@end@japanese@ldf@hook\relax
5216   \let\ProvidesLanguage\bxjs@save@ProvidesLanguage
5217   \let\LdfInit\bxjs@save@LdfInit

```

```

5218 \ifx t\bxjs@japanese@ldf@skipped
5219   \ClassWarningNoLine\bxjs@clsname
5220     {Loading japanese.ldf is skipped}%

```

ダミーの言語定義。

```

5221   \ifundef\l@japanese{\chardef\l@japanese\z@}{}%
5222   \let\datejapanese\empty\let\captionjapanese\empty
5223   \let\extrajapanese\empty\let\noextrajapanese\empty
5224   \main@language{japanese}%
5225 \fi}
5226 \g@addto@macro\bxjs@begin@document@hook{%
5227   \let\bxjs@begin@japanese@ldf@hook\relax
5228   \let\bxjs@end@japanese@ldf@hook\relax}
5229 \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。

```

5230 \bxjs@set@dupload@proc{geometry.sty}{%
5231   \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。

```

5232 \if l@jsEngine
5233   \pandocSkipLoadPackage{xeCJK}
5234   \providecommand*\setCJKmainfont{\setmainfont}
5235 \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。

```

5236 \if x@jsEngine
5237   \expandafter\g@addto@macro\csname opt@xeCJK.sty\endcsname{%
5238     ,space}
5239 \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。

```

5240 \ifjswitheTeX
5241 \@onlypreamble\bxjs@info@or@ban
5242 \def\bxjs@info@or@ban#1{%
5243   \PackageInfo\bxjs@clsname
5244   {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。

```

5245 \unless\ifbxjs@jp@or@indent
5246   \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。

```

5247 \IfFileExists{parskip.sty}{%
5248   \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。

```

5249 }{%else
5250   \eappto\bxjs@endpreamble@hook{%
5251     \parindent=\the\parindent\relax
5252     \parskip=\the\parskip\relax}}
5253 \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。

```

5254 \ifbxjs@jp@or@secnumdepth\else
5255   \bxjs@info@or@ban{secnumdepth}
5256   \eappto\bxjs@endpreamble@hook{%
5257     \c@secnumdepth=\the\c@secnumdepth\relax}
5258 \fi

```

■**block-heading** について \paragraph、\subparagraph を別行見出しに変える処理を抑止する。

※ 2.7.1 版以前では別行見出し変更が既定で有効であった。

```

5259 \ifbxjs@jp@or@block@heading\else
5260   \let\bxjs@frozen@paragraph\paragraph
5261   \let\bxjs@frozen@subparagraph\subparagraph
5262   \bxjs@info@or@ban{block-heading}
5263   \appto\bxjs@endpreamble@hook{%
5264     \let\oldparagraph\undefined
5265     \let\paragraph\bxjs@frozen@paragraph
5266     \let\subparagraph\bxjs@frozen@subparagraph}
5267 \fi

```

以上。

```

5268 \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。

```
5269 \g@addto@macro\bxjs@begin@document@hook{%
5270   \@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。

```
5271   \ifx\oldparagraph\@undefined\else
5272     \@tempwatrue
5273   \fi
```

`\paragraph` が番号付きの場合は、マーク変更が必要である。

```
5274   \ifnum\c@secnumdepth>3
5275     \@tempwatrue
5276   \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。

```
5277   \if@tempwa\ifx\jsParagraphMark\bxjs@org@paragraph@mark
5278     \let\jsParagraphMark\@empty
5279   \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。

```
5280 \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。

```
5281 \ifnum0\if x\jsEngine1\fi\if 1\jsEngine1\fi>\z@
5282   \catcode"3000=\active
5283   \begingroup \catcode`\!=7
5284   \protected\gdef!!!!3000{\pandocZWSpace}
5285   \endgroup
```

```

5286 \else\ifx\DeclareUnicodeCharacter\@undefined\else
5287 \DeclareUnicodeCharacter{3000}{\bxjs@zsp@char}
5288 \bxjs@protected\def\bxjs@zsp@char{\pandocZWSpace}
5289 \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。

```

5290 \if j@jsEngine
5291 \bxjs@fix@hyperref@unicode{false}
5292 \else
5293 \bxjs@fix@hyperref@unicode{true}
5294 \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)。

```

5295 \ifbxjs@jp@fix@strong\ifbxjs@jp@strong@cmd
5296 \let\orgtextbf\textbf
5297 \DeclareRobustCommand\pandocTextbf[1]{%
5298 \begingroup
5299 \let\textbf\orgtextbf
5300 \strong{#1}%
5301 \endgroup}%
5302 \g@addto@macro\bxjs@begin@document@hook{%
5303 \let\textbf\pandocTextbf}
5304 \fi\fi

```

`\strong` の書体を設定する。

```

5305 \jsAtEndOfClass{%
5306 \ifx\strongfontdeclare\@undefined\else
5307 \ifcase\bxjs@jp@strong
5308 \or \strongfontdeclare{\sffamily}%
5309 \or \strongfontdeclare{\sffamily\bfseries}%
5310 \fi
5311 \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。

```

5312 \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。

```
5313 \ifbxjs@bxghost@available
5314   \RequirePackage[verb]{bxghost}[2020/01/31]% v0.3.0
5315   \let\bxjs@eghostguarded\eghostguarded
5316 \else
5317 \chardef\bxjs@eghost@c=23
5318 \ifx j\jsEngine \xspcode\bxjs@eghost@c=3
5319 \else\ifx l\jsEngine \ltjsetparameter{alxspmode={\bxjs@eghost@c,3}}
5320 \else\ifx x\jsEngine %no-op
5321 \else \let\bxjs@eghost@c\undefined
5322 \fi\fi\fi
5323 \ifx\bxjs@eghost@c\undefined\else
5324   \font\bxjs@eghost@f=ec-lmr10 at 1.23456pt
5325   \def\bxjs@pan@eghost{\bgroup\bxjs@eghost@f\bxjs@eghost@c\egroup}
5326   \def\bxjs@eghostguarded#1{%
5327     \bxjs@pan@eghost\null#1\null\bxjs@pan@eghost}
5328 \fi
5329 \fi
5330 \ifx\bxjs@eghostguarded\undefined\else
5331   \let\orgtexttt\texttt
5332   \DeclareRobustCommand\pandocTexttt[1]{%
5333     \ifmmode \nfss@text{\ttfamily #1}%
5334     \else
5335       \ifvmode \leavevmode \fi
5336       \bxjs@eghostguarded{\begingroup\ttfamily#1\endgroup}%
5337     \fi}
5338   \g@addto@macro\bxjs@begin@document@hook{%
5339     \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。

