

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”)





v2.7 [2022/03/30]

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

1	はじめに	3
2	オプション	11
3	和文フォントの変更	41
4	フォントサイズ	41
5	レイアウト	47
5.1	ページレイアウト	48
6	改ページ (日本語 TeX 開発コミュニティ版のみ)	63
7	ページスタイル	64
8	文書のマークアップ	67
8.1	表題	67
8.2	章・節	73
8.3	リスト環境	85
8.4	パラメータの設定	92
8.5	フロート	94
8.6	キャプション	95
9	フォントコマンド	96

10	相互参照	99
10.1	目次の類	99
10.2	参考文献	104
10.3	索引	106
10.4	脚注	107
11	段落の頭へのグルー挿入禁止	110
12	いろいろなロゴ	114
13	amsmath との衝突の回避	114
14	初期設定	115
付録 A	和文ドライバの仕様 	119
付録 B	和文ドライバ：minimal 	120
B.1	補助マクロ	120
B.2	(u)pTeX 用の設定	123
B.3	pdfTeX 用の処理	127
B.4	X _Y TeX 用の処理	128
B.5	後処理（エンジン共通）	129
付録 C	和文ドライバ：standard 	132
C.1	準備	132
C.2	和文ドライバパラメタ	132
C.3	共通処理 (1)	133
C.4	pTeX 用設定	141
C.5	pdfTeX 用設定：CJK + bxcjkatype	145
C.6	X _Y TeX 用設定：xeCJK + zxjatype	147
C.7	LuaTeX 用設定：LuaTeX-ja	149
C.8	共通処理 (2)	153
付録 D	和文ドライバ：modern 	154
D.1	フォント設定	154
D.2	fixltx2e 読込	154
D.3	和文カテゴリコード	155
D.4	完了	155
付録 E	和文ドライバ：pandoc 	155
E.1	準備	155
E.2	和文ドライバパラメタ	156
E.3	dupload システム	157

E.4	lang 変数	158
E.5	geometry 変数	161
E.6	CJKmainfont 変数	161
E.7	Option clash 対策	161
E.8	レイアウト上書き禁止	162
E.9	paragraph のマーク	163
E.10	全角空白文字	163
E.11	hyperref 対策	164
E.12	Pandoc 要素に対する和文用の補正	164
E.13	ifPDFTeX スイッチ	166
E.14	完了	167
付録 F	補助パッケージ一覧 	167
付録 G	補助パッケージ：bxjscompat 	167
G.1	準備	167
G.2	X _Y TeX 部分	168
G.3	LuaTeX 部分	169
G.4	完了	170
付録 H	補助パッケージ：bxjscjkat 	170
H.1	準備	170
H.2	和文カテゴリコードの設定	171
H.3	ギリシャ・キリル文字の扱い	172
H.4	初期設定	179
H.5	完了	180
付録 I	補助パッケージ：bxjspandoc 	180
I.1	準備	180
I.2	パッケージオプション	180
I.3	パッケージ読込の阻止	181
I.4	fixltx2e パッケージ	181
I.5	cmap パッケージ	181
I.6	microtype パッケージ	182
I.7	Unicode 文字変換対策	182
I.8	PandoLa モジュール	183
I.9	完了	183

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。インストール時のモジュール指定は以下のようである。

<code><article></code>	<code>bxjsarticle.cls</code>	短いレポート（章なし）
<code><report></code>	<code>bxjsreport.cls</code>	長いレポート（章あり）
<code><book></code>	<code>bxjsbook.cls</code>	書籍用
<code><slide></code>	<code>bxjsslide.cls</code>	スライド用

本ドキュメントクラスは奥村晴彦氏および日本語 TeX 開発コミュニティによる「pL^AT_EX 2_ε 新ドキュメントクラス」に改変を加えたものである。本ドキュメントクラスに関する説明は全てこの形式の枠の中に記す。枠の外にあるものは原版著者による原版に対する解説である。

これは L^AT_EX3 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 T_EX 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、私のものもそれに準じて modified BSD とすることにします。

[2016-07-13] 日本語 T_EX 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 upL^AT_EX 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。

<code><article></code>	<code>jsarticle.cls</code>	論文・レポート用
<code><book></code>	<code>jsbook.cls</code>	書籍用
<code><report></code>	<code>jsreport.cls</code>	レポート用
<code><jspf></code>	<code>jspf.cls</code>	某学会誌用
<code><kiyou></code>	<code>kiyou.cls</code>	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、`pLATEX` や `LATEX` の不都合な点に対して、クラスファイル内で独自に対策を施していました。しかし、2016 年以降、コミュニティ版 `pLATEX` が次第に対策コードをカーネル内に取り込むようになりました。そこで、新しい `pLATEX` カーネルと衝突しないように、日付が古い場合だけパッチをあてる場合があります。この処理に使用するフラグを定義します。

```
16 \newif\ifjsc@needsp@tch
17 \jsc@needsp@tchfalse
```

■BXJS クラス特有の設定

長さ値の指定で式を利用可能にするため `calc` を読み込む。

```
18 \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。

```
19 \RequirePackage{keyval}
```

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`、`ifpdf`

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`

`\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。)

```
20 \def\jsAtEndOfClass{%
21   \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。

```
22 \IfFileExists{bxjscompat.sty}{%
23   \RequirePackage{bxjscompat}%
24 }
```

`\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等：`\jsArticle=bxjsarticle`、`\jsBook=bxjsbook`、`\jsReport=bxjsreport`、`\jsSlide=bxjsslide`。

```
25 \let\jsArticle=a
26 \let\jsBook=b
27 \let\jsReport=r
28 \let\jsSlide=s
29 %<article>\let\jsDocClass\jsArticle
```

```

30 %<book>\let\jsDocClass\jsBook
31 %<report>\let\jsDocClass\jsReport
32 %<slide>\let\jsDocClass\jsSlide

\jsEngine [暗黙文字トークン] エンジン (TEX の種類) の種別: j = pTEX 系、x = XYTEX、p =
pdfTEX (含 DVI モード)、l = LuaTEX、J = NTT jTEX、O = Omega 系、n = 以上の何
れでもない。
33 \let\jsEngine=n
34 \def\bxjs@test@engine#1#2{%
35   \edef\bxjs@tmpa{\string#1}%
36   \edef\bxjs@tmpb{\meaning#1}%
37   \ifx\bxjs@tmpa\bxjs@tmpb #2\fi}
38 \bxjs@test@engine\kanjiskip{\let\jsEngine=j}
39 \bxjs@test@engine\jintercharskip{\let\jsEngine=J}
40 \bxjs@test@engine\Omegaversion{\let\jsEngine=O}
41 \bxjs@test@engine\XeTeXversion{\let\jsEngine=x}
42 \bxjs@test@engine\pdftexversion{\let\jsEngine=p}
43 \bxjs@test@engine\luatexversion{\let\jsEngine=l}

\ifjsWithupTeX [スイッチ] エンジンが (内部漢字コードが Unicode の) upTEX であるか。
44 \newif\ifjsWithupTeX
45 \ifx\ucs\@undefined\else \ifnum\ucs"3000="3000
46   \jsWithupTeXtrue
47 \fi\fi
48 \let\if@jsc@uplatex\ifjsWithupTeX

\ifjsWithpTeXng [スイッチ] エンジンが pTEX-ng であるか。
49 \newif\ifjsWithpTeXng
50 \bxjs@test@engine\ngbanner{\jsWithpTeXngtrue}

\ifjsWitheTeX [スイッチ] エンジンが  $\epsilon$ -TEX 拡張をもつか。
51 \newif\ifjsWitheTeX
52 \bxjs@test@engine\eTeXversion{\jsWitheTeXtrue}

非サポートのエンジンの場合は強制終了させる。
※ NTT jTEX と Omega 系。
53 \let\bxjs@tmpa\relax
54 \ifx J\jsEngine \def\bxjs@tmpa{NTT-jTEX}\fi
55 \ifx O\jsEngine \def\bxjs@tmpa{Omega}\fi
56 \ifx\bxjs@tmpa\relax \expandafter\@gobble
57 \else
58   \ClassError\bxjs@clsname
59   {The engine in use (\bxjs@tmpa) is not supported}
60   {It's a fatal error. I'll quit right now.}
61   \expandafter\@firstofone
62 \fi{\endinput\@@end}

LuaTEX の場合、本クラス用の Lua モジュールを用意する。
63 \ifx l\jsEngine

```

```

64 \directlua{ bxjs = {} }
65 \fi

```

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

```

66 \ifjsWithTeX \let\bxjs@protected\protected
67 \else \let\bxjs@protected\empty
68 \fi

```

`\bxjs@robust@def` 無引数の頑強な命令を定義する。

```

69 \ifjsWithTeX
70 \def\bxjs@robust@def{\protected\def}
71 \else
72 \def\bxjs@robust@def{\DeclareRobustCommand*}
73 \fi

```

`\ifjsInPdfMode` [スイッチ] pdfTeX / LuaTeX が PDF モードで動作しているか。
 ※ LuaTeX 0.8x 版でのプリミティブ名変更に対応。

TODO: `iftex` パッケージ (+ 自前実装) に置き換える。

```

74 \newif\ifjsInPdfMode
75 \@nameuse{ImposeOldLuaTeXBehavior}
76 \let\bxjs@tmpa\PackageWarningNoLine
77 \let\PackageWarningNoLine\PackageInfo % suppress warning
78 \RequirePackage{ifpdf}
79 \let\PackageWarningNoLine\bxjs@tmpa
80 \@nameuse{RevokeOldLuaTeXBehavior}
81 \let\ifjsInPdfMode\ifpdf

```

`\ifbxjs@explIII` [スイッチ] `expl3` がカーネルに組み込まれているか。

```

82 \newif\ifbxjs@explIII
83 \@ifl@t@r{fmtversion{2020/02/02}}{\bxjs@explIIItrue}{}

```

`\ifbxjs@TUenc` [スイッチ] L^AT_EX の既定のフォントエンコーディングが TU であるか。

※ 2017 年 1 月以降の L^AT_EX カーネルにおいて「Unicode を表す L^AT_EX 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 L^AT_EX を Xe_LTeX または LuaTeX で動かしている場合は、既定のエンコーディングが TU になる。それ以外の場合は、既定のエンコーディングは OT1 である。

```

84 \newif\ifbxjs@TUenc
85 \def\bxjs@tmpa{TU}\edef\bxjs@tmpb{\f@encoding}
86 \ifx\bxjs@tmpa\bxjs@tmpb
87 \bxjs@TUenctrue
88 \fi

```

`\ifbxjs@old@hook@system` [スイッチ] L^AT_EX 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。

```

89 \newif\ifbxjs@old@hook@system
90 \@ifl@t@r{fmtversion{2020/10/01}}{\bxjs@old@hook@systemtrue}

```

`\bxjs@CGHN` L^AT_EX 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^AT_EX 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^AT_EX では“package/after/PKG”に展開される。

```
91 \ifl@t@r\fmtversion{2021/11/15}{%
92   \def\bxjs@CGHN#1{#1}%
93 }{%else
94   \def\bxjs@CGHN#1{\bxjs@CGHN@a#1//}%
95   \def\bxjs@CGHN@a#1/#2/#3//{#1/#3/#2}}
```

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

```
96 \@gobbletwo\if\if \def\bxjs@cond#1\fi{%
97   #1\expandafter\@firstoftwo
98   \else\expandafter\@secondoftwo
99   \fi}
```

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

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

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

```
102 \def\bxjs@csletcs#1#2{%
103   \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。ただし片方が空の場合は , を入れない。完全展開可能。

```
104 \def\bxjs@catopt#1#2{%
105   #1\if\relax#1\relax\else\if\relax#2\relax\else,\fi\fi#2}
```

`\bxjs@ifplus` `\@ifstar` の + 版。

```
106 \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。

```
107 \def\bxjs@trim#1{\expandafter\bxjs@trim@a#1\@nil#1}
108 \def\bxjs@trim@a{\futurelet\bxjs@tmpb\bxjs@trim@b}
109 \def\bxjs@trim@b{\bxjs@cond\ifx\bxjs@tmpb\@sptoken\fi
110   {\bxjs@trim@c\bxjs@trim@a}{\bxjs@trim@d}}
111 \def\bxjs@trim@c#1 {#1}
112 \def\bxjs@trim@d#1\@nil{\bxjs@trim@e#1\@nil: \@nil\@nnil}
113 \def\bxjs@trim@e#1 \@nil#2\@nnil{\bxjs@cond\ifx\@nnil#2\@nnil\fi
114   {\bxjs@trim@f#1\@nnil}{\bxjs@trim@e#1\@nil: \@nil\@nnil}}
115 \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。

```

116 \def\bxjs@set@array@from@clist#1#2{%
117   \@tempcnta\z@
118   \@for\bxjs@tmpa:=\@empty#2\do{%
119     \bxjs@trim\bxjs@tmpa \bxjs@cslet{#1/\the\@tempcnta}\bxjs@tmpa
120     \advance\@tempcnta\@ne}
121   \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。

```

122 \let\c@bxjs@tempcnta\@tempcnta
123 \def\bxjs@gset@tempcnta{\setcounter{bxjs@tempcnta}}

```

\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）。

```

124 \def\jsSetQHLlength#1#2{%
125   \begingroup
126   \bxjs@parse@qh{#2}%
127   \ifx\bxjs@tmpb\relax
128     \setlength\@tempdima{#2}%
129     \xdef\bxjs@g@tmpa{\the\@tempdima}%
130   \else \global\let\bxjs@g@tmpa\bxjs@tmpb
131   \fi
132   \endgroup
133   #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^AT_EX 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。

```

134 \if j\jsEngine \def\bxjs@parse@qh@units{zw,zh}
135 \else \def\bxjs@parse@qh@units{trueQ,trueH,Q,H,zw,zh}
136 \fi
137 \def\bxjs@parse@qh#1{%
138   \let\bxjs@tmpb\relax
139   \@for\bxjs@tmpa:=\bxjs@parse@qh@units\do{%
140     \ifx\bxjs@tmpb\relax
141       \edef\bxjs@next{{\bxjs@tmpa}{#1}}%
142       \expandafter\bxjs@parse@qh@a\csname bxjs@unit@\bxjs@tmpa\expandafter
143       \endcsname\bxjs@next
144     \fi}}
145 \def\bxjs@parse@qh@a#1#2#3{%
146   \def\bxjs@next##1#2\@nil##2\@nnil{\bxjs@parse@qh@b{##1}{##2}#1}%
147   \bxjs@next#3\@nil#2\@nil\@nnil}
148 \def\bxjs@parse@qh@b#1#2#3{%
149   \ifx\@nnil#2\@nnil\else

```

```

150 \ifx#3\relax
151 \ClassError\bxjs@clsname
152 {You cannot use '\bxjs@tmpa' here}{\@ehc}%
153 \def\bxjs@tmpb{Opt}%
154 \else
155 \@tempdimb#3\relax \@tempdimb#1\@tempdimb
156 \edef\bxjs@tmpb{\the\@tempdimb}%
157 \fi
158 \fi}

今の段階では Q/H だけが使用可能。

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

```

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

```

160 \@onlypreamble\bxjs@begin@document@hook
161 \let\bxjs@begin@document@hook\@empty
162 \AtBeginDocument{\bxjs@begin@document@hook}

```

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

```

163 \@onlypreamble\bxjs@post@option@hook
164 \let\bxjs@post@option@hook\@empty

```

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

```

165 \@onlypreamble\bxjs@pre@jadriver@hook
166 \let\bxjs@pre@jadriver@hook\@empty

```

一時的な手続き用の制御綴。

```

167 \@onlypreamble\bxjs@tmpdo
168 \@onlypreamble\bxjs@tmpdo@a
169 \@onlypreamble\bxjs@tmpdo@b
170 \@onlypreamble\bxjs@tmpdo@c
171 \@onlypreamble\bxjs@tmpdo@d

```

`\jsInhibitGlue` は `\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。

```

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

```

■環境検査 🐛 \TeX 処理系のバージョンがサポート対象であることを検査する。

※現状での処理系バージョン要件は「 \TeX は 0.997 版（2007 年頃）以上」という現実離れしたものになっている。

TODO: 3.0 版において、もっと現実的なバージョン要件を定める予定。多分「本体」と「標準和文ドライバ」で条件を分けることになる。

```

174 \@tempswatrue
175 \if x\jsEngine
176 \ifdim\the\XeTeXversion\XeTeXrevision\p@<0.997\p@
177 \@tempswafalse \fi
178 \fi

```

非サポートのバージョン場合は強制終了させる。

```
179 \if@tempswa \expandafter\@gobble
180 \else
181   \ClassError\bxjs@clsname
182   {The engine in use is all too old}
183   {It's a fatal error. I'll quit right now.}
184   \expandafter\@firstofone
185 \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。

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

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。

```
196 \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。

```
197 \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。

```
198 %<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。

```
199 %<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。

```
200 %<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。

201 `\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 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載しては 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 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、 \LaTeX 2_ϵ の `b5paper` は $250\text{ mm} \times 176\text{ mm}$ ですが、 $\text{p}\text{\LaTeX 2}_\epsilon$ の `b5paper` は $257\text{ mm} \times 182\text{ mm}$ になっています。ここでは $\text{p}\text{\LaTeX 2}_\epsilon$ にならって JIS に従いました。

デフォルトは `a4paper` です。

`b5var` (B5 変形, $182\text{ mm} \times 230\text{ mm}$), `a4var` (A4 変形, $210\text{ mm} \times 283\text{ mm}$) を追加しました。

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

```
202 \@onlypreamble\bxjs@setpaper
203 \def\bxjs@setpaper#1{\def\bxjs@param@paper{#1}}
204 \newif\ifbxjs@iso@bsize
205 \DeclareOption{iso-bsize}{\bxjs@iso@bsizetrue}
206 \@onlypreamble\bxjs@setpaper@bsize
207 \def\bxjs@setpaper@bsize#1{\edef\bxjs@param@paper{%
208   b#1\ifbxjs@iso@bsize paper\else jfi}}
209 \DeclareOption{a3paper}{\bxjs@setpaper{a3paper}}
210 \DeclareOption{a4paper}{\bxjs@setpaper{a4paper}}
211 \DeclareOption{a5paper}{\bxjs@setpaper{a5paper}}
212 \DeclareOption{a6paper}{\bxjs@setpaper{a6paper}}
213 \DeclareOption{b4paper}{\bxjs@setpaper@bsize{4}}
214 \DeclareOption{b5paper}{\bxjs@setpaper@bsize{5}}
215 \DeclareOption{b6paper}{\bxjs@setpaper@bsize{6}}
216 \DeclareOption{a4j}{\bxjs@setpaper{a4paper}}
217 \DeclareOption{a5j}{\bxjs@setpaper{a5paper}}
218 \DeclareOption{b4j}{\bxjs@setpaper{b4j}}
219 \DeclareOption{b5j}{\bxjs@setpaper{b5j}}
220 \DeclareOption{a4var}{\bxjs@setpaper{{210truemm}{283truemm}}}
221 \DeclareOption{b5var}{\bxjs@setpaper{{182truemm}{230truemm}}}
222 \DeclareOption{letterpaper}{\bxjs@setpaper{letterpaper}}
223 \DeclareOption{legalpaper}{\bxjs@setpaper{legalpaper}}
224 \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。

```
225 \@for\bxjs@tmpa:={%
226   a0,a1,a2,c0,c1,c2,c3,c4,c5,c6,ansia,ansib,ansic,ansid,ansie%
227 }\do{\edef\bxjs@next{%
228   \noexpand\DeclareOption{\bxjs@tmpa paper}%
229   {\noexpand\bxjs@setpaper{\bxjs@tmpa paper}}%
230 }\bxjs@next}
231 \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。

```
232 \@for\bxjs@tmpa:={0,1,2,3}\do{\edef\bxjs@next{%
233   \noexpand\DeclareOption{b\bxjs@tmpa paper}%
234   {\noexpand\bxjs@setpaper@bsize{\bxjs@tmpa}}%
235 }\bxjs@next}
```

Pandoc では用紙サイズ指定について「後ろに paper を付けた名前のオプション」を指定する。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、「用紙サイズを custom とすると何も設定しない」ようにするため custompaper というオプションを用意する。

```
236 \DeclareOption{a4varpaper}{\bxjs@setpaper{{210truecm}{283truecm}}}
237 \DeclareOption{b5varpaper}{\bxjs@setpaper{{182truecm}{230truecm}}}
238 \DeclareOption{screenpaper}{\bxjs@setpaper{screen}}
239 \DeclareOption{custompaper}{}
```

■横置き 用紙の縦と横の長さを入れ換えます。

```
240 \newif\if@landscape
241 \@landscapefalse
242 \DeclareOption{landscape}{\@landscapetrue}
```

■slide オプション slide を新設しました。

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

```
243 \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。

```
244 %<!slide>\@slidefalse
245 %<slide>\@slidetrue
```

■サイズオプション 10pt, 11pt, 12pt のほかに、8pt, 9pt, 14pt, 17pt, 21pt, 25pt, 30pt, 36pt, 43pt を追加しました。これは等比数列になるように選んだものです（従来の

20pt も残しました)。`\@ptsize` の定義が変だったのでご迷惑をおかけしましたが、標準的なドキュメントクラスと同様にポイント数から 10 を引いたものに直しました。

[2003-03-22] 14Q オプションを追加しました。

[2003-04-18] 12Q オプションを追加しました。

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

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

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

```
246 \newcommand{\@ptsize}{0}
247 \def\bxjs@param@basefontsize{10pt}
248 \def\jsUnusualPtSize{-20}
```

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

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

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

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

```
250 \jsSetQHLength\@tempdima{#1}%
251 \edef\bxjs@param@basefontsize{\the\@tempdima}%
252 \ifdim\@tempdima=10pt \long\def\@ptsize{0}%
253 \else\ifdim\@tempdima=10.95pt \long\def\@ptsize{1}%
254 \else\ifdim\@tempdima=12pt \long\def\@ptsize{2}%
255 \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。

```
256 \def\bxjs@setjbasefontsize#1{%
257 \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。

```
258 \newif\ifjsc@mag
259 \newif\ifjsc@mag@xreal
260 %\let\jsc@magscale\@undefined

261 \DeclareOption{8pt}{\bxjs@setbasefontsize{8pt}}
262 \DeclareOption{9pt}{\bxjs@setbasefontsize{9pt}}
263 \DeclareOption{10pt}{\bxjs@setbasefontsize{10pt}}
264 \DeclareOption{11pt}{\bxjs@setbasefontsize{10.95pt}}
265 \DeclareOption{12pt}{\bxjs@setbasefontsize{12pt}}
266 \DeclareOption{14pt}{\bxjs@setbasefontsize{14.4pt}}
267 \DeclareOption{17pt}{\bxjs@setbasefontsize{17.28pt}}
```

```

268 \DeclareOption{20pt}{\bxjs@setbasefontsize{20pt}}
269 \DeclareOption{21pt}{\bxjs@setbasefontsize{20.74pt}}
270 \DeclareOption{25pt}{\bxjs@setbasefontsize{24.88pt}}
271 \DeclareOption{30pt}{\bxjs@setbasefontsize{29.86pt}}
272 \DeclareOption{36pt}{\bxjs@setbasefontsize{35.83pt}}
273 \DeclareOption{43pt}{\bxjs@setbasefontsize{43pt}}
274 \DeclareOption{12Q}{\bxjs@setjbasefontsize{3mm}}
275 \DeclareOption{14Q}{\bxjs@setjbasefontsize{3.5mm}}
276 \DeclareOption{10ptj}{\bxjs@setjbasefontsize{10pt}}
277 \DeclareOption{10.5ptj}{\bxjs@setjbasefontsize{10.5pt}}
278 \DeclareOption{11ptj}{\bxjs@setjbasefontsize{11pt}}
279 \DeclareOption{12ptj}{\bxjs@setjbasefontsize{12pt}}

```

JS クラス互換の magstyle 設定オプション。

```

280 \DeclareOption{usemag}{\let\bxjs@magstyle\bxjs@magstyle@@usemag}
281 \DeclareOption{nomag}{\let\bxjs@magstyle\bxjs@magstyle@@nomag}
282 \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 pL^AT_EX 2_ε 本体で行います (plcore.dtx 参照)。オプション `tombow` で日付きのトンボ, オプション `tombo` で日付なしのトンボを出力します。これらはアスキー版のままです。カウンタ `\hour`, `\minute` は pL^AT_EX 2_ε 本体で宣言されています。

取りあえず、pT_EX 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。

```

283 \if j\jsEngine
284 \hour\time \divide\hour by 60\relax
285 \@tempcnta\hour \multiply\@tempcnta 60\relax
286 \minute\time \advance\minute-\@tempcnta
287 \DeclareOption{tombow}{%
288   \tombowtrue \tombowdatetrue
289   \setlength{\@tombowwidth}{.1\p@}%
290   \@bannertoken{%
291     \jobname\space(\number\year-\two@digits\month-\two@digits\day
292     \space\two@digits\hour:\two@digits\minute)}}%
293   \maketombowbox}
294 \DeclareOption{tombo}{%
295   \tombowtrue \tombowdatefalse
296   \setlength{\@tombowwidth}{.1\p@}%
297   \maketombowbox}
298 \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。

```

299 \if j\jsEngine

```

```

300 \DeclareOption{mentuke}{%
301   \tombowtrue \tombowdatefalse
302   \setlength{\@tombowwidth}{\z@}%
303   \maketombowbox}
304 \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。

```

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

```

■二段組 `twocolumn` で二段組になります。

```

308 \DeclareOption{onecolumn}{\@twocolumnfalse}
309 \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。

```

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

```

■右左起こし 書籍では章は通常は奇数ページ起こしになりますが，横組ではこれを `openright` と表すことにしてあります。 `openany` で偶数ページからでも始まるようになります。

[2017-02-24] `openright` は横組では奇数ページ起こし，縦組では偶数ページ起こしを表します。ややこしいですが，これは \LaTeX の標準クラスが西欧の横組事情しか考慮せずに，奇数ページ起こしと右起こしを一緒にしてしまったせいです。縦組での奇数ページ起こしと横組での偶数ページ起こしも表現したいので，`jsclasses` では新たに `openleft` も追加しました。

```

312 %<book|report>\DeclareOption{openright}{\@openrighttrue\@openleftfalse}
313 %<book|report>\DeclareOption{openleft}{\@openlefttrue\@openrightfalse}
314 %<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` \LaTeX の `eqnarray` 環境では `&` でできるアキが大きすぎるようですので，少し小さくします。また，中央の要素も `\displaystyle` にします。

```

315 \def\eqnarray{%
316   \stepcounter{equation}%
317   \def\@currentlabel{\p@equation\theequation}%
318   \global\@eqnswtrue
319   \m@th
320   \global\@eqcnt\z@
321   \tabskip\@centering
322   \let\\\@eqnocr
323   $$$\everycr{}\halign to\displaywidth\bgroup
324     \hskip\@centering$\displaystyle\tabskip\z@skip{##}$\@eqnse1

```



```

325      &\global\@eqcnt\@ne \hfil$\displaystyle{##{}}$\hfil
326      &\global\@eqcnt\tw@ $\displaystyle{##}$\hfil\tabskip\@centering
327      &\global\@eqcnt\thr@@ \hb@xt@\z@\bgroup\hss##\egroup
328      \tabskip\z@skip
329      \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。

```

330 \DeclareOption{leqno}{\input{leqno.clo}}
331 \DeclareOption{fleqn}{\input{fleqn.clo}}
332 % fleqn 用の eqnarray 環境の再定義
333 \def\eqnarray{%
334   \stepcounter{equation}%
335   \def\@currentlabel{\p@equation\theequation}%
336   \global\@eqnswtrue\m@th
337   \global\@eqcnt\z@
338   \tabskip\mathindent
339   \let\=\@eqnocr
340   \setlength\abovedisplayskip{\topsep}%
341   \ifvmode
342     \addtolength\abovedisplayskip{\partopsep}%
343   \fi
344   \addtolength\abovedisplayskip{\parskip}%
345   \setlength\belowdisplayskip{\abovedisplayskip}%
346   \setlength\belowdisplayshortskip{\abovedisplayskip}%
347   \setlength\abovedisplayshortskip{\abovedisplayskip}%
348   $$\everycr{}\halign to\linewidth% $$
349   \bgroup
350   \hskip\@centering$\displaystyle\tabskip\z@skip{##}$\@eqnsele
351   &\global\@eqcnt\@ne \hfil$\displaystyle{##{}}$\hfil
352   &\global\@eqcnt\tw@
353   $\displaystyle{##}$\hfil \tabskip\@centering
354   &\global\@eqcnt\thr@@ \hb@xt@\z@\bgroup\hss##\egroup
355   \tabskip\z@skip\cr
356   }}

```

■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。

```

357 % \DeclareOption{openbib}{%
358 %   \AtEndOfPackage{%
359 %     \renewcommand\@openbib@code{%
360 %       \advance\leftmargin\bibindent
361 %       \itemindent -\bibindent
362 %       \listparindent \itemindent
363 %       \parsep \z@}%
364 %     \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。AMSTeX 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：

```
365 %\let\bxjs@enablejfam\undefined
```

enablejfam オプションの処理。

```
366 \def\bxjs@kv@enablejfam@true{\let\bxjs@enablejfam=t}
367 \def\bxjs@kv@enablejfam@false{\let\bxjs@enablejfam=f}
368 \def\bxjs@kv@enablejfam@default{\let\bxjs@enablejfam\undefined}
369 \define@key{bxjs}{enablejfam}[true]{%
370   \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。

```
371 \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 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。

```
372 \newif\ifjsDraft
373 \DeclareOption{draft}{\jsDrafttrue \overfullrule=5pt }
374 \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, dvi2pdf などでも出力出来るようになる）が使えます。

[2018-02-04] winjis オプションはコソソリ削除しました。代替として、同等なものをパッケージ化 (winjis.sty) して、GitHub にはコソ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。

```
375 \newif\ifbxjs@papersize
376 \bxjs@papersizetrue
377 \DeclareOption{nopapersize}{\bxjs@papersizefalse}
378 \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。

```
379 \newif\if@english
380 \@englishfalse
381 \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_EX 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。

```

382 \newif\if@jslogo \@jslogofalse
383 \DeclareOption{jslogo}{\@jslogotrue}
384 \DeclareOption{nojslogo}{\@jslogofalse}

```

■複合設定オプション

TODO: `\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 $\mathrm{T}_{\mathrm{E}}\mathrm{X}$ 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 128pt 以上の場合でも動作するように修正した。

```

385 \mathchardef\bxjs@isc@ll=128
386 \mathchardef\bxjs@isc@sl=259
387 \def\bxjs@isc@sl@h{65539 }
388 \def\bxjs@invscale#1#2{%
389   \begingroup \@tempdima=#1\relax \@tempdimb#2\p@\relax
390   \ifdim\@tempdima<\bxjs@isc@ll\p@
391     \@tempcnta\@tempdima \multiply\@tempcnta\@cclvi
392     \divide\@tempcnta\@tempdimb \multiply\@tempcnta\@cclvi
393   \else
394     \@tempcnta\@tempdima \divide\@tempcnta\@tempdimb
395     \multiply\@tempcnta\p@ \let\bxjs@isc@sl\bxjs@isc@sl@h
396   \fi
397   \@tempcntb\p@ \divide\@tempcntb\@tempdimb
398   \advance\@tempcnta-\@tempcntb \advance\@tempcnta-\tw@
399   \@tempdimb\@tempcnta\@ne
400   \advance\@tempcnta\@tempcntb \advance\@tempcnta\@tempcntb
401   \advance\@tempcnta\bxjs@isc@sl \@tempdimc\@tempcnta\@ne
402   \@whiledim\@tempdimb<\@tempdimc\do{%
403     \@tempcntb\@tempdimb \advance\@tempcntb\@tempdimc
404     \advance\@tempcntb\@ne \divide\@tempcntb\@tw@
405     \ifdim #2\@tempcntb>\@tempdima
406       \advance\@tempcntb\m@ne \@tempdimc=\@tempcntb\@ne
407     \else \@tempdimb=\@tempcntb\@ne \fi}%
408   \xdef\bxjs@gtmpa{\the\@tempdimb}%
409   \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 $\mathrm{L}^{\mathrm{A}}\mathrm{T}_{\mathrm{E}}\mathrm{X}$ 用の既定テンプレートを用いて他形式から $\mathrm{L}^{\mathrm{A}}\mathrm{T}_{\mathrm{E}}\mathrm{X}$ (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。

```

410 \DeclareOption{pandoc}{%
411   \bxjs@apply@pandoc@opt}
412 \onlypreamble\bxjs@apply@pandoc@opt

```

```
413 \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。

```
414 \g@addto@macro\bxjs@post@option@hook{%
415   \bxjs@oldfontcommandstrue
416   \setkeys{bxjs}{ja=pandoc}%
417   \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）。

```
418 \ifx\bxjs@driver@opt\undefined
419   \def\bxjs@driver@opt{dvipdfmx}%
420   \bxjs@dvi@opttrue
421 \fi
422 \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。

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

■エンジン・ドライバオプション

`\bxjs@engine@given` オプションで明示されたエンジンの種別。

```
427 %\let\bxjs@engine@given\undefined
```

`\bxjs@engine@opt` 明示されたエンジンのオプション名。

```
428 %\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^AT_EX の慣習に従って「ドライバの指定」とみなすべきだから。

```
429 \DeclareOption{autodetect-engine}{%
430   \let\bxjs@engine@given=*}
431 \DeclareOption{latex}{%
432   \def\bxjs@engine@opt{latex}%
433   \let\bxjs@engine@given=n}
434 \DeclareOption{platex}{%
435   \def\bxjs@engine@opt{platex}%
436   \let\bxjs@engine@given=j}
437 \DeclareOption{uplatex}{%
438   \def\bxjs@engine@opt{uplatex}%
439   \let\bxjs@engine@given=u}
```

```

440 \DeclareOption{xelatex}{%
441   \def\bxjs@engine@opt{xelatex}%
442   \let\bxjs@engine@given=x}
443 \DeclareOption{pdflatex}{%
444   \def\bxjs@engine@opt{pdflatex}%
445   \let\bxjs@engine@given=p}
446 \DeclareOption{lualatex}{%
447   \def\bxjs@engine@opt{lualatex}%
448   \let\bxjs@engine@given=l}
449 \DeclareOption{platex-ng}{%
450   \def\bxjs@engine@opt{platex-ng}%
451   \let\bxjs@engine@given=g}
452 \DeclareOption{platex-ng*}{%
453   \def\bxjs@engine@opt{platex-ng*}%
454   \let\bxjs@platexng@nodrv=t%
455   \let\bxjs@engine@given=g}

```

`\bxjs@driver@given` オプションで明示されたドライバの種類。

```

456 %\let\bxjs@driver@given\@undefined
457 \let\bxjs@driver@@dvimode=0
458 \let\bxjs@driver@@dviPDFmx=1
459 \let\bxjs@driver@@pdfmode=2
460 \let\bxjs@driver@@xetex=3
461 \let\bxjs@driver@@dvips=4
462 \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。

```

463 %\let\bxjs@driver@opt\@undefined

464 \DeclareOption{dvips}{%
465   \def\bxjs@driver@opt{dvips}%
466   \let\bxjs@driver@given\bxjs@driver@@dvips}
467 \DeclareOption{dviout}{%
468   \def\bxjs@driver@opt{dviout}%
469   \let\bxjs@driver@given\bxjs@driver@@dvimode}
470 \DeclareOption{xdvi}{%
471   \def\bxjs@driver@opt{xdvi}%
472   \let\bxjs@driver@given\bxjs@driver@@dvimode}
473 \DeclareOption{dviPDFmx}{%
474   \def\bxjs@driver@opt{dviPDFmx}%
475   \let\bxjs@driver@given\bxjs@driver@@dviPDFmx}
476 \DeclareOption{nodvidriver}{%
477   \def\bxjs@driver@opt{nodvidriver}%
478   \let\bxjs@driver@given\bxjs@driver@@none}
479 \DeclareOption{pdftex}{%
480   \def\bxjs@driver@opt{pdftex}%
481   \let\bxjs@driver@given\bxjs@driver@@pdfmode}
482 \DeclareOption{luatex}{%
483   \def\bxjs@driver@opt{luatex}%

```

```

484 \let\bxjs@driver@given\bxjs@driver@@pdfmode}
485 \DeclareOption{xetex}{%
486 \def\bxjs@driver@opt{xetex}%
487 \let\bxjs@driver@given\bxjs@driver@@xetex}

dvipdfmx-if-dvi は 2.0 版より非推奨となった。
488 \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: 互換用オプションを分離する (2.0 版で?)。

`\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 489 \@onlypreamble\bxjs@depre@opt
490 \def\bxjs@depre@opt#1#2{%
491 \ClassWarningNoLine\bxjs@clsname
492 {The old option '#1' is DEPRECATED\MessageBreak
493 and may be abolished in future!\MessageBreak
494 You should instead write:\MessageBreak
495 \space\space #2}}
496 \@onlypreamble\bxjs@depre@opt@do
497 \def\bxjs@depre@opt@do#1#2{%
498 \bxjs@depre@opt{#1}{#2}%
499 \setkeys{bxjs}{#2}}

```

`\ifbxjs@bigcode` 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。

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

`nobigcode` / `bigcode` オプションの定義。

```

501 \DeclareOption{nobigcode}{%
502 \bxjs@bigcodefalse}
503 \DeclareOption{bigcode}{%
504 \bxjs@bigcodetrue}

```

`\ifbxjs@oldfontcommands` `\allowoldfontcommands` を既定で有効にするか。

```
505 \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。

```

506 \DeclareOption{nooldfontcommands}{%
507 \bxjs@oldfontcommandsfalse}

```

```

508 \DeclareOption{oldfontcommands}{%
509   \bxjs@oldfontcommandstrue}


```

■JS クラスのオプションで無効なもの  `ltjsclasses` に倣って警告を出す。

```

510 \DeclareOption{winjis}{%
511   \ClassWarningNoLine\bxjs@clsname
512     {This class does not support `winjis' option}%
513 }
514 \DeclareOption{mingoth}{%
515   \ClassWarningNoLine\bxjs@clsname
516     {This class does not support `mingoth' option}%
517 }
518 \DeclareOption{jis}{%
519   \ClassWarningNoLine\bxjs@clsname
520     {This class does not support `jis' option}%
521 }

```

■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。

```

522 \DeclareOption*{%
523   \def\bxjs@next{\bxjs@safe@setkeys\bxjs}%
524   \expandafter\bxjs@next\expandafter{\CurrentOption}}

```

`\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。

```

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

```

`\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>]` に等値する（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。

```

529 \@onlypreamble\bxjs@declare@enum@option
530 \def\bxjs@declare@enum@option#1#2#3{%
531   \bxjs@csletcs{bxjs@#2}{bxjs@#2@@#3}%
532   \define@key{bxjs}{#1}{%
533     \expandafter\ifx\csname bxjs@#2@@##1\endcsname\relax
534     \bxjs@error@keyval{#1}{##1}%
535     \else \bxjs@csletcs{bxjs@#2}{bxjs@#2@@##1}%
536     \fi}}

```

`\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。


```

537 \@onlypreamble\bxjs@declare@bool@option
538 \def\bxjs@declare@bool@option#1#2#3{%
539   \csname newif\expandafter\endcsname\csname ifbxjs@#2\endcsname
540   \@nameuse{bxjs@#2#3}%
541   \define@key{bxjs}{#1}[true]{%
542     \expandafter\ifx\csname bxjs@#2##1\endcsname\relax
543       \bxjs@error@keyval{#1}{##1}%
544     \else \@nameuse{bxjs@#2##1}%
545     \fi}}

\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。
546 \def\bxjs@set@keyval#1#2#3{%
547   \bxjs@csletcs{bxjs@next}{bxjs@kv@#1@#2}%
548   \ifx\bxjs@next\relax
549     \bxjs@error@keyval{#1}{#2}%
550     #3%
551   \else \bxjs@next
552   \fi}
553 \@onlypreamble\bxjs@error@keyval
554 \def\bxjs@error@keyval#1#2{%
555   \ClassError\bxjs@clsname
556   {Invalid value '#2' for option #1}\@ehc}

\jsScale 〔実数値マクロ〕 和文スケール値。
557 \def\jsScale{0.924715}

\bxjs@base@opt 明示された base オプションの値。
558 %\let\bxjs@base@opt\@undefined

    base オプションの処理。
559 \define@key{bxjs}{base}{%
560   \edef\bxjs@base@opt{#1}%
561   \bxjs@setbasefontsize{#1}}
562 \define@key{bxjs}{fontsize}{\setkeys{bxjs}{base=#1}}

\bxjs@jbase@opt 明示された jbase オプションの値。
563 %\let\bxjs@jbase@opt\@undefined

    jbase オプションの処理。
564 \define@key{bxjs}{jbase}{\edef\bxjs@jbase@opt{#1}}
565 \define@key{bxjs}{jafontsize}{\setkeys{bxjs}{jbase=#1}}

\bxjs@scale@opt 明示された scale オプションの値。
566 %\let\bxjs@scale@opt\@undefined

    scale オプションの処理。
567 \define@key{bxjs}{scale}{%
568   \edef\bxjs@scale@opt{#1}%

```

```

569 \let\jsScale\bxjs@scale@opt}
570 \define@key{bxjs}{jafontscale}{\setkeys{bxjs}{scale=#1}}

    noscale オプションの処理。
TODO: noscale は 3.0 版で廃止の予定。
571 \DeclareOption{noscale}{\bxjs@depre@opt@do{noscale}{scale=1}}

\bxjs@param@mag mag オプションの値。
572 \let\bxjs@param@mag\relax

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

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

\bxjs@jadriver 和文ドライバの名前。
575 \let\bxjs@jadriver\relax
576 %\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: jadriver は 3.0 版で廃止の予定。
    ※単なる ja という指定は無視される (Pandoc 対策)。
577 \define@key{bxjs}{jadriver}{%
578   \bxjs@depre@opt{jadriver}{ja=#1}\edef\bxjs@jadriver@opt{#1}}
579 \define@key{bxjs}{ja}[\relax]{%
580   \ifx\relax#1\else\edef\bxjs@jadriver@opt{#1}\fi}

\jsJaFont 和文フォント設定の名前。
581 \let\jsJaFont\@empty

    jafont オプションの処理。
582 \define@key{bxjs}{jafont}{\edef\jsJaFont{#1}}

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

    japaram オプションの処理。
584 \define@key{bxjs}{japaram}{%
585   \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。
586 \define@key{bxjs}{pandoc}[]{%
587   \ExecuteOptions{pandoc}%
588   \edef\jsJaParam{\bxjs@catopt\jsJaParam{#1}}}
589 \define@key{bxjs}{pandoc+}[]{%
590   \ExecuteOptions{pandoc+}%
591   \edef\jsJaParam{\bxjs@catopt\jsJaParam{#1}}}

```

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

```
592 \let\bxjs@magstyle@@mag=m
593 \let\bxjs@magstyle@@real=r
594 \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。

```
595 \let\bxjs@magstyle@@usemag\bxjs@magstyle@@mag
596 \let\bxjs@magstyle@@nomag\bxjs@magstyle@@real
597 \bxjs@cslet{bxjs@magstyle@@nomag*}\bxjs@magstyle@@xreal
```

`\bxjs@magstyle@@default` は既定の値を表す。

```
598 \let\bxjs@magstyle@@default\bxjs@magstyle@@usemag
599 \ifx l\jsEngine \ifnum\luatexversion>86
600   \let\bxjs@magstyle@@default\bxjs@magstyle@@xreal
601 \fi\fi
602 \ifjsWithpTeXng
603   \let\bxjs@magstyle@@default\bxjs@magstyle@@xreal
604 \fi
605 \let\bxjs@magstyle\bxjs@magstyle@@default
```

magstyle オプションの処理。

```
606 \define@key{bxjs}{magstyle}{%
607   \bxjs@csletcs{bxjs@magstyle}{bxjs@magstyle@@#1}%
608   \ifx\bxjs@magstyle\relax
609     \bxjs@error@keyval{magstyle}{#1}%
610     \let\bxjs@magstyle\bxjs@magstyle@@default
611   \fi}
```

`\bxjs@geometry` geometry オプションの指定値。

```
612 \let\bxjs@geometry@@class=c
613 \let\bxjs@geometry@@user=u
614 \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。

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

`\ifbxjs@dvi@opt` dvi オプションが指定されたか。

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

DVI モードのドライバとドライバ種別との対応。

```
617 \let\bxjs@dvidriver@@dvipdfmx=\bxjs@driver@@dvipdfmx
618 \let\bxjs@dvidriver@@dvips=\bxjs@driver@@dvips
619 \let\bxjs@dvidriver@@dviout=\bxjs@driver@@dvimode
620 \let\bxjs@dvidriver@@xdvi=\bxjs@driver@@dvimode
621 \let\bxjs@dvidriver@@nodvidriver=\bxjs@driver@@none
```

dvi オプションの処理。

```

622 \define@key{bxjs}{dvi}{%
623   \bxjs@csletcs{bxjs@tmpa}{bxjs@dvidriver@@#1}%
624   \ifx\bxjs@tmpa\relax
625     \bxjs@error@keyval{dvi}{#1}%
626   \else

\bxjs@driver@given を未定義にしていることに注意。

627   \def\bxjs@driver@opt{#1}%
628   \let\bxjs@driver@given\@undefined
629   \bxjs@dvi@opttrue
630 \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。
631 \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。
632 \newif\ifbxjs@force@chapterabstract
633 %<book>\bxjs@force@chapterabstracttrue

        layout オプションの処理。
634 \@namedef{bxjs@kv@layout@v1}{%
635 %<book>\bxjs@layout@buggyhmargintrue
636 %<report>\bxjs@force@chapterabstracttrue
637 }
638 \@namedef{bxjs@kv@layout@v2}{%
639 %<book>\bxjs@layout@buggyhmarginfalse
640 %<report>\bxjs@force@chapterabstractfalse
641 }
642 \define@key{bxjs}{layout}{%
643   \bxjs@set@keyval{layout}{#1}{}}

\bxjs@textwidth@limit  textwidth-limit の指定値。
644 %\let\bxjs@textwidth@limit@opt\@undefined
645 \define@key{bxjs}{textwidth-limit}{%
646   \bxjs@depre@opt{textwidth-limit}{textwidth=#1zw}%
647   \edef\bxjs@textwidth@limit@opt{#1}}

\bxjs@textwidth@opt  textwidth の指定値。
648 %\let\bxjs@textwidth@opt\@undefined
649 \define@key{bxjs}{textwidth}{\edef\bxjs@textwidth@opt{#1}}
650 \define@key{bxjs}{line_length}{\setkeys{bxjs}{textwidth=#1}}

\bxjs@number@of@lines@opt  number-of-lines の指定値。
651 %\let\bxjs@number@of@lines@opt\@undefined
652 \define@key{bxjs}{number-of-lines}{\edef\bxjs@number@of@lines@opt{#1}}
653 \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。

```
654 %\let\bxjs@paragraph@mark\@undefined
655 \define@key{bxjs}{paragraph-mark}{%
656   \edef\bxjs@paragraph@mark{#1}}
```

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

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

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

```
658 \bxjs@declare@bool@option{jaspacercmd}{jaspace@cmd}{true}
659 \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。

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

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

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

`\bxjs@everyparhook` everyparhook の指定値。

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

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

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

`\ifbxjs@usezw` [スイッチ] use-zw の指定値。

TODO: zw/nozw は 3.0 版で廃止の予定。

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

`\ifbxjs@disguise@js` [スイッチ] disguise-js の指定値。

TODO: js/nojs は 3.0 版で廃止の予定。

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

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

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

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

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

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

```
683 \bxjs@declare@bool@option{plautopatch}{plautopatch}{false}
684 \g@addto@macro\bxjs@plautopatchfalse{\def\bxjs@plautopatchgiven{false}}
```

■オプションの実行

L^AT_EX の実装では、クラスやパッケージのオプションのトークン列の中に { } 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る(仕様は変わらない)。

※クラスに `\DeclareOption*` がある場合は `\@unusedoptions` は常に空のままであることを利用している。

```
685 \let\bxjs@org@removeelement\@removeelement
686 \def\@removeelement#1#2#3{%
687   \def\reserved@a{#2}%
688   \ifx\reserved@a\@empty \let#3\@empty
689   \else \bxjs@org@removeelement{#1}{#2}{#3}%
690   \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。

```
691 %<article>\ExecuteOptions{a4paper,oneside,onecolumn,notitlepage,final}
692 %<book>\ExecuteOptions{a4paper,twoside,onecolumn,titlepage,openright,final}
693 %<report>\ExecuteOptions{a4paper,oneside,onecolumn,titlepage,openany,final}
694 %<slide>\ExecuteOptions{36pt,a4paper,landscape,oneside,onecolumn,titlepage,final}
695 \ProcessOptions\relax
696 \bxjs@post@option@hook
```

後処理

```
697 \if@slide
698   \def\maybeblue{\@ifundefined{ver@color.sty}{\color{blue}}{}}
699 \fi
700 \if@landscape
701   \setlength\@tempdima {\paperheight}
702   \setlength\paperheight{\paperwidth}
703   \setlength\paperwidth {\@tempdima}
704 \fi
```

■グローバルオプションの整理

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、やはり後のパッケージの読み込み処理で不具合を起こすようである (ProcessOptions* がエラーになる)。従って、このようなオプションは除外することにする。

```
705 \def\bxjs@tmpdo{%
706   \def\bxjs@tmpa{\@gobble}%
707   \expandafter\bxjs@tmpdo@a\@classoptionslist,\@nil,%
708   \let\@classoptionslist\bxjs@tmpa}
709 \def\bxjs@tmpdo@a#1,{%
710   \ifx\@nil#1\relax\else
711     \bxjs@tmpdo@b#1{\@nil
712       \if@tempswa \edef\bxjs@tmpa{\bxjs@tmpa,#1}\fi
713       \expandafter\bxjs@tmpdo@a
714       \fi}
715 \def\bxjs@tmpdo@b#1#2{\bxjs@tmpdo@c}
716 \def\bxjs@tmpdo@c#1\@nil{%
717   \ifx\@nil#1\@nil \@tempwatrue \else \@tempwafalse \fi}
718 \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。

```
719 \@expandtwoargs\@removeelement
720 {papersize}\@classoptionslist\@classoptionslist
721 \@expandtwoargs\@removeelement
722 {10pt}\@classoptionslist\@classoptionslist
723 \@expandtwoargs\@removeelement
724 {noscale}\@classoptionslist\@classoptionslist
```

■使用エンジンの検査・自動判定 ユーザが uplatex オプションの有無により指定したエンジンが、実際に使われているもの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^AT_EX/ upL^AT_EX を自動判別するオプション autodetect-engine を新設しました。upL^AT_EX 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 ot_f パッケージにも uplatex オプションが渡るようにします。

このコードを削除。

[2016-11-11] pL^AT_EX の場合は、オプション uplatex が指定されていれば必ずエラーを出します。autodetect-engine が有効になっていてもエラーを出しますが、これは ot_f パッケージに uplatex オプションが渡ってしまうのを防ぐためです。

正規化前の和文ドライバの値を \bxjs@jadriver に設定する。

```
725 \ifx\bxjs@jadriver@opt\@undefined\else
726   \let\bxjs@jadriver\bxjs@jadriver@opt
```

727 \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。

```
728 \let\bxjs@tmpb\jsEngine
729 \ifx j\bxjs@tmpb\ifjsWithpTeXng
730   \let\bxjs@tmpb=g
731 \fi\fi
732 \ifx j\bxjs@tmpb\ifjsWithupTeX
733   \let\bxjs@tmpb=u
734 \fi\fi
735 \ifx p\bxjs@tmpb\ifjsInPdfMode\else
736   \let\bxjs@tmpb=n
737 \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。)

```
738 \ifx *\bxjs@engine@given
739   \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^AT_EX 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。

```
740 \ifx j\bxjs@engine@given
741   \g@addto@macro\@classoptionslist{,latex}
742 \else\ifx u\bxjs@engine@given
743   \g@addto@macro\@classoptionslist{,uplatex}
744 \fi\fi
745 \fi
746 \ifx\bxjs@engine@given\@undefined\else
747   \ifx\bxjs@engine@given\bxjs@tmpb\else
748     \ClassError\bxjs@clsname
749       {Option '\bxjs@engine@opt' used on wrong engine}\@ehc
750   \fi
751 \fi
```

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

```
752 \ifjsWithpTeXng
753   \g@addto@macro\@classoptionslist{,uplatex}
754 \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。

```
755 \@tempwatrue
756 \ifx \bxjs@driver@given\@undefined\else
757   \ifjsInPdfMode
758     \ifx\bxjs@driver@given\bxjs@driver@@pdfmode\else
759       \@tempwafalse
760     \fi
761   \else\ifx x\jsEngine
```



```

762 \ifx\bxjs@driver@given\bxjs@driver@@xetex\else
763 \@tempswafalse
764 \fi
765 \else
766 \ifx\bxjs@driver@given\bxjs@driver@@pdfmode
767 \@tempswafalse
768 \else\ifx\bxjs@driver@given\bxjs@driver@@xetex
769 \@tempswafalse
770 \fi\fi
771 \ifjsWithpTeXng\ifx\bxjs@driver@given\bxjs@driver@@dvipdfmx\else
772 \@tempswafalse
773 \fi\fi
774 \fi\fi
775 \fi
776 \if@tempswa\else
777 \ClassError\bxjs@clsname
778 {Option '\bxjs@driver@opt' used on wrong engine}\@ehc
779 \fi

```

DVI 出力のエンジンである場合の追加処理。

```

780 \ifjsInPdfMode \@tempswafalse
781 \else\ifx x\jsEngine \@tempswafalse
782 \else\ifjsWithpTeXng \@tempswafalse
783 \else \@tempswatrue
784 \fi\fi\fi
785 \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)。

```

786 \ifx\bxjs@driver@opt\@undefined
787 \if \ifbxjs@explIII T\else\ifx\bxjs@jadriver@opt\@undefined F\else T\fi\fi T%
788 \ClassWarningNoLine\bxjs@clsname
789 {A driver option is MISSING!!\MessageBreak
790 You should properly specify one of the valid\MessageBreak
791 driver options according to the DVI driver\MessageBreak
792 that is in use:\MessageBreak
793 \@spaces dvips, dvipdfmx, dviout, xdvi,\MessageBreak
794 \@spaces nodvidriver}
795 \fi
796 \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。)

```

797 \ifbxjs@dvi@opt
798 \edef\bxjs@next{%
799 \let\noexpand\bxjs@driver@given
800 \csname bxjs@dvidriver@@\bxjs@driver@opt\endcsname
801 \noexpand\g@addto@macro\noexpand\@classoptionslist
802 {,\bxjs@driver@opt}%

```

```

803     }\bxjs@next
804   \fi
805 \fi

```

エンジンが p_TE_X-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。

```

806 \ifjsWithpTeXng
807   \ifx\bxjs@driver@given\bxjs@driver@@dvipdfmx
808     \let\bxjs@platexng@nodrv\@undefined
809   \else\ifx t\bxjs@platexng@nodrv\else
810     \g@addto@macro\@classoptionslist{,dvipdfmx}
811   \fi\fi
812 \fi

```

ドライバが nodviddriver 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。

```

813 \ifx\bxjs@driver@given\bxjs@driver@@none
814   \bxjs@papersizefalse
815 \fi

```

■その他の BXJS 特有の後処理 ☞ 標準の和文ドライバの名前の定数。

```

816 \def\bxjs@@minimal{minimal}
817 \def\bxjs@@standard{standard}
818 \def\bxjs@@pandoc{pandoc}
819 \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)p_TE_X である場合は standard に変える。

※ (u)p_TE_X 以外で ja を省略するのは 2.0 版より非推奨となった。

```

820 \ifx\bxjs@jadriver\relax
821   \ifx j\jsEngine
822     \let\bxjs@jadriver\bxjs@@standard
823   \else
824     \ClassWarningNoLine\bxjs@clsname
825     {The option 'ja' is MISSING!!\MessageBreak
826     So 'ja=minimal' is assumed as fallback, but\MessageBreak
827     such implicit setting is now DEPRECATED!\MessageBreak
828     You should write 'ja=minimal' explicitly,\MessageBreak
829     if it is intended}
830     \let\bxjs@jadriver\bxjs@@minimal
831   \fi
832 \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、ifpdf。

※Pandoc モードでは plautopatch の既定値を真とする。

```

833 \ifx\bxjs@jadriver\bxjs@@pandoc \ifx\bxjs@plautopatch@given\@undefined
834   \ifjsWitheTeX

```

```

835 \bxjs@plautopatchtrue
836 \fi\fi\fi
837 \ifx j\jsEngine \ifbxjs@plautopatch
838 \RequirePackage{plautopatch}[2018/08/22]%v0.3
839 \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。

```

840 \ifx\bxjs@jadriver@opt\@undefined\else
841 \ifx\bxjs@engine@given\@undefined
842 \ClassError\bxjs@clsname
843 {An engine option must be explicitly given}%
844 {When you use a Japanese-driver you must specify a correct\MessageBreak
845 engine option.\MessageBreak\@ehc}
846 \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。)

```

847 \ifx\bxjs@magstyle@@default\bxjs@magstyle@@mag\else
848 \ifx\bxjs@magstyle\bxjs@magstyle@@mag
849 \let\bxjs@magstyle\bxjs@magstyle@@default
850 \ClassError\bxjs@clsname
851 {The engine does not support 'magstyle=usemag'}%
852 {LuaTeX v0.87 or later no longer supports the "mag" feature of TeX.\MessageBreak
853 The default value 'nomag*' is used instead.\MessageBreak \@ehc}
854 \fi
855 \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。

```

856 \ifx\bxjs@jbase@opt\@undefined\else
857 \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。

```

858 \jsSetQHLength\@tempdima{\bxjs@jbase@opt}%
859 \bxjs@invscale\@tempdima\jsScale
860 \bxjs@setbasefontsize{\@tempdima}%
861 \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。

```

862 \ifx\bxjs@scale@opt\@undefined\else
863 \ClassWarningNoLine\bxjs@clsname
864 {Redundant 'scale' option is ignored}%
865 \fi
866 \jsSetQHLength\@tempdima{\bxjs@jbase@opt}%
867 \@tempdimb=\bxjs@param@basefontsize\relax

```

```

868 \edef\jsScale{\strip@pt\@tempdimb}%
869 \bxjs@invscale\@tempdima\jsScale
870 \edef\jsScale{\strip@pt\@tempdima}%
871 \fi
872 \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。

```

873 \let\Cjascale\jsScale

```

8bit 欧文 T_EX の場合は、高位バイトをアクティブ化しておく。（和文を含むマクロ定義を
 通用させるため。）

```

874 \if \if p\jsEngine T\else\if n\jsEngine T\else F\fi\fi T
875 \@tempcnta="80 \loop \ifnum\@tempcnta<"100
876 \catcode\@tempcnta\active
877 \advance\@tempcnta\@ne
878 \repeat
879 \fi

```

js オプション指定時は、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。

```

880 \ifbxjs@disguise@js
881 %<book|report>\def\bxjs@js@clsname{jsbook}
882 %<!book&!report>\def\bxjs@js@clsname{jsarticle}
883 \@namedef{ver@\bxjs@js@clsname.cls}{2001/01/01 (bxjs)}
884 \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。

```

885 \g@addto@macro\@classoptionslist{,nosetpagesize}

```

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

```

886 \ifbxjs@oldfontcommands
887 \AtEndOfClass{\allowoldfontcommands}
888 \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^AT_EX 2_ε
 はトンボ出力幅を両側に 1 インチとっていますので、dvips 使用時に

-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^AT_EX 2_ε 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)。

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。

```
889 \newdimen\stockwidth \newdimen\stockheight
890 \begingroup\expandafter\expandafter\expandafter\endgroup
891 \expandafter\ifx\csname iftombow\expandafter\endcsname\csname iftrue\endcsname
892 % \newdimen\stockwidth \newdimen\stockheight
893 \setlength{\stockwidth}{\paperwidth}
894 \setlength{\stockheight}{\paperheight}
895 \advance \stockwidth 2in
896 \advance \stockheight 2in
897 \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。

```
898 %<slide>\def\n@baseline{13}%
899 %<!slide>\ifdim\bxjs@param@basefontsize<10pt \def\n@baseline{15}%
900 %<!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。

```
901 \ifx\bxjs@magstyle\bxjs@magstyle@@mag
902 \jsc@magtrue
903 \else\ifx\bxjs@magstyle\bxjs@magstyle@@xreal
904 \jsc@mag@xrealtrue
905 \fi\fi
```

サイズの変更は T_EX 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)/(10pt) × 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。

```
906 \ifx\bxjs@param@mag\relax
907   \@tempdima=\bxjs@param@basefontsize
908   \advance\@tempdima.001pt \multiply\@tempdima25
909   \divide\@tempdima16384\relax \@tempcnta\@tempdima\relax
910   \edef\bxjs@param@mag{\the\@tempcnta}
911 \else
912 % mag 値が直接指定された場合
913   \bxjs@gset@tempcnta{\bxjs@param@mag}
914   \ifnum\@tempcnta<\z@ \@tempcnta=\z@ \fi
915 % 有効な mag 値の範囲は 1--32768
916   \edef\bxjs@param@mag{\the\@tempcnta}
917   \advance\@tempcnta100000
918   \def\bxjs@tmpa#1#2#3#4#5\@nil{\@tempdima=#2#3#4.#5\p@}
919   \expandafter\bxjs@tmpa\the\@tempcnta\@nil
920   \edef\bxjs@param@basefontsize{\the\@tempdima}
921 \fi
922 \@tempcnta\bxjs@param@mag \advance\@tempcnta100000
923 \def\bxjs@tmpa#1#2#3#4\@nil{\@tempdima=#2#3.#4\p@}
924 \expandafter\bxjs@tmpa\the\@tempcnta\@nil
925 \edef\jsc@magscale{\strip@pt\@tempdima}
926 \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。

\mag する場合（現状はこれが既定）にコードの変更を低減するために、以下では必要に応じて、\jsc@mpt を \p@? と書く。その上で、\mag する場合は ? を無視して \p@ と解釈させ、\mag しない場合は ? を英字扱いにして \p@? という制御綴を \jsc@mpt と同値にする。※（多分 2.0 版あたりで）JS クラスに合わせるため \p@? 表記を止める予定。

```
927 \newdimen\jsc@mpt
928 \newdimen\jsc@mmm
929 \ifjsc@mag
930   \jsc@mpt=1\p@
931   \jsc@mmm=1mm
932   \catcode`\?=9 % \p@? read as \p@
```

```

933 \else
934   \jsc@empt=\jsc@magscale\p@
935   \jsc@mmm=\jsc@magscale mm
936   \catcode\?=11 \let\p@?\jsc@empt
937 \fi
938 \chardef\bxjs@qmc=\catcode\?\relax
939 \g@addto@macro\bxjs@pre@jadriver@hook{\catcode\?=12\relax}

```

ここで p_TE_X 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。

```

940 \newdimen\jsZw
941 \jsZw=10\jsc@empt \jsZw=\jsScale\jsZw
942 \ifbxjs@usezw
943   \providecommand*\zw{\jsZw}
944 \fi

```

\zwspace 全角幅の水平空き。

```

945 \def\zwspace{\hspace{\jsZw\relax}

```

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

```

946 \ifjsc@mag@xreal
947   \RequirePackage{type1cm}
948   \let\jsc@invscale\bxjs@invscale

```

ムニャムニャムニャ……。

```

949   \ifbxjs@TUenc
950     \expandafter\let\csname TU/lmr/m/n/10\endcsname\relax
951   \else
952     \expandafter\let\csname OT1/cmr/m/n/10\endcsname\relax
953   \fi
954   \expandafter\let\csname OMX/cmex/m/n/10\endcsname\relax
955   \let\jsc@get@external@font\get@external@font
956   \def\get@external@font{%
957     \jsc@preadjust@extract@font
958     \jsc@get@external@font}
959   \def\jsc@fstrunc#1{%
960     \edef\jsc@tmpa{\strip@pt#1}%
961     \expandafter\jsc@fstrunc@a\jsc@tmpa.****\@nil}
962   \def\jsc@fstrunc@a#1.#2#3#4#5#6\@nil{%
963     \if#5*\else
964       \edef\jsc@tmpa{#1%
965         \ifnum#2#3>\z@ .#2\ifnum#3>\z@ #3\fi\fi}%
966     \fi}
967   \def\jsc@preadjust@extract@font{%
968     \let\jsc@req@size\f@size
969     \dimen@f@size\p@ \jsc@invscale\dimen@\jsc@magscale
970     \advance\dimen@.005pt\relax \jsc@fstrunc\dimen@

```

```

971 \let\jsc@ref@size\jsc@tmpa
972 \let\f@size\jsc@ref@size}
973 \def\execute@size@function#1{%
974 \let\jsc@c@ref@size\f@size
975 \let\f@size\jsc@req@size
976 \csname s@fct@#1\endcsname}
977 \let\jsc@DeclareErrorFont\DeclareErrorFont
978 \def\DeclareErrorFont#1#2#3#4#5{%
979 \@tempdimc#5\p@ \@tempdimc\jsc@magscale\@tempdimc
980 \edef\jsc@tmpa{{#1}{#2}{#3}{#4}{\strip@pt\@tempdimc}}
981 \expandafter\jsc@DeclareErrorFont\jsc@tmpa}
982 \def\gen@sfcnt{%
983 \edef\mandatory@arg{\mandatory@arg\jsc@c@ref@size}%
984 \empty@sfcnt}
985 \def\genb@sfcnt{%
986 \edef\mandatory@arg{%
987 \mandatory@arg\expandafter\genb@x\jsc@c@ref@size..\@}%
988 \empty@sfcnt}
989 \ifbxjs@TUenc\else
990 \DeclareErrorFont{OT1}{cmr}{m}{n}{10}
991 \fi
992 \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 993 \def\jsc@smallskip{\vspace\jsc@smallskipamount}
\jsc@bigskip 994 %\def\jsc@medskip{\vspace\jsc@medskipamount}
995 %\def\jsc@bigskip{\vspace\jsc@bigskipamount}

```