```
5340   \ifbxjs@bxghost@available\else
5341   \expandafter\def\expandafter\verb\expandafter{%
5342     \expandafter\bxjs@pan@eghost\verb}
5343   \g@addto@macro\verb@egroup{\bxjs@pan@eghost}
5344   \fi
5345 \fi
5346 \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。

```
5347 \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。
```

```
5348 \def\bxjs@check@frontier{%
5349   \expandafter\bxjs@check@frontier@a\@currnamestack\noindent...\@nil}
5350 \def\bxjs@check@frontier@a#1#2#3#4#5\@nil#6{%
5351   \ifx\noindent#4#6\fi}
```

```
\bxjs@unforge@ifPDFTeX \ifPDFTeX を偽（正しい値）にする。
```

```
5352 \@onlypreamble\bxjs@unforge@ifPDFTeX
5353 \def\bxjs@unforge@ifPDFTeX{\global\bxjs@csletcs{ifPDFTeX}{iffalse}}
```

```
\bxjs@forge@ifPDFTeX \ifPDFTeX を真（偽装した値）にする。
```

```
5354 \@onlypreamble\bxjs@forge@ifPDFTeX
5355 \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。
```

```
5356 \def\bxjs@unload@forge@ifPDFTeX{%
5357   \bxjs@unforge@ifPDFTeX
5358   \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。

```
5359 \jsAtEndOfClass{\bxjs@forge@ifPDFTeX}
5360 \ifjsWitheTeX
5361   \AtBeginOfEveryFile{\bxjs@check@frontier\bxjs@unforge@ifPDFTeX}%
5362   \AtEndOfEveryFile{\bxjs@check@frontier\bxjs@forge@ifPDFTeX}%
5363   \g@addto@macro\bxjs@endpreamble@hook{\bxjs@unload@forge@ifPDFTeX}
5364 \else
5365   \g@addto@macro\bxjs@begin@document@hook{\bxjs@unload@forge@ifPDFTeX}
5366 \fi
5367 \fi
```

## E.14 完了

おしまい。

```
5368 %</pandoc>
```

和文ドライバ実装はここまで。

```
5369 %</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。

```
5370 %<*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

```
5371 %<*compat>
```

```
5372 \def\bxac@pkgname{bxjscompat}
```

`\bxjx@engine` エンジンの種別。

```
5373 \let\bxac@engine=n
```

```
5374 \def\bxac@do#1#2{%
```

```
5375 \edef\bxac@tmpa{\string#1}%
```

```
5376 \edef\bxac@tmpb{\meaning#1}%
```

```
5377 \ifx\bxac@tmpa\bxac@tmpb #2\fi}
```

```
5378 \bxac@do\kanjiskip{\let\bxac@engine=j}
```

```
5379 \bxac@do\XeTeXversion{\let\bxac@engine=x}
```

```
5380 \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。

```
5381 \ifx\jsAtEndOfClass\undefined
```

```
5382 \let\bxac@delayed@if@bxjs\@firstofone
```

```
5383 \else \let\bxac@delayed@if@bxjs\jsAtEndOfClass
```

```
5384 \fi
```

`\ImposeOldLuaTeXBehavior` `\ImposeOldLuaTeXBehavior` は 0.85 版以降の LuaTeX を一時的に pdfTeX と互換であるように見せかける。`\RevokeOldLuaTeXBehavior` で元に戻すことができる。

※エンジンが LuaTeX 以外の場合は何もしない。

```
5385 \newif\ifbxac@in@old@behavior
```

```
5386 \let\ImposeOldLuaTeXBehavior\relax
```

```
5387 \let\RevokeOldLuaTeXBehavior\relax
```

## G.2 8bit 欧文 TeX

```
5388 \ifx n\bxac@engine
```

和文を含むマクロ定義を通用させるため、高位バイトをアクティブ化しておく。

```
5389 \@tempcnta="80 \loop \ifnum\@tempcnta<"100
5390 \catcode\@tempcnta\active
5391 \advance\@tempcnta\@ne
5392 \repeat
```

以上。

```
5393 \fi
```

## G.3 XeTeX

```
5394 \ifx x\bxac@engine
```

■文字クラスの設定 XeTeX の文字クラス (`\XeTeXcharclass`) の Unicode 規定に基づく設定は、初期の版ではフォーマットに組み込まれていたが、2016/02/01 以降の 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、XeTeX に「文字間トークン挿入」の機能が導入されたのは 0.997 版 (2007 年頃) からのようだ。

ただし xeCJK が読込済ならば (そちらが適切に設定しているはずなので) 何もしない。

```
5395 \ifx\XeTeXcharclass\@undefined\else
5396 \bxac@delayed@if@bxjs{%
5397 \@ifpackageloaded{xeCJK}{}\@else
```

設定が未実行の状態ならば、設定用のファイルを読む。

```
5398 \ifx\xe@alloc@intercharclass\@undefined\else
5399 \ifnum\xe@alloc@intercharclass=\z@
5400 \PackageInfo\bxac@pkgname
5401 {Setting up interchar class for CJK...\@gobble}%
5402 \InputIfFileExists{load-unicode-xetex-classes.tex}{%
5403 \xe@alloc@intercharclass=3
5404 }{%else
5405 \PackageWarning\bxac@pkgname
5406 {Cannot find file 'load-unicode-xetex-classes.tex'%
5407 \@gobble}%
5408 }%
5409 \fi\fi
```

フォーマット組込だった時代の設定は不完全なところがあるので補正する。

```
5410 \ifnum\XeTeXcharclass"3041=\z@
5411 \PackageInfo\bxac@pkgname
5412 {Adjusting interchar class for CJK...\@gobble}%
5413 \@for\bxac@tmpb:={%
5414 3041,3043,3045,3047,3049,3063,3083,3085,3087,308E,%
5415 3095,3096,30A1,30A3,30A5,30A7,30A9,30C3,30E3,30E5,%
```

```

5416      30E7,30EE,30F5,30F6,30FC,31F0,31F1,31F2,31F3,31F4,%
5417      31F5,31F6,31F7,31F8,31F9,31FA,31FB,31FC,31FD,31FE,%
5418      31FF%
5419      }\do{\XeTeXcharclass"\bxac@tmpb=\@ne}%
5420      \fi
5421    }%
5422  }
5423 \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。

```

5424 \chardef\bxac@tmpb=11
5425 \def\bxac@do#1#2{%
5426   \@tempcnta=#1\relax
5427   \unless\ifnum\catcode\@tempcnta=\bxac@tmpb
5428     \chardef\bxac@tmpa=#2\relax
5429     \@whilenum{\@tempcnta<\bxac@tmpa}\do{%
5430       \catcode\@tempcnta\bxac@tmpb \advance\@tempcnta\@ne}%
5431     \fi}
5432 \bxac@do{"4E00}{"9FCD}

```

以上。