```

\jsc@smallskipamount
\jsc@medskipamount 996 \newskip\jsc@smallskipamount
\jsc@bigskipamount 997 \jsc@smallskipamount=3\jsc@mpt plus 1\jsc@mpt minus 1\jsc@mpt
998 %\newskip\jsc@medskipamount
999 %\jsc@medskipamount =6\jsc@mpt plus 2\jsc@mpt minus 2\jsc@mpt
1000 %\newskip\jsc@bigskipamount
1001 %\jsc@bigskipamoun =12\jsc@mpt plus 4\jsc@mpt minus 4\jsc@mpt

```

`\paperwidth`, `\paperheight` を `\mag` にあわせてスケールしておきます (☆)。

[2016-07-11] 新しく追加した `\stockwidth`, `\stockheight` も `\mag` にあわせてスケールします。

[2017-01-11] トンボオプションが指定されているとき「だけ」`\stockwidth`, `\stockheight` が定義されています。

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

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

```
1002 % \ifpapersize
1003 %   \setlength{\@tempdima}{\paperwidth}
1004 %   \setlength{\@tempdimb}{\paperheight}
1005 %   \iftombow
1006 %     \advance \@tempdima 2truein
1007 %     \advance \@tempdimb 2truein
1008 %   \fi
1009 %   \AtBeginDvi{\special{pdf: pagesize width \the\@tempdima\space height \the\@tempdimb}}
1010 % \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。

\@ 欧文といえば、L^AT_EX の `\def\@{\spacefactor\@m}` という定義（\@m は 1000）では `I watch TV\@.` と書くと V とピリオドのペアカーニングが効かなくなります。そこで、次のような定義に直し、`I watch TV.\@` と書くことにします。

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

BXJS クラスでの変更点：

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

```
1011 \chardef\bxjs@periodchar=`\
1012 \bxjs@protected\def\bxjs@SE{%
1013   \ifnum\spacefactor<\@m \spacefactor\@m
1014   \else \spacefactor\sfcode\bxjs@periodchar
1015   \fi}
1016 \ifbxjs@fix@at@cmd
1017   \def\@{\bxjs@SE{}}
1018 \fi
```

4 フォントサイズ

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

```
\@setfontsize{\normalsize}{10}{16}
```

のようにして行います。これは

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

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

<code>\@vpt</code>	5	<code>\@vipt</code>	6	<code>\@viipt</code>	7
<code>\@viipt</code>	8	<code>\@ixpt</code>	9	<code>\@xpt</code>	10
<code>\@xipt</code>	10.95	<code>\@xiipt</code>	12	<code>\@xivpt</code>	14.4

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

`\kanjiskip` は `pLATEX 2ε` で `0pt plus .4pt minus .5pt` に設定していますが、これはそもそも文字サイズの変更に応じて変わるべきものです。それに、プラスになったりマイナスになったりするの、追い出しと追い込みの混在が生じ、統一性を欠きます。なるべく追い出しになるようにプラスの値だけにしたいところですが、ごくわずかなマイナスは許すことにしました。

`\xkanjiskip` については、四分つまり全角の 1/4 を標準として、追い出すために三分あるいは二分まで延ばすのが一般的ですが、ここでは Times や Palatino のスペースがほぼ四分であることに着目して、これに一致させています。これなら書くときにスペースを空けても空けなくても同じ出力になります。

`\parindent` については、0 (以下) でなければ全角幅 (1zw) に直します。

[2008-02-18] `english` オプションで `\parindent` を 1em にしました。

`\setfontsize` `\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 `\setfontsize` に対してパッチを当てるように変更。

```
1019 \def\bxjs@tmpa{\def\setfontsize##1##2##3%
1020 \expandafter\bxjs@tmpa\expandafter{%
1021   \setfontsize{#1}{#2}{#3}%
1022 % 末尾にコードを追加
1023   \expandafter\def\expandafter\size@update\expandafter{%
1024     \size@update
1025     \jsFontSizeChanged}%
1026 }
```

`\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。

```
1027 \newcommand*\jsFontSizeChanged{%
1028   \jsZw=\f@size\p@
1029   \jsZw=\jsScale \jsZw
1030   \ifdim\parindent>\z@
1031     \if@english \parindent=1em
1032     \else       \parindent=1\jsZw
```

```

1033 \fi
1034 \fi\relax
1035 \jsResetDimen}

```

`\jsResetDimen` ユーザ定義用のフック。

```

1036 \newcommand*\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。

```

1037 \ifjsc@mag
1038 \let\jsc@setfontsize\@setfontsize
1039 \else
1040 \def\jsc@setfontsize#1#2#3{%
1041 \@setfontsize#1{#2\jsc@mpt}{#3\jsc@mpt}}
1042 % microtype 対策
1043 \ifjsWithTeX\if j\jsEngine\else
1044 \def\jsc@setfontsize#1#2#3{%
1045 \edef\bxjs@sfs@next{%
1046 \unexpanded{\@setfontsize#1}%
1047 {\the\dimexpr#2\jsc@mpt\relax}{\the\dimexpr#3\jsc@mpt\relax}%
1048 }\bxjs@sfs@next}
1049 \fi\fi
1050 \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。

```

1051 \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。

```

1052 \newif\ifnarrowbaselines
1053 \if@english
1054   \narrowbaselinestrue
1055 \fi
1056 \def\narrowbaselines{%
1057   \narrowbaselinestrue
1058   \skip0=\abovedisplayskip
1059   \skip2=\abovedisplayshortskip
1060   \skip4=\belowdisplayskip
1061   \skip6=\belowdisplayshortskip
1062 % 一時的に警告を無効化する
1063   \let\bxjs@save@nomath\@nomath
1064   \let\@nomath\@gobble
1065   \@currsize\selectfont
1066   \let\@nomath\bxjs@save@nomath
1067   \abovedisplayskip=\skip0
1068   \abovedisplayshortskip=\skip2
1069   \belowdisplayskip=\skip4
1070   \belowdisplayshortskip=\skip6\relax}
1071 \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。

```

1072 \def\bxjs@if@narrowbaselines{%
1073   \ifnarrowbaselines\expandafter\@firstoftwo
1074   \else \expandafter\@secondoftwo
1075   \fi
1076 }

```

`\normalsize` 標準のフォントサイズと行送りを選ぶコマンドです。

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

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

microtype 対策のため if 文を避ける。

```

1077 \renewcommand{\normalsize}{%
1078   \bxjs@if@narrowbaselines{%
1079     \jsc@setfontsize\normalsize\@xpt\@xipt

```

```

1080 }{%else
1081   \jsc@setfontsize\normalsize\@xpt{\n@baseline}%
1082 }%

```

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_EX 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。

```

1083 \abovedisplayskip 11\p@? \@plus3\p@? \@minus4\p@?
1084 \abovedisplayshortskip \z@ \@plus3\p@?
1085 \belowdisplayskip 9\p@? \@plus3\p@? \@minus4\p@?
1086 \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。

```

1087 \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。

```

1088 %</class>
1089 %<*class|minijs>
1090 %% initialize
1091 \normalsize
1092 %</class|minijs>
1093 %<*class>

```

\Cht 基準となる長さの設定をします。 p_IA_TE_X 2_ε 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 (lzw) 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_TE_X 系以外では未定義なのでここで定義する。

```

1094 \ifx\Cht\@undefined \newdimen\Cht \fi
1095 \ifx\Cdp\@undefined \newdimen\Cdp \fi
1096 \ifx\Cwd\@undefined \newdimen\Cwd \fi
1097 \ifx\Cvs\@undefined \newdimen\Cvs \fi
1098 \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。

```

1099 \setlength\Cht{0.88\jsZw}
1100 \setlength\Cdp{0.12\jsZw}
1101 \setlength\Cwd{1\jsZw}
1102 \setlength\Cvs{\baselineskip}
1103 \setlength\Chs{1\jsZw}

```

\small \small も \normalsize と同様に設定します。 行送りは, \normalsize が 16 ポイントな

ら、割合からすれば $16 \times 0.9 = 14.4$ ポイントになりますが、`\small` の使われ方を考えて、ここでは和文 13 ポイント、欧文 11 ポイントとします。また、`\topsep` と `\parsep` は、元はそれぞれ 4 ± 2 , 2 ± 1 ポイントでしたが、ここではゼロ (`\z@`) にしました。

microtype 対策のため if 文を避ける。後の `\footnotesize` も同様。

```

1104 \newcommand{\small}{%
1105   \bxjs@if@narrowbaselines{%
1106     %<!kiyou>   \jsc@setfontsize\small\@ixpt{11}%
1107     %<kiyou>    \jsc@setfontsize\small{8.8888}{11}%
1108   }{%else
1109     %<!kiyou>   \jsc@setfontsize\small\@ixpt{13}%
1110     %<kiyou>    \jsc@setfontsize\small{8.8888}{13.2418}%
1111   }%
1112   \abovedisplayskip 9\p@? \@plus3\p@? \@minus4\p@?
1113   \abovedisplayshortskip \z@ \@plus3\p@?
1114   \belowdisplayskip \abovedisplayskip
1115   \belowdisplayshortskip \belowdisplayskip
1116   \def\@listi{\leftmargin\leftmargini
1117             \topsep \z@
1118             \parsep \z@
1119             \itemsep \parsep}}

```

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

```

1120 \newcommand{\footnotesize}{%
1121   \bxjs@if@narrowbaselines{%
1122     %<!kiyou>   \jsc@setfontsize\footnotesize\@viipt{9.5}%
1123     %<kiyou>    \jsc@setfontsize\footnotesize{8.8888}{11}%
1124   }{%else
1125     %<!kiyou>   \jsc@setfontsize\footnotesize\@viipt{11}%
1126     %<kiyou>    \jsc@setfontsize\footnotesize{8.8888}{13.2418}%
1127   }%
1128   \abovedisplayskip 6\p@? \@plus2\p@? \@minus3\p@?
1129   \abovedisplayshortskip \z@ \@plus2\p@?
1130   \belowdisplayskip \abovedisplayskip
1131   \belowdisplayshortskip \belowdisplayskip
1132   \def\@listi{\leftmargin\leftmargini
1133             \topsep \z@
1134             \parsep \z@
1135             \itemsep \parsep}}

```

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

`\LARGE`
`\huge`
`\Huge`
`\HUGE`

[2004-11-03] \HUGE を追加。

```
1136 \newcommand{\scriptsize}{\jsc@setfontsize\scriptsize\@viipt\@viipt}
1137 \newcommand{\tiny}{\jsc@setfontsize\tiny\@vpt\@vpt}
1138 \if@twocolumn
1139 %<!kiyou> \newcommand{\large}{\jsc@setfontsize\large\@xiipt{\n@baseline}}
1140 %<kiyou> \newcommand{\large}{\jsc@setfontsize\large{11.111}{\n@baseline}}
1141 \else
1142 %<!kiyou> \newcommand{\large}{\jsc@setfontsize\large\@xiipt{17}}
1143 %<kiyou> \newcommand{\large}{\jsc@setfontsize\large{11.111}{17}}
1144 \fi
1145 %<!kiyou>\newcommand{\Large}{\jsc@setfontsize\Large\@xivpt{21}}
1146 %<kiyou>\newcommand{\Large}{\jsc@setfontsize\Large{12.222}{21}}
1147 \newcommand{\LARGE}{\jsc@setfontsize\LARGE\@xviipt{25}}
1148 \newcommand{\huge}{\jsc@setfontsize\huge\@xxpt{28}}
1149 \newcommand{\Huge}{\jsc@setfontsize\Huge\@xxvpt{33}}
1150 \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。

```
1151 \everydisplay=\expandafter{\the\everydisplay \narrowbaselines}
```

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

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。『pL^AT_EX₂_ε 美文書作成入門』（1997 年）では \sffamily \fontseries{sbc} として新ゴ M と合わせましたが、\fontseries{sbc} はちょっと幅が狭いように感じました。

```
1152 % \newcommand{\headfont}{\bfseries}
1153 \newcommand{\headfont}{\sffamily}
1154 % \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 でしたが、2zw にしました。
\columnseprule このスペースの中央に \columnseprule の幅の罫線が引かれます。

```
1155 %<!kiyou>\setlength\columnsep{2\Cwd}
1156 %<kiyou>\setlength\columnsep{28truebp}
1157 \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 1158 \setlength\lineskip{1\jsc@mp}{  
1159 \setlength\normallineskip{1\jsc@mp}{  
\normallineskiplimit 1160 \setlength\lineskiplimit{1\jsc@mp}{  
1161 \setlength\normallineskiplimit{1\jsc@mp}{
```

`\baselinestretch` 実際の行送りが `\baselineskip` の何倍かを表すマクロです。たとえば

```
\renewcommand{\baselinestretch}{2}
```

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

```
1162 \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。

```
1163 \setlength\parskip{\z@}  
1164 \if@slide  
1165 \setlength\parindent{0\p@}  
1166 \else  
1167 \setlength\parindent{1\Cwd}  
1168 \fi
```

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

```
\@highpenalty 1169 \@lowpenalty 51  
1170 \@medpenalty 151  
1171 \@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。

```
1172 % \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。

```
1173 % \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。

```
1174 \onlypreamble\bxjs@bd@pre@geometry@hook
1175 \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。

```
1176 \ifjsc@mag
1177 \mag=\bxjs@param@mag
1178 \fi
1179 \setlength{\topskip}{10\p@?}
```

`\jsSetQHLength` のための和文単位の定義。

```
1180 \def\bxjs@unit@trueQ{0.25trueem}\let\bxjs@unit@trueH\bxjs@unit@trueQ
1181 \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。

```
1182 \@tempswafalse
1183 \def\bxjs@tmpdo{\@ifnextchar\bgroup\bxjs@tmpdo@a\remove@to@nnil}
1184 \def\bxjs@tmpdo@a#1{\edef\bxjs@tmpa{#1}%
1185   \@ifnextchar\bgroup\bxjs@tmpdo@b\remove@to@nnil}
1186 \def\bxjs@tmpdo@b#1{\edef\bxjs@tmpa{\bxjs@tmpa,#1}%
1187   \@ifnextchar\@nnil\bxjs@tmpdo@c\remove@to@nnil}
1188 \def\bxjs@tmpdo@c\@nnil{\@tempswatrue
1189   \edef\bxjs@param@paper{papersize={\bxjs@tmpa}}}
1190 \expandafter\bxjs@tmpdo\bxjs@param@paper\@nnil
```

`W,H` の形式について。

```
1191 \if@tempswa\else
1192   \def\bxjs@tmpa{\@nil,\@nil}
1193   \def\bxjs@tmpdo#1,#2,#3\@nnil{%
1194     \def\bxjs@tmpb{#3}\ifx\bxjs@tmpa\bxjs@tmpb
1195       \@tempswatrue\edef\bxjs@param@paper{papersize={#1,#2}}\fi}
1196   \expandafter\bxjs@tmpdo\bxjs@param@paper,\@nil,\@nil\@nnil
1197 \fi
```

`W*H` の形式について。

```
1198 \if@tempswa\else
1199   \def\bxjs@tmpa{\@nil*\@nil}
1200   \def\bxjs@tmpdo#1*#2*#3\@nnil{%
1201     \def\bxjs@tmpb{#3}\ifx\bxjs@tmpa\bxjs@tmpb
1202       \@tempswatrue\edef\bxjs@param@paper{papersize={#1,#2}}\fi}
1203   \expandafter\bxjs@tmpdo\bxjs@param@paper*\@nil*\@nil\@nnil
1204 \fi
```

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

```
1205 \edef\bxjs@layout@paper{%
1206   \ifjsc@mag truedimen,\fi
1207   \if@landscape landscape,\fi
1208   \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。

```
1209 %<*article|report>
1210 \def\bxjs@layout@base{%
1211   headheight=\topskip,footskip=0.03367\paperheight,%
1212   headsep=\footskip-\topskip,includeheadfoot,%
1213 }
1214 \edef\bxjs@layout{\bxjs@layout@base
1215   hscale=0.76,hmarginratio=1:1,%
1216   vscale=0.83,vmarginratio=1:1,%
1217 }
1218 %</article|report>
1219 %<*book>
1220 \def\bxjs@layout@base{%
1221   headheight=\topskip,headsep=6\jsc@mmm,nofoot,includeheadfoot,%
1222 }
1223 \ifbxjs@layout@buggyhmargin    %---
1224 % アレ
1225 \edef\bxjs@layout{\bxjs@layout@base
1226   hmargin=36\jsc@mmm,hmarginratio=1:1,%
1227   vscale=0.83,vmarginratio=1:1,%
1228 }
1229 \else    %---
1230 % 非アレ
1231 \edef\bxjs@layout{\bxjs@layout@base
1232   hmargin=18\jsc@mmm,%
1233   vscale=0.83,vmarginratio=1:1,%
1234 }
1235 \fi    %---
1236 %</book>
1237 %<*slide>
1238 \def\bxjs@layout@base{%
1239   noheadfoot,%
1240 }
1241 \edef\bxjs@layout{\bxjs@layout@base
1242   hscale=0.9,hmarginratio=1:1,%
1243   vscale=0.944,vmarginratio=1:1,%
1244 }
1245 %</slide>
```

`textwidth` オプションの設定を反映する。

```
1246 %<!*book>
1247 \ifx\bxjs@textwidth@opt\undefined\else
1248   \jsSetQHLlength\@tempdima{\bxjs@textwidth@opt}
```

```

1249 \edef\bxjs@layout{\bxjs@layout width=\the\@tempdima,}
1250 \fi
1251 %</!book>
1252 \ifx\bxjs@number@of@lines@opt\@undefined\else
1253 \bxjs@gsset@tempcnta{\bxjs@number@of@lines@opt}
1254 \edef\bxjs@layout{\bxjs@layout lines=\the\@tempcnta,}
1255 \fi

```

`\fullwidth` [寸法レジスタ] ヘッダ・フッタ領域の横幅。

```
1256 \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。

```

1257 %<*book>
1258 \newcommand\jsTextWidthLimit{40}
1259 \@tempdima=\jsTextWidthLimit\Cwd
1260 \ifx\bxjs@textwidth@limit@opt\@undefined\else
1261 \bxjs@gsset@tempcnta{\bxjs@textwidth@limit@opt}
1262 \@tempdima=\@tempcnta\Cwd
1263 \fi
1264 \ifx\bxjs@textwidth@opt\@undefined\else
1265 \jsSetQHLlength\@tempdima{\bxjs@textwidth@opt}
1266 \fi
1267 \edef\bxjs@textwidth@limit{\the\@tempdima}
1268 \ifdim\@tempdima=\jsTextWidthLimit\Cwd\else
1269 \bxjs@invscale\@tempdima{\strip@pt\Cwd}
1270 \long\edef\jsTextWidthLimit{\strip@pt\@tempdima}
1271 \fi
1272 %</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。

```

1273 \def\bxjs@preproc@layout{%
1274 \edef\bxjs@save@ht@strutbox{\the\ht\strutbox}\ht\strutbox=10\jsc@mppt}

```

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

```
1275 \def\bxjs@postproc@layout{%
```

`geometry` のドライバを再設定する。

```

1276 \ifx\bxjs@geometry@driver\relax\else
1277 \let\Gm@driver\bxjs@geometry@driver
1278 \fi

```

`\ht\strutbox` の値を元に戻す。

```
1279 \ht\strutbox=\bxjs@save@ht@strutbox\relax
```

\textwidth の値を補正する。

```
1280 \ifbxjs@whole@zw@lines
1281   \@tempdimb=\textwidth
1282   \if@twocolumn \@tempdima=2\Cwd \else \@tempdima=1\Cwd \fi
1283   \advance\textwidth.005pt\relax
1284   \divide\textwidth\@tempdima \multiply\textwidth\@tempdima
1285   \advance\@tempdimb-\textwidth
1286   \advance\oddsidemargin 0.5\@tempdimb
1287   \advance\evensidemargin 0.5\@tempdimb
1288 \fi
1289 \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。

```
1290 %<*book>
1291   \@tempdima=\bxjs@textwidth@limit\relax
1292   \ifbxjs@whole@zw@lines
1293     \advance\@tempdima.005pt\relax
1294     \divide\@tempdima\Cwd \multiply\@tempdima\Cwd
1295   \fi
1296   \ifdim\textwidth>\@tempdima
1297     \textwidth=\@tempdima
1298     \addtolength\evensidemargin{\fullwidth-\textwidth}
1299   \fi
1300 %</book>
```

\textheight 関連の調整。

```
1301 \@tempdimb=\textheight
1302 \advance\textheight-\topskip
1303 \advance\textheight.005pt\relax
1304 \divide\textheight\baselineskip \multiply\textheight\baselineskip
1305 \advance\textheight\topskip
1306 \advance\@tempdimb-\textheight
1307 \advance\topmargin0.5\@tempdimb
```

\headheight 関連の調整。

```
1308 \@tempdima=\topskip
1309 \advance\headheight\@tempdima
1310 \advance\topmargin-\@tempdima
```

marginpar 関連の調整。

```
1311 \setlength\marginparsep{\columnsep}
1312 \setlength\marginparpush{\baselineskip}
1313 \setlength\marginparwidth{\paperwidth-\oddsidemargin-1truein%
1314   -\textwidth-10\jsc@mmm-\marginparsep}
1315 \ifbxjs@whole@zw@lines
1316   \divide\marginparwidth\Cwd \multiply\marginparwidth\Cwd
1317 \fi
```

連動する変数。

```

1318 \maxdepth=.5\topskip
1319 \stockwidth=\paperwidth
1320 \stockheight=\paperheight
1321 }

```

`\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。

```

1322 \edef\jsGeometryOptions{%
1323 \bxjs@layout@paper,\bxjs@layout}

```

■ geometry パッケージを読み込む

ムニャムニャ。

```

1324 \def\bxjs@geometry@name{geometry}
1325 \ifbxjs@old@hook@system
1326 \let\bxjs@apply@bd@pre@geometry@hook\AtBeginDocument
1327 \else
1328 \def\bxjs@apply@bd@pre@geometry@hook{%
1329 \AddToHook{begindocument}{\bxjs@geometry@name}}
1330 \fi

```

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

```

1331 \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。

```

1332 \ifbxjs@papersize
1333 \ifx\bxjs@driver@given\bxjs@driver@@dvipdfmx
1334 \PassOptionsToPackage{dvipdfm}{geometry}
1335 \else\ifx\bxjs@driver@given\bxjs@driver@@dvimode
1336 \PassOptionsToPackage{dvipdfm}{geometry}
1337 \fi\fi
1338 \let\bxPapersizeSpecialDone=t
1339 \else
1340 \PassOptionsToPackage{driver=none}{geometry}
1341 \fi

```

ここで `geometry` を読み込む。

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

```

1342 \bxjs@apply@bd@pre@geometry@hook{\bxjs@bd@pre@geometry@hook}
1343 \bxjs@apply@bd@pre@geometry@hook{\ImposeOldLuaTeXBehavior}
1344 \bxjs@preproc@layout
1345 \edef\bxjs@next{%
1346 \noexpand\RequirePackage[\bxjs@layout@paper,\bxjs@layout]{geometry}%

```

```

1347 } \bxjs@next
1348 \bxjs@apply@bd@pre@geometry@hook{\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。

```

1349 \let\bxjs@geometry@driver\Gm@driver
1350 \bxjs@postproc@layout

```

geometry のドライバ自動判別に対する前処理。

```

1351 \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。

```

1352 \ifpackagelater{geometry}{2010/02/12}{\}%else
1353 \PackageError\bxjs@clsname
1354 {Your 'geometry' package is too old (< v5.0)}%
1355 {\@ehc}%
1356 \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。

```

1357 \ifjsWithpTeXng
1358 \ifx\Gm@driver\@empty
1359 \def\Gm@driver{pdftex}%
1360 \fi
1361 \fi}

```

`\setpagelayout` ページレイアウト設定のためのユーザ命令。

```

1362 \def\setpagelayout{%
1363 \bxjs@ifplus{\bxjs@setpagelayout@a\tw@}{\}%else
1364 \@ifstar{\bxjs@setpagelayout@a\@ne}{\bxjs@setpagelayout@a\z@}}
1365 \def\bxjs@setpagelayout@a#1#2{%
1366 \ifcase#1% modify
1367 \def\bxjs@next{\ifjsc@mag truedimen,\fi #2}%
1368 \or% reset(*)
1369 \def\bxjs@next{reset,\bxjs@layout@paper,#2}%
1370 \or% semireset(+)
1371 \def\bxjs@next{reset,\bxjs@layout@paper,\bxjs@layout@base,#2}%
1372 \fi
1373 \bxjs@preproc@layout
1374 \edef\bxjs@next{%
1375 \noexpand\geometry{\bxjs@next}%
1376 }\bxjs@next
1377 \bxjs@postproc@layout}

```

■ geometry パッケージを読み込まない 🐼

geometry=user の場合の処理。

```
1378 \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。

```
1379 \g@addto@macro\bxjs@begin@document@hook{%
1380   \ifdim\textwidth=.5\maxdimen
1381     \ClassError\bxjs@clsname
1382       {Page layout is not properly set}%
1383       {\@ehd}%
1384   \fi}
1385 \def\jsUseMinimalPageLayout{%
1386   \setlength{\textwidth}{6.5in}%
1387   \setlength{\textheight}{8in}}
```

`\setpagelayout` はとりあえず無効にしておく。

```
1388 \let\bxjs@geometry@driver\relax
1389 \def\setpagelayout{%
1390   \bxjs@ifplus{\bxjs@pagelayout@a}{%else
1391     \@ifstar{\bxjs@pagelayout@a}{\bxjs@pagelayout@a}}
1392 \def\bxjs@pagelayout@a#1{%
1393   \ClassError\bxjs@clsname
1394     {Command '\string\setpagelayout' is not supported,\MessageBreak
1395     because 'geometry' value is not 'class'}\@eha}
1396 %
1397 \fi\fi
```

■JS クラスと共通処理の開始

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。

※ `diff` が崩壊するのを避けるためオリジナルのコードを無効化した状態で挿入しておく。

```
1398 %<*jsclasses>
```

■縦方向のスペース

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

[2003-06-26] `\headheight` はヘッダの高さで、元は 12pt でしたが、新ドキュメントクラスでは `\topskip` と等しくしていました。ところが、`fancyhdr` パッケージで `\headheight`

が小さいとおかしいことになるようですので、2 倍に増やしました。代わりに、版面の上下揃えの計算では `\headheight` ではなく `\topskip` を使うことにしました。

[2016-08-17] 圈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。

```
1399 \setlength\topskip{1.38zw}%% from 10\jsc@mpt (2016-08-17)
1400 \if@slide
1401   \setlength\headheight{0\jsc@mpt}
1402 \else
1403   \setlength\headheight{20\jsc@mpt}%% from 2\topskip (2016-08-17); from \topskip (2003-
      06-26)
1404 \fi
```

`\footskip` `\footskip` は本文領域下端とフッタ下端との距離です。標準クラスファイルでは、book で 0.35in (約 8.89mm), book 以外で 30pt (約 10.54mm) となっていました, ここでは A4 判のときちょうど 1cm となるように、`\paperheight` の 0.03367 倍 (最小 `\baselineskip`) としました。書籍については、フッタは使わないことにして、ゼロにしました。

```
1405 %<*article|kiyou>
1406 \if@slide
1407   \setlength\footskip{0pt}
1408 \else
1409   \setlength\footskip{0.03367\paperheight}
1410   \ifdim\footskip<\baselineskip
1411     \setlength\footskip{\baselineskip}
1412   \fi
1413 \fi
1414 %</article|kiyou>
1415 %<jspf>\setlength\footskip{9\jsc@mmm}
1416 %<*book>
1417 \if@report
1418   \setlength\footskip{0.03367\paperheight}
1419   \ifdim\footskip<\baselineskip
1420     \setlength\footskip{\baselineskip}
1421   \fi
1422 \else
1423   \setlength\footskip{0pt}
1424 \fi
1425 %</book>
1426 %<*report>
1427 \setlength\footskip{0.03367\paperheight}
1428 \ifdim\footskip<\baselineskip
1429   \setlength\footskip{\baselineskip}
1430 \fi
1431 %</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)。

```
1432 %<*article>
1433 \if@slide
1434   \setlength\headsep{0\jsc@empt}
1435   \addtolength\headsep{-\topskip}%% added (2016-10-08)
1436   \addtolength\headsep{10\jsc@empt}%% added (2016-10-08)
1437 \else
1438   \setlength\headsep{\footskip}
1439   \addtolength\headsep{-\topskip}
1440 \fi
1441 %</article>
1442 %<*book>
1443 \if@report
1444   \setlength\headsep{\footskip}
1445   \addtolength\headsep{-\topskip}
1446 \else
1447   \setlength\headsep{6\jsc@mmm}
1448   \addtolength\headsep{-\topskip}%% added (2016-10-08)
1449   \addtolength\headsep{10\jsc@empt}%% added (2016-10-08)
1450 \fi
1451 %</book>
1452 %<*report>
1453 \setlength\headsep{\footskip}
1454 \addtolength\headsep{-\topskip}
1455 %</report>
1456 %<*jspf>
1457 \setlength\headsep{9\jsc@mmm}
1458 \addtolength\headsep{-\topskip}
1459 %</jspf>
1460 %<*kiyou>
1461 \setlength\headheight{0\jsc@empt}
1462 \setlength\headsep{0\jsc@empt}
1463 \addtolength\headsep{-\topskip}%% added (2016-10-08)
1464 \addtolength\headsep{10\jsc@empt}%% added (2016-10-08)
1465 %</kiyou>
```

`\maxdepth` `\maxdepth` は本文最下行の最大の深さで, plain $\mathrm{T}_{\mathrm{E}}\mathrm{X}$ や $\mathrm{L}^{\mathrm{A}}\mathrm{T}_{\mathrm{E}}\mathrm{X}$ 2.09 では 4pt に固定でした。 $\mathrm{L}^{\mathrm{A}}\mathrm{T}_{\mathrm{E}}\mathrm{X}2_{\mathrm{e}}$ では `\maxdepth + \topskip` を本文フォントサイズの 1.5 倍にしたいのですが, `\topskip` は本文フォントサイズ (ここでは 10pt) に等しいので, 結局 `\maxdepth` は `\topskip` の半分の値 (具体的には 5pt) にします。

```
1466 \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。

```
1467 \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。

```
1468 %<*article>
1469 \if@slide
1470   \setlength\fullwidth{0.9\paperwidth}
1471 \else
1472   \setlength\fullwidth{0.76\paperwidth}
1473 \fi
1474 \if@twocolumn \@tempdima=2zw \else \@tempdima=1zw \fi
1475 \divide\fullwidth\@tempdima \multiply\fullwidth\@tempdima
1476 \setlength\textwidth{\fullwidth}
1477 %</article>
1478 %<*book>
1479 \if@report
1480   \setlength\fullwidth{0.76\paperwidth}
1481 \else
1482   \setlength\fullwidth{\paperwidth}
1483   \addtolength\fullwidth{-36\jsc@mmm}
1484 \fi
1485 \if@twocolumn \@tempdima=2zw \else \@tempdima=1zw \fi
1486 \divide\fullwidth\@tempdima \multiply\fullwidth\@tempdima
1487 \setlength\textwidth{\fullwidth}
1488 \if@report \else
1489   \if@twocolumn \else
1490     \ifdim \fullwidth>40zw
1491       \setlength\textwidth{40zw}
1492     \fi
1493   \fi
1494 \fi
1495 %</book>
1496 %<*report>
1497 \setlength\fullwidth{0.76\paperwidth}
1498 \if@twocolumn \@tempdima=2zw \else \@tempdima=1zw \fi
1499 \divide\fullwidth\@tempdima \multiply\fullwidth\@tempdima
1500 \setlength\textwidth{\fullwidth}
1501 %</report>
1502 %<*jspf>
1503 \setlength\fullwidth{50zw}
```

```

1504 \addtolength\fullwidth{8\jsc@mmm}
1505 \setlength\textwidth{\fullwidth}
1506 %</jspf>
1507 %<*kiyou>
1508 \setlength\fullwidth{48zw}
1509 \addtolength\fullwidth{\columnsep}
1510 \setlength\textwidth{\fullwidth}
1511 %</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)。

```

1512 %<*article|book|report>
1513 \if@slide
1514   \setlength{\textheight}{0.95\paperheight}
1515 \else
1516   \setlength{\textheight}{0.83\paperheight}
1517 \fi
1518 \addtolength{\textheight}{-10\jsc@mpt}%% from -\topskip (2016-10-08); from -
    \headheight (2003-06-26)
1519 \addtolength{\textheight}{-\headsep}
1520 \addtolength{\textheight}{-\footskip}
1521 \addtolength{\textheight}{-\topskip}
1522 \divide\textheight\baselineskip
1523 \multiply\textheight\baselineskip
1524 %</article|book|report>
1525 %<jspf>\setlength{\textheight}{51\baselineskip}
1526 %<kiyou>\setlength{\textheight}{47\baselineskip}
1527 \addtolength{\textheight}{\topskip}
1528 \addtolength{\textheight}{0.1\jsc@mpt}
1529 %<jspf>\setlength{\mathindent}{10\jsc@mmm}

```

`\flushbottom` [2016-07-18] `\textheight` に念のため 0.1 ポイント余裕を持たせているのと同様に、`\flushbottom` にも余裕を持たせます。元の $\text{\LaTeX 2}_{\epsilon}$ での完全な `\flushbottom` の定