```
5433 \fi
```

## G.4 LuaTeX

```
5434 \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。

```

5435 %\unless\ifnum\luatexversion<80 \ifnum\luatexversion<85
5436 % \chardef\pdftexversion=200
5437 % \def\pdftexrevision{0}
5438 % \let\pdftexbanner\luatexbanner
5439 %\fi\fi

```

`\ImposeOldLuaTeXBehavior` 0.85 版以降であるかを検査する。

```

\RevokeOldLuaTeXBehavior 5440 \begingroup\expandafter\expandafter\expandafter\endgroup
5441 \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。

```

5442 \def\bxac@ob@list{%
5443   \do{\let}\pdfoutput{\outputmode}%
5444   \do{\let}\pdfpagewidth{\pagewidth}%
5445   \do{\let}\pdfpageheight{\pageheight}%
5446   \do{\protected\edef}\pdfhorigin{{\pdfvariable horigin}}%
5447   \do{\protected\edef}\pdfvorigin{{\pdfvariable vorigin}}%
5448 \def\bxac@ob@do#1#2{\begingroup
5449   \expandafter\bxac@ob@do@a\csname bxac@\string#2\endcsname{#1}#2}
5450 \def\bxac@ob@do@a#1#2#3#4{\endgroup

```

```

5451 \ifbxac@in@old@behavior \let#1#3\relax #2#3#4\relax
5452 \else \let#3#1\relax \let#1\@undefined
5453 \fi}
5454 \protected\def\ImposeOldLuaTeXBehavior{%
5455 \unless\ifbxac@in@old@behavior
5456 \bxac@in@old@behaviortrue
5457 \let\do\bxac@ob@do \bxac@ob@list
5458 \fi}
5459 \protected\def\RevokeOldLuaTeXBehavior{%
5460 \ifbxac@in@old@behavior
5461 \bxac@in@old@behaviorfalse
5462 \let\do\bxac@ob@do \bxac@ob@list
5463 \fi}
5464 \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。

```

5465 \directlua{
5466 local function range(cs, ce, cc, ff)
5467   if ff or not tex.getcatcode(cs) == cc then
5468     local setcc = tex.setcatcode
5469     for c = cs, ce do setcc(c, cc) end
5470   end
5471 end
5472 range(0x3400, 0x4DB5, 11, false)
5473 \ifnum\luatexversion>64
5474 range(0x4DB5, 0x4DBF, 11, true)
5475 range(0x4E00, 0x9FCC, 11, false)
5476 range(0x9FCD, 0x9FFF, 11, true)
5477 range(0xAC00, 0xD7A3, 11, false)
5478 range(0x20000, 0x2A6D6, 11, false)
5479 range(0x2A6D7, 0x2A6FF, 11, true)
5480 range(0x2A700, 0x2B734, 11, false)
5481 range(0x2B735, 0x2B73F, 11, true)
5482 range(0x2B740, 0x2B81D, 11, false)
5483 range(0x2B81E, 0x2B81F, 11, true)
5484 range(0x2B820, 0x2CEA1, 11, false)
5485 range(0x2CEA2, 0x2FFFD, 11, true)
5486 \fi
5487 }

```

以上。

```
5488 \fi
```

## G.5 完了

おしまい。

```
5489 %</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

```
5490 %<*cjkcat>
5491 \def\bxjx@pkgname{bxjscjkat}
5492 \newcount\bxjx@canta
5493 \@onlypreamble\bxjx@tmpdo
5494 \@onlypreamble\bxjx@tmpdo@a
5495 \@onlypreamble\bxjx@tmpdo@b
```

\bxjx@engine エンジンの種別。

```
5496 \let\bxjx@engine=n
5497 \def\bxjx@tmpdo#1#2{%
5498   \edef\bxjx@tmpa{\string#1}%
5499   \edef\bxjx@tmpb{\meaning#1}%
5500   \ifx\bxjx@tmpa\bxjx@tmpb #2\fi}
5501 \bxjx@tmpdo\kanjiskip{\let\bxjx@engine=j}
5502 \bxjx@tmpdo\enablecjktoken{%
5503   \ifx\ucs\@undefined\else \ifnum\ucs"3000="3000
5504     \let\bxjx@engine=u\fi\fi}
5505 \bxjx@tmpdo\XeTeXversion{\let\bxjx@engine=x}
5506 \bxjx@tmpdo\pdftexversion{\let\bxjx@engine=p}
5507 \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。

```
5508 \def\bxjx@tmpdo#1#2{%
5509   \if#1\bxjx@engine
5510     \ifpackageloaded{#2}{\fi}%else
5511     \PackageError\bxjx@pkgname
5512       {Package '#2' must be loaded}%
5513       {Package loading is aborted.\MessageBreak\@ehc}%
5514     \endinput}
5515 \fi}
5516 \bxjx@tmpdo{p}{bxcjkatype}
5517 \bxjx@tmpdo{x}{xeCJK}
5518 \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。

```
5519 \ifx\TextOrMath\@undefined
5520   \RequirePackage{fixltx2e}
5521 \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 LuaT<sub>E</sub>X-ja と（ほぼ）等価なものに変更する。

※ LuaT<sub>E</sub>X-ja との相違点：A830、A960、1B000。

```
5522 \if u\bxjx@engine
5523 \@for\bxjx@tmpa:={%
5524 0080,0100,0180,0250,02B0,0300,0500,0530,0590,0600,%
5525 0700,0750,0780,07C0,0800,0840,0860,08A0,0900,0980,%
5526 0A00,0A80,0B00,0B80,0C00,0C80,0D00,0D80,0E00,0E80,%
5527 0F00,1000,10A0,1200,1380,13A0,1400,1680,16A0,1700,%
5528 1720,1740,1760,1780,1800,18B0,1900,1950,1980,19E0,%
5529 1A00,1A20,1AB0,1B00,1B80,1BC0,1C00,1C50,1C80,1CC0,%
5530 1CD0,1D00,1D80,1DC0,1E00,1F00,2440,27C0,27F0,2800,%
5531 2A00,2C00,2C60,2C80,2D00,2D30,2D80,2DE0,2E00,4DC0,%
5532 A4D0,A500,A640,A6A0,A700,A720,A800,A830,A840,A880,%
5533 A8E0,A900,A930,A980,A9E0,AA00,AA60,AA80,AAE0,AB00,%
5534 AB30,AB70,ABCO,DB80,DC00,E000,FBO0,FB50,FE00,%
5535 FE70,FFFO,%
5536 10000,10080,10100,10140,10190,101D0,10280,102A0,%
5537 102E0,10300,10330,10350,10380,103A0,10400,10450,%
5538 10480,104B0,10500,10530,10600,10800,10840,10860,%
5539 10880,108E0,10900,10920,10980,109A0,10A00,10A60,%
5540 10A80,10AC0,10B00,10B40,10B60,10B80,10C00,10C80,%
5541 10E60,11000,11080,110D0,11100,11150,11180,111E0,%
5542 11200,11280,112B0,11300,11400,11480,11580,11600,%
5543 11660,11680,11700,118A0,11A00,11A50,11AC0,11C00,%
5544 11C70,11D00,12000,12400,12480,13000,14400,16800,%
5545 16A40,16AD0,16B00,16F00,1BC00,1BCA0,1D000,1D100,%
5546 1D200,1D300,1D360,1D400,1D800,1E000,1E800,1E900,%
5547 1EE00,1F000,1F030,1FOA0,1F300,1F600,1F650,1F680,%
5548 1F700,1F780,1F800,1F900,E0000,E0100,F0000,100000,%
5549 00C0%
5550 }\do{%
5551 \@tempcnta="\bxjx@tmpa\relax
5552 \@tempcntb\@tempcnta \advance\@tempcntb\m@ne
5553 \chardef\bxjx@tmpb\kcatcode\@tempcntb
5554 \kcatcode\@tempcnta=15 \kcatcode\@tempcntb\bxjx@tmpb}
5555 \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。

```

5556 \onlypreamble\bxjx@grkcyr@list
5557 \def\bxjx@grkcyr@list{%
5558 \do{0391}{LGR}{\textAlpha}{A}%           % GR. C. L. ALPHA
5559 \do{0392}{LGR}{\textBeta}{B}%           % GR. C. L. BETA
5560 \do{0393}{LGR}{\textGamma}{\Gamma}%     % GR. C. L. GAMMA
5561 \do{0394}{LGR}{\textDelta}{\Delta}%     % GR. C. L. DELTA
5562 \do{0395}{LGR}{\textEpsilon}{E}%       % GR. C. L. EPSILON
5563 \do{0396}{LGR}{\textZeta}{Z}%          % GR. C. L. ZETA
5564 \do{0397}{LGR}{\textEta}{H}%           % GR. C. L. ETA
5565 \do{0398}{LGR}{\textTheta}{\Theta}%     % GR. C. L. THETA
5566 \do{0399}{LGR}{\textIota}{I}%          % GR. C. L. IOTA
5567 \do{039A}{LGR}{\textKappa}{K}%         % GR. C. L. KAPPA
5568 \do{039B}{LGR}{\textLambda}{\Lambda}%   % GR. C. L. LAMDA
5569 \do{039C}{LGR}{\textMu}{M}%           % GR. C. L. MU
5570 \do{039D}{LGR}{\textNu}{N}%           % GR. C. L. NU
5571 \do{039E}{LGR}{\textXi}{\Xi}%         % GR. C. L. XI
5572 \do{039F}{LGR}{\textOmicron}{O}%     % GR. C. L. OMICRON
5573 \do{03A0}{LGR}{\textPi}{\Pi}%          % GR. C. L. PI
5574 \do{03A1}{LGR}{\textRho}{P}%           % GR. C. L. RHO
5575 \do{03A3}{LGR}{\textSigma}{\Sigma}%    % GR. C. L. SIGMA
5576 \do{03A4}{LGR}{\textTau}{T}%          % GR. C. L. TAU
5577 \do{03A5}{LGR}{\textUpsilon}{\Upsilon}% % GR. C. L. UPSILON
5578 \do{03A6}{LGR}{\textPhi}{\Phi}%        % GR. C. L. PHI
5579 \do{03A7}{LGR}{\textChi}{X}%          % GR. C. L. CHI
5580 \do{03A8}{LGR}{\textPsi}{\Psi}%        % GR. C. L. PSI
5581 \do{03A9}{LGR}{\textOmega}{\Omega}%    % GR. C. L. OMEGA
5582 \do{03B1}{LGR}{\textalpha}{\alpha}%    % GR. S. L. ALPHA
5583 \do{03B2}{LGR}{\textbeta}{\beta}%     % GR. S. L. BETA
5584 \do{03B3}{LGR}{\textgamma}{\gamma}%    % GR. S. L. GAMMA
5585 \do{03B4}{LGR}{\textdelta}{\delta}%    % GR. S. L. DELTA
5586 \do{03B5}{LGR}{\textepsilon}{\epsilon}% % GR. S. L. EPSILON
5587 \do{03B6}{LGR}{\textzeta}{\zeta}%     % GR. S. L. ZETA
5588 \do{03B7}{LGR}{\texteta}{\eta}%        % GR. S. L. ETA
5589 \do{03B8}{LGR}{\texttheta}{\theta}%    % GR. S. L. THETA
5590 \do{03B9}{LGR}{\textiota}{\iota}%     % GR. S. L. IOTA
5591 \do{03BA}{LGR}{\textkappa}{\kappa}%    % GR. S. L. KAPPA
5592 \do{03BB}{LGR}{\textlambda}{\lambda}%  % GR. S. L. LAMDA
5593 \do{03BC}{LGR}{\textmu}{\mu}%         % GR. S. L. MU
5594 \do{03BD}{LGR}{\textnu}{\nu}%         % GR. S. L. NU
5595 \do{03BE}{LGR}{\textxi}{\xi}%         % GR. S. L. XI
5596 \do{03BF}{LGR}{\textomicron}{o}%       % GR. S. L. OMICRON
5597 \do{03C0}{LGR}{\textpi}{\pi}%         % GR. S. L. PI

```



5598 \do{03C1}{LGR}{\textrho}{\rho}% % GR. S. L. RHO  
5599 \do{03C2}{LGR}{\textvarsigma}{\varsigma}% % GR. S. L. FINAL SIGMA  
5600 \do{03C3}{LGR}{\textsigma}{\sigma}% % GR. S. L. SIGMA  
5601 \do{03C4}{LGR}{\texttau}{\tau}% % GR. S. L. TAU  
5602 \do{03C5}{LGR}{\textupsilon}{\upsilon}% % GR. S. L. UPSILON  
5603 \do{03C6}{LGR}{\textphi}{\phi}% % GR. S. L. PHI  
5604 \do{03C7}{LGR}{\textchi}{\chi}% % GR. S. L. CHI  
5605 \do{03C8}{LGR}{\textpsi}{\psi}% % GR. S. L. PSI  
5606 \do{03C9}{LGR}{\textomega}{\omega}% % GR. S. L. OMEGA  
5607 \do{0401}{T2A}{\CYRYO}{-}% % CY. C. L. IO  
5608 \do{0410}{T2A}{\CYRA}{-}% % CY. C. L. A  
5609 \do{0411}{T2A}{\CYRB}{-}% % CY. C. L. BE  
5610 \do{0412}{T2A}{\CYRV}{-}% % CY. C. L. VE  
5611 \do{0413}{T2A}{\CYRG}{-}% % CY. C. L. GHE  
5612 \do{0414}{T2A}{\CYRD}{-}% % CY. C. L. DE  
5613 \do{0415}{T2A}{\CYRE}{-}% % CY. C. L. IE  
5614 \do{0416}{T2A}{\CYRZH}{-}% % CY. C. L. ZHE  
5615 \do{0417}{T2A}{\CYRZ}{-}% % CY. C. L. ZE  
5616 \do{0418}{T2A}{\CYRI}{-}% % CY. C. L. I  
5617 \do{0419}{T2A}{\CYRISHRT}{-}% % CY. C. L. SHORT I  
5618 \do{041A}{T2A}{\CYRK}{-}% % CY. C. L. KA  
5619 \do{041B}{T2A}{\CYRL}{-}% % CY. C. L. EL  
5620 \do{041C}{T2A}{\CYRM}{-}% % CY. C. L. EM  
5621 \do{041D}{T2A}{\CYRN}{-}% % CY. C. L. EN  
5622 \do{041E}{T2A}{\CYRO}{-}% % CY. C. L. O  
5623 \do{041F}{T2A}{\CYRP}{-}% % CY. C. L. PE  
5624 \do{0420}{T2A}{\CYRR}{-}% % CY. C. L. ER  
5625 \do{0421}{T2A}{\CYRS}{-}% % CY. C. L. ES  
5626 \do{0422}{T2A}{\CYRT}{-}% % CY. C. L. TE  
5627 \do{0423}{T2A}{\CYRU}{-}% % CY. C. L. U  
5628 \do{0424}{T2A}{\CYRF}{-}% % CY. C. L. EF  
5629 \do{0425}{T2A}{\CYRH}{-}% % CY. C. L. HA  
5630 \do{0426}{T2A}{\CYRC}{-}% % CY. C. L. TSE  
5631 \do{0427}{T2A}{\CYRCH}{-}% % CY. C. L. CHE  
5632 \do{0428}{T2A}{\CYRSH}{-}% % CY. C. L. SHA  
5633 \do{0429}{T2A}{\CYRSHCH}{-}% % CY. C. L. SHCHA  
5634 \do{042A}{T2A}{\CYRHRDSN}{-}% % CY. C. L. HARD SIGN  
5635 \do{042B}{T2A}{\CYRERY}{-}% % CY. C. L. YERU  
5636 \do{042C}{T2A}{\CYRSFTSN}{-}% % CY. C. L. SOFT SIGN  
5637 \do{042D}{T2A}{\CYREREV}{-}% % CY. C. L. E  
5638 \do{042E}{T2A}{\CYRYU}{-}% % CY. C. L. YU  
5639 \do{042F}{T2A}{\CYRYA}{-}% % CY. C. L. YA  
5640 \do{0430}{T2A}{\cyra}{-}% % CY. S. L. A  
5641 \do{0431}{T2A}{\cyrb}{-}% % CY. S. L. BE  
5642 \do{0432}{T2A}{\cyrv}{-}% % CY. S. L. VE  
5643 \do{0433}{T2A}{\cyrg}{-}% % CY. S. L. GHE  
5644 \do{0434}{T2A}{\cyrd}{-}% % CY. S. L. DE  
5645 \do{0435}{T2A}{\cyre}{-}% % CY. S. L. IE  
5646 \do{0436}{T2A}{\cyrzh}{-}% % CY. S. L. ZHE