義は

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

ですが、次のようにします。

```
1530 \def\flushbottom{%  
1531   \def\@textbottom{\vskip \z@ \@plus.1\jcs@empt}%  
1532   \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。

```
1533 \setlength\marginparsep{\columnsep}  
1534 \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。TEX は上・左マージンに `1truein` を挿入しますが、トンボ関係のオプションが指定されると pLATEX 2_ε (`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。

```
1535 \setlength{\oddsidemargin}{\paperwidth}  
1536 \addtolength{\oddsidemargin}{-\fullwidth}  
1537 \setlength{\oddsidemargin}{.5\oddsidemargin}  
1538 \iftombow  
1539   \addtolength{\oddsidemargin}{-1in}  
1540 \else  
1541   \addtolength{\oddsidemargin}{-\inv@mag in}  
1542 \fi  
1543 \setlength{\evensidemargin}{\oddsidemargin}  
1544 \if@mparswitch  
1545   \addtolength{\evensidemargin}{\fullwidth}  
1546   \addtolength{\evensidemargin}{-\textwidth}  
1547 \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。

```
1548 \setlength\marginparwidth{\paperwidth}  
1549 \addtolength\marginparwidth{-\oddsidemargin}  
1550 \addtolength\marginparwidth{-\inv@mag in}  
1551 \addtolength\marginparwidth{-\textwidth}  
1552 \addtolength\marginparwidth{-10\jcs@mmm}  
1553 \addtolength\marginparwidth{-\marginparsep}  
1554 \@tempdima=1zw  
1555 \divide\marginparwidth\@tempdima  
1556 \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つたので、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)。

```

1557 \setlength\topmargin{\paperheight}
1558 \addtolength\topmargin{-\textheight}
1559 \if@slide
1560   \addtolength\topmargin{-\headheight}
1561 \else
1562   \addtolength\topmargin{-10\jsc@empt}%% from -\topskip (2016-10-08); from -
      \headheight (2003-06-26)
1563 \fi
1564 \addtolength\topmargin{-\headsep}
1565 \addtolength\topmargin{-\footskip}
1566 \setlength\topmargin{0.5\topmargin}
1567 %<kiyou>\setlength\topmargin{81truebp}
1568 \iftombow
1569   \addtolength\topmargin{-1in}
1570 \else
1571   \addtolength\topmargin{-\inv@mag in}
1572 \fi
1573 %</jsclasses>

```

■脚注

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

ここは元々は

```
{\footnotesize\global\setlength\footnotesep{\baselineskip}}
```

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

```
1574 \footnotesep=11\p@? \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。

```
1575 \setlength{\skip\footins}{16\p@? \@plus 5\p@? \@minus 2\p@?}
```

■フロート関連 フロート (図, 表) 関連のパラメータは L^AT_EX 2_ε 本体で定義されていますが、ここで設定変更します。本文ページ (本文とフロートが共存するページ) とフロートだけのページで設定が異なります。ちなみに、カウンタは内部では `\c@` を名前に冠したマクロになっています。

`\c@topnumber` `topnumber` カウンタは本文ページ上部のフロートの最大数です。
 [2003-08-23] ちょっと増やしました。
 1576 `\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。
 1577 `\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。
 1578 `\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。
 1579 `\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。
 1580 `\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。
 1581 `\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。
 1582 `\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。
 1583 `\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。
 1584 `\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。
 1585 `\renewcommand{\dblfloatpagefraction}{.8}`

`\floatsep` `\floatsep` はページ上部・下部のフロート間の距離です。`\textfloatsep` はページ上部・下部のフロートと本文との距離です。`\intextsep` は本文の途中に出力されるフロートと本文との距離です。
 1586 `\setlength\floatsep {12\p@? \@plus 2\p@? \@minus 2\p@?}`
 1587 `\setlength\textfloatsep{20\p@? \@plus 2\p@? \@minus 4\p@?}`
 1588 `\setlength\intextsep {12\p@? \@plus 2\p@? \@minus 2\p@?}`

`\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` 1589 `\setlength\dblfloatsep {12\p@? \@plus 2\p@? \@minus 2\p@?}`
 1590 `\setlength\dbltextfloatsep{20\p@? \@plus 2\p@? \@minus 4\p@?}`

`\@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 1591 \setlength\@fptop{0\p@? \@plus 1fil}
          1592 \setlength\@fpsep{8\p@? \@plus 2fil}
          1593 \setlength\@fpbot{0\p@? \@plus 1fil}

\@dblftop 段抜きフロートについての値です。
\@dblfpsep 1594 \setlength\@dblftop{0\p@? \@plus 1fil}
\@dblfpbot 1595 \setlength\@dblfpsep{8\p@? \@plus 2fil}
          1596 \setlength\@dblfpbot{0\p@? \@plus 1fil}

```

6 改ページ（日本語 T_EX 開発コミュニティ版のみ）

`\pltx@cleartorightpage` [2017-02-24] コミュニティ版 pL_AT_EX 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。

```

1597 %\def\pltx@cleartorightpage{\clearpage\if@twoside
1598 % \ifodd\c@page
1599 % \iftdir
1600 % \hbox{}\thispagestyle{empty}\newpage
1601 % \if@twocolumn\hbox{}\newpage\fi
1602 % \fi
1603 % \else
1604 % \ifydir
1605 % \hbox{}\thispagestyle{empty}\newpage
1606 % \if@twocolumn\hbox{}\newpage\fi
1607 % \fi
1608 % \fi\fi}
1609 %\def\pltx@cleartoleftpage{\clearpage\if@twoside
1610 % \ifodd\c@page
1611 % \ifydir
1612 % \hbox{}\thispagestyle{empty}\newpage
1613 % \if@twocolumn\hbox{}\newpage\fi
1614 % \fi
1615 % \else
1616 % \iftdir
1617 % \hbox{}\thispagestyle{empty}\newpage
1618 % \if@twocolumn\hbox{}\newpage\fi
1619 % \fi
1620 % \fi\fi}

```

```

1621 \def\pltx@cleartooddpage{\clearpage\if@twoside
1622   \ifodd\c@page\else
1623     \hbox{}\thispagestyle{empty}\newpage
1624     \if@twocolumn\hbox{}\newpage\fi
1625   \fi\fi}
1626 \def\pltx@cleartoevenpage{\clearpage\if@twoside
1627   \ifodd\c@page
1628     \hbox{}\thispagestyle{empty}\newpage
1629     \if@twocolumn\hbox{}\newpage\fi
1630   \fi\fi}

```

BXJS クラスでは `\iftkdir` 等が使えないので、横組を仮定した定義を用いる。

```

1631 \let\pltx@cleartorightpage\pltx@cleartooddpage
1632 \let\pltx@cleartoleftpage\pltx@cleartoevenpage

```

`\cleardoublepage` [2017-02-24] コミュニティ版 p \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。

```

1633 %<*book|report>
1634 \if@openleft
1635   \let\cleardoublepage\pltx@cleartoleftpage
1636 \else\if@openright
1637   \let\cleardoublepage\pltx@cleartorightpage
1638 \fi\fi
1639 %</book|report>

```

7 ページスタイル

ページスタイルとして、 \LaTeX 2_ε (欧文版) の標準クラスでは `empty`, `plain`, `headings`, `myheadings` があります。このうち `empty`, `plain` スタイルは \LaTeX 2_ε 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` で変更する場合はこれにあたります。しかし、同一ページに複数の `\markboth` があると、おかしい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^AT_EX 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。

```
1640 % \def\ps@empty{%
1641 %   \let\mkboth\gobbletwo
1642 %   \let\@oddhead\empty
1643 %   \let\@oddfoot\empty
1644 %   \let\@evenhead\empty
1645 %   \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。

```
1646 \def\ps@plainfoot{%
1647   \let\mkboth\gobbletwo
1648   \let\@oddhead\empty
1649   \def\@oddfoot{\normalfont\hfil\thepage\hfil}%
1650   \let\@evenhead\empty
1651   \let\@evenfoot\@oddfoot}
1652 \def\ps@plainhead{%
1653   \let\mkboth\gobbletwo
1654   \let\@oddfoot\empty
1655   \let\@evenfoot\empty
1656   \def\@evenhead{%
1657     \ifmparswitch \hss \fi
1658     \hbox to \fullwidth{\textbf{\thepage}\hfil}%
1659     \ifmparswitch\else \hss \fi}%
1660   \def\@oddhead{%
1661     \hbox to \fullwidth{\hfil\textbf{\thepage}}\hss}}
1662 %<book>\let\ps@plain\ps@plainhead
1663 %<!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。

```
1664 %<*article|slide>
1665 \if@twoside
1666   \def\ps@headings{%
1667     \let\@oddfoot\empty
1668     \let\@evenfoot\empty
```

```

1669 \def\@evenhead{\if@mparswitch \hss \fi
1670 \underline{\hbox to \fullwidth{\textbf{\thepage}\hfil\leftmark}}}%
1671 \if@mparswitch\else \hss \fi}%
1672 \def\@oddhead{%
1673 \underline{%
1674 \hbox to \fullwidth{{\rightmark}\hfil\textbf{\thepage}}}\hss}%
1675 \let\@mkboth\markboth
1676 \def\sectionmark##1{\markright{%
1677 \ifnum \c@secnumdepth >\z@ \bxjs@label@sect{section}\hskip1\jsZw\fi
1678 ##1}{}}%
1679 \def\subsectionmark##1{\markright{%
1680 \ifnum \c@secnumdepth >\@ne \bxjs@label@sect{subsection}\hskip1\jsZw\fi
1681 ##1}}%
1682 }
1683 \else % if not twoside
1684 \def\ps@headings{%
1685 \let\@oddfoot\@empty
1686 \def\@oddhead{%
1687 \underline{%
1688 \hbox to \fullwidth{{\rightmark}\hfil\textbf{\thepage}}}\hss}%
1689 \let\@mkboth\markboth
1690 \def\sectionmark##1{\markright{%
1691 \ifnum \c@secnumdepth >\z@ \bxjs@label@sect{section}\hskip1\jsZw\fi
1692 ##1}}%
1693 \fi
1694 %</article|slide>

```

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

\autoxspacing は未定義の可能性があるので、「\autoxspacing が定義済なら実行する」マクロ \bxjs@maybe@autoxspacing を代わりに用いる。

```

1695 %<*book|report>
1696 \def\bxjs@maybe@autoxspacing{%
1697 \ifx\autoxspacing\undefined\else \autoxspacing \fi}
1698 \newif\if@omit@number
1699 \def\ps@headings{%
1700 \let\@oddfoot\@empty
1701 \let\@evenfoot\@empty
1702 \def\@evenhead{%
1703 \if@mparswitch \hss \fi
1704 \underline{\hbox to \fullwidth{\bxjs@maybe@autoxspacing
1705 \textbf{\thepage}\hfil\leftmark}}}%
1706 \if@mparswitch\else \hss \fi}%
1707 \def\@oddhead{\underline{\hbox to \fullwidth{\bxjs@maybe@autoxspacing
1708 {\if@twoside\rightmark\else\leftmark\fi}\hfil\textbf{\thepage}}}\hss}%
1709 \let\@mkboth\markboth

```

```

1710 \def\chaptermark##1{\markboth{%
1711 \ifnum \c@secnumdepth >\m@ne
1712 \if@mainmatter
1713 \if@omit@number\else
1714 \@chapapp\thechapter\@chappos\hskip1\jsZw
1715 \fi
1716 \fi
1717 \fi
1718 ##1}{}}%
1719 \def\sectionmark##1{\markright{%
1720 \ifnum \c@secnumdepth >\z@ \bxjs@label@sect{section}\hskip1\jsZw\fi
1721 ##1}}}%
1722 %</book|report>

```

最後は学会誌の場合です。

```

1723 %<*jspf>
1724 \def\ps@headings{%
1725 \def\@oddfoot{\normalfont\hfil\thepage\hfil}
1726 \def\@evenfoot{\normalfont\hfil\thepage\hfil}
1727 \def\@oddhead{\normalfont\hfil \@title \hfil}
1728 \def\@evenhead{\normalfont\hfil プラズマ・核融合学会誌\hfil}}
1729 %</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。

```

1730 \def\ps@myheadings{%
1731 \let\@oddfoot\@empty\let\@evenfoot\@empty
1732 \def\@evenhead{%
1733 \if@mparswitch \hss \fi%
1734 \hbox to \fullwidth{\thepage\hfil\leftmark}%
1735 \if@mparswitch\else \hss \fi}%
1736 \def\@oddhead{%
1737 \hbox to \fullwidth{\rightmark\hfil\thepage}\hss}%
1738 \let\@mkboth\@gobbletwo
1739 %<book|report> \let\chaptermark\@gobble
1740 \let\sectionmark\@gobble
1741 %<!book&!report> \let\subsectionmark\@gobble
1742 }

```

8 文書のマークアップ

8.1 表題

`\title` これらは L^AT_EX 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 1743 % \newcommand*{\title}[1]{\gdef\@title{#1}}

```

```

1744 % \newcommand*{\author}[1]{\gdef\@author{#1}}
1745 % \newcommand*{\date}[1]{\gdef\@date{#1}}
1746 % \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。

本体を `\jsSubtitle` として定義する。

```

1747 \newcommand*{\jsSubtitle}[1]{\gdef\bxjs@subtitle{#1}}
1748 %\let\bxjs@subtitle\@undefined

\title にフックを入れる。

1749 \renewcommand*{\title}[1]{\bxjs@decl@subtitle\gdef\@title{#1}}
1750 \g@addto@macro\bxjs@begin@document@hook{\bxjs@decl@subtitle}
1751 \def\bxjs@decl@subtitle{%
1752   \global\let\bxjs@decl@subtitle\relax
1753   \ifx\subtitle\@undefined
1754     \global\let\subtitle\jsSubtitle
1755   \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。

```

1756 \def\bxjs@annihilate@subtitle{%
1757   \ifx\subtitle\jsSubtitle \global\let\subtitle\relax \fi
1758   \global\let\jsSubtitle\relax}

```

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

```

\author 1759 %<*jspf>
1760 \newcommand*{\etitle}[1]{\gdef\@etitle{#1}}
\keywords 1761 \newcommand*{\eauthor}[1]{\gdef\@eauthor{#1}}
1762 \newcommand*{\keywords}[1]{\gdef\@keywords{#1}}
1763 \newcommand*{\email}[1]{\gdef\authors@mail{#1}}
1764 \newcommand*{\AuthorsEmail}[1]{\gdef\authors@mail{author's e-mail:\ #1}}
1765 %</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。

```

1766 \def\plainifnotempty{%
1767   \ifx \@oddhead \@empty
1768     \ifx \@oddfoot \@empty

```

```

1769     \else
1770         \thispagestyle{plainfoot}%
1771     \fi
1772 \else
1773     \thispagestyle{plainhead}%
1774 \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。

```

1775 %<*article|book|report|slide>
1776 \if@titlepage
1777     \newcommand{\maketitle}{%
1778         \begin{titlepage}%
1779             \let\footnotesize\small
1780             \let\footnoterule\relax
1781             \let\footnote\thanks
1782             \null\vfil
1783             \if@slide
1784                 {\footnotesize \@date}%
1785                 \begin{center}
1786                     \mbox{} \[\!1\!Zw]
1787                     \large
1788                     {\maybeblue\hrule height0\p@? depth2\p@?\relax}\par
1789                     \jsc@smallskip
1790                     \@title
1791                     \ifx\bxjs@subtitle\@undefined\else
1792                         \par\vskip\z@
1793                         {\small \bxjs@subtitle\par}
1794                     \fi
1795                     \jsc@smallskip
1796                     {\maybeblue\hrule height0\p@? depth2\p@?\relax}\par
1797                     \vfill
1798                     {\small \@author}%
1799                 \end{center}
1800             \else
1801                 \vskip 60\p@?
1802                 \begin{center}%
1803                     {\LARGE \@title \par}%
1804                     \ifx\bxjs@subtitle\@undefined\else
1805                         \vskip5\p@?
1806                         {\normalsize \bxjs@subtitle\par}
1807                     \fi
1808                 \vskip 3em%
1809                 {\large
1810                     \lineskip .75em

```

```

1811         \begin{tabular}[t]{c}%
1812             \@author
1813         \end{tabular}\par}%
1814         \vskip 1.5em
1815         {\large \@date \par}%
1816     \end{center}%
1817     \fi
1818     \par
1819     \@thanks\vfil\null
1820 \end{titlepage}%
1821 \setcounter{footnote}{0}%
1822 \global\let\thanks\relax
1823 \global\let\maketitle\relax
1824 \global\let\@thanks\@empty
1825 \global\let\@author\@empty
1826 \global\let\@date\@empty
1827 \global\let\@title\@empty
1828 \global\let\title\relax
1829 \global\let\author\relax
1830 \global\let\date\relax
1831 \global\let\and\relax
1832 \bxjs@annihilate@subtitle
1833 }%
1834 \else
1835     \newcommand{\maketitle}{\par
1836         \begingroup
1837             \renewcommand\thefootnote{\@fnsymbol\c@footnote}%
1838             \def\@makefnmark{\rlap{\@textsuperscript{\normalfont\@thefnmark}}}%
1839             \long\def\@makefntext##1{\advance\leftskip 3\jsZw
1840                 \parindent 1\jsZw\noindent
1841                 \llap{\@textsuperscript{\normalfont\@thefnmark}\hskip0.3\jsZw}##1}%
1842             \if@twocolumn
1843                 \ifnum \col@number=\@ne
1844                     \@maketitle
1845                 \else
1846                     \twocolumn[\@maketitle]%
1847                 \fi
1848             \else
1849                 \newpage
1850                 \global\@topnum\z@ % Prevents figures from going at top of page.
1851                 \@maketitle
1852             \fi
1853             \plainifnotempty
1854             \@thanks
1855         \endgroup
1856         \setcounter{footnote}{0}%
1857         \global\let\thanks\relax
1858         \global\let\maketitle\relax
1859         \global\let\@thanks\@empty

```

```

1860 \global\let\@author\@empty
1861 \global\let\@date\@empty
1862 \global\let\@title\@empty
1863 \global\let\title\relax
1864 \global\let\author\relax
1865 \global\let\date\relax
1866 \global\let\and\relax
1867 \bxjs@annihilate@subtitle
1868 }

```

\@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。

```

1869 \def\@maketitle{%
1870 \newpage\null
1871 \vskip 2em
1872 \begin{center}%
1873 \let\footnote\thanks
1874 {\LARGE \@title \par}%
1875 \ifx\bxjs@subtitle\@undefined\else
1876 \vskip3\p@?
1877 {\normalsize \bxjs@subtitle\par}
1878 \fi
1879 \vskip 1.5em
1880 {\large
1881 \lineskip .5em
1882 \begin{tabular}[t]{c}%
1883 \@author
1884 \end{tabular}\par}%
1885 \vskip 1em
1886 {\large \@date}%
1887 \end{center}%
1888 \par\vskip 1.5em
1889 %<article|slide> \ifvoid\@abstractbox\else\centerline{\box\@abstractbox}\vskip1.5em\fi
1890 }
1891 \fi
1892 %</article|book|report|slide>
1893 %<*jspf>
1894 \newcommand{\maketitle}{\par
1895 \begingroup
1896 \renewcommand\thefootnote{\@fnsymbol\c@footnote}%
1897 \def\@makefnmark{\rlap{\@textsuperscript{\normalfont\@thefnmark}}}%
1898 \long\def\@makefntext##1{\advance\leftskip 3\jsZw
1899 \parindent 1\jsZw\noindent
1900 \llap{\@textsuperscript{\normalfont\@thefnmark}\hskip0.3\jsZw}##1}%
1901 \twocolumn[\@maketitle]%
1902 \plainifnotempty
1903 \@thanks
1904 \endgroup
1905 \setcounter{footnote}{0}%
1906 \global\let\thanks\relax

```

```

1907 \global\let\maketitle\relax
1908 \global\let\@thanks\@empty
1909 \global\let\@author\@empty
1910 \global\let\@date\@empty
1911 % \global\let\@title\@empty % \@title は柱に使う
1912 \global\let\title\relax
1913 \global\let\author\relax
1914 \global\let\date\relax
1915 \global\let\and\relax
1916 \ifx\authors@mail\@undefined\else{%
1917   \def\@makefntext{\advance\leftskip 3\jsZw \parindent -3\jsZw}%
1918   \footnotetext[0]{\itshape\authors@mail}%
1919 }\fi
1920 \global\let\authors@mail\@undefined}
1921 \def\@maketitle{%
1922   \newpage\null
1923   \vskip 6em % used to be 2em
1924   \begin{center}
1925     \let\footnote\thanks
1926     \ifx\@title\@undefined\else{\LARGE\headfont\@title\par}\fi
1927     \lineskip .5em
1928     \ifx\@author\@undefined\else
1929       \vskip 1em
1930       \begin{tabular}[t]{c}%
1931         \@author
1932       \end{tabular}\par
1933     \fi
1934     \ifx\@etitle\@undefined\else
1935       \vskip 1em
1936       {\large \@etitle \par}%
1937     \fi
1938     \ifx\@eauthor\@undefined\else
1939       \vskip 1em
1940       \begin{tabular}[t]{c}%
1941         \@eauthor
1942       \end{tabular}\par
1943     \fi
1944     \vskip 1em
1945     \@date
1946   \end{center}
1947   \vskip 1.5em
1948   \centerline{\box\@abstractbox}
1949   \ifx\@keywords\@undefined\else
1950     \vskip 1.5em
1951     \centerline{\parbox{157\jsc@mmm}{\textsf{Keywords:}}\ \small\@keywords}}
1952   \fi
1953   \vskip 1.5em}
1954 %</jspf>

```


8.2 章・節

ムニャムニャ……。

`\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。

```
1955 \def\bxjs@label@sect#1{%
1956   \expandafter\ifx\csname label#1\endcsname\relax
1957     \csname the#1\endcsname
1958   \else \csname label#1\endcsname
1959   \fi}
1960 \def\@secntformat#1{\bxjs@label@sect{#1}\quad}
```

`\@secapp` 節番号の接頭辞。

`\@secpos` 節番号の接尾辞。

```
1961 \ifnum\bxjs@label@section=\bxjs@label@section@@compat\else
1962 \def\@secapp{\presectionname}
1963 \def\@secpos{\postsectionname}
1964 \fi
```

`\labelsection` 節番号の出力書式。

```
1965 \ifnum\bxjs@label@section=\bxjs@label@section@@modern
1966 \def\labelsection{\@secapp\thesection\@secpos}
1967 \fi
```

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

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

それぞれの引数の意味は次の通りです。

名 ユーザレベルコマンドの名前です (例: section)。

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

字下げ 見出しの字下げ量です。

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

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

スタイル 見出しの文字スタイルの設定です。

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

別見出し 目次や柱に出力する見出しです。

見出し 見出しです。

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

次は `\@startsection` の定義です。情報処理学会論文誌スタイルファイル (`ipsjcommon.sty`) を参考にさせていただきましたが、完全に行送りが `\baselineskip` の整数倍にならなくてもいいから前の行と重ならないようにしました。

```
1968 \def\@startsection#1#2#3#4#5#6{%
1969   \if@noskipsec \leavevmode \fi
1970   \par
1971 % 見出し上の空きを \@tempskipa にセットする
1972   \@tempskipa #4\relax
1973 % \@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
1974   \if@english \@afterindentfalse \else \@afterindenttrue \fi
1975 % 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
1976   \ifdim \@tempskipa <\z@
1977     \@tempskipa -\@tempskipa \@afterindentfalse
1978   \fi
1979   \if@nobreak
1980 %   \everypar{\everyparhook}% これは間違い
1981     \everypar{}%
1982   \else
1983     \addpenalty\@secpenalty
1984 % 次の行は削除
1985 %   \addvspace\@tempskipa
1986 % 次の \noindent まで追加
1987     \ifdim \@tempskipa >\z@
1988       \if@slide\else
1989         \null
1990         \vspace*{-\baselineskip}%
1991       \fi
1992       \vskip\@tempskipa
1993     \fi
1994   \fi
1995   \noindent
1996 % 追加終わり
1997   \@ifstar
1998     {\@ssect{#3}{#4}{#5}{#6}}%
1999     {\@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のように、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 everyparhook=compat` である場合にのみ直後のトークンを実行する。

```

2000 \ifnum\bxjs@everyparhook=\bxjs@everyparhook@@compat
2001 \let\bxjs@if@ceph\@firstofone
2002 \else \let\bxjs@if@ceph\@gobble
2003 \fi

2004 \def\@sect#1#2#3#4#5#6[#7]#8{%
2005 \ifnum #2>\c@secnumdepth
2006 \let\@svsec\@empty
2007 \else
2008 \refstepcounter{#1}%
2009 \protected@edef\@svsec{\@secntformat{#1}\relax}%
2010 \fi
2011 % 見出し後の空きを \@tempskipa にセット
2012 \@tempskipa #5\relax
2013 % 条件判断の順序を入れ替えました
2014 \ifdim \@tempskipa<\z@
2015 \def\@svsechd{%
2016 #6{\hskip #3\relax
2017 \@svsec #8}%
2018 \csname #1mark\endcsname{#7}%
2019 \addcontentsline{toc}{#1}{%
2020 \ifnum #2>\c@secnumdepth \else
2021 \protect\numberline{\bxjs@label@sect{#1}}%
2022 \fi
2023 #7}}% 目次にフルネームを載せるなら #8
2024 \else
2025 \begingroup
2026 \interlinepenalty \@M % 下から移動
2027 #6{%
2028 \@hangfrom{\hskip #3\relax\@svsec}%
2029 % \interlinepenalty \@M % 上に移動
2030 #8\@@par}%
2031 \endgroup
2032 \csname #1mark\endcsname{#7}%
2033 \addcontentsline{toc}{#1}{%
2034 \ifnum #2>\c@secnumdepth \else
2035 \protect\numberline{\bxjs@label@sect{#1}}%
2036 \fi
2037 #7}}% 目次にフルネームを載せるならここは #8
2038 \fi
2039 \@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。

```

2040 \def\@xsect#1{%
2041 % 見出しの後ろの空きを \@tempskipa にセット

```

```

2042 \@tempskipa #1\relax
2043 % 条件判断の順序を変えました
2044 \ifdim \@tempskipa<\z@
2045 \nobreakfalse
2046 \global\@noskipsectrue
2047 \everypar{%
2048 \if@noskipsec
2049 \global\@noskipsecfalse
2050 {\setbox\z@\lastbox}%
2051 \clubpenalty\@M
2052 \begingroup \svsechd \endgroup
2053 \unskip
2054 \@tempskipa #1\relax
2055 \hskip -\@tempskipa
2056 \else
2057 \clubpenalty \@clubpenalty
2058 \everypar\expandafter{\bxjs@if@ceph\everyparhook}%
2059 \fi\bxjs@if@ceph\everyparhook}%
2060 \else
2061 \par \nobreak
2062 \vskip \@tempskipa
2063 \@afterheading
2064 \fi
2065 \if@slide
2066 {\vskip\if@twocolumn-5\jsc@empt\else-6\jsc@empt\fi
2067 \maybeblue\hrule height0\jsc@empt depth1\jsc@empt
2068 \vskip\if@twocolumn 4\jsc@empt\else 7\jsc@empt\fi\relax}%
2069 \fi
2070 \par % 2000-12-18
2071 \ignorespaces}
2072 \def\@ssect#1#2#3#4#5{%
2073 \@tempskipa #3\relax
2074 \ifdim \@tempskipa<\z@
2075 \def\@svsechd{#4{\hskip #1\relax #5}}%
2076 \else
2077 \begingroup
2078 #4{%
2079 \@hangfrom{\hskip #1}%
2080 \interlinepenalty \@M #5\@@par}%
2081 \endgroup
2082 \fi
2083 \@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^AT_EX 本体で定義済みです。

```

\subsectionmark 2084 \newcommand*\chaptermark[1]{}
\subsubsectionmark 2085 % \newcommand*\sectionmark[1]{}

\paragraphmark
\subparagraphmark

```

```

2086 % \newcommand*{\subsectionmark}[1]{}
2087 % \newcommand*{\subsubsectionmark}[1]{}
2088 % \newcommand*{\paragraphmark}[1]{}
2089 % \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。

```

2090 %<!book&!report>\setcounter{secnumdepth}{3}
2091 %<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。これは第

`\c@section` 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 2092 \newcounter{part}
2093 %<book|report>\newcounter{chapter}
\c@subsubsection 2094 %<book|report>\newcounter{section}[chapter]
\c@paragraph 2095 %<!book&!report>\newcounter{section}
\c@subparagraph 2096 \newcounter{subsection}[section]
2097 \newcounter{subsubsection}[subsection]
2098 \newcounter{paragraph}[subsubsection]
2099 \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。

```

2100 \renewcommand{\thepart}{\@Roman\c@part}
2101 %<*&!book&!report>
2102 \ifnum\bxjs@label@section=\bxjs@label@section@@compat
2103 \renewcommand{\thesection}{\presectionname\@arabic\c@section\postsectionname}
2104 \renewcommand{\thesubsection}{\@arabic\c@section.\@arabic\c@subsection}
2105 \else
2106 \renewcommand{\thesection}{\@arabic\c@section}
2107 \renewcommand{\thesubsection}{\thesection.\@arabic\c@subsection}
2108 \fi
2109 %</&!book&!report>
2110 %<*&book|report>
2111 \renewcommand{\thechapter}{\@arabic\c@chapter}
2112 \renewcommand{\thesection}{\thechapter.\@arabic\c@section}
2113 \renewcommand{\thesubsection}{\thesection.\@arabic\c@subsection}

```

```

2114 %</book|report>
2115 \renewcommand{\thesubsubsection}{%
2116   \thesubsection.\@arabic\c@subsubsection}
2117 \renewcommand{\theparagraph}{%
2118   \thesubsubsection.\@arabic\c@paragraph}
2119 \renewcommand{\thesubparagraph}{%
2120   \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。

```

2121 %<book|report>\newcommand{\@chapapp}{\prechaptername}
2122 %<book|report>\newcommand{\@chappos}{\postchaptername}

```

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

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

[2017-03-05] `\frontmatter` と `\mainmatter` の2つの命令は、改丁または改ページした後で `\pagenumbering{...}` でノンブルを1にリセットします。長い間 `\frontmatter` は `openany` のときに単なる改ページとしていましたが、これではノンブルをリセットする際に偶奇逆転が起こる場合があります。 `openany` かどうかに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 \LaTeX の標準クラス 2017/03/05 も参照)。

```

2123 %<*book|report>
2124 \newcommand\frontmatter{%
2125   \pltx@cleartooddpage
2126   \@mainmatterfalse
2127   \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。

```

2128 \newcommand\mainmatter{%
2129   \pltx@cleartooddpage
2130   \@mainmattertrue
2131   \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。

```

2132 \newcommand\backmatter{%
2133   \if@openleft
2134     \cleardoublepage
2135   \else\if@openright
2136     \cleardoublepage
2137   \else
2138     \clearpage
2139   \fi\fi
2140   \@mainmatterfalse}
2141 %</book|report>

```

■部

`\part` 新しい部を始めます。

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

```
\secdef{星なし}{星あり}
```

星なし * のない形の定義です。

星あり * のある形の定義です。

`\secdef` は次のようにして使います。

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

まず `book` と `report` のクラス以外です。

```
2142 %<!*book&!report>
2143 \newcommand\part{%
2144   \if@noskipsec \leavevmode \fi
2145   \par
2146   \addvspace{4ex}%
2147   \if@english \@afterindentfalse \else \@afterindenttrue \fi
2148   \secdef\@part\@spart}
2149 %</!*book&!report>
```

`book` および `report` クラスの場合は、少し複雑です。

```
2150 %<*book|report>
2151 \newcommand\part{%
2152   \if@openleft
2153     \cleardoublepage
2154   \else\if@openright
2155     \cleardoublepage
2156   \else
2157     \clearpage
2158   \fi\fi
2159   \thispagestyle{empty}% 欧文用標準スタイルでは plain
2160   \if@twocolumn
2161     \onecolumn
2162     \@restonecoltrue
2163   \else
2164     \@restonecolfalse
2165   \fi
2166   \null\vfil
2167   \secdef\@part\@spart}
2168 %</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。

```
2169 %<*!book&!report>
2170 \def\@part[#1]#2{%
2171   \ifnum \c@secnumdepth >\m@ne
2172     \refstepcounter{part}%
2173     \addcontentsline{toc}{part}{%
2174       \prepartname\thepart\postpartname\hspace{1\jsZw}#1}%
2175   \else
2176     \addcontentsline{toc}{part}{#1}%
2177   \fi
2178   \markboth{}{}%
2179   {\parindent\z@
2180     \raggedright
2181     \interlinepenalty \@M
2182     \normalfont
2183     \ifnum \c@secnumdepth >\m@ne
2184       \Large\headfont\prepartname\thepart\postpartname
2185       \par\nobreak
2186     \fi
2187     \huge \headfont #2%
2188     \markboth{}{}\par}%
2189   \nobreak
2190   \vskip 3ex
2191   \@afterheading}
2192 %</!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。

```
2193 %<*book|report>
2194 \def\@part[#1]#2{%
2195   \ifnum \c@secnumdepth >-2\relax
2196     \refstepcounter{part}%
2197     \addcontentsline{toc}{part}{%
2198       \prepartname\thepart\postpartname\hspace{1\jsZw}#1}%
2199   \else
2200     \addcontentsline{toc}{part}{#1}%
2201   \fi
2202   \markboth{}{}%
2203   {\centering
2204     \interlinepenalty \@M
2205     \normalfont
2206     \ifnum \c@secnumdepth >-2\relax
2207       \huge\headfont \prepartname\thepart\postpartname
2208       \par\vskip20\p@?
2209     \fi
2210     \Huge \headfont #2\par}%
2211   \@endpart}
2212 %</book|report>
```