```

5647 \do{0437}{T2A}{\cyrz}{}% % CY. S. L. ZE
5648 \do{0438}{T2A}{\cyri}{}% % CY. S. L. I
5649 \do{0439}{T2A}{\cyrishrt}{}% % CY. S. L. SHORT I
5650 \do{043A}{T2A}{\cyrk}{}% % CY. S. L. KA
5651 \do{043B}{T2A}{\cyrl}{}% % CY. S. L. EL
5652 \do{043C}{T2A}{\cyrn}{}% % CY. S. L. EM
5653 \do{043D}{T2A}{\cyro}{}% % CY. S. L. EN
5654 \do{043E}{T2A}{\cyro}{}% % CY. S. L. O
5655 \do{043F}{T2A}{\cyrp}{}% % CY. S. L. PE
5656 \do{0440}{T2A}{\cyrr}{}% % CY. S. L. ER
5657 \do{0441}{T2A}{\cyrs}{}% % CY. S. L. ES
5658 \do{0442}{T2A}{\cyrt}{}% % CY. S. L. TE
5659 \do{0443}{T2A}{\cyru}{}% % CY. S. L. U
5660 \do{0444}{T2A}{\cyrf}{}% % CY. S. L. EF
5661 \do{0445}{T2A}{\cyrh}{}% % CY. S. L. HA
5662 \do{0446}{T2A}{\cyrc}{}% % CY. S. L. TSE
5663 \do{0447}{T2A}{\cyrch}{}% % CY. S. L. CHE
5664 \do{0448}{T2A}{\cyrsh}{}% % CY. S. L. SHA
5665 \do{0449}{T2A}{\cyrshch}{}% % CY. S. L. SHCHA
5666 \do{044A}{T2A}{\cyrhrdsn}{}% % CY. S. L. HARD SIGN
5667 \do{044B}{T2A}{\cyrery}{}% % CY. S. L. YERU
5668 \do{044C}{T2A}{\cyrftsnn}{}% % CY. S. L. SOFT SIGN
5669 \do{044D}{T2A}{\cyrerev}{}% % CY. S. L. E
5670 \do{044E}{T2A}{\cyryu}{}% % CY. S. L. YU
5671 \do{044F}{T2A}{\cyrya}{}% % CY. S. L. YA
5672 \do{0451}{T2A}{\cyryo}{}% % CY. S. L. IO
5673 \do{00A7}{TS1}{\textsection}{\mathsection}% SECTION SYMBOL
5674 \do{00A8}{TS1}{\textasciidieresis}{}% % DIAERESIS
5675 \do{00B0}{TS1}{\textdegree}{\mathdegree}% % DEGREE SIGN
5676 \do{00B1}{TS1}{\textpm}{\pm}% % PLUS-MINUS SIGN
5677 \do{00B4}{TS1}{\textasciicute}{}% % ACUTE ACCENT
5678 \do{00B6}{TS1}{\textparagraph}{\mathparagraph}% PILCROW SIGN
5679 \do{00D7}{TS1}{\texttimes}{\times}% % MULTIPLICATION SIGN
5680 \do{00F7}{TS1}{\textdiv}{\div}% % DIVISION SIGN
5681 }

```

`\mathdegree` 面倒なので補っておく。

```
5682 \providecommand*{\mathdegree}{\circ}
```

`\ifbxjx@gcc@cjkk` [スイッチ] 「特定 CJK 曖昧文字」を和文扱いにするか。

```
5683 \newif\ifbxjx@gcc@cjkk
```

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

```
5684 \newcommand*\greekasCJK{%
```

```
5685 \bxjx@gcc@cjkktrue}
```

`\nogreekasCJK` [公開命令] 「特定 CJK 曖昧文字」を欧文扱いにする。

```
5686 \newcommand*\nogreekasCJK{%
```

```
5687 \bxjx@gcc@cjkkfalse}
```

`\bxjx@fake@grk \bxjx@fake@grk{〈出力文字〉}{〈基準文字〉}`： ラテン文字で代用される数式ギリシャ文字の出力を行う。〈基準文字〉 (`mathchardef` の制御綴) の数式クラスと数式ファミリーを引き継いで、〈出力文字〉 (ASCII 文字トークン) の文字コードの数式文字を出力する。例えば、`\Pi` の意味が `\mathchar"7005` である場合、`\bxjx@fake@grk{B}{\Pi}` は `\mathchar"7042` を実行する。

※フォントパッケージ使用時の再定義を考慮して、〈基準文字〉が `mathchardef` であるかを検査し、そうでない場合はフォールバックとして単に 〈出力文字〉 を実行する。

```
5688 \def\bxjx@tmpdo#1\relax{%
5689   \def\bxjx@fake@grk##1##2{%
5690     \expandafter\bxjx@fake@grk@a\meaning##2#1\@nil{##1}{##2}}%
5691   \def\bxjx@fake@grk@a##1#1##2\@nil##3##4{%
5692     \ifx\##1\%
5693       \bxjx@cnta##4\divide\bxjx@cnta\@cclvi
5694       \multiply\bxjx@cnta\@cclvi \advance\bxjx@cnta`##3\relax
5695       \mathchar\bxjx@cnta
5696     \else ##3\fi}
5697 }\expandafter\bxjx@tmpdo\string\mathchar\relax
```

#### ■pdfTeX・upTeX の場合

```
5698 \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 L<sup>A</sup>T<sub>E</sub>X においても、必ず「`inputenc` が明示的に読み込まれた」状態になる。

```
5699 \@ifpackageloaded{inputenc}{}{%else
5700   \RequirePackage[utf8]{inputenc}}
5701 \def\bxjx@tmpa{utf8}
5702 \ifx\bxjx@tmpa\inputencdoingname
5703   \PackageWarningNoLine\bxjx@pkgname
5704     {Input encoding changed to utf8}%
5705   \inputencoding{utf8}%
5706 \fi
```

upTeX の場合に、「特定曖昧 CJK 文字」を含むブロックの和文カテゴリコードを変更する。

```
5707 \if u\bxjx@engine
5708 \kcatcode"0370=15
5709 \kcatcode"0400=15
5710 \kcatcode"0500=15
5711 \fi
```

各文字について `\DeclareUnicodeCharacter` を実行する。

```
5712 \def\bxjx@tmpdo#1{%
5713   \@tempcnta=#1\relax
```

```
5714 \expandafter\bxjx@tmpdo@a\csname bxjx@KC/\the\@tempcnta\endcsname{#1}}
5715 \def\bxjx@tmpdo@a#1#2#3#4#5{%
```

引数 =  $\langle$ bxjx@KC/ $\langle$ 符号値 $\rangle$  $\rangle$  $\langle$ fontenc $\rangle$  $\langle$ LICR $\rangle$  $\langle$ 数式 LICR $\rangle$

“数式中の動作”を決定する。 $\langle$ 数式 LICR $\rangle$  が空 (数式非対応) なら警告を出す。

```
5716 \ifx\#5\%
5717 \def\bxjx@tmpa{\@inmathwarn#4}%
```

$\langle$ 数式 LICR $\rangle$  が英字である場合は  $\backslash$ bxjx@fake@grk で出力する。大文字なら  $\backslash$ Pi、小文字なら  $\backslash$ pi を基準文字にする。

```
5718 \else\ifcat A\noexpand#5%
5719 \edef\bxjx@tmpa{\noexpand\bxjx@fake@grk{#5}%
5720 {\ifnum\uccode`#5=`#5\noexpand\Pi\else\noexpand\pi\fi}}%
```

それ以外は  $\langle$ 数式 LICR $\rangle$  をそのまま実行する。

```
5721 \else \def\bxjx@tmpa{#5}%
5722 \fi\fi
5723 \def\bxjx@tmpb{\bxjx@tmpdo@b{#1}{#2}{#3}{#4}}%
5724 \expandafter\bxjx@tmpb\expandafter{\bxjx@tmpa}}
```

以降はエンジン種別で分岐する。up $\TeX$  の場合。

```
5725 \if u\bxjx@engine
5726 \def\bxjx@tmpdo@b#1#2#3#4#5{%
```

引数 =  $\langle$ bxjx@KC/ $\langle$ 符号値 $\rangle$  $\rangle$  $\langle$ fontenc $\rangle$  $\langle$ LICR $\rangle$  $\langle$ 数式中の動作 $\rangle$

当該の Unicode 文字の動作は「テキストでは  $\langle$ LICR $\rangle$ 、数式では  $\langle$ 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  $\backslash$ 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  $\backslash$ bxjx@KC/... を流用している。

```
5727 \kchardef#1=\@tempcnta
5728 \DeclareTextCommandDefault{#4}{\bxjx@ja@or@not{#1}{#3}{#4}}%
5729 \DeclareUnicodeCharacter{#2}{\TextOrMath{#4}{#5}}
```

pdf $\TeX$  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  $\backslash$ UTF $\langle$ 符号値 $\rangle$  を使う ( $\backslash$ UTF は bxcjkatype の命令)。 $\backslash$ bxjx@KC/... は使わないが定義済にする必要がある。

```
5730 \else\if p\bxjx@engine
5731 \def\bxjx@tmpdo@b#1#2#3#4#5{%
5732 \mathchardef#1=\@tempcnta
5733 \DeclareTextCommandDefault{#4}{\bxjx@ja@or@not{\UTF{#2}}{#3}{#4}}%
5734 \DeclareUnicodeCharacter{#2}{\TextOrMath{#4}{#5}}
5735 \fi\fi
```