\@spart 番号を付けない部です。


```

2213 %<*!book&!report>
2214 \def\@spart#1{%
2215     \parindent \z@ \raggedright
2216     \interlinepenalty \@M
2217     \normalfont
2218     \huge \headfont #1\par}%
2219 \nobreak
2220 \vskip 3ex
2221 \@afterheading}
2222 %</!book&!report>
2223 %<*book|report>
2224 \def\@spart#1{%
2225     \centering
2226     \interlinepenalty \@M
2227     \normalfont
2228     \Huge \headfont #1\par}%
2229 \@endpart}
2230 %</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^AT_EX では `classes.dtx` v1.4b (2000/05/19) で修正されています。

```

2231 %<*book|report>
2232 \def\@endpart{\vfil\newpage
2233     \if@twoside
2234     \if@openleft %% added (2017/02/24)
2235         \null\thispagestyle{empty}\newpage
2236     \else\if@openright %% added (2016/12/13)
2237         \null\thispagestyle{empty}\newpage
2238     \fi\fi %% added (2016/12/13, 2017/02/24)
2239 \fi
2240 \if@restonecol
2241     \twocolumn
2242 \fi}
2243 %</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。

```

2244 %<*book|report>
2245 \newcommand{\chapter}{%
2246     \if@openleft\cleardoublepage\else
2247     \if@openright\cleardoublepage\else\clearpage\fi\fi
2248     \plainifnotempty % 元: \thispagestyle{plain}

```

```

2249 \global\@topnum\z@
2250 \if@english \afterindentfalse \else \afterindenttrue \fi
2251 \secdef
2252     {\@omit@numberfalse\@chapter}%
2253     {\@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。

```

2254 \def\@chapter[#1]#2{%
2255     \ifnum \c@secnumdepth >\m@ne
2256         \if@mainmatter
2257             \refstepcounter{chapter}%
2258             \typeout{\@chapapp\thechapter\@chappos}%
2259             \addcontentsline{toc}{chapter}%
2260                 {\protect\numberline
2261 %             %{\if@english\thechapter\else\@chapapp\thechapter\@chappos\fi}%
2262                 {\@chapapp\thechapter\@chappos}%
2263                 #1}%
2264         \else\addcontentsline{toc}{chapter}{#1}\fi
2265     \else
2266         \addcontentsline{toc}{chapter}{#1}%
2267     \fi
2268     \chaptermark{#1}%
2269     \addtocontents{lof}{\protect\addvspace{10\jsc@mpt}}%
2270     \addtocontents{lot}{\protect\addvspace{10\jsc@mpt}}%
2271     \if@twocolumn
2272         \@topnewpage[\@makechapterhead{#2}]%
2273     \else
2274         \@makechapterhead{#2}%
2275         \@afterheading
2276     \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。

```

2277 \def\@makechapterhead#1{%
2278     \vspace*{2\Cvs}% 欧文は 50pt
2279     {\parindent \z@ \raggedright \normalfont
2280         \ifnum \c@secnumdepth >\m@ne
2281             \if@mainmatter
2282                 \huge\headfont \@chapapp\thechapter\@chappos
2283                 \par\nobreak
2284                 \vskip \Cvs % 欧文は 20pt
2285             \fi
2286         \fi
2287         \interlinepenalty\@M
2288         \Huge \headfont #1\par\nobreak
2289         \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。

```

2290 \def\@schapter#1{%

```

```

2291 \chaptermark{#1}%
2292 \if@twocolumn
2293   \@topnewpage[\@makeschapterhead{#1}]%
2294 \else
2295   \@makeschapterhead{#1}\@afterheading
2296 \fi}

```

`\@makeschapterhead` 番号なしの章見出しです。

```

2297 \def\@makeschapterhead#1{%
2298   \vspace*{2\Cvs}% 欧文は 50pt
2299   {\parindent \z@ \raggedright
2300    \normalfont
2301    \interlinepenalty\@M
2302    \Huge \headfont #1\par\nobreak
2303    \vskip 3\Cvs}} % 欧文は 40pt
2304 %</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。

```

2305 \if@twocolumn
2306   \newcommand{\section}{%
2307     %<jspf>\ifx\maketitle\relax\else\maketitle\fi
2308     \@startsection{section}{1}{\z@}%
2309     %<!kiyou> {0.6\Cvs}{0.4\Cvs}%
2310     %<kiyou> {\Cvs}{0.5\Cvs}%
2311     % {\normalfont\large\headfont\@secapp}}
2312     {\normalfont\large\headfont\raggedright}}
2313 \else
2314   \newcommand{\section}{%
2315     \if@slide\clearpage\fi
2316     \@startsection{section}{1}{\z@}%
2317     {\Cvs \@plus.5\Cdp \@minus.2\Cdp}% 前アキ
2318     {.5\Cvs \@plus.3\Cdp}% 後アキ
2319     % {\normalfont\Large\headfont\@secapp}}
2320     {\normalfont\Large\headfont\raggedright}}
2321 \fi

```

`\subsection` 同上です。

```

2322 \if@twocolumn
2323   \newcommand{\subsection}{\@startsection{subsection}{2}{\z@}%
2324     {\z@}{\if@slide .4\Cvs \else \z@ \fi}%
2325     {\normalfont\normalsize\headfont}}
2326 \else
2327   \newcommand{\subsection}{\@startsection{subsection}{2}{\z@}%
2328     {\Cvs \@plus.5\Cdp \@minus.2\Cdp}% 前アキ

```

```

2329     {.5\Cvs \@plus.3\Cdp}% 後アキ
2330     {\normalfont\large\headfont}}
2331 \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)。

```

2332 \if@twocolumn
2333   \newcommand{\subsubsection}{\@startsection{subsubsection}{3}{\z@}%
2334     {\z@}{\if@slide .4\Cvs \else \z@ \fi}%
2335     {\normalfont\normalsize\headfont}}
2336 \else
2337   \newcommand{\subsubsection}{\@startsection{subsubsection}{3}{\z@}%
2338     {\Cvs \@plus.5\Cdp \@minus.2\Cdp}%
2339     {\if@slide .5\Cvs \@plus.3\Cdp \else \z@ \fi}%
2340     {\normalfont\normalsize\headfont}}
2341 \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` は standard 和文ドライバが読み込まれた場合は `\jchar` と同義になるが、
それ以外は何もしない。

```

2342 \newcommand\jsParagraphMark{\relax\jsJaChar{■}}
2343 \let\bxjs@org@paragraph@mark\jsParagraphMark
2344 \ifx\bxjs@paragraph@mark\@empty
2345   \let\jsParagraphMark\@empty
2346 \else\ifx\bxjs@paragraph@mark\@undefined\else
2347   \long\edef\jsParagraphMark{\noexpand\jsJaChar{\bxjs@paragraph@mark}}
2348 \fi\fi
2349 \let\jsJaChar\@empty
2350 \if@twocolumn
2351   \newcommand{\paragraph}{\@startsection{paragraph}{4}{\z@}%
2352     {\z@}{\if@slide .4\Cvs \else -1\jsZw\fi}% 改行せず 1\jsZw のアキ
2353     <jspf> {\normalfont\normalsize\headfont}}
2354     <!jspf> {\normalfont\normalsize\headfont\jsParagraphMark}}
2355 \else
2356   \newcommand{\paragraph}{\@startsection{paragraph}{4}{\z@}%

```

```

2357      {0.5\Cvs \@plus.5\Cdp \@minus.2\Cdp}%
2358      {\if@slide .5\Cvs \@plus.3\Cdp \else -1\jsZw\fi}% 改行せず 1\jsZw のアキ
2359 %<jspf>      {\normalfont\normalsize\headfont}}
2360 %<!jspf>      {\normalfont\normalsize\headfont\jsParagraphMark}}
2361 \fi

```

`\subparagraph` 見出しの後ろで改行されません。

```

2362 \if@twocolumn
2363   \newcommand{\subparagraph}{\@startsection{subparagraph}{5}{\z@}%
2364     {\z@}{\if@slide .4\Cvs \@plus.3\Cdp \else -1\jsZw\fi}%
2365     {\normalfont\normalsize\headfont}}
2366 \else
2367   \newcommand{\subparagraph}{\@startsection{subparagraph}{5}{\z@}%
2368     {\z@}{\if@slide .5\Cvs \@plus.3\Cdp \else -1\jsZw\fi}%
2369     {\normalfont\normalsize\headfont}}
2370 \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 `\leftmargin k` に設定します。

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

[2002-05-11] 3zw に変更しました。

[2005-03-19] 二段組は 2zw に戻しました。

```

2371 \if@slide
2372   \setlength\leftmargini{1\jsZw}
2373 \else
2374   \if@twocolumn
2375     \setlength\leftmargini{2\jsZw}
2376   \else
2377     \setlength\leftmargini{3\jsZw}
2378   \fi
2379 \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 2380 \if@slide
\leftmarginv 2381   \setlength\leftmarginii {1\jsZw}
2382   \setlength\leftmarginiii {1\jsZw}
\leftmarginvi 2383   \setlength\leftmarginiv {1\jsZw}
2384   \setlength\leftmarginv {1\jsZw}
2385   \setlength\leftmarginvi {1\jsZw}
2386 \else
2387   \setlength\leftmarginii {2\jsZw}
2388   \setlength\leftmarginiii {2\jsZw}

```

```

2389 \setlength\leftmarginiv {2\jsZw}
2390 \setlength\leftmarginv {1\jsZw}
2391 \setlength\leftmarginvi {1\jsZw}
2392 \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。

```

2393 \setlength \labelsep {0.5\jsZw} % .5em
2394 \setlength \labelwidth{\leftmargini}
2395 \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。

```

2396 \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 2397 \@beginparpenalty -\@lowpenalty
\@itempenalty 2398 \@endparpenalty -\@lowpenalty
2399 \@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 $\pm_{0.1}^{0.2} \backslash baselineskip$ を思い切って外しました。

```

2400 \def\@listi{\leftmargin\leftmargini
2401 \parsep \z@
2402 \topsep 0.5\baselineskip
2403 \itemsep \z@ \relax}
2404 \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）。

```

2405 \@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 2406 \def\@listii{\leftmargin\leftmarginii
\@listiv 2407 \labelwidth\leftmarginii \advance\labelwidth-\labelsep
2408 \topsep \z@
\@listv 2409 \parsep \z@
\@listvi 2410 \itemsep\parsep}
2411 \def\@listiii{\leftmargin\leftmarginiii
2412 \labelwidth\leftmarginiii \advance\labelwidth-\labelsep
2413 \topsep \z@
2414 \parsep \z@

```

```

2415 \itemsep\parsep}
2416 \def\@listiv {\leftmargin\leftmarginiv
2417             \labelwidth\leftmarginiv
2418             \advance\labelwidth-\labelsep}
2419 \def\@listv {\leftmargin\leftmarginv
2420             \labelwidth\leftmarginv
2421             \advance\labelwidth-\labelsep}
2422 \def\@listvi {\leftmargin\leftmarginvi
2423             \labelwidth\leftmarginvi
2424             \advance\labelwidth-\labelsep}

```

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

\theenumi 出力する番号の書式を設定します。これらは L^AT_EX 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。

```

2425 \renewcommand{\theenumi}{\@arabic\c@enumi}
2426 \renewcommand{\theenumii}{\@alph\c@enumii}
2427 \renewcommand{\theenumiii}{\@roman\c@enumiii}
2428 \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。

※現状の zxjatype の \inhibitglue の実装には「前後のグルーを消してしまう」という不備があって、そのため enumii の出力が異常になるという不具合があった。zxjatype を修正するまでの回避策として、サイズがゼロの罫 (\bxjs@dust) でガードしておく。

```

2429 \def\bxjs@dust{\vrule\@width\z@\@height\z@\@depth\z@}
2430 \newcommand*{\jsInJaParen}[1]{%
2431   \bxjs@dust\jsInhibitGlue (#1) \jsInhibitGlue\bxjs@dust}
2432 \newcommand{\labelenumi}{\theenumi.}
2433 \newcommand{\labelenumii}{\jsInJaParen{\theenumii}}
2434 \newcommand{\labelenumiii}{\theenumiii.}
2435 \newcommand{\labelenumiv}{\theenumiv.}

```

\p@enumii \p@enum*n* は \ref コマンドで enumerate 環境の第 *n* レベルの項目が参照されるときを書く式です。これも第 2 レベルは和文用かっこにしました。

```

\p@enumiv 2436 \renewcommand{\p@enumii}{\theenumi}
2437 \renewcommand{\p@enumiii}{\theenumi\jsInhibitGlue (\theenumii )}
2438 \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 2439 \newcommand\labelitemi{\textbullet}
\labelitemiii 2440 \newcommand\labelitemii{\normalfont\bfseries \textendash}
\labelitemiv 2441 \newcommand\labelitemiii{\textasteriskcentered}
\labelitemv 2442 \newcommand\labelitemiv{\textperiodcentered}
```

■description 環境

`description` 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。

```
2443 \newenvironment{description}{%
2444   \list{}{%
2445     \labelwidth=\leftmargin
2446     \labelsep=1\jsZw
2447     \advance \labelwidth by -\labelsep
2448     \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。

```
2449 \newcommand*\descriptionlabel[1]{\normalfont\headfont #1\hfil}
```

■概要

`abstract` 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` `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。

```
2450 %<*book|report>
2451 \newenvironment{chapterabstract}{%
2452   \begin{list}{}{%
2453     \listparindent=1\jsZw
2454     \itemindent=\listparindent
```



```

2455 \rightmargin=0pt
2456 \leftmargin=5\jsZw\item[]\end{list}\vspace{\baselineskip}}
2457 %</book|report>

```

“普通の” abstract 環境の定義。

```

2458 %<*article|report|slide>
2459 \newbox\@abstractbox
2460 \if@titlepage
2461 \newenvironment{abstract}{%
2462 \titlepage
2463 \null\vfil
2464 \@beginparpenalty\@lowpenalty
2465 \begin{center}%
2466 \headfont \abstractname
2467 \@endparpenalty\@M
2468 \end{center}%

```

BXJS クラスでは、概要の最初の段落に段落下げが入るようにする。

```

2469 \par}%
2470 {\par\vfil\null\endtitlepage}
2471 \else
2472 \newenvironment{abstract}{%
2473 \if@twocolumn
2474 \ifx\maketitle\relax
2475 \section*{\abstractname}%
2476 \else
2477 \global\setbox\@abstractbox\hbox\bgroup
2478 \begin{minipage}[b]{\textwidth}
2479 \small\parindent1\jsZw
2480 \begin{center}%
2481 {\headfont \abstractname\vspace{-.5em}\vspace{\z@}}%
2482 \end{center}%
2483 \list{}{%
2484 \listparindent\parindent
2485 \itemindent \listparindent
2486 \rightmargin \leftmargin}%
2487 \item\relax
2488 \fi
2489 \else
2490 \small
2491 \begin{center}%
2492 {\headfont \abstractname\vspace{-.5em}\vspace{\z@}}%
2493 \end{center}%
2494 \list{}{%
2495 \listparindent\parindent
2496 \itemindent \listparindent
2497 \rightmargin \leftmargin}%
2498 \item\relax
2499 \fi}{\if@twocolumn

```

```

2500     \ifx\maketitle\relax
2501     \else
2502     \endlist\end{minipage}\egroup
2503     \fi
2504     \else
2505     \endlist
2506     \fi}
2507 \fi
2508 %</article|report|slide>
2509 %<*jspf>
2510 \newbox\@abstractbox
2511 \newenvironment{abstract}{%
2512   \global\setbox\@abstractbox\hbox\bgroup
2513   \begin{minipage}[b]{157\jsc@mmm}{\sffamily Abstract}\par
2514     \small
2515     \if@english \parindent6\jsc@mmm \else \parindent1\jsZw \fi}%
2516   {\end{minipage}\egroup}
2517 %</jspf>

```

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

```

2518 %<*book|report>
2519 \ifbxjs@force@chapterabstract
2520   \let\abstract\chapterabstract
2521   \let\endabstract\endchapterabstract
2522 \fi
2523 %</book|report>

```

■キーワード

keywords 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。

```

2524 %<*jspf>
2525 %\newbox\@keywordsbox
2526 %\newenvironment{keywords}{%
2527 %   \global\setbox\@keywordsbox\hbox\bgroup
2528 %   \begin{minipage}[b]{1570\jsc@mmm}{\sffamily Keywords:}\par
2529 %     \small\parindent0\jsZw}%
2530 %   {\end{minipage}\egroup}
2531 %</jspf>

```

■verse 環境

verse 詩のための verse 環境です。

```

2532 \newenvironment{verse}{%
2533   \let \\\=\@centercr
2534   \list{}{%
2535     \itemsep \z@

```

```

2536 \itemindent -2\jsZw % 元: -1.5em
2537 \listparindent\itemindent
2538 \rightmargin \z@
2539 \advance\leftmargin 2\jsZw}% 元: 1.5em
2540 \item\relax}{\endlist}

```

■quotation 環境

quotation 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。

```

2541 \newenvironment{quotation}{%
2542 \list{}{%
2543 \listparindent\parindent
2544 \itemindent\listparindent
2545 \rightmargin \z@}%
2546 \item\relax}{\endlist}

```

■quote 環境

quote 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。

```

2547 \newenvironment{quote}%
2548 {\list{}{\rightmargin\z@}\item\relax}{\endlist}

```

■定理など ltthm.dtx 参照。たとえば次のように定義します。

```

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

```

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

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

```

2549 \def\@begintheorem#1#2{\trivlist\labelsep=1\jsZw
2550 \item[\hskip \labelsep{\headfont #1\ #2}]}
2551 \def\@opargbegintheorem#1#2#3{\trivlist\labelsep=1\jsZw
2552 \item[\hskip \labelsep{\headfont #1\ #2 (#3)}]}

```

titlepage 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^AT_EX 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^AT_EX の標準クラス 2017/02/15 に合わせましたが、こちらで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。

```

2553 \newenvironment{titlepage}{%

```

```

2554 %<book>      \pltx@cleartooddpage %% 2017-02-24
2555      \if@twocolumn
2556          \@restonecoltrue\onecolumn
2557      \else
2558          \@restonecolfalse\newpage
2559      \fi
2560      \thispagestyle{empty}%
2561      \ifodd\c@page\setcounter{page}\@ne\else\setcounter{page}\z@\fi %% 2017-02-
2562          24
2563      }%
2564      {\if@restonecol\twocolumn \else \newpage \fi
2565      \if@twoside\else
2566          \setcounter{page}\@ne
2567      \fi}

```

■付録

`\appendix` 本文と付録を分離するコマンドです。

```

2567 %<!*book&!report>
2568 \newcommand{\appendix}{\par
2569     \setcounter{section}{0}%
2570     \setcounter{subsection}{0}%
2571     \ifnum\bxjs@label@section=\bxjs@label@section@@compat
2572         \gdef\presectionname{\appendixname}%
2573         \gdef\postsectionname{}}%
2574     % \gdef\thesection{\@Alph\c@section}% [2003-03-02]
2575     \gdef\thesection{\presectionname\@Alph\c@section\postsectionname}%
2576     \gdef\thesubsection{\@Alph\c@section.\@arabic\c@subsection}%
2577     \else
2578         \gdef\@secapp{\appendixname}%
2579         \gdef\@secpos{}}%
2580     \gdef\thesection{\@Alph\c@section}%
2581     \fi}
2582 %</!*book&!report>
2583 %<*book|report>
2584 \newcommand{\appendix}{\par
2585     \setcounter{chapter}{0}%
2586     \setcounter{section}{0}%
2587     \gdef\@chapapp{\appendixname}%
2588     \gdef\@chappos{}}%
2589     \gdef\thechapter{\@Alph\c@chapter}}
2590 %</book|report>

```

8.4 パラメータの設定

■array と tabular 環境

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

2591 \setlength\arraycolsep{5\p@?}

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

2592 \setlength\tabcolsep{6\p@?}

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

2593 \setlength\arrayrulewidth{.4\p@}

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

2594 \setlength\doublerulesep{2\p@}

■tabbing 環境

\tabbingsep \' コマンドで入るアキです。

2595 \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。

2596 \skip\@mpfootins = \skip\footins

■framebox 環境

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

\fbxrule \fbbox, \framebox の罫線の幅です。

2597 \setlength\fbxsep{3\p@?}

2598 \setlength\fbxrule{.4\p@}

■equation と eqnarray 環境

\theequation 数式番号を出力するコマンドです。

2599 %<!book&!report>\renewcommand \theequation {\@arabic\c@equation}

2600 %<*book|report>

2601 \@addtoreset{equation}{chapter}

2602 \renewcommand\theequation

2603 {\ifnum \c@chapter>\z@ \thechapter.\fi \@arabic\c@equation}

2604 %</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。

2605 % \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。

2606 % \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。

```
2607 % \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。

```
2608 %<*&book&!report>
2609 \newcounter{figure}
2610 \renewcommand \thefigure {\@arabic\c@figure}
2611 %</!book&!report>
2612 %<*&book|report>
2613 \newcounter{figure}[chapter]
2614 \renewcommand \thefigure
2615     {\ifnum \c@chapter>\z@ \thechapter.\fi \@arabic\c@figure}
2616 %</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。
`\ftype@figure` した。

```
\ext@figure 2617 \def\fps@figure{tbp}
2618 \def\ftype@figure{1}
\fnum@figure 2619 \def\ext@figure{lof}
2620 \def\fnum@figure{\figurename\nobreak\thefigure}
```

figure * 形式は段抜きのフロートです。

```
figure* 2621 \newenvironment{figure}%
2622     {\@float{figure}}%
2623     {\endfloat}
2624 \newenvironment{figure*}%
2625     {\@dblfloat{figure}}%
2626     {\enddblfloat}
```

■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。

```
2627 %<!*book&!report>
2628 \newcounter{table}
2629 \renewcommand\thetable{\@arabic\c@table}
2630 %</!*book&!report>
2631 %<*book|report>
2632 \newcounter{table}[chapter]
2633 \renewcommand \thetable
2634     {\ifnum \c@chapter>\z@ \thechapter.\fi \@arabic\c@table}
2635 %</book|report>
```

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

```
\ext@table 2636 \def\fps@table{tbp}
2637 \def\ftype@table{2}
\fnun@table 2638 \def\ext@table{lot}
2639 \def\fnun@table{\tablename\nobreak\thetable}
```

`table *` は段抜きのフロートです。

```
table* 2640 \newenvironment{table}%
2641     {\@float{table}}%
2642     {\end@float}
2643 \newenvironment{table*}%
2644     {\dblfloat{table}}%
2645     {\enddblfloat}
```

8.6 キャプション

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

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

```
2646 \newlength\abovecaptionskip
2647 \newlength\belowcaptionskip
2648 \setlength\abovecaptionskip{5\p@?} % 元: 10\p@
2649 \setlength\belowcaptionskip{5\p@?} % 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。

```
2650 %<!*jspf>
2651 % \long\def\@makecaption#1#2{{\small
2652 %     \advance\leftskip10\jsc@mmm
```

```

2653 % \advance\rightskip10\jsc@mmm
2654 % \vskip\abovecaptionskip
2655 % \sbox\@tempboxa{#1\hskip1\jsZw\relax #2}%
2656 % \ifdim \wd\@tempboxa >\hsize
2657 % #1\hskip1\jsZw\relax #2\par
2658 % \else
2659 % \global \@minipagefalse
2660 % \hb@xt@\hsize{\hfil\box\@tempboxa\hfil}%
2661 % \fi
2662 % \vskip\belowcaptionskip}}
2663 \long\def\@makecaption#1#2{{\small
2664 \advance\leftskip .0628\linewidth
2665 \advance\rightskip .0628\linewidth
2666 \vskip\abovecaptionskip
2667 \sbox\@tempboxa{#1\zwspace#2}%
2668 \ifdim \wd\@tempboxa <\hsize \centering \fi
2669 #1\zwspace#2\par
2670 \vskip\belowcaptionskip}}
2671 %</!jspf>
2672 %<*jspf>
2673 \long\def\@makecaption#1#2{%
2674 \vskip\abovecaptionskip
2675 \sbox\@tempboxa{\small\sffamily #1\quad #2}%
2676 \ifdim \wd\@tempboxa >\hsize
2677 {\small\sffamily
2678 \list{#1}{%
2679 \renewcommand{\makelabel}[1]{##1\hfil}
2680 \itemsep \z@
2681 \itemindent \z@
2682 \labelsep \z@
2683 \labelwidth 11\jsc@mmm
2684 \listparindent\z@
2685 \leftmargin 11\jsc@mmm}\item\relax #2\endlist}
2686 \else
2687 \global \@minipagefalse
2688 \hb@xt@\hsize{\hfil\box\@tempboxa\hfil}%
2689 \fi
2690 \vskip\belowcaptionskip}
2691 %</jspf>

```

9 フォントコマンド

ここでは L^AT_EX 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 `\scrDeclareOldFontCommand` 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なりました。

[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。

```
2692 \newif\if@jsc@warnoldfontcmd
2693 \@jsc@warnoldfontcmdtrue
2694 \newif\if@jsc@warnoldfontcmdexception
2695 \@jsc@warnoldfontcmdexceptionfalse
```

`\jsc@DeclareOldFontCommand`

```
2696 \newcommand*{\jsc@DeclareOldFontCommand}[3]{%
2697   \g@addto@macro\bxjs@oldfontcmd@list{\do#1}%
2698   \DeclareOldFontCommand{#1}{%
2699     \bxjs@oldfontcmd{#1}#2%
2700   }{%
2701     \bxjs@oldfontcmd{#1}#3%
2702   }%
2703 }
2704 \DeclareRobustCommand*{\jsc@warnoldfontcmd}[1]{%
2705   \ClassInfo\bxjs@clsname
2706   {Old font command '\string#1' is used!!\MessageBreak
2707     The first occurrence is}%
2708 }
```

`\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。

```
2709 \newcommand*{\allowoldfontcommands}{%
2710   \@jsc@warnoldfontcmdexceptiontrue}
2711 \newcommand*{\disallowoldfontcommands}{%
2712   \@jsc@warnoldfontcmdexceptionfalse}

2713 \let\bxjs@oldfontcmd@list\@empty
2714 \def\bxjs@oldfontcmd#1{%
2715   \expandafter\bxjs@oldfontcmd@a\csname bxjs@ofc/\string#1\endcsname#1}
2716 \def\bxjs@oldfontcmd@a#1#2{%
2717   \if@jsc@warnoldfontcmdexception\else
2718     \global\@jsc@warnoldfontcmdfalse
2719     \ifx#1\relax
2720       \global\let#1=t%
2721       \jsc@warnoldfontcmd{#2}%
2722     \fi
2723   \fi}
```

```

2724 \def\bxjs@warnoldfontcmd@final{%
2725 % \par
2726 \global\let\bxjs@warnoldfontcmd@final\@empty
2727 \let\@tempa\@empty
2728 \def\do##1{%
2729   \expandafter\ifx\csname bxjs@ofc/\string##1\endcsname\relax\else
2730     \edef\@tempa{\@tempa \space\string##1}\fi}
2731 \bxjs@oldfontcmd@list
2732 \ifx\@tempa\@empty\else
2733   \ClassWarningNoLine\bxjs@clsname
2734     {Some old font commands were used in text:\MessageBreak
2735       \space\@tempa\MessageBreak
2736       You should note, that since 1994 LaTeX2e provides a\MessageBreak
2737       new font selection scheme called NFSS2 with several\MessageBreak
2738       new, combinable font commands. The
2739       class provides\MessageBreak
2740       the old font commands only for compatibility}
2741 \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。

```

2742 \def\bxjs@warnoldfontcmd@kick@final{%
2743   \g@addto@macro\clearpage{\bxjs@warnoldfontcmd@final}}
2744 \AtEndDocument{\bxjs@warnoldfontcmd@kick@final}

```

`\mc` フォントファミリーを変更します。

```

\gt 2745 \jsc@DeclareOldFontCommand{\mc}{\normalfont\mcfamily}{\mathmc}
\rm 2746 \jsc@DeclareOldFontCommand{\gt}{\normalfont\gtfamily}{\mathgt}
2747 \jsc@DeclareOldFontCommand{\rm}{\normalfont\rmfamily}{\mathrm}
\sff 2748 \jsc@DeclareOldFontCommand{\sf}{\normalfont\sffamily}{\mathsf}
\tt 2749 \jsc@DeclareOldFontCommand{\tt}{\normalfont\ttfamily}{\mathtt}

```

`\bf` ボールドシリーズにします。通常のミディウムシリーズに戻すコマンドは `\mdseries` です。

```

2750 \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。

```

2751 \jsc@DeclareOldFontCommand{\it}{\normalfont\itshape}{\mathit}
2752 \jsc@DeclareOldFontCommand{\sl}{\normalfont\slshape}{\@nomath\sl}
2753 \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 2754 \DeclareRobustCommand*\cal{\@fontswitch\relax\mathcal}
2755 \DeclareRobustCommand*\mit{\@fontswitch\relax\mathnormal}

```

10 相互参照

10.1 目次の類

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

```
\contentsline{section}{タイトル}{ページ}
```

たとえば `\section` に見出し番号が付く場合、上の「タイトル」は

```
\numberline{番号}{見出し}
```

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

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

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

この「番号」は `\thefigure` コマンドで生成された図番号です。

`table` 環境も同様です。

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

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

という書式です。

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

インデント 左側の字下げ量です。

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

`\@pnumwidth` ページ番号の入る箱の幅です。

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

`\@dotsep` 点の間隔です (単位 `mu`)。

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

```
2756 \newcommand\@pnumwidth{1.55em}
2757 \newcommand\@tocrmarg{2.55em}
2758 \newcommand\@dotsep{4.5}
2759 %<!book&!report>\setcounter{tocdepth}{2}
2760 %<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)

```
2761 \newdimen\jsc@tocl@width
2762 \newcommand{\tableofcontents}{%
2763 %<*book|report>
2764 \settowidth\jsc@tocl@width{\headfont\prechaptername\postchaptername}%
2765 \settowidth\@tempdima{\headfont\appendixname}%
2766 \ifdim\jsc@tocl@width<\@tempdima \setlength\jsc@tocl@width{\@tempdima}\fi
2767 \ifdim\jsc@tocl@width<2\jsZw \divide\jsc@tocl@width by 2 \advance\jsc@tocl@width 1\jsZw\fi
2768 \if@twocolumn
2769 \@restonecoltrue\onecolumn
2770 \else
2771 \@restonecolfalse
2772 \fi
2773 \chapter*{\contentsname}%
2774 \mkboth{\contentsname}{}%
2775 %</book|report>
2776 %<*!book&!report>
2777 \settowidth\jsc@tocl@width{\headfont\presectionname\postsectionname}%
2778 \settowidth\@tempdima{\headfont\appendixname}%
2779 \ifdim\jsc@tocl@width<\@tempdima\relax\setlength\jsc@tocl@width{\@tempdima}\fi
2780 \ifdim\jsc@tocl@width<2\jsZw \divide\jsc@tocl@width by 2 \advance\jsc@tocl@width 1\jsZw\fi
2781 \section*{\contentsname}%
2782 \mkboth{\contentsname}{\contentsname}%
2783 %</!book&!report>
2784 \@starttoc{toc}%
2785 %<book|report> \if@restonecol\twocolumn\fi
2786 }
```

`\l@part` 部の目次です。

```
2787 \newcommand*{\l@part}[2]{%
2788 \ifnum \c@tocdepth >-2\relax
2789 %<!book&!report> \addpenalty\@secpenalty
2790 %<book|report> \addpenalty{-\@highpenalty}%
2791 \addvspace{2.25em \@plus\p@?}%
2792 \begingroup
2793 \parindent \z@
2794 % \@pnumwidth should be \@tocrmarg
2795 % \rightskip \@pnumwidth
2796 \rightskip \@tocrmarg
2797 \parfillskip -\rightskip
2798 {\leavevmode
2799 \large \headfont
2800 \setlength\@lnumwidth{4\jsZw}%
2801 #1\hfil \hb@xt@\@pnumwidth{\hss #2}}\par
2802 \nobreak
2803 %<book|report> \global\@nobreaktrue
2804 %<book|report> \everypar{\global\@nobreakfalse\everypar{}}%
```

```

2805 \endgroup
2806 \fi}

```

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

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