以上の処理を「特定 CJK 曖昧文字」の各々に適用する。

```
5736 \let\do\bxjx@tmpdo \bxjx@grkcyr@list
```

$\backslash$ bxjx@DeclareUnicodeCharacter  $\backslash$ bxjx@DeclareUnicodeCharacter を変更して、「特定 CJK 曖昧文字」の場合に再定義を抑制したもの。

```
5737 \@onlypreamble\bxjx@org@DeclareUnicodeCharacter
```

```

5738 \let\bxjx@org@DeclareUnicodeCharacter\DeclareUnicodeCharacter
5739 \onlypreamble\bxjx@DeclareUnicodeCharacter
5740 \def\bxjx@DeclareUnicodeCharacter#1#2{%
5741   \count@=#1\relax
5742   \expandafter\ifx\csname bxjx@KC/\the\count@\endcsname\relax
5743     \bxjx@org@DeclareUnicodeCharacter{#1}{#2}%
5744   \else
5745     \wlog{ \space\space skipped defining Unicode char U+#1}%
5746   \fi}

```

`\bxjx@ja@or@not` `\bxjx@ja@or@not{〈和文用定義〉}{〈対象 fontenc〉}{〈LICR〉}` : `\[no]greekasCJK` の状態に応じて和文または欧文で文字を出力する。

```

5747 \def\bxjx@ja@or@not#1#2#3{%
\greekasCJK の場合は、無条件に (和文用定義) を実行する。
5748   \ifbxjx@gcc@cjk #1%
\nogreekasCJK の場合は、対象のエンコーディングに変更して LICR を実行するが、その
エンコーディングが未定義の場合は (フォールバックとして) 和文用定義を使う。
5749   \else\expandafter\ifx\csname T@#2\endcsname\relax #1%
5750   \else \UseTextSymbol{#2}{#3}%
5751   \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。

```

5752 \begingroup
5753 \toks@\expandafter{\DeclareFontEncoding@{#1}{#2}{#3}}
5754 \xdef\next{\def\noexpand\DeclareFontEncoding@##1##2##3{%
5755   \noexpand\bxjx@swap@DUC@cmd
5756   \the\toks@
5757   \noexpand\bxjx@swap@DUC@cmd}}
5758 \endgroup\next
5759 \def\bxjx@swap@DUC@cmd{%
5760   \let\bxjx@tmpa\DeclareUnicodeCharacter
5761   \let\DeclareUnicodeCharacter\bxjx@DeclareUnicodeCharacter
5762   \let\bxjx@DeclareUnicodeCharacter\bxjx@tmpa
5763   \let\bxjx@tmpa\relax}

```

以上。

### ■Xe<sub>La</sub>TeX・Lua<sub>La</sub>TeX の場合

```
5764 \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。

```

5765 \def\bxjx@tmpdo#1{%
5766   \bxjx@cna=#1\relax
5767   \begingroup
5768     \lcode`~=\bxjx@cna
5769   \lowercase{\endgroup

```

```
5770 \bxjx@tmpdo@a{~}{#1}}
5771 \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、upL<sup>A</sup>T<sub>E</sub>X 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。

```
5772 \ifx\#5\\\let\bxjx@tmpa\relax
5773 \else\ifcat A\noexpand#5%
5774 \edef\bxjx@tmpa{\noexpand\bxjx@fake@grk{#5}%
5775 {\ifnum\uccode`#5=`#5\noexpand\Pi\else\noexpand\pi\fi}}%
5776 \else \def\bxjx@tmpa{#5}%
5777 \fi\fi
5778 \ifx\bxjx@tmpa\relax\else
5779 \mathcode\bxjx@cmta"8000 \let#1\bxjx@tmpa
5780 \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。

```
5781 \mathchardef\bxjx@tmpa="119
5782 \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。

Lua<sub>T</sub>E<sub>X</sub> の場合は、Lua<sub>T</sub>E<sub>X</sub>-ja の jacharrange の設定を変更する。

※ “範囲 2” がギリシャ・キリル文字、“範囲 8” が Latin-1 の記号。

```
5783 \if l\bxjx@engine
5784 \protected\def\greekasCJK{%
5785 \bxjx@gcc@cjktrue
5786 \ltjsetparameter{jacharrange={+2, +8}}
5787 \protected\def\nogreekasCJK{%
5788 \bxjx@gcc@cjkfalse
5789 \ltjsetparameter{jacharrange={-2, -8}}
5790 \fi
```

X<sub>Ǝ</sub>L<sup>A</sup>T<sub>E</sub>X の場合、xeCJK は X<sub>Ǝ</sub>L<sup>A</sup>T<sub>E</sub>X 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。

```
5791 \if x\bxjx@engine
5792 \let\bxjx@gcc@cjk@list\@empty
5793 \def\do#1#2#3#4{%
5794 \edef\bxjx@gcc@cjk@list{\bxjx@gcc@cjk@list
5795 \noexpand\XeTeXcharclass"#1\bxjx@cmta}}
5796 \bxjx@grkcyr@list
5797 \protected\def\greekasCJK{%
5798 \bxjx@gcc@cjktrue
5799 \bxjx@cmta=\@ne \bxjx@gcc@cjk@list}
5800 \protected\def\nogreekasCJK{%
5801 \bxjx@gcc@cjkfalse
5802 \bxjx@cmta=\z@ \bxjx@gcc@cjk@list}
5803 \fi
```

以上。

```
5804 \fi\fi
```

## H.4 初期設定

「特定 CJK 曖昧文字」を欧文扱いにする。

```
5805 \nogreekasCJK
```

## H.5 完了

おしまい。

```
5806 %</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。

### I.1 準備

```
5807 %<*ancpandoc>
```

```
5808 %% このファイルは日本語文字を含みます.
```

```
5809 \def\bxjsp@pkgnam{bxjspandoc}
```

`\bxjsp@engine` エンジンの種別。

```
5810 \let\bxjsp@engine=n
```

```
5811 \@onlypreamble\bxjsp@do
```

```
5812 \def\bxjsp@do#1#2{%
```

```
5813 \edef\bxjsp@tmpa{\string#1}%
```

```
5814 \edef\bxjsp@tmpb{\meaning#1}%
```

```
5815 \ifx\bxjsp@tmpa\bxjsp@tmpb #2\fi}
```

```
5816 \bxjsp@do\kanjiskip{\let\bxjsp@engine=j}
```

```
5817 \bxjsp@do\XeTeXversion{\let\bxjsp@engine=x}
```

```
5818 \bxjsp@do\pdftexversion{\let\bxjsp@engine=p}
```

```
5819 \bxjsp@do\luatexversion{\let\bxjsp@engine=l}
```