```

2807 %<*book|report>
2808 \newcommand*{\l@chapter}[2]{%
2809 \ifnum \c@tocdepth >\m@ne
2810 \addpenalty{-\@highpenalty}%
2811 \addvspace{1.0em \@plus\p@}%
2812 % \vskip 1.0em \@plus\p@ % book.cls では↑がこうなっている
2813 \begingroup
2814 \parindent\z@
2815 % \rightskip\@pnumwidth
2816 \rightskip\@tocrmarg
2817 \parfillskip-\rightskip
2818 \leavevmode\headfont
2819 % % \if@english\setlength\@l@numwidth{5.5em}\else\setlength\@l@numwidth{4.683\jsZw}\fi
2820 \setlength\@l@numwidth{\jsc@tocl@width}\advance\@l@numwidth 2.683\jsZw
2821 \advance\leftskip\@l@numwidth \hskip-\leftskip
2822 #1\nobreak\hfil\nobreak\hbox to\@pnumwidth{\hss#2}\par
2823 \penalty\@highpenalty
2824 \endgroup
2825 \fi}
2826 %</book|report>

```

\l@section 節の目次です。

```

2827 %<!*book&!report>
2828 \newcommand*{\l@section}[2]{%
2829 \ifnum \c@tocdepth >\z@
2830 \addpenalty{\@secpenalty}%
2831 \addvspace{1.0em \@plus\p@}%
2832 \begingroup
2833 \parindent\z@
2834 % \rightskip\@pnumwidth
2835 \rightskip\@tocrmarg
2836 \parfillskip-\rightskip
2837 \leavevmode\headfont
2838 % % \setlength\@l@numwidth{4\jsZw}% 元 1.5em [2003-03-02]
2839 \setlength\@l@numwidth{\jsc@tocl@width}\advance\@l@numwidth 2\jsZw
2840 \advance\leftskip\@l@numwidth \hskip-\leftskip
2841 #1\nobreak\hfil\nobreak\hbox to\@pnumwidth{\hss#2}\par
2842 \endgroup
2843 \fi}
2844 %</!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。

```

2845 %<book|report> % \newcommand*{\l@section}{\@dottedtocline{1}{1\jsZw}{3.683\jsZw}}

```

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

\l@section さらに下位レベルの目次項目の体裁です。あまり使ったことがありませんので、要修正かも
\l@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 2846 %<!*book&!report>
2847 % \newcommand*{\l@section} {\@dottedtocline{2}{1.5em}{2.3em}}
2848 % \newcommand*{\l@subsubsection}{\@dottedtocline{3}{3.8em}{3.2em}}
2849 % \newcommand*{\l@paragraph} {\@dottedtocline{4}{7.0em}{4.1em}}
2850 % \newcommand*{\l@subparagraph} {\@dottedtocline{5}{10em}{5em}}
2851 %
2852 % \newcommand*{\l@section} {\@dottedtocline{2}{1zw}{3zw}}
2853 % \newcommand*{\l@subsubsection}{\@dottedtocline{3}{2\jsZw}{3\jsZw}}
2854 % \newcommand*{\l@paragraph} {\@dottedtocline{4}{3\jsZw}{3\jsZw}}
2855 % \newcommand*{\l@subparagraph} {\@dottedtocline{5}{4\jsZw}{3\jsZw}}
2856 %
2857 \newcommand*{\l@section}{%
2858     \@tempdima\jsc@tocl@width \advance\@tempdima -1\jsZw
2859     \@dottedtocline{2}{\@tempdima}{3\jsZw}}
2860 \newcommand*{\l@subsubsection}{%
2861     \@tempdima\jsc@tocl@width \advance\@tempdima 0\jsZw
2862     \@dottedtocline{3}{\@tempdima}{4\jsZw}}
2863 \newcommand*{\l@paragraph}{%
2864     \@tempdima\jsc@tocl@width \advance\@tempdima 1\jsZw
2865     \@dottedtocline{4}{\@tempdima}{5\jsZw}}
2866 \newcommand*{\l@subparagraph}{%
2867     \@tempdima\jsc@tocl@width \advance\@tempdima 2\jsZw
2868     \@dottedtocline{5}{\@tempdima}{6\jsZw}}
2869 %</!*book&!report>
2870 %<!*book|report>
2871 % \newcommand*{\l@section} {\@dottedtocline{2}{3.8em}{3.2em}}
2872 % \newcommand*{\l@subsubsection}{\@dottedtocline{3}{7.0em}{4.1em}}
2873 % \newcommand*{\l@paragraph} {\@dottedtocline{4}{10em}{5em}}
2874 % \newcommand*{\l@subparagraph} {\@dottedtocline{5}{12em}{6em}}
2875 \newcommand*{\l@section}{%
2876     \@tempdima\jsc@tocl@width \advance\@tempdima -1\jsZw
2877     \@dottedtocline{1}{\@tempdima}{3.683\jsZw}}
2878 \newcommand*{\l@subsubsection}{%
2879     \@tempdima\jsc@tocl@width \advance\@tempdima 2.683\jsZw
2880     \@dottedtocline{2}{\@tempdima}{3.5\jsZw}}
2881 \newcommand*{\l@subsubsection}{%
2882     \@tempdima\jsc@tocl@width \advance\@tempdima 6.183\jsZw
2883     \@dottedtocline{3}{\@tempdima}{4.5\jsZw}}
2884 \newcommand*{\l@paragraph}{%
2885     \@tempdima\jsc@tocl@width \advance\@tempdima 10.683\jsZw
2886     \@dottedtocline{4}{\@tempdima}{5.5\jsZw}}
2887 \newcommand*{\l@subparagraph}{%
2888     \@tempdima\jsc@tocl@width \advance\@tempdima 16.183\jsZw
```

```

2889 \dottedtocline{5}{\@tempdima}{6.5\jsZw}}
2890 %</book|report>

```

`\numberline` 欧文版 L^AT_EX では `\numberline{...}` は幅 `\@tempdima` の箱に左詰めで出力する命令ですが、アスキー版では `\@tempdima` の代わりに `\@lnumwidth` という変数で幅を決めるように再定義しています。後続文字が全角か半角かでスペースが変わらないように `\hspace` を入れておきました。

```

2891 \newdimen\@lnumwidth
2892 \def\numberline#1{\hb@xt@\@lnumwidth{#1\hfil}\hspace{0pt}}

```

`\dottedtocline` L^AT_EX 本体 (ltsect.dtx 参照) での定義と同じですが、`\@tempdima` を `\@lnumwidth` に `\jsTocLine` 変えています。

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

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

とします。

```

2893 \def\jsTocLine{\leaders\hbox{%
2894   $\m@th \mkern \@dotsep mu\hbox{.}\mkern \@dotsep mu$}\hfill}
2895 \def\dottedtocline#1#2#3#4#5{\ifnum #1>\c@tocdepth \else
2896   \vskip \z@ \@plus.2\p@?
2897   {\leftskip #2\relax \rightskip \@tocrmarg \parfillskip -\rightskip
2898    \parindent #2\relax\@afterindenttrue
2899    \interlinepenalty\@M
2900    \leavevmode
2901    \@lnumwidth #3\relax
2902    \advance\leftskip \@lnumwidth \null\nobreak\hskip -\leftskip
2903    {#4}\nobreak
2904    \jsTocLine \nobreak\hb@xt@\@pnumwidth{%
2905      \hfil\normalfont \normalcolor #5\par}\fi}

```

■ 図目次と表目次

`\listoffigures` 図目次を出力します。

```

2906 \newcommand{\listoffigures}{%
2907 %<*book|report>
2908   \if@twocolumn\@restonecoltrue\onecolumn
2909   \else\@restonecolfalse\fi
2910   \chapter*{\listfigurename}%
2911   \@mkboth{\listfigurename}{}%
2912 %</book|report>
2913 %<!*book&!report>
2914   \section*{\listfigurename}%
2915   \@mkboth{\listfigurename}{\listfigurename}%
2916 %</*!book&!report>
2917   \@starttoc{lof}%

```

```

2918 %<book|report> \if@restonecol\twocolumn\fi
2919 }

```

`\l@figure` 図目次の項目を出力します。

```

2920 \newcommand*{\l@figure}{\@dottedtocline{1}{1\jsZw}{3.683\jsZw}}

```

`\listoftables` 表目次を出力します。

```

2921 \newcommand{\listoftables}{%
2922 %<*book|report>
2923 \if@twocolumn\@restonecoltrue\onecolumn
2924 \else\@restonecolfalse\fi
2925 \chapter*{\listtablename}%
2926 \@mkboth{\listtablename}{}%
2927 %</book|report>
2928 %<!*book&!report>
2929 \section*{\listtablename}%
2930 \@mkboth{\listtablename}{\listtablename}%
2931 %<!/book&!report>
2932 \@starttoc{lot}%
2933 %<book|report> \if@restonecol\twocolumn\fi
2934 }

```

`\l@table` 表目次は図目次と同じです。

```

2935 \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。

```

2936 \newdimen\bibindent
2937 \setlength\bibindent{2\jsZw}

```

`thebibliography` 参考文献リストを出力します。

[2016-07-16] L^AT_EX 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。

```

2938 \newenvironment{thebibliography}[1]{%
2939 \@jsc@warnoldfontcmdexceptiontrue
2940 \global\let\presectionname\relax
2941 \global\let\postsectionname\relax
2942 %<article|slide> \section*{\refname}\@mkboth{\refname}{\refname}%
2943 %<*kiyou>
2944 \vspace{1.5\baselineskip}
2945 \subsubsection*{\refname}\@mkboth{\refname}{\refname}%
2946 \vspace{0.5\baselineskip}
2947 %</kiyou>
2948 %<book|report> \chapter*{\bibname}\@mkboth{\bibname}{}%
2949 %<book|report> \addcontentsline{toc}{chapter}{\bibname}%

```



```

2950 \list{\@biblabel{\@arabic\c@enumiv}}%
2951 {\settowidth\labelwidth{\@biblabel{#1}}%
2952 \leftmargin\labelwidth
2953 \advance\leftmargin\labelsep
2954 \@openbib@code
2955 \usecounter{enumiv}%
2956 \let\p@enumiv\@empty
2957 \renewcommand\theenumiv{\@arabic\c@enumiv}}%
2958 %<kiyou> \small
2959 \sloppy
2960 \clubpenalty4000
2961 \@clubpenalty\clubpenalty
2962 \widowpenalty4000%
2963 \sfcode`.\@m}
2964 {\def\@noitemerr
2965 {\@latex@warning{Empty `thebibliography' environment}}}%
2966 \endlist}

```

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

```

2967 \newcommand{\newblock}{\hskip .11em\@plus.33em\@minus.07em}

```

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

```

2968 \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。

```

2969 % \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。かつ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。

```

2970 % \def\@citex[#1]#2{\leavevmode
2971 % \let\@citea\@empty
2972 % \@cite{\@for\@citeb:=#2\do
2973 % {\@citea\def\@citea{, \inhibitglue\penalty\@m\ }%
2974 % \edef\@citeb{\expandafter\@firstofone\@citeb\@empty}}%
2975 % \if@filesw\immediate\write\@auxout{\string\citation{\@citeb}}\fi
2976 % \ifundefined{b@\@citeb}{\mbox{\normalfont\bfseries ?}}%
2977 % \G@refundefinedtrue
2978 % \@latex@warning
2979 % {Citation `{\@citeb' on page \thepage \space undefined}}%
2980 % {\@cite@ofmt{\csname b@\@citeb\endcsname}}}{#1}}
2981 % \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。

```
2982 % \DeclareRobustCommand\cite{\unskip
2983 %   \@ifnextchar [{\@tempwattrue\@citex}{\@tempwafalse\@citex[]}]
2984 % \def\@cite#1#2{${\hbox{\scriptsize{#1}\if@tempwa
2985 %   , \jsInhibitGlue\ #2\fi}) }}$}
```

10.3 索引

`theindex` 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)。

```
2986 \newenvironment{theindex}{% 索引を 3 段組で出力する環境
2987   \if@twocolumn
2988     \onecolumn\@restonecolfalse
2989   \else
2990     \clearpage\@restonecoltrue
2991   \fi
2992   \columnseprule.4pt \columnsep 2\jsZw
2993   \ifx\multicols\@undefined
2994 %<book|report>      \twocolumn[\@makeschapterhead{\indexname}%
2995 %<book|report>      \addcontentsline{toc}{chapter}{\indexname}]%
2996 %<!book&!report>    \def\presectionname{}\def\postsectionname{%
2997 %<!book&!report>    \twocolumn[\section*{\indexname}]%
2998   \else
2999     \ifdim\textwidth<\fullwidth
3000       \setlength{\evensidemargin}{\oddsidemargin}
3001       \setlength{\textwidth}{\fullwidth}
3002       \setlength{\linewidth}{\fullwidth}
3003 %<book|report>      \begin{multicols}{3}[\chapter*{\indexname}%
3004 %<book|report>      \addcontentsline{toc}{chapter}{\indexname}]%
3005 %<!book&!report>    \def\presectionname{}\def\postsectionname{%
3006 %<!book&!report>    \begin{multicols}{3}[\section*{\indexname}]%
3007   \else
3008 %<book|report>      \begin{multicols}{2}[\chapter*{\indexname}%
3009 %<book|report>      \addcontentsline{toc}{chapter}{\indexname}]%
3010 %<!book&!report>    \def\presectionname{}\def\postsectionname{%
3011 %<!book&!report>    \begin{multicols}{2}[\section*{\indexname}]%
3012   \fi
3013   \fi
3014 %<book|report>      \@mkboth{\indexname}{}%
3015 %<!book&!report>    \@mkboth{\indexname}{\indexname}%
3016   \plainifnotempty % \thispagestyle{plain}
3017   \parindent\z@
3018   \parskip\z@ \@plus .3\p@?\relax
3019   \let\item\@idxitem
3020   \raggedright
3021   \footnotesize\narrowbaselines
3022 }
```

```

3023 \ifx\multicols\@undefined
3024 \if@restonecol\onecolumn\fi
3025 \else
3026 \end{multicols}
3027 \fi
3028 \clearpage
3029 }

```

`\@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 3030 \newcommand{\@idxitem}{\par\hangindent 4\jsZw} % 元 40pt
\subsubitem 3031 \newcommand{\subitem}{\@idxitem \hspace*{2\jsZw}} % 元 20pt
3032 \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。

```

3033 \newcommand{\indexspace}{\par \vskip 10\p@? \@plus5\p@? \@minus3\p@?\relax}

```

`\seename` 索引の `\see`, `\seealso` コマンドで出力されるものです。デフォルトはそれぞれ *see*, *see also*

`\alsoname` という英語ですが、ここではとりあえず両方とも「→」に変えました。⇒ (\rightarrow)
などでもいいでしょう。

```

3034 \newcommand\seename{\if@english see\else →\fi}
3035 \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。p_AT_EX 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。

```

3036 \ifx\pltx@foot@penalty\@undefined
3037 \let\footnotes@ve=\footnote
3038 \def\footnote{\jsInhibitGlue\footnotes@ve}
3039 \let\footnotemarks@ve=\footnotemark
3040 \def\footnotemark{\jsInhibitGlue\footnotemarks@ve}
3041 \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_TE_X 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_AT_EX の変更に追随しました (Thanks: 角藤さん)。p_AT_EX 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_TE_X 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 `newttext` や `newpTtext` の使用時におかしくなっています。これらのパッケージは内部で `\thefootnote` を再定義していますので、気になる場合はパッケージを読み込むときに `defaultsups` オプションを付けてください (qa:57284, qa:57287)。

```
3042 \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。

```
3043 % \def\thefootnote{\ifnum\c@footnote>\z@ 注\kern0.1\jsZw\@arabic\c@footnote\fi}
```

`\footnoterule` 本文と脚注の間の罫線です。

```
3044 \renewcommand{\footnoterule}{%
3045   \kern-2.6\p@? \kern-.4\p@
3046   \hrule width .4\columnwidth
3047   \kern 2.6\p@?}
```

`\c@footnote` 脚注番号は章ごとにリセットされます。

```
3048 %<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_EX and TUG NEWS*, Vol. 2, No. 4 (1993), p. 9)

[2016-08-25] コミュニティ版 p_AT_EX 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_AT_EX のバグ修正に追随しました。

[2016-11-29] 古い p_AT_EX 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。

```
3049 \long\def\@footnotetext{%
3050   \insert\footins\bgroup
3051     \normalfont\footnotesize
3052     \interlinepenalty\interfootnotelinepenalty
3053     \splittopskip\footnotesep
3054     \splitmaxdepth \dp\strutbox \floatingpenalty \@MM
3055     \hsize\columnwidth \@parboxrestore
3056     \protected@edef\@currentlabel{%
3057       \csname p@footnote\endcsname\@thefnmark
```

```

3058     }%
3059     \color@begingroup
3060     \@makefntext{%
3061         \rule{z@}{\footnotesep\ignorespaces}%
3062         \futurelet\jsc@next\jsc@fo@t}
3063 \def\jsc@fo@t{\ifcat\bgroup\noexpand\jsc@next \let\jsc@next\jsc@f@t
3064             \else \let\jsc@next\jsc@f@t\fi \jsc@next}
3065 \def\jsc@f@t{\bgroup\aftergroup\jsc@@foot\let\jsc@next}
3066 \def\jsc@f@t#1{#1\jsc@@foot}
3067 \def\jsc@@foot{\@finalstrut\strutbox\color@endgroup\egroup
3068     \ifx\pltx@foot@penalty\undefined\else
3069         \ifhmode\null\fi
3070         \ifnum\pltx@foot@penalty=z@\else
3071             \penalty\pltx@foot@penalty
3072             \pltx@foot@penalty\z@
3073         \fi
3074     \fi}

```

`\@makefntext` 実際に脚注を出力する命令です。`\@makefnmark` は脚注の番号を出力する命令です。ここでは脚注が左端から一定距離に来るようにしてあります。

```

3075 \newcommand\@makefntext[1]{%
3076     \advance\leftskip 3\jsZw
3077     \parindent 1\jsZw
3078     \noindent
3079     \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。

```

3080 % \def\@xfootnotenext[#1]{%
3081 %     \begingroup
3082 %         \ifnum#1>z@
3083 %             \csname c@\@mpfn\endcsname #1\relax
3084 %             \unrestored@protected@xdef\@thefnmark{\thempfn}%
3085 %         \else
3086 %             \unrestored@protected@xdef\@thefnmark{}%
3087 %         \fi
3088 %     \endgroup
3089 %     \@footnotetext}

```

ここまでのコードは JS クラスを踏襲する。

11 段落の頭へのグルー挿入禁止

段落頭のかぎカッコなどを見かけ 1 字半下げから全角 1 字下げに直します。

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

```
3090 \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。

```
3091 \def\everyparhook{\jsInhibitGlueAtParTop}
3092 \ifnum\bxjs@everyparhook=\bxjs@everyparhook@@compat
3093 \g@addto@macro\bxjs@begin@document@hook{\everypar{\everyparhook}}
3094 \fi
```

[2016-07-18] `\inhibitglue` の発行対象を `\inhibitxspcode` が 2 に設定されているものすべてに拡大しました。

[2016-12-01] すぐ上の変更で `\@tempa` を使っていたのがよくなかったので、プレフィックスを付けて `\jsc@tempa` にしました (forum:2085)。

[2017-02-13] `\jsc@tempa` は実はテンポラリではなく「この処理専用のユニーク制御綴」である必要があります。間違って別の箇所でする危険性が高いので、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。

```
3095 \ifx j\jsEngine
3096 \def\@inhibitglue{%
3097   \futurelet\@let@token\@@inhibitglue}
3098 \begingroup
3099 \let\GDEF=\gdef
3100 \let\CATCODE=\catcode
3101 \let\ENDGROUP=\endgroup
3102 \CATCODE`k=12
3103 \CATCODE`a=12
3104 \CATCODE`n=12
3105 \CATCODE`j=12
3106 \CATCODE`i=12
3107 \CATCODE`c=12
3108 \CATCODE`h=12
3109 \CATCODE`r=12
3110 \CATCODE`t=12
```

```

3111 \CATCODE`e=12
3112 \GDEF\KANJI@CHARACTER{kanji character }
3113 \ENDGROUP
3114 \def\@@inhibitglue{%
3115   \expandafter\expandafter\expandafter\jsc@inhibitglue\expandafter\meaning\expandafter\@let@to
3116 \expandafter\def\expandafter\jsc@inhibitglue\expandafter#\expandafter1\KANJI@CHARACTER#2#3\jsc
3117   \def\jsc@ig@temp{#1}%
3118   \ifx\jsc@ig@temp\@empty
3119     \ifnum\the\inhibitxspcode`#2=2\relax
3120       \inhibitglue
3121     \fi
3122   \fi}
3123 \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。

```

3124 \ifnum\bxjs@everyparhook=\bxjs@everyparhook@@compat

```

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

まず、環境の直後の段落です。

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

```

3125 \def\@doendpe{%
3126   \@endpetrue
3127   \def\par{%
3128     \@restorepar\clubpenalty\@clubpenalty\everypar{\everyparhook}\par\@endpefalse}%
3129   \everypar{\setbox\z@\lastbox}\everypar{\everyparhook}\@endpefalse\everyparhook}}

```

[2017-08-31] `minipage` 環境にも対策します。

```

3130 \def\@setminipage{%
3131   \@minipagetrue
3132   \everypar{\@minipagefalse\everypar{\everyparhook}}}%
3133 }

```

`\item` 命令の直後です。

```

3134 \def\@item[#1]{%
3135   \if@noparitem
3136     \@donoparitem
3137   \else
3138     \if@inlabel
3139       \indent \par
3140     \fi
3141     \ifhmode
3142       \unskip\unskip \par
3143     \fi
3144     \if@newlist

```

```

3145     \if@nobreak
3146         \@nbitem
3147     \else
3148         \addpenalty\@beginparpenalty
3149         \addvspace\@topsep
3150         \addvspace{-\parskip}%
3151     \fi
3152 \else
3153     \addpenalty\@itempenalty
3154     \addvspace\itemsep
3155 \fi
3156 \global\@inlabeltrue
3157 \fi
3158 \everypar{%
3159     \@minipagefalse
3160     \global\@newlistfalse
3161     \if@inlabel
3162         \global\@inlabelfalse
3163         {\setbox\z@\lastbox
3164         \ifvoid\z@
3165             \kern-\itemindent
3166         \fi}%
3167     \box\@labels
3168     \penalty\z@
3169 \fi
3170 \if@nobreak
3171     \@nobreakfalse
3172     \clubpenalty \@M
3173 \else
3174     \clubpenalty \@clubpenalty
3175     \everypar{\everyparhook}%
3176 \fi
3177 \everyparhook}%
3178 \if@noitemarg
3179     \@noitemargfalse
3180     \if@nmbrlist
3181         \refstepcounter\@listctr
3182     \fi
3183 \fi
3184 \sbox\@tempboxa{\makelabel{#1}}%
3185 \global\setbox\@labels\hbox{%
3186     \unhbox\@labels
3187     \hskip \itemindent
3188     \hskip -\labelwidth
3189     \hskip -\labelsep
3190     \ifdim \wd\@tempboxa >\labelwidth
3191         \box\@tempboxa
3192     \else
3193         \hbox to\labelwidth {\unhbox\@tempboxa}%

```



```

3194 \fi
3195 \hskip \labelsep}%
3196 \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。

```

3197 \def\@afterheading{%
3198 \@nbreaktrue
3199 \everypar{%
3200 \if@nbreak
3201 \@nbreakfalse
3202 \clubpenalty \@M
3203 \if@afterindent \else
3204 {\setbox\z@\lastbox}%
3205 \fi
3206 \else
3207 \clubpenalty \@clubpenalty
3208 \everypar{\everyparhook}%
3209 \fi\everyparhook}}

```

「`\everyparhook` 用のパッチ処理」はここまで。

```

3210 \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` を読みこんだ場合に `lltjcore.sty` によって上書きされるのを防ぐため遅延させる。

```

3211 \ifnum\bxjs@everyparhook=\bxjs@everyparhook@@none\else
3212 \AtEndOfPackage{%
3213 \def\@gnewline #1{%
3214 \ifvmode
3215 \@nolnerr
3216 \else
3217 \unskip \reserved@e {\reserved@f#1}\nbreak \hfil \break \null
3218 \jsInhibitGlue \ignorespaces
3219 \fi}
3220 }
3221 \fi

```

12 いろいろなロゴ

L^AT_EX 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。

※ $\small{}$ 、 $\small{}$ の制御綴は定義しない。

```
3222 \if@jslogo
3223   \IfFileExists{jslogo.sty}{%
3224     \RequirePackage{jslogo}%
3225   }{%
3226     \ClassWarningNoLine{bxjs@clsname}
3227     {The package 'jslogo' is not installed.\MessageBreak
3228      It is included in the recent release of\MessageBreak
3229      the 'jsclasses' bundle}
3230   }
3231 \fi
```

13 amsmath との衝突の回避

$\lt{x@ifnextchar}$ 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^AT_EX の \ProvidesFile で悪さをする例が F_TE_X で報告されています。これを避けるための tDB さんのフィックスを挿入しておきます。副作用がありましたらお知らせください。

この現象については私の TeX 掲示板 4273～, 16058～ で議論がありました。なお、AMS 関係のパッケージを読み込む際に psamsfonts オプションを与えても回避できます (Thanks: しっぱ愛好家さん)。

[2016-11-19] 本家の ltclass.dtx 2004/01/28 v1.1g で修正されているのでコメントアウトしました。

```
3232 %\let\lt{x@ifnextchar}\@ifnextchar
3233 %\def\ProvidesFile#1{%
3234 %  \begingroup
3235 %    \catcode`\ 10 %
3236 %    \ifnum \endlinechar<256 %
3237 %      \ifnum \endlinechar>\m@ne
3238 %        \catcode\endlinechar 10 %
3239 %      \fi
3240 %    \fi
3241 %    \@makeother\/%
```

```

3242 % \makeother\&%
3243 % \ltx@ifnextchar[{\@providesfile{#1}}{\@providesfile{#1}[]}]

```

14 初期設定

■いろいろな語

```

\prepartname
\postpartname 3244 \newcommand{\prepartname}{\if@english Part~\else 第\fi}
\prechaptername 3245 \newcommand{\postpartname}{\if@english\else 部\fi}
3246 %<book|report>\newcommand{\prechaptername}{\if@english Chapter~\else 第\fi}
\postchaptername 3247 %<book|report>\newcommand{\postchaptername}{\if@english\else 章\fi}
\presectionname 3248 \newcommand{\presectionname}{}% 第
\postsectionname 3249 \newcommand{\postsectionname}{}% 節

\contentsname

\listfigurename 3250 \newcommand{\contentsname}{\if@english Contents\else 目次\fi}
\listtablename 3251 \newcommand{\listfigurename}{\if@english List of Figures\else 図目次\fi}
3252 \newcommand{\listtablename}{\if@english List of Tables\else 表目次\fi}

\refname
\bibname 3253 \newcommand{\refname}{\if@english References\else 参考文献\fi}
3254 \newcommand{\bibname}{\if@english Bibliography\else 参考文献\fi}
\indexname 3255 \newcommand{\indexname}{\if@english Index\else 索引\fi}

\figurename
\tablename 3256 %<!jspf>\newcommand{\figurename}{\if@english Fig.~\else 図\fi}
3257 %<jspf>\newcommand{\figurename}{Fig.~}
3258 %<!jspf>\newcommand{\tablename}{\if@english Table~\else 表\fi}
3259 %<jspf>\newcommand{\tablename}{Table~}

\appendixname
\abstractname 3260 % \newcommand{\appendixname}{\if@english Appendix~\else 付録\fi}
3261 \newcommand{\appendixname}{\if@english \else 付録\fi}
3262 %<!book>\newcommand{\abstractname}{\if@english Abstract\else 概要\fi}

```

■**今日の日付** L^AT_EX で処理した日付を出力します。jarticle などと違って、標準を西暦にし、余分な空白が入らないように改良しました。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。

```

3263 \@onlypreamble\bxjs@decl@Seireki@cmds

```

```

3264 \@tempswafalse
3265 \if p\jsEngine \@tempwattrue \fi
3266 \if n\jsEngine \@tempwattrue \fi
3267 \bxjs@cond\if@tempswa\fi{%
3268 % 欧文 8bitTeX の場合
3269 \newif\ifjsSeireki \jsSeirekitrue
3270 \def\bxjs@decl@Seireki@cmds{%
3271   \def\西暦{\jsSeirekitrue}%
3272   \def\和暦{\jsSeirekifalse\bxjs@wareki@used}}
3273 \def\Seireki{\jsSeirekitrue}
3274 \def\Wareki{\jsSeirekifalse\bxjs@wareki@used}
3275 \def\bxjs@if@use@seireki{\bxjs@cond\ifjsSeireki\fi}
3276 \def\bxjs@iai{\noexpand~}
3277 }{%
3278 \newif\if 西暦 \西暦 true
3279 \def\bxjs@decl@Seireki@cmds{%
3280   \def\西暦{\西暦 true}%
3281   \def\和暦{\西暦 false\bxjs@wareki@used}}
3282 \def\Seireki{\西暦 true}
3283 \def\Wareki{\西暦 false\bxjs@wareki@used}
3284 \def\bxjs@if@use@seireki{\bxjs@cond\if 西暦\fi}
3285 \let\bxjs@iai@empty
3286 }
3287 \bxjs@decl@Seireki@cmds
3288 \let\bxjs@unxp@firstofone \let\bxjs@onxp@firstofone
3289 \bxjs@test@engine\unexpanded{%
3290   \let\bxjs@unxp\unexpanded \def\bxjs@onxp{\unexpanded\expandafter}}

```

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

```

3291 \newif\ifbxjs@bxwareki@avail
3292 \IfFileExists{bxwareki.sty}{%
3293   \RequirePackage{bxwareki}[]%
3294   \bxjs@bxwareki@availtrue}{%

```

\bxjs@wareki@used 和暦が非対応の場合に警告を出す。

```

3295 \ifbxjs@bxwareki@avail \let\bxjs@wareki@used\empty
3296 \else
3297   \bxjs@robust@def\bxjs@wareki@used{%
3298     \global\let\bxjs@wareki@used\empty
3299     \ClassWarning\bxjs@clsname
3300       {Wareki mode is not supported, since\MessageBreak
3301         'bxwareki' is unavailable, reported}}
3302   \g@addto@macro\bxjs@begin@document@hook{%
3303     \let\bxjs@wareki@used\empty}
3304 \fi

```

\jayear 和暦における年の表記（元号 + 年数）。

\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。

```

3305 \ifbxjs@bxwareki@avail
3306   \let\jyear\warekiyear
3307   \def\bxjs@tmpa{H}\ifx\bxjs@tmpa\warekigengoinitial
3308     \newcount\heisei \heisei=\value{warekiyear}
3309   \fi

    bxwareki が使えない場合は西暦表示にする。

3310 \else
3311   \edef\jyear{\the\year \bxjs@iai}
3312 \fi

```

\today 英語、西暦、和暦で場合分けをする。

```

3313 \let\bxjs@next\relax
3314 \ifbxjs@bxwareki@avail \ifx\warekigengo\@empty\else
3315   \def\bxjs@next{\bxjs@onxp{\warekitoday}}
3316 \fi\fi
3317 \edef\bxjs@today{%
3318   \if@english
3319     \ifcase\month\or
3320       January\or February\or March\or April\or May\or June\or
3321       July\or August\or September\or October\or November\or December\fi
3322     \space\number\day, \number\year
3323   \else
3324     \ifx\bxjs@next\relax \expandafter\@firstoftwo
3325     \else \noexpand\bxjs@if@use@seireki
3326     \fi {%
3327       \number\year\bxjs@iai\bxjs@unxp{年}%
3328       \bxjs@iai\number\month\bxjs@iai\bxjs@unxp{月}%
3329       \bxjs@iai\number\day\bxjs@iai\bxjs@unxp{日}%
3330     }\bxjs@next}%
3331   \fi}
3332 \let\today\bxjs@today

```

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

```

3333 \g@addto@macro\bxjs@begin@document@hook{%
3334   \ifx\bbl@jpn@maybekansuji\@undefined\else
3335     \bxjs@decl@Seireki@cmds
3336     \g@addto@macro\datejapanese{%
3337       \let\today\bxjs@today}%
3338   \fi}

```

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

```

3339 \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。

```
3340 %<slide>\pagestyle{empty}%
3341 %<article|report>\pagestyle{plain}%
3342 %<book>\pagestyle{headings}%
3343 \pagenumbering{arabic}
3344 \if@twocolumn
3345   \twocolumn
3346   \sloppy
3347   \flushbottom
3348 \else
3349   \onecolumn
3350   \raggedbottom
3351 \fi
3352 %<*slide>
3353 \renewcommand\familydefault{\sfdefault}
3354 \raggedright
3355 %</slide>
```

■BXJS 独自の追加処理 🍷

フックを実行する。