`\bxjsp@begin@document@hook` 文書本体開始時フック。

```
5820 \@onlypreamble\bxjsp@begin@document@hook
```

```
5821 \let\bxjsp@begin@document@hook\@empty
```

```
5822 \AtBeginDocument{\bxjsp@begin@document@hook}
```

`\ifbxjsp@babel@used` [スイッチ] Babel が読み込まれたか。

```

5823 \newif\ifbxjsp@babel@used
5824 \g@addto@macro\bxjsp@begin@document@hook{%
5825   \@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。

```

5826 \newif\ifbxjsp@english
5827 \DeclareOption{english}{\bxjsp@englishtrue}

```

オプション定義はおしまい。

```

5828 \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。

```

5829 \@onlypreamble\pandocSkipLoadFile
5830 \newcommand*\pandocSkipLoadFile[1]{%
5831   \expandafter\bxjsp@skip@load@file@a\csname ver@#1\endcsname{#1}}
5832 \def\bxjsp@skip@load@file@a#1#2{%
5833   \ifx#1\relax
5834     \def#1{2001/01/01}%
5835     \PackageInfo\bxjsp@pkgname
5836       {File '#2' marked as loaded@gobble}%
5837   \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。

```

5838 \@onlypreamble\pandocSkipLoadPackage
5839 \newcommand*\pandocSkipLoadPackage[1]{%
5840   \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 L<sup>A</sup>T<sub>E</sub>X ではこれで警告が出る。これを抑止する。

L<sup>A</sup>T<sub>E</sub>X カーネルが新しい場合は `fixltx2e` を読込済にする。

```

5841 \ifx\@IncludeInRelease\@undefined\else
5842   \pandocSkipLoadPackage{fixltx2e}
5843 \fi

```



## 1.5 cmap パッケージ

エンジンが (u)pL<sup>A</sup>T<sub>E</sub>X のときに cmap パッケージが読み込まれるのを阻止する。(実際は警告が出るだけで無害であるが。)

```
5844 \if j\bxjsp@engine
5845   \pandocSkipLoadPackage{cmap}
5846 \fi
```

## 1.6 microtype パッケージ

警告が多すぎなので消す。

```
5847 \if j\bxjsp@engine \else
5848   \PassOptionsToPackage{verbose=silent}{microtype}
5849 \fi
```

エンジンが (u)pL<sup>A</sup>T<sub>E</sub>X のときに microtype パッケージが読み込まれるのを阻止し、さらにテンプレートで使われている命令を通すためにダミーの定義を行う。

※昔は standard ドライバでこの処理を行っていたが、元来は Pandoc 用の処理なので、1.5 版で pandoc に移動。

```
5850 \if j\bxjsp@engine
5851   \pandocSkipLoadPackage{microtype}
5852   \newcommand*{UseMicrotypeSet}[2][]{ }
5853 \fi
```

## 1.7 Unicode 文字変換対策

Pandoc で L<sup>A</sup>T<sub>E</sub>X 形式に書き出す場合は、元データ中の一部の Unicode 文字を「L<sup>A</sup>T<sub>E</sub>X の表記」に置き換える。その中には日本語文書で問題になるものが含まれる。

```
…→\ldots{ } ‘→` ’→' “→`` ”→''
```

日本語 L<sup>A</sup>T<sub>E</sub>X では「L<sup>A</sup>T<sub>E</sub>X の表記」は欧文扱い、Unicode 文字は和文扱いとして使い分ける習慣があるので、このような置換が行われるのは好ましくない。

これらの置換のうち、後の 4 つは Pandoc の `--no-tex-ligatures` オプションを指定すれば抑止できるが、「…」の置換を抑止する機能はないようである。そこで、「\ldots を『…』に戻す」という処置を行う。

`\pandocLdots` Pandoc 用の `\ldots` の実装。非数式である場合は代わりに … を実行する。

※以前は「Pandoc が必ず `\ldots{ }` の形で書き出す」ことを利用して後続に `{ }` があるかで「元が … であるか」を判断していた。ところが、Pandoc 2.7 版で `{ }` を必ずしも付けなくなったため、1.9f 版で非数式の `\ldots` を全て … に戻す動作に変更した。

```
5854 \DeclareRobustCommand{\pandocLdots}{%
5855   \let\bxjsp@do\bxjsp@ja@ellipsis
5856   \ifmmode \let\bxjsp@do\bxjsp@org@ldots
```

```

5857 \else\ifbxjsp@babel@used
5858   \expandafter\ifx\csname bxjsp@ld/\language\endcsname\relax
5859     \let\bxjsp@do\bxjsp@org@ldots \fi
5860   \fi\fi \bxjsp@do}
5861 \@namedef{bxjsp@ld/japanese}{1}
5862 \def\bxjsp@ja@ellipsis{…}
5863 \let\bxjsp@org@ldots\ldots

```

`\ldots` の実装を `\pandocLdots` に置き換える。

```

5864 \g@addto@macro\bxjsp@begin@document@hook{%
5865   \let\bxjsp@org@ldots\pandocLdots

```

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。

```

5866 \long\def\bxjsp@tmpa{\ldots}%
5867 \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。

```

5868   \ifnum0\ifbxjsp@english\ifbxjsp@babel@used\else1\fi\fi=0
5869     \let\ldots\pandocLdots
5870   \fi
5871 \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  $X_{\text{L}}\text{T}_{\text{E}}\text{X}$ ・ $\text{L}\text{u}\text{a}\text{T}_{\text{E}}\text{X}$  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。

```

5872 \chardef\bxjsp@cc@other=12
5873 \@onlypreamble\bxjsp@makeother@range
5874 \def\bxjsp@makeother@range#1#2{%
5875   \@tempcnta"#1\relax \@tempcntb"#2\relax
5876   \loop\ifnum\@tempcnta<\@tempcntb
5877     \catcode\@tempcnta\bxjsp@cc@other
5878     \advance\@tempcnta\@ne
5879   \repeat}
5880 \ifnum0\if x\bxjsp@engine1\fi\if 1\bxjsp@engine1\fi>0
5881   \catcode"1F23B=\bxjsp@cc@other
5882   \bxjsp@makeother@range{9FCD}{A000}
5883   \bxjsp@makeother@range{1B002}{1B170}
5884   \bxjsp@makeother@range{2B820}{2EBF0}
5885 \fi

```

## 1.8 PandoLa モジュール

インストール済であれば読み込む。

```

5886 \IfFileExists{bxpandola.sty}{%
5887   \RequirePackage{bxpandola}\relax

```

```
5888 \PackageInfo\bxjsp@pkgnam  
5889 {PandoLa module is loaded\@gobble}  
5890 }{}
```

## 1.9 完了

おしまい。

```
5891 %</ancpandoc>
```

補助パッケージ実装はここまで。

```
5892 %</anc>
```