```
3356 \bxjs@pre@jadriver@hook
```

和文ドライバのファイルを読み込む。

```
3357 \input{bxjsja-\bxjs@jadriver.def}
```

おしまい。

```
3358 %</class>
```

以上です。

付録 A 和文ドライバの仕様

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

- `\jsDocClass` [文字トークンの `let`] 文書クラスの種類を示し、次のいずれかと一致する (`\if` で判定可能)。
 - `\jsArticle` `bxjsarticle` クラス
 - `\jsBook` `bxjsbook` クラス
 - `\jsReport` `bxjsreport` クラス
 - `\jsSlide` `bxjsslide` クラス
- `\jsEngine` [文字トークンの `let`] 使用されているエンジンの種別。 (`\if` で判定可能)。
 - `p` `pdfTeX` (DVI モードも含む)
 - `l` `LuaTeX` (〃)
 - `x` `XYTeX`
 - `j` `pTeX` または `upTeX`
 - `n` 以上の何れでもない
- `\ifjsWithupTeX` [スイッチ] 使用されているエンジンが `upTeX` であるか。
- `\ifjsWitheTeX` [スイッチ] 使用されているエンジンが `eε-TeX` 拡張であるか。
- `\ifjsInPdfMode` [スイッチ] 使用されているエンジンが (`pdfTeX`・`LuaTeX` の) PDF モードであるか。
- `\jsUnusualPtSize` [整数定数を表す文字列のマクロ] 基底フォントサイズが 10pt、11pt、12pt のいずれでもない場合の `\@ptsize` の値。 (`\@ptsize` 自体があまり有用でないと思われる。)
- `\jsScale` [実数を表す文字列のマクロ] 和文フォントサイズの要求サイズに対するスケール。クラスオプション `scale` で指定される。(既定値は 0.924715。)
- `\jsJaFont` [マクロ] 和文フォント設定を表す文字列。クラスオプション `jafont` で指定された値。
- `\jsJaParam` [マクロ] 和文モジュールに渡すパラメタを表す文字列。この値が何を表すかは決まっておらず、各々の和文モジュールが独自に解釈する。クラスオプション `japaram` で指定された値。
- `\jsInhibitGlue` [マクロ] `\inhibitglue` という命令が定義されていればそれを実行し、そうでなければ何もしない。JS クラスで `\inhibitglue` を用いている箇所は全て `\jsInhibitGlue` に置き換えられている。従って、`\inhibitglue` は未定義でも動作するが、その実装がある場合は BXJS クラスはそれを活用する。
- `\jsInhibitGlueAtParTop` [マクロ] ★ 段落先頭におけるカギ括弧の位置調整を行うマクロ。全ての段落先頭で呼び出される。
- `\jsZw` [内部寸法値] 「現在の全角幅」を表す変数。JS クラスで `zw` 単位で設定されている長さパラメタはこの変数を単位として設定されている。この変数の値は実際に

用いられる「和文フォント」のメトリックに基づくのではなく、機械的に `\jsScale` × (フォントサイズ) であると定められている (フォントサイズ変更の度に再設定される)。従って、「和文コンポーネント」はこの設定と辻褄が合うように和文フォントサイズを調整する必要がある。ほとんどの場合、和文フォントを NFSS で規定する際に `\jsScale` の値をスケール値として与えれば上手くいく。

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

以下で標準で用意されている和文ドライバの実装を示す。

```
3359 %<*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-jb 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: 本来は「minimal にすら依存しない」はずのものが minimal のコード中に書かれているような気がする……。

B.1 補助マクロ

```
3360 %<*minimal>
```

```
3361 %% このファイルは日本語文字を含みます
```

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

```
3362 \def\DeclareJaTextFontCommand#1#2{%
```

```
3363   \DeclareRobustCommand#1[1]{%
```

```
3364     \relax
```

```
3365     \ifmmode \expandafter\nfss@text \fi
```

```
3366     {#2#1}}%
```

```
3367 }
```

```
\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。
```

```
3368 \def\DeclareJaMathFontCommand#1#2{%
```

```
3369   \DeclareRobustCommand#1[1]{%
```

```
3370     \relax
```

```
3371     \ifmmode\else \non@alpherr{#1\space}\fi
```



```

3372 \nfss@text{\fontfamily\familydefault
3373 \fontseries{m}\fontshape{n}\selectfont\relax
3374 #2##1}%
3375 }%
3376 }

```

`\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。

```

3377 \long\def\bxjs@@CSsfdefault{\sfdefault}%
3378 \onlypreamble\bxjs@if@sf@default
3379 \def\bxjs@if@sf@default#1{%
3380 \ifx\familydefault\bxjs@@CSsfdefault#1\fi
3381 \g@addto@macro\bxjs@begin@document@hook{%
3382 \ifx\familydefault\bxjs@@CSsfdefault#1\fi}%
3383 }

```

`\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。

```

3384 \@tempdima\p@ \bxjs@invscale\@tempdima\jsScale
3385 \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。

```

3386 \def\jsLetHeadChar#1#2{%
3387 \begingroup
3388 \escapechar=`\ %
3389 \let\bxjs@tmpa={% brace-match-hack
3390 \bxjs@let@hchar@exp#2}%
3391 \endgroup
3392 \let#1\bxjs@g@tmpa}
3393 \def\bxjs@let@hchar@exp{%
3394 \futurelet\@let@token\bxjs@let@hchar@exp@a}
3395 \def\bxjs@let@hchar@exp@a{%
3396 \bxjs@cond\ifcat\noexpand\@let@token\bgroup\fi{% 波括弧
3397 \bxjs@let@hchar@out\let\relax
3398 }{\bxjs@cond\ifcat\noexpand\@let@token\@sptoken\fi{% 空白
3399 \bxjs@let@hchar@out\let\space%
3400 }{\bxjs@cond\if\noexpand\@let@token\@backslashchar\fi{% バックスラッシュ
3401 \bxjs@let@hchar@out\let\@backslashchar
3402 }{\bxjs@let@hchar@exp@b}}}}
3403 \def\bxjs@let@hchar@exp@b#1{%
3404 \expandafter\bxjs@let@hchar@exp@c\string#1?\@nil#1}

```

```

3405 \def\bxjs@let@hchar@exp@c#1#2\@nil{%
3406 %\message{<#1#2>}%
3407 \bxjs@cond@if#1\@backslashchar\fi{% 制御綴
3408 \bxjs@cond\expandafter\ifx\noexpand\@let@token\@let@token\fi{%
3409 \bxjs@let@hchar@out\let\relax
3410 }{%else
3411 \expandafter\bxjs@let@hchar@exp
3412 }%
3413 }{%else
3414 \bxjs@let@hchar@chr#1%
3415 }}
3416 \def\bxjs@let@hchar@chr#1{%
3417 \bxjs@let@hchar@out\def{{#1}}}
3418 \def\bxjs@let@hchar@out#1#2{%
3419 \global#1\bxjs@gtmpa#2\relax
3420 \toks@\bgroup}% skip to right brace

```

UTF-8 のバイト列を扱うコード。

```

3421 \chardef\bxjs@let@hchar@csta=128
3422 \chardef\bxjs@let@hchar@cstb=192
3423 \chardef\bxjs@let@hchar@cstc=224
3424 \chardef\bxjs@let@hchar@cstd=240
3425 \chardef\bxjs@let@hchar@cste=248
3426 \let\bxjs@let@hchar@chr@ue@a\bxjs@let@hchar@chr
3427 \def\bxjs@let@hchar@chr@ue#1{%
3428 \@tempcnta=#1\relax
3429 %\message{\the\@tempcnta}%
3430 \bxjs@cond@ifnum\@tempcnta<\bxjs@let@hchar@csta\fi{%
3431 \bxjs@let@hchar@chr@ue@a#1%
3432 }{\bxjs@cond@ifnum\@tempcnta<\bxjs@let@hchar@cstb\fi{%
3433 \bxjs@let@hchar@out\let\relax
3434 }{\bxjs@cond@ifnum\@tempcnta<\bxjs@let@hchar@cstc\fi{%
3435 \bxjs@let@hchar@chr@ue@b
3436 }{\bxjs@cond@ifnum\@tempcnta<\bxjs@let@hchar@cstd\fi{%
3437 \bxjs@let@hchar@chr@ue@c
3438 }{\bxjs@cond@ifnum\@tempcnta<\bxjs@let@hchar@cste\fi{%
3439 \bxjs@let@hchar@chr@ue@d
3440 }{%else
3441 \bxjs@let@hchar@out\let\relax
3442 }}}}
3443 \def\bxjs@let@hchar@chr@ue@a#1{%
3444 \bxjs@let@hchar@out\def{{#1}}}
3445 \def\bxjs@let@hchar@chr@ue@b#1#2{%
3446 \bxjs@let@hchar@out\def{{#1#2}}}
3447 \def\bxjs@let@hchar@chr@ue@c#1#2#3{%
3448 \bxjs@let@hchar@out\def{{#1#2#3}}}
3449 \def\bxjs@let@hchar@chr@ue@d#1#2#3#4{%
3450 \bxjs@let@hchar@out\def{{#1#2#3#4}}}

```

B.2 (u)pTeX 用の設定

```
3451 \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。

```
3452 \def\bxjs@let@hchar@chr@pp#1#2{%
3453   \expandafter\bxjs@let@hchar@chr@pp@a\meaning#2\relax#1#2}
3454 \def\bxjs@let@hchar@chr@pp@a#1#2\relax#3#4{%
3455   \%message{(\meaning#3:\meaning#4)}%
3456   \bxjs@cond\if#1k\fi{%
3457     \bxjs@let@hchar@out\def{#4}}%
3458   }{%else
3459     \bxjs@let@hchar@chr@ue#3#4%
3460   }}
3461 \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。

```
3462 \edef\jsc@JYn{\ifjsWithupTeX JY2\else JY1\fi}
3463 \edef\jsc@JTn{\ifjsWithupTeX JT2\else JT1\fi}
3464 \edef\jsc@pfx@{\ifjsWithupTeX u\fi}
```

`\bxjs@declarefontshape` は標準の和文フォント宣言である。後で `\bxjs@scale` を求めるため一旦マクロにしておく。`\bxjs@sizereference` は全角幅を測定する時に参照するフォント。

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

```
3465 \@onlypreamble\bxjs@declarefontshape
3466 \ifjsWithupTeX
3467 \def\bxjs@declarefontshape{%
3468   \DeclareFontShape{JY2}{mc}{m}{n}{<->s*[\bxjs@scale]upjpnrm-h}{}%
3469   \DeclareFontShape{JY2}{gt}{m}{n}{<->s*[\bxjs@scale]upjpngt-h}{}%
3470   \DeclareFontShape{JT2}{mc}{m}{n}{<->s*[\bxjs@scale]upjpnrm-v}{}%
3471   \DeclareFontShape{JT2}{gt}{m}{n}{<->s*[\bxjs@scale]upjpngt-v}{}%
3472 }
3473 \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。

```
3474 \else
3475 \def\bxjs@declarefontshape{%
3476   \DeclareFontShape{JY1}{mc}{m}{n}{<->s*[\bxjs@scale]jis}{}%
3477 }
```

```

3477 \DeclareFontShape{JY1}{gt}{m}{n}{<->s*[\bxjs@scale]jisg}{}%
3478 \DeclareFontShape{JT1}{mc}{m}{n}{<->s*[\bxjs@scale]tmin10}{}%
3479 \DeclareFontShape{JT1}{gt}{m}{n}{<->s*[\bxjs@scale]tgoth10}{}%
3480 }
3481 \def\bxjs@sizereference{jis}
3482 \fi

```

既に使用されている標準和文フォント定義がもしあれば取り消す。

```

3483 \def\bxjs@next#1/#2/#3/#4/#5\relax{%
3484   \def\bxjs@tmpb{#5}}
3485 \ifjsWithpTeXng \def\bxjs@tmpb{10}%
3486 \else
3487 \expandafter\expandafter\expandafter\bxjs@next
3488 \expandafter\string\the\jfont\relax
3489 \fi
3490 \@for\bxjs@tmpa:={\jsc@JYn/mc/m/n,\jsc@JYn/gt/m/n,%
3491                  \jsc@JTn/mc/m/n,\jsc@JTn/gt/m/n}\do
3492   {\expandafter\let\csname\bxjs@tmpa/10\endcsname=\@undefined
3493    \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。

```

3494 \begingroup
3495 % 参照用フォント (\bxjs@sizereference) の全角空白の幅を取得
3496 \font\bxjs@tmpa=\bxjs@sizereference\space at 10pt
3497 \setbox\z@\hbox{\bxjs@tmpa\char\jis"2121\relax}
3498 % 幅が丁度 10pt なら補正は不要
3499 \ifdim\wd\z@=10pt
3500   \global\let\bxjs@scale\jsScale
3501 \else
3502 % (10*s)/(10*f) として計算、\bxjs@invscale は BXJS で定義
3503   \edef\bxjs@tmpa{\strip@pt\wd\z@}
3504   \@tempdima=10pt \@tempdima=\jsScale\@tempdima
3505   \bxjs@invscale\@tempdima\bxjs@tmpa
3506   \xdef\bxjs@scale{\strip@pt\@tempdima}
3507 \fi
3508 \endgroup
3509 %\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。

```

3510 \bxjs@declarefontshape

    フォント代替の明示的定義。

3511 \DeclareFontShape{\jsc@JYn}{mc}{m}{it}{<->ssub*mc/m/n}{%

```

```

3512 \DeclareFontShape{\jsc@JYn}{mc}{m}{sl}{<->ssub*mc/m/n}{f}
3513 \DeclareFontShape{\jsc@JYn}{mc}{m}{sc}{<->ssub*mc/m/n}{f}
3514 \DeclareFontShape{\jsc@JYn}{gt}{m}{it}{<->ssub*gt/m/n}{f}
3515 \DeclareFontShape{\jsc@JYn}{gt}{m}{sl}{<->ssub*gt/m/n}{f}
3516 \DeclareFontShape{\jsc@JYn}{mc}{bx}{it}{<->ssub*gt/m/n}{f}
3517 \DeclareFontShape{\jsc@JYn}{mc}{bx}{sl}{<->ssub*gt/m/n}{f}
3518 \DeclareFontShape{\jsc@JYn}{gt}{bx}{it}{<->ssub*gt/m/n}{f}
3519 \DeclareFontShape{\jsc@JYn}{gt}{bx}{sl}{<->ssub*gt/m/n}{f}
3520 \DeclareFontShape{\jsc@JYn}{mc}{b}{n}{<->ssub*mc/bx/n}{f}
3521 \DeclareFontShape{\jsc@JYn}{mc}{b}{it}{<->ssub*mc/bx/n}{f}
3522 \DeclareFontShape{\jsc@JYn}{mc}{b}{sl}{<->ssub*mc/bx/n}{f}
3523 \DeclareFontShape{\jsc@JYn}{gt}{b}{n}{<->ssub*gt/bx/n}{f}
3524 \DeclareFontShape{\jsc@JYn}{gt}{b}{it}{<->ssub*gt/bx/n}{f}
3525 \DeclareFontShape{\jsc@JYn}{gt}{b}{sl}{<->ssub*gt/bx/n}{f}
3526 \DeclareFontShape{\jsc@JTn}{mc}{m}{it}{<->ssub*mc/m/n}{f}
3527 \DeclareFontShape{\jsc@JTn}{mc}{m}{sl}{<->ssub*mc/m/n}{f}
3528 \DeclareFontShape{\jsc@JTn}{mc}{m}{sc}{<->ssub*mc/m/n}{f}
3529 \DeclareFontShape{\jsc@JTn}{gt}{m}{it}{<->ssub*gt/m/n}{f}
3530 \DeclareFontShape{\jsc@JTn}{gt}{m}{sl}{<->ssub*gt/m/n}{f}
3531 \DeclareFontShape{\jsc@JTn}{mc}{bx}{it}{<->ssub*gt/m/n}{f}
3532 \DeclareFontShape{\jsc@JTn}{mc}{bx}{sl}{<->ssub*gt/m/n}{f}
3533 \DeclareFontShape{\jsc@JTn}{gt}{bx}{it}{<->ssub*gt/m/n}{f}
3534 \DeclareFontShape{\jsc@JTn}{gt}{bx}{sl}{<->ssub*gt/m/n}{f}
3535 \DeclareFontShape{\jsc@JTn}{mc}{b}{n}{<->ssub*mc/bx/n}{f}
3536 \DeclareFontShape{\jsc@JTn}{mc}{b}{it}{<->ssub*mc/bx/n}{f}
3537 \DeclareFontShape{\jsc@JTn}{mc}{b}{sl}{<->ssub*mc/bx/n}{f}
3538 \DeclareFontShape{\jsc@JTn}{gt}{b}{n}{<->ssub*gt/bx/n}{f}
3539 \DeclareFontShape{\jsc@JTn}{gt}{b}{it}{<->ssub*gt/bx/n}{f}
3540 \DeclareFontShape{\jsc@JTn}{gt}{b}{sl}{<->ssub*gt/bx/n}{f}

```

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。

```

3541 \@ifl@t@r\fmtversion{2020/10/01}
3542   {\jsc@needspace@tchfalse}{\jsc@needspace@tchtrue}
3543   \ifjsc@needspace@tch          % --- for 2020-02-02 or older BEGIN
3544   \ifx\@rmfamilyhook\@undefined % old
3545   \DeclareRobustCommand\rmfamily
3546     {\not@math@alphabet\rmfamily\mathrm
3547     \romanfamily\rmdefault\kanjifamily\mcdefault\selectfont}
3548   \DeclareRobustCommand\sffamily
3549     {\not@math@alphabet\sffamily\mathsf
3550     \romanfamily\sfddefault\kanjifamily\gtdefault\selectfont}
3551   \DeclareRobustCommand\ttfamily
3552     {\not@math@alphabet\ttfamily\mathtt
3553     \romanfamily\ttddefault\kanjifamily\gtdefault\selectfont}
3554   \g@addto@macro\bxjs@begin@document@hook{%
3555     \ifx\mweights@init\@undefined\else % mweights.sty is loaded
3556     % my definitions above should have been overwritten, recover it!

```

```

3557 % \selectfont is executed twice but I don't care about speed...
3558 \expandafter\g@addto@macro\csname rmfamily \endcsname
3559 {\kanjifamily\mcdefault\selectfont}%
3560 \expandafter\g@addto@macro\csname sffamily \endcsname
3561 {\kanjifamily\gtdefault\selectfont}%
3562 \expandafter\g@addto@macro\csname ttfamily \endcsname
3563 {\kanjifamily\gtdefault\selectfont}%
3564 \fi}
3565 \else % 2020-02-02
3566 \g@addto@macro\@rmfamilyhook
3567 {\prepare@family@series@update@kanji{mc}\mcdefault}
3568 \g@addto@macro\@sffamilyhook
3569 {\prepare@family@series@update@kanji{gt}\gtdefault}
3570 \g@addto@macro\@ttfamilyhook
3571 {\prepare@family@series@update@kanji{gt}\gtdefault}
3572 \fi
3573 \else % --- for 2020-02-02 or older END & for 2020-10-01 BEGIN
3574 \AddToHook{rmfamily}%
3575 {\prepare@family@series@update@kanji{mc}\mcdefault}
3576 \AddToHook{sffamily}%
3577 {\prepare@family@series@update@kanji{gt}\gtdefault}
3578 \AddToHook{ttfamily}%
3579 {\prepare@family@series@update@kanji{gt}\gtdefault}
3580 \fi % --- for 2020-10-01 END
3581 \ifx\DeclareFixJFMCJKTextFontCommand\undefined
3582 \DeclareJaTextFontCommand{\textmc}{\mcfamily}
3583 \DeclareJaTextFontCommand{\textgt}{\gtfamily}
3584 \fi
3585 \bxjs@if@sf@default{%
3586 \renewcommand\kanjifamilydefault{\gtdefault}}

```

念のため。

```

3587 \selectfont

```

これ以降では、`\bxjs@parse@qh` の処理は p_TE_X 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）。

```

3588 \def\bxjs@parse@qh#1{\let\bxjs@tmpb\relax}
3589 \let\bxjs@parse@qh@a\undefined
3590 \let\bxjs@parse@qh@b\undefined

```

■パラメタの設定

```

3591 \prebreakpenalty\jis"2147=10000
3592 \postbreakpenalty\jis"2148=10000
3593 \prebreakpenalty\jis"2149=10000
3594 \inhibitxspcode`!=1
3595 \inhibitxspcode`〒=2
3596 \xspcode`+=3
3597 \xspcode`%=3

```

"80～"FF の範囲の \spcode を 3 に変更。

```
3598 \@tempcnta="80 \@whilenum\@tempcnta<"100 \do{%
3599   \xspcode\@tempcnta=3\advance\@tempcnta\@ne}
```

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

```
3600 \let\jsInhibitGlueAtParTop\@inhibitglue
```

\jsResetDimen は空のままでよい。

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

```
3601 \begingroup
3602 \catcode`\!=0
```

\bxjs@ptex@dir 現在の組方向：t=縦、y=横、?=その他。

```
3603 \gdef\bxjs@ptex@dir{%
3604   !iftdir t%
3605   !else!ifydir y%
3606   !else ?%
3607   !fi!fi}
```

新版の pTeX で脚注番号の周囲の空きが過大になる現象への対処。

※現在の pLaTeX カーネルでは対処が既に行われている。ここでは、\@makefnmark の定義が古いものであった場合に、新しいものに置き換える。

```
3608 % 古い \@makefnmark の定義
3609 \long\def\bxjs@tmpa{\hbox{%
3610   !ifydir \@textsuperscript{\normalfont\@thefnmark}%
3611   !else\hbox{\yoko\@textsuperscript{\normalfont\@thefnmark}}!fi}}
3612 \ifx\@makefnmark\bxjs@tmpa
3613 \long\gdef\@makefnmark{%
3614   !ifydir \hbox{\hbox{\@textsuperscript{\normalfont\@thefnmark}}\hbox{}}%
3615   !else\hbox{\yoko\@textsuperscript{\normalfont\@thefnmark}}!fi}
3616 \fi
```

エスケープ文字の変更はここまで。

```
3617 \endgroup
```

■minijs パッケージのブロック やっておく。

```
3618 \@namedef{ver@minijs.sty}{}
```

B.3 pdfTeX 用の処理

```
3619 \else\ifx p\jsEngine
```

\jsLetHeadChar を UTF-8 バイト列に対応させる。

```
3620 \let\bxjs@let@hchar@chr\bxjs@let@hchar@chr@ue
```

ムニャムニャ。

```

3621 \@onlypreamble\bxjs@CJK@loaded
3622 \def\bxjs@CJK@loaded{%
3623   \def\@footnotemark{%
3624     \leavevmode
3625     \ifhmode
3626       \edef\x@sf{\the\spacefactor}%
3627       \ifdim\lastkern>\z@\ifdim\lastkern<5sp\relax
3628         \unkern\unkern
3629         \ifdim\lastskip>\z@ \unskip \fi
3630       \fi\fi
3631       \nobreak
3632     \fi
3633     \@makefnmark
3634     \ifhmode \spacefactor\x@sf \fi
3635   \relax}%
3636 \let\bxjs@CJK@loaded\relax
3637 }
3638 \g@addto@macro\bxjs@begin@document@hook{%
3639   \@ifpackageloaded{CJK}{%
3640     \bxjs@CJK@loaded
3641   }{}%
3642 }

```

B.4 X₃TeX 用の処理

```

3643 \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。

```

3644 \def\bxjs@let@hchar@chr#1{%
3645   \@tempcnta`#1\relax \divide\@tempcnta"800\relax
3646   \bxjs@cond\ifnum\@tempcnta=27 \fi{%
3647     \bxjs@let@hchar@chr@xe
3648   }\{\bxjs@let@hchar@out\def{{#1}}}}
3649 \def\bxjs@let@hchar@chr@xe#1{%
3650   \lccode`0=`#1\relax
3651   \lowercase{\bxjs@let@hchar@out\def{{0}}}}

```

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

```

3652 \@onlypreamble\bxjs@do@precisetext
3653 \ifx\XeTeXgenerateactualtext\@undefined\else
3654   \def\bxjs@do@precisetext{%
3655     \XeTeXgenerateactualtext=\@ne}
3656 \fi

```

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

TODO: バージョン要件を見直して暫定措置を解除する。

```

3657 \@onlypreamble\bxjs@do@simplejasetup
3658 \def\bxjs@do@simplejasetup{%
3659   \@namedef{bxjs@zeroglue/0.0pt}{T}%

```



```

3660 \ifnum\XeTeXinterchartokenstate>\z@
3661 \else\expandafter\ifx\csname bxjs@zeroglue/\the\XeTeXlinebreakskip\endcsname\relax\else
3662 \jsSimpleJaSetup
3663 \ClassInfo\bxjs@clsname
3664 {\string\jsSimpleJaSetup' is applied@gobble}%
3665 \fi\fi}

```

`\jsSimpleJaSetup` 日本語出力用の超簡易的な設定。

```

3666 \newcommand*{\jsSimpleJaSetup}{%
3667 \XeTeXlinebreaklocale "ja"\relax
3668 \XeTeXlinebreakskip=0pt plus 1pt minus 0.1pt\relax
3669 \XeTeXlinebreakpenalty=0\relax}

```

B.5 後処理 (エンジン共通)

```

3670 \fi\fi\fi

simplejasetup オプションの処理。
3671 \ifx\bxjs@do@simplejasetup\@undefined\else
3672 \g@addto@macro\bxjs@begin@document@hook{%
3673 \ifbxjs@simplejasetup
3674 \bxjs@do@simplejasetup
3675 \fi}
3676 \fi

precisetext オプションの処理。
3677 \ifbxjs@precisetext
3678 \ifx\bxjs@do@precisetext\@undefined
3679 \ClassWarning\bxjs@clsname
3680 {The current engine does not support the\MessageBreak
3681 'precise-text' option@gobble}
3682 \else
3683 \bxjs@do@precisetext
3684 \fi
3685 \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。

```

3686 \g@addto@macro\bxjs@begin@document@hook{%
3687 \ifx\jsInhibitGlueAtParTop\@empty
3688 \def\bxjs@tmpa{\jsInhibitGlueAtParTop}%
3689 \ifx\everyparhook\bxjs@tmpa
3690 \let\everyparhook\@empty
3691 \fi
3692 \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。

```

3693 \ifnum\bxjs@everyparhook=\bxjs@everyparhook@@modern

```

まず `\everypar` を“乗っ取る”処理を行う。

```
3694 \let\bxjs@everypar\everypar
3695 \newtoks\everypar
3696 \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。

```
3697 \bxjs@everypar{\the\expandafter\everypar\everyparhook}%
3698 \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。

```
3699 \ifbxjs@fancyhdr
```

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

```
3700 \@onlypreamble\bxjs@adjust@fancyhdr
3701 \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。

```
3702 \def\bxjs@tmpa{\fancyplain{}{\sl\rightmark}\strut}%
3703 \def\bxjs@tmpb{\fancyplain{}{\rightmark}\strut}%
3704 \ifx\f@ncyelh\bxjs@tmpa \global\let\f@ncyelh\bxjs@tmpb \fi
3705 \ifx\f@ncyerh\bxjs@tmpa \global\let\f@ncyerh\bxjs@tmpb \fi
3706 \ifx\f@ncyolh\bxjs@tmpa \global\let\f@ncyolh\bxjs@tmpb \fi
3707 \ifx\f@ncyorh\bxjs@tmpa \global\let\f@ncyorh\bxjs@tmpb \fi
3708 \def\bxjs@tmpa{\fancyplain{}{\sl\leftmark}\strut}%
3709 \def\bxjs@tmpb{\fancyplain{}{\leftmark}\strut}%
3710 \ifx\f@ncyelh\bxjs@tmpa \global\let\f@ncyelh\bxjs@tmpb \fi
3711 \ifx\f@ncyerh\bxjs@tmpa \global\let\f@ncyerh\bxjs@tmpb \fi
3712 \ifx\f@ncyolh\bxjs@tmpa \global\let\f@ncyolh\bxjs@tmpb \fi
3713 \ifx\f@ncyorh\bxjs@tmpa \global\let\f@ncyorh\bxjs@tmpb \fi
3714 \def\bxjs@tmpa{\rm\thepage\strut}%
3715 \def\bxjs@tmpb{\thepage\strut}%
3716 \ifx\f@ncyecf\bxjs@tmpa \global\let\f@ncyecf\bxjs@tmpb \fi
3717 \ifx\f@ncyocf\bxjs@tmpa \global\let\f@ncyocf\bxjs@tmpb \fi
```

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

```
3718 \ifx\fullwidth\undefined\else \ifdim\textwidth<\fullwidth
3719 \setlength{\@tempdima}{\fullwidth-\textwidth}%
3720 \edef\bxjs@tmpa{\noexpand\fancyhfoffset[EL,OR]{\the\@tempdima}%
3721 }\bxjs@tmpa
3722 \fi\fi
```

```

3723 \PackageInfo\bxjs@clsname
3724 {Patch to fancyhdr is applied\@gobble}}

```

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

```

3725 \def\bxjs@pagestyle@hook{%
3726   \@ifpackageloaded{fancyhdr}{%
3727     \bxjs@adjust@fancyhdr
3728     \global\let\bxjs@adjust@fancyhdr\relax
3729   }{}}

```

`\pagestyle` にフックを入れ込む。

```

3730 \let\bxjs@org@pagestyle\pagestyle
3731 \def\pagestyle{%
3732   \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。

```

3733 \g@addto@macro\bxjs@begin@document@hook{%
3734   \bxjs@pagestyle@hook
3735   \global\let\bxjs@pagestyle@hook\relax}

```

終わり。

```

3736 \fi

```

■和文空白命令

```

3737 \ifbxjs@jaspace@cmd

```

`\jaenspace` 半角幅の水平空き。

```

3738 \def\jaenspace{\hskip.5\jsZw\relax}

```

`\jathinspace` 和欧文間空白を入れるユーザ命令。

※ `minimal` ではダミー定義。

```

3739 \def\jathinspace{\hskip\z@skip}

```

`_` 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。

```

3740 \def\_ {\zwspace}

```

`\jaspace` `jlreq` クラスと互換の命令。

```

3741 \DeclareRobustCommand*{\jaspace}[1]{%
3742   \expandafter\ifx\csname bxjs@jaspace@@#1\endcsname\relax
3743   \ClassError\bxjs@clsname
3744     {Unknown jaspaces: #1}{\@eha}%
3745   \else
3746     \csname bxjs@jaspace@@#1\endcsname
3747   \fi}
3748 \def\bxjs@jaspace@@zenkaku{\hskip 1\jsZw\relax}
3749 \def\bxjs@jaspace@@nibu{\hskip .5\jsZw\relax}
3750 \def\bxjs@jaspace@@shibu{\hskip .25\jsZw\relax}

```

終わり。

```
3751 \fi
```

以上で終わり。

```
3752 %</minimal>
```

付録 C 和文ドライバ：standard 🍷

標準のドライバ。

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

■和文フォント指定の扱い

C.1 準備

まず minimal ドライバを読み込む。

```
3753 %<*standard>
```

```
3754 %% このファイルは日本語文字を含みます
```

```
3755 \input{bxjsja-minimal.def}
```

simplejasetup は standard では無効になる。

```
3756 \bxjs@simplejasetupfalse
```

C.2 和文ドライバパラメタ

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

```
\ifbxjs@jp@jismmiv 2004JIS 字形を優先させるか。
```

```
3757 \newif\ifbxjs@jp@jismmiv
```

jis2004 オプションの処理。

```
3758 \bxjs@cslet{bxjs@kv@jis2004@true}\bxjs@jp@jismmivtrue
```

```
3759 \bxjs@cslet{bxjs@kv@jis2004@false}\bxjs@jp@jismmivfalse
```

```
3760 \define@key{bxjsStd}{jis2004}[true]{%
```

```
3761 \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。

```
3762 \newif\ifbxjs@jp@units
```

`units` オプションの処理。

```
3763 \let\bxjs@kv@units@true\bxjs@jp@unitstrue
```

```
3764 \let\bxjs@kv@units@false\bxjs@jp@unitsfalse
```

```
3765 \define@key{bxjsStd}{units}[true]{%
```

```
3766 \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。

```
3767 \let\bxjs@jp@font\@empty
```

`font` オプションの処理。

```
3768 \define@key{bxjsStd}{font}{%
```

```
3769 \edef\bxjs@jp@font{#1}}
```

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

```
3770 \newif\ifbxjs@jp@strong@cmd \bxjs@jp@strong@cmdtrue
```

`strong-cmd` オプションの処理。

```
3771 \let\bxjs@kv@strongcmd@true\bxjs@jp@strong@cmdtrue
```

```
3772 \let\bxjs@kv@strongcmd@false\bxjs@jp@strong@cmdfalse
```

```
3773 \define@key{bxjs}{strong-cmd}[true]{\bxjs@set@keyval{strongcmd}{#1}{}}
```

実際の `japaram` の値を適用する。

```
3774 \def\bxjs@next#1{\bxjs@safe@setkeys{bxjsStd}{#1}}
```

```
3775 \expandafter\bxjs@next\expandafter{\jsJaParam}
```

C.3 共通処理 (1)

■**jis2004 パラメタ** `jis2004` パラメタが有効の場合は、グローバルオプションに `jis2004` を追加する。

※`otf` や `luatexja-preset` 等のパッケージがこのオプションを利用する。

```
3776 \@onlypreamble\bxjs@apply@mmiv
```

```
3777 \def\bxjs@apply@mmiv{%
```

```
3778 \g@addto@macro\@classoptionslist{,jis2004}
```

```
3779 % \ifpackagewith 判定への対策
```

```
3780 \PassOptionsToPackage{jis2004}{otf}
```

```
3781 \global\let\bxjs@apply@mmiv\relax}
```

```
3782 \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。

```
3783 \if j\jsEngine
```

```
3784 \bxjs@jp@unitsfalse
```

```
3785 \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。

```

3786 \ifbxjs@jp@units
3787   \IfFileExists{bxcalc.sty}{%
3788     \RequirePackage{bxcalc}[2018/01/28]%v1.0a
3789     \ifx\usepTeXunits\@undefined
3790       \PackageWarningNoLine{bxjs@clsname}
3791         {Cannot support pTeX units (zw etc.), since\MessageBreak
3792           the package 'bxcalc' is too old}%
3793       \bxjs@jp@unitsfalse
3794     \else \usepTeXunits
3795     \fi
3796   }{%else
3797     \PackageWarningNoLine{bxjs@clsname}
3798       {Cannot support pTeX units (zw etc.), since\MessageBreak
3799         the package 'bxcalc' is unavailable}%
3800     \bxjs@jp@unitsfalse
3801   }
3802 \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。

```

3803 \ifbxjs@jp@units
3804 \def\bxjs@parse@qh#1{\let\bxjs@tmpb\relax}
3805 \let\bxjs@parse@qh@a\@undefined
3806 \let\bxjs@parse@qh@b\@undefined
3807 \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。

```

3808 \ifbxjs@jp@units
3809   \def\bxjs@let@lenexpr#1#2{%
3810     \edef#1{#2}%
3811     \expandafter\CUXParseExpr\expandafter#1\expandafter{#1}}
3812 \else
3813   \def\bxjs@let@lenexpr{\edef}
3814 \fi

```

■\strong 命令の補填

\strong 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 strongenv する。

※既に利用可能である場合は何もしない。

```

3815 \ifbxjs@jp@strong@cmd\jsAtEndOfClass{%
3816   \ifx\strong\@undefined\ifx\strongenv\@undefined
3817     \DeclareRobustCommand{\strongenv}{\bxjs@strong@font}%
3818     \DeclareTextFontCommand{\strong}{\strongenv}%

```

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。

```

3819 \newcommand*{\strongfontdeclare}{\bxjs@strongfontdeclare}%
3820 \newcount\bxjs@strong@level
3821 \bxjs@protected\def\bxjs@strongfontdeclare#1{%
3822   \bxjs@set@array@from@clist{bxjs@strong}{#1}%
3823   \bxjs@strong@level\z@}%
3824 \bxjs@strongfontdeclare{\bfseries}%
3825 \def\bxjs@strong@font{%
3826   \bxjs@csletcs{bxjs@tmpa}{bxjs@strong/\the\bxjs@strong@level}%
3827   \ifx\bxjs@tmpa\relax
3828     \advance\bxjs@strong@level\m@ne \bxjs@strong@font
3829   \else \advance\bxjs@strong@level\@ne \bxjs@tmpa
3830   \fi}%
3831 \fi\fi
3832 }\fi

```

■共通命令の実装 `\jQ` 等の「単位」系の共通命令を実装する。まず ε -TeX 拡張が使えるか検査する。

```
3833 \ifjsWitheTeX
```

使える場合は、「`\dimexpr` 外部寸法表記`\relax`」の形式（これは内部値なので単位として使える）で各命令定義する。

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

```

\jH 3834 \@tempdima=0.25mm
3835 \protected\edef\jQ{\dimexpr\the\@tempdima\relax}
3836 \let\jH\jQ

```

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

```

\trueH 3837 \ifjsc@mag
3838   \@tempdimb=\jsBaseFontSize\relax
3839   \edef\bxjs@tmpa{\strip@pt\@tempdimb}%
3840   \@tempdima=2.5mm
3841   \bxjs@invscale\@tempdima\bxjs@tmpa
3842   \protected\edef\trueQ{\dimexpr\the\@tempdima\relax}
3843   \@tempdima=10pt
3844   \bxjs@invscale\@tempdima\bxjs@tmpa
3845   \protected\edef\bxjs@truept{\dimexpr\the\@tempdima\relax}
3846 \else \let\trueQ\jQ \let\bxjs@truept\p@
3847 \fi
3848 \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。

```

3849 \@tempdima\trueQ \bxjs@invscale\@tempdima\jsScale
3850 \protected\edef\ascQ{\dimexpr\the\@tempdima\relax}
3851 \@tempdima\bxjs@truept \bxjs@invscale\@tempdima\jsScale

```

```

3852 \protected\edef\ascpt{\dimexpr\the\@tempdima\relax}
3853 \fi

```

`\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。

```

3854 \def\jafontsize#1#2{%
3855   \begingroup
3856     \bxjs@jafontsize@a{#1}%
3857     \@tempdimb\jsInverseScale\@tempdima
3858     \bxjs@jafontsize@a{#2}%
3859     \xdef\bxjs@g@tmpa{%
3860       \noexpand\fontsize{\the\@tempdimb}{\the\@tempdima}}%
3861   \endgroup\bxjs@g@tmpa}
3862 \def\bxjs@jafontsize@a#1{%
3863   \bxjs@parse@qh{#1}%
3864   \ifx\bxjs@tmpb\relax \def\bxjs@tmpb{#1}\fi
3865   \@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。(エンジン依存のコード。)

`\bxjs@kanjiskip` 和文間空白の量を表すテキスト。

```

3866 \def\bxjs@kanjiskip{0pt}

```

`\setkanjiskip` 和文間空白の量を設定する。

```

3867 \newcommand*\setkanjiskip[1]{%
3868   \bxjs@let@lenexpr\bxjs@kanjiskip{#1}%
3869   \bxjs@reset@kanjiskip}

```

`\getkanjiskip` 和文間空白の量を表すテキストに展開する。

```

3870 \newcommand*\getkanjiskip{%
3871   \bxjs@kanjiskip}

```

`\ifbxjs@kanjiskip@enabled` 和文間空白の挿入が有効か。ただし pTeX では自身の `\(no)autospacing` での制御を用いるのでこの変数は常に真とする。

```

3872 \newif\ifbxjs@kanjiskip@enabled \bxjs@kanjiskip@enabledtrue

```

`\bxjs@enable@kanjiskip` 和文間空白の挿入を有効／無効にする。(pTeX 以外)

```

\bxjs@disable@kanjiskip 3873 \bxjs@robust@def\bxjs@enable@kanjiskip{%
3874   \bxjs@kanjiskip@enabledtrue
3875   \bxjs@reset@kanjiskip}
3876 \bxjs@robust@def\bxjs@disable@kanjiskip{%
3877   \bxjs@kanjiskip@enabledfalse
3878   \bxjs@reset@kanjiskip}

```

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

```

3879 \bxjs@robust@def\bxjs@reset@kanjiskip{%
3880   \ifbxjs@kanjiskip@enabled
3881     \setlength{\@tempskipa}{\bxjs@kanjiskip}%

```



```

3882 \else \@tempskipa\z@
3883 \fi
3884 \bxjs@apply@kanjiskip}

```

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

```

\setxkanjiskip 3885 \def\bxjs@xkanjiskip{Opt}
\getxkanjiskip 3886 \newcommand*\setxkanjiskip[1]{%
\ifbxjs@xkanjiskip@enabled 3887 \bxjs@let@lenexpr\bxjs@xkanjiskip{#1}%
3888 \bxjs@reset@xkanjiskip}
\bxjs@enable@xkanjiskip 3889 \newcommand*\getxkanjiskip{%
3890 \bxjs@xkanjiskip}
\bxjs@disable@xkanjiskip 3891 \newif\ifbxjs@xkanjiskip@enabled \bxjs@xkanjiskip@enabledtrue
\bxjs@reset@xkanjiskip 3892 \bxjs@robust@def\bxjs@enable@xkanjiskip{%
3893 \bxjs@xkanjiskip@enabledtrue
3894 \bxjs@reset@xkanjiskip}
3895 \bxjs@robust@def\bxjs@disable@xkanjiskip{%
3896 \bxjs@xkanjiskip@enabledfalse
3897 \bxjs@reset@xkanjiskip}
3898 \bxjs@robust@def\bxjs@reset@xkanjiskip{%
3899 \ifbxjs@xkanjiskip@enabled
3900 \setlength{\@tempskipa}{\bxjs@xkanjiskip}%
3901 \else \@tempskipa\z@
3902 \fi
3903 \bxjs@apply@xkanjiskip}

```

`\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。

```

3904 \g@addto@macro\jsResetDimen{%
3905 \bxjs@reset@kanjiskip
3906 \bxjs@reset@xkanjiskip}
3907 \let\bxjs@apply@kanjiskip\relax
3908 \let\bxjs@apply@xkanjiskip\relax

```

■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、 $\mathrm{T}_{\mathrm{E}}\mathrm{X}$ Live の `kanji-config-updmap` コマンドで使う“ファミリー”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ーを表す。

`\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。

```

3909 \@onlypreamble\bxjs@adjust@jafont
3910 \def\bxjs@adjust@jafont#1{%
3911 \ifx\jsJaFont\bxjs@auto
3912 \bxjs@get@kanjiEmbed
3913 \ifx\bxjs@jaEmbed\relax
3914 \let\bxjs@tmpa\@empty
3915 \else

```

```

3916 \let\bxjs@tmpa\bxjs@jaEmbed
3917 \ifx\bxjs@jaVariant\bxjs@zhziv
3918 \bxjs@apply@mmiv
3919 \fi
3920 \fi
3921 \else
3922 \let\bxjs@tmpa\jsJaFont
3923 \fi
3924 \if f#1\ifx\bxjs@tmpa\bxjs@noEmbed
3925 \ClassWarningNoLine\bxjs@clsname
3926 {Option 'jafont=noEmbed' is ignored, because it is\MessageBreak
3927 not available on the current situation}%
3928 \let\bxjs@tmpa@empty
3929 \fi\fi
3930 }
3931 \def\bxjs@@auto{auto}
3932 \def\bxjs@noEmbed{noEmbed}
3933 \def\bxjs@zhziv{-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。

```

3934 \let\bxjs@jaEmbed\relax
3935 \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。

```

3936 \@onlypreamble\bxjs@get@kanjiEmbed
3937 \def\bxjs@get@kanjiEmbed{%
3938 \begingroup\setbox\z@=\hbox{%
3939 \global\let\bxjs@tmpdo@empty
3940 \def\bxjs@next##1##2##3{%
3941 \def##1###1##3 ###2\@nil###3\@nnil{%
3942 \ifx$###1$\gdef##2{###2}\fi}%
3943 \g@addto@macro\bxjs@tmpdo{%
3944 \expandafter##1\bxjs@tmpa\@nil##3 \@nil\@nnil}}%
3945 \bxjs@next\bxjs@tmpdo@a\bxjs@g@tmpa{kanjiEmbed}%
3946 \bxjs@next\bxjs@tmpdo@b\bxjs@g@tmpa{jaEmbed}%
3947 \bxjs@next\bxjs@tmpdo@c\bxjs@g@tmpb{kanjiVariant}%
3948 \bxjs@next\bxjs@tmpdo@d\bxjs@g@tmpb{jaVariant}%
3949 %
3950 \global\let\bxjs@g@tmpa\relax
3951 \global\let\bxjs@g@tmpb\relax
3952 \endlinechar\m@ne
3953 \let\do\@makeother\dospecials
3954 \catcode32=10 \catcode12=10 %form-feed
3955 \let\bxjs@tmpa@empty
3956 \openin\@inputcheck="|kpsewhich updmap.cfg"\relax
3957 \ifeof\@inputcheck\else
3958 \read\@inputcheck to\bxjs@tmpa

```

```

3959     \closein\@inputcheck
3960     \fi
3961     \ifx\bxjs@tmpa\@empty\else
3962         \openin\@inputcheck="\bxjs@tmpa"\relax
3963         \@tempwattrue
3964         \loop\if@tempwa
3965             \read\@inputcheck to\bxjs@tmpa
3966             \bxjs@tmpdo
3967             \ifeof\@inputcheck \@tempwafalse \fi
3968         \repeat
3969     \fi
3970 } \endgroup
3971 \let\bxjs@jaEmbed\bxjs@g@tmpa
3972 \let\bxjs@jaVariant\bxjs@g@tmpb
3973 }

```

`\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。

```

3974 \@onlypreamble\bxjs@resolve@jafont@paren
3975 \def\bxjs@resolve@jafont@paren#1{%
3976     \def\bxjs@tmpb{\let#1}%
3977     \expandafter\bxjs@resolve@jafont@paren@a#1\@nil()\@nil\@nnil#1}
3978 \@onlypreamble\bxjs@resolve@jafont@paren@a
3979 \def\bxjs@resolve@jafont@paren@a#1(#2)#3\@nil#4\@nnil#5{%
3980     \ifx\relax#4\relax \bxjs@tmpb#5%
3981     \else
3982         \edef\bxjs@tmpa{#1\bxjs@jafont@paren{#2}#3}%
3983         \bxjs@tmpb\bxjs@tmpa
3984     \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。

```

3985 \newcommand*\jachar[1]{%
3986     \begingroup
3987         \jsLetHeadChar\bxjs@tmpa{#1}%
3988         \ifx\bxjs@tmpa\relax
3989             \ClassWarningNoLine\bxjs@clsname
3990                 {Illegal argument given to \string\jachar}%
3991         \else
3992             \expandafter\bxjs@jachar\expandafter{\bxjs@tmpa}%
3993         \fi
3994     \endgroup}

```

`\jsJaChar` を `\jachar` と等価にする。

```

3995 \let\jsJaChar\jachar

```

下請けの `\bxjs@jachar` の実装はエンジンにより異なる。

```
3996 \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。

```
3997 \PassOptionsToPackage{setpagesize=false}{hyperref}
```

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

```
3998 \@onlypreamble\bxjs@fix@hyperref@unicode
3999 \def\bxjs@fix@hyperref@unicode#1{%
4000   \PassOptionsToPackage{bxjs/hook=#1}{hyperref}%
4001   \@namedef{KV@Hyp@bxjs/hook}##1{%
4002     \KV@Hyp@unicode{##1}%
4003     \def\KV@Hyp@unicode###1{%
4004       \expandafter\ifx\csname if##1\expandafter\endcsname
4005         \csname if###1\endcsname\else
4006         \ClassWarningNoLine\bxjs@clsname
4007           {Blcoked hyperref option 'unicode=####1'}%
4008       \fi
4009     }%
4010   }%
4011 }
```

`\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。

```
4012 \@onlypreamble\jsCheckHyperrefUnicode
4013 \let\jsCheckHyperrefUnicode\@empty
4014 \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。

```
4015 \@onlypreamble\bxjs@check@hyperref@unicode
4016 \def\bxjs@check@hyperref@unicode#1{%
4017   \g@addto@macro\jsCheckHyperrefUnicode{%
4018     \@tempwafalse
4019     \begingroup
4020       \expandafter\ifx\csname ifHy@unicode\endcsname\relax
4021         \aftergroup\@tempwattrue \fi
4022       \expandafter\ifx\csname ifHy@unicode\expandafter\endcsname
4023         \csname if#1\endcsname
4024       \aftergroup\@tempwattrue \fi
4025     \endgroup
4026     \if@tempswa\else
4027       \ClassError\bxjs@clsname
4028         {The value of hyperref 'unicode' key is not suitable\MessageBreak
4029         for the present engine (must be #1)}%
4030       {\@ehc}%
4031     \fi}}
```

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

```

4032 \@onlypreamble\bxjs@urgent@special
4033 \@onlypreamble\bxjs@urgent@special@a

```

LaTeX カーネルの新フック管理が導入済かを調べる。未導入の古い版である場合。

```

4034 \ifbxjs@old@hook@system
4035 \def\bxjs@urgent@special#1{%
4036   \AtBeginDvi{\special{#1}}%
4037   \g@addto@macro\bxjs@begin@document@hook{%
4038     \@ifpackageloaded{atbegshi}{%
4039       \begingroup
4040         \toks\z@{\special{#1}}%
4041         \toks\tw@\expandafter{\AtBegShi@HookFirst}%
4042         \xdef\AtBegShi@HookFirst{\the\toks@\the\toks\tw@}%
4043       \endgroup
4044     }{}%
4045   }%
4046 }

```

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。

```

4047 \else
4048   \def\bxjs@urgent@special#1{%
4049     \bxjs@urgent@special@a
4050     \AddToHook{shipout/firstpage}[pxjahyper/enc]{\special{#1}}
4051   \def\bxjs@urgent@special@a{%
4052     \DeclareHookRule{shipout/firstpage}{pxjahyper/enc}{<}{hyperref}%
4053     \global\let\bxjs@urgent@special@a\relax
4054 \fi

```

C.4 pTeX 用設定

```

4055 \if j\jsEngine

```

■共通命令の実装

```

4056 \def\bxjs@apply@kanjiskip{%
4057   \kanjiskip\@tempskipa}
4058 \def\bxjs@apply@xkanjiskip{%
4059   \xkanjiskip\@tempskipa}

```

\jaJaChar のサブマクロ。

```

4060 \def\bxjs@jachar#1{%
4061   \bxjs@jachar@a#1...\@nil}
4062 \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。

```

4063   \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。

```

4064 \else\ifx.#3%
4065 \@tempcnta`#1 \multiply\@tempcnta64
4066 \advance\@tempcnta`#2 \advance\@tempcnta-"3080
4067 \bxjs@jachar@b
4068 \else\ifx.#4%
4069 \@tempcnta`#1 \multiply\@tempcnta64
4070 \advance\@tempcnta`#2 \multiply\@tempcnta64
4071 \advance\@tempcnta`#3 \advance\@tempcnta-"E2080
4072 \bxjs@jachar@b
4073 \else
4074 \@tempcnta`#1 \multiply\@tempcnta64
4075 \advance\@tempcnta`#2 \multiply\@tempcnta64
4076 \advance\@tempcnta`#3 \multiply\@tempcnta64
4077 \advance\@tempcnta`#4 \advance\@tempcnta-"3C82080
4078 \bxjs@jachar@b
4079 \fi\fi\fi}

```

符号値が \@tempcnta の和文文字を出力する処理。

```

4080 \ifjsWithupTeX
4081 \def\bxjs@jachar@b{\kchar\@tempcnta}
4082 \else
4083 \def\bxjs@jachar@b{%
4084 \ifx\bxUInt\@undefined\else
4085 \bxUInt{\@tempcnta}%
4086 \fi}
4087 \fi

```

和欧文間空白の命令 \jathinspace の実装。

```

4088 \ifbxjs@jaspace@cmd
4089 \def\jathinspace{\hskip\xkanjiskip}
4090 \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。

```

4091 \ifbxjs@jp@jismmiv
4092 \PassOptionsToPackage{prefer2004jis}{pxchfon}
4093 \PassOptionsToPackage{prefer2004jis}{pxbabel}
4094 \fi

```

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

```

4095 \let\bxjs@jafont@paren\@firstofone
4096 \let\bxjs@tmpa\jsJaFont
4097 \ifx\bxjs@tmpa\bxjs@@auto
4098 \let\bxjs@tmpa\@empty
4099 \else\ifx\bxjs@tmpa\bxjs@@noEmbed

```

```

4100 \def\bxjs@tmpa{noembed}
4101 \fi\fi
4102 \bxjs@resolve@jafont@paren\bxjs@tmpa
4103 \edef\bxjs@tmpa{\bxjs@catopt\bxjs@tmpa\bxjs@jp@font}
4104 \ifx\bxjs@tmpa\@empty\else
4105 \edef\bxjs@next{%
4106 \noexpand\RequirePackage[\bxjs@tmpa]{pxchfon}[2010/05/12]% v0.5
4107 }\bxjs@next
4108 \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……。)

```

4109 \begingroup
4110 \global\let\bxjs@g@tmpa\relax
4111 \catcode`\|=0 \catcode`\|=12
4112 |def|bxjs@tmpdo@a#1|@nil{%
4113 |bxjs@tmpdo@a#1|@nil\RequirePackage|@nnil}%
4114 |def|bxjs@tmpdo@a#1\RequirePackage#2|@nnil{%
4115 |ifx$#1$|bxjs@tmpdo@b#2|@nil keyval|@nnil |fi}%
4116 |catcode`\|=0 \catcode`\|=12
4117 \def\bxjs@tmpdo@b#1keyval#2\@nnil{%
4118 \ifx$#2$\else
4119 \xdef\bxjs@g@tmpa{%
4120 \noexpand\PassOptionsToPackage{scale=\jsScale}{otf}}%
4121 \fi}
4122 \@firstofone{%
4123 \catcode10=12 \endlinechar\m@ne
4124 \let\do@makeother \dospecials \catcode32=10
4125 \openin\@inputcheck=otf.sty\relax
4126 \@tempswatrue
4127 \loop\if@tempswa
4128 \ifeof\@inputcheck \@tempswafalse \fi
4129 \if@tempswa
4130 \read\@inputcheck to\bxjs@next
4131 \expandafter\bxjs@tmpdo\bxjs@next\@nil
4132 \fi
4133 \repeat
4134 \closein\@inputcheck
4135 \endgroup}
4136 \bxjs@g@tmpa

```

■**hyperref 対策** unicode にしてはいけない。

```

4137 \ifbxjs@hyperref@enc
4138 \bxjs@check@hyperref@unicode{false}

```

暫定的なナニカ。

```
4139 \ifjsWithupTeX\ifbxjs@old@hook@system\else
4140   \IfFileExists{pxjahyper-uni.def}{%
4141     \AddToHook{\bxjs@CGHN{package/hyperref/after}}{\input{pxjahyper-
      uni.def}}
4142   }{}
4143 \fi\fi
4144 \fi
```

tounicode special 命令を出力する。

```
4145 \if \ifx\bxjs@driver@given\bxjs@driver@@dvipdfmx T%
4146   \else\ifjsWithpTeXng T\else F\fi\fi T%
4147 \IfFileExists{pxjahyper-enc.sty}{\@tempwatrue}{\@tempwafalse}
4148 \if@tempswa
4149   \RequirePackage{pxjahyper-enc}[2020/10/05]%v0.6
4150   \ifbxjs@bigcode\else \suppressbigcode \fi
4151 \else
4152   \ifnum\jis"2121="A1A1 %euc
4153     \bxjs@urgent@special{pdf:tounicode EUC-UCS2}
4154   \else\ifnum\jis"2121="8140 %sjis
4155     \bxjs@urgent@special{pdf:tounicode 90ms-RKSJ-UCS2}
4156   \else\ifnum\jis"2121="3000 %uptex
4157     \ifbxjs@bigcode
4158       \bxjs@urgent@special{pdf:tounicode UTF8-UTF16}
4159       \PassOptionsToPackage{bigcode}{pxjahyper}
4160     \else
4161       \bxjs@urgent@special{pdf:tounicode UTF8-UCS2}
4162       \PassOptionsToPackage{nobigcode}{pxjahyper}
4163     \fi
4164   \fi\fi\fi
4165   \let\bxToUnicodeSpecialDone=t
4166 \fi
4167 \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。

```
4168 \ifx f\bxjs@enablejfam\else
4169   \@enablejfamtrue
4170 \fi
```

実際に和文用の数式ファミリの設定を行う。

```
4171 \if@enablejfam
4172   \DeclareSymbolFont{mincho}{\jsc@JYn}{mc}{m}{n}
4173   \DeclareSymbolFontAlphabet{\mathmc}{mincho}
4174   \SetSymbolFont{mincho}{bold}{\jsc@JYn}{gt}{m}{n}
4175   \jfam\symmincho
4176   \DeclareMathAlphabet{\mathgt}{\jsc@JYn}{gt}{m}{n}
4177   \g@addto@macro\bxjs@begin@document@hook{%
4178     \ifx\reDeclareMathAlphabet\undefined\else
```



```

4179 \reDeclareMathAlphabet{\mathrm}{\@mathrm}{\@mathmc}%
4180 \reDeclareMathAlphabet{\mathbf}{\@mathbf}{\@mathgt}%
4181 \reDeclareMathAlphabet{\mathsf}{\@mathsf}{\@mathgt}%
4182 \fi}
4183 \fi

```

C.5 pdfTeX 用設定：CJK + bxcjkjatype

```
4184 \else\if p\jsEngine
```

■**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。

```

4185 \bxjs@adjust@jafont{f}
4186 \let\bxjs@jafont@paren\@firstofone
4187 \bxjs@resolve@jafont@paren\bxjs@tmpa
4188 \edef\bxjs@tmpa{\bxjs@catopt\bxjs@tmpa\bxjs@jp@font}
4189 \edef\bxjs@tmpa{\bxjs@catopt\bxjs@tmpa{whole}}
4190 \ifx\bxjs@jadriver\bxjs@@pandoc\else
4191 \edef\bxjs@tmpa{\bxjs@catopt\bxjs@tmpa{autotilde}}
4192 \fi
4193 \edef\bxjs@next{%
4194 \noexpand\RequirePackage[\bxjs@tmpa]{bxcjkjatype}[2013/10/15]% v0.2c
4195 }\bxjs@next
4196 \bxjs@cjk@loaded

```

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

```

4197 \ifbxjs@hyperref@enc
4198 \PassOptionsToPackage{unicode}{hyperref}
4199 \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。

```

4200 \ifx\bxcjkjatypeHyperrefPatchDone\@undefined
4201 \begingroup
4202 \CJK@input{UTF8.bdg}
4203 \endgroup
4204 \g@addto@macro\pdfstringdefPreHook{%
4205 \@nameuse{CJK@UTF8Binding}%
4206 }
4207 \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。

```

4208 \ifx\bxckjatypeHyperrefPatchDone\@undefined
4209 \g@addto@macro\pdfstringdefPreHook{%
4210   \ifx~\bxjs@@CJKtilde
4211     \let\bxjs@org@LetUnexpandableSpace\HyPsd@LetUnexpandableSpace
4212     \let\HyPsd@LetUnexpandableSpace\bxjs@LetUnexpandableSpace
4213     \let~\@empty
4214   \fi
4215 }
4216 \def\bxjs@@CJKtilde{\CJKecglue\ignorespaces}
4217 \def\bxjs@@tildecmd{~}
4218 \def\bxjs@LetUnexpandableSpace#1{%
4219   \def\bxjs@tmpa{#1}\ifx\bxjs@tmpa\bxjs@@tildecmd\else
4220     \bxjs@org@LetUnexpandableSpace#1%
4221   \fi}
4222 \fi

```

■共通命令の実装

```

4223 \newskip\jsKanjiSkip
4224 \newskip\jsXKanjiSkip
4225 \ifx\CJKecglue\@undefined
4226   \def\CJKtilde{\CJK@global\def~{\CJKecglue\ignorespaces}}
4227 \fi
4228 \let\autospacing\bxjs@enable@kanjiskip
4229 \let\noautospacing\bxjs@disable@kanjiskip
4230 \protected\def\bxjs@CJKglue{\hskip\jsKanjiSkip}
4231 \def\bxjs@apply@kanjiskip{%
4232   \jsKanjiSkip\@tempskipa
4233   \let\CJKglue\bxjs@CJKglue}
4234 \let\autoxspacing\bxjs@enable@xkanjiskip
4235 \let\noautoxspacing\bxjs@disable@xkanjiskip
4236 \protected\def\bxjs@CJKecglue{\hskip\jsXKanjiSkip}
4237 \def\bxjs@apply@xkanjiskip{%
4238   \jsXKanjiSkip\@tempskipa
4239   \let\CJKecglue\bxjs@CJKecglue}

```

\jachar のサブマクロの実装。

```

4240 \def\bxjs@jachar#1{%
4241   \CJKforced{#1}}

```

和欧文間空白の命令 \jathinspace の実装。

```

4242 \ifbxjs@jaspace@cmd
4243   \protected\def\jathinspace{\CJKecglue}
4244 \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。

```

4245 \ifx t\bxjs@enablejfam
4246   \ClassWarningNoLine\bxjs@clsname
4247   {You cannot use 'enablejfam=true', since the\MessageBreak

```

```

4248     CJK package does not support Japanese math}
4249 \fi

```

C.6 X_YTeX 用設定：xeCJK + zxjatype

```

4250 \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。

```

4251 \RequirePackage{zxjatype}
4252 \PassOptionsToPackage{no-math}{fontspec}%!
4253 \PassOptionsToPackage{xetex}{graphicx}%!
4254 \PassOptionsToPackage{xetex}{graphics}%!
4255 \ifx\zxJaFamilyName\undefined
4256     \ClassError{bxjs}{clsname
4257         {xeCJK or zxjatype is too old}}\@ehc
4258 \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。

```

4259 \bxjs@adjust@jafont{f}
4260 \let\bxjs@jafont@paren\@gobble
4261 \bxjs@resolve@jafont@paren\bxjs@tmpa
4262 \edef\bxjs@tmpa{\bxjs@catopt\bxjs@tmpa\bxjs@jp@font}
4263 \ifx\bxjs@tmpa\@empty
4264     \setCJKmainfont[BoldFont=HaranoAjiGothic-Medium.otf]{HaranoAjiMincho-
        Regular.otf}
4265     \setCJKsansfont[BoldFont=HaranoAjiGothic-Medium.otf]{HaranoAjiGothic-
        Medium.otf}
4266 \else
4267     \edef\bxjs@next{%
4268         \noexpand\RequirePackage[\bxjs@tmpa]{zxjafont}[2013/01/28]% v0.2a
4269     }\bxjs@next
4270 \fi

```

■hyperref 対策 unicode オプションの指定に関する話。

X_YTeX 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_YTeX の場合にもパッケージ側で文字コード変換を行う方が望ましいと考えている。実際、unicode を無効にしていると警告が出て強制的に有効化される。一方で、過去 (r35125 まで) の xdvipdfmx では、文字列を UTF-16 に変換した状態で与えるのは不正と見なしていて警告が発生する。

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

※ T_EX の小数の精度は十進で 4 桁までしか保証されないなので、\strcmp を利用して文字列で比較している。(整数部が多桁になっても大丈夫。) しかし実は、\strcmp プリミティブが

追加されたのは 0.9994 版（2009 年 6 月）かららしい。

TODO: バージョン要件を見直して暫定措置を解除する。

```
4271 \ifx\strcmp\@undefined\else % 未定義なら条件を満たさない
4272 \ifnum\strcmp{\the\XeTeXversion\XeTeXrevision}{0.99992}>\m@ne
4273   \ifbxjs@hyperref@enc
4274     \PassOptionsToPackage{unicode}{hyperref}
4275   \fi
4276 \fi
4277 \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。

```
4278 \AtEndOfPackage{%
4279 \def\@gnewline #1{%
4280   \ifvmode \@nolnerr
4281   \else
4282     \unskip \reserved@a {\reserved@a#1}\nobreak \hfil \break \null
4283     \nobreak \hskip-1sp\hskip1sp\relax
4284     \ignorespaces
4285   \fi}
4286 }
```

■共通命令の実装

```
4287 \newskip\jsKanjiSkip
4288 \newskip\jsXKanjiSkip
4289 \ifx\CJKecglue\@undefined
4290   \def\CJKtilde{\CJK@global\def~{\CJKecglue\ignorespaces}}
4291 \fi
4292 \let\autospacing\bxjs@enable@kanjiskip
4293 \let\noautospacing\bxjs@disable@kanjiskip
4294 \protected\def\bxjs@CJKglue{\hskip\jsKanjiSkip}
4295 \def\bxjs@apply@kanjiskip{%
4296   \jsKanjiSkip\@tempskipa
4297   \xeCJKsetup{CJKglue={\bxjs@CJKglue}}}
4298 \let\autoxspacing\bxjs@enable@xkanjiskip
4299 \let\noautoxspacing\bxjs@disable@xkanjiskip
4300 \protected\def\bxjs@CJKecglue{\hskip\jsXKanjiSkip}
4301 \def\bxjs@apply@xkanjiskip{%
4302   \jsXKanjiSkip\@tempskipa
4303   \xeCJKsetup{CJKecglue={\bxjs@CJKecglue}}}
```

`\mcfamily`、`\gtfamily` は本来は zxjatype の方で定義すべきであろうが、現状は暫定的にここで定義する。

```

4304 \ifx\mcfamily\@undefined
4305   \protected\def\mcfamily{\CJKfamily{\CJKrmdefault}}
4306   \protected\def\gtfamily{\CJKfamily{\CJKsfdefault}}
4307 \fi

```

\jachar のサブマクロの実装。

```

4308 \def\bxjs@jachar#1{%
4309   \xeCJKDeclareCharClass{CJK}{`#1}\relax
4310   #1}

```

\jathinspace の実装。

```

4311 \ifbxjs@jaspace@cmd
4312   \protected\def\jathinspace{\CJKecglue}
4313 \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。

```

4314 \ifx t\bxjs@enablejfam
4315   \@enablejfamtrue
4316 \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。

```

4317 \if@enablejfam
4318   \xeCJKsetup{CJKmath=true}
4319 \fi

```

C.7 LuaTeX 用設定：LuaTeX-ja

```

4320 \else\if 1\jsEngine

```

■LuaTeX-ja パッケージの読込 luatexja とともに luatexja-fontspec パッケージを読み込む。

luatexja は自前の \zw（これは実際の現在和文フォントに基づく値を返す）を定義するので、\zw の定義を消しておく。なお、レイアウト定義の「全角幅」は「規定」に基づく \jsZw であることに注意が必要。

※ 1.0b 版から「graphics パッケージに pdftex オプションを渡す」処理を行っていたが、1.4 版で廃止された。

```

4321 \let\zw\@undefined
4322 \RequirePackage{luatexja}
4323 \edef\bxjs@next{%
4324   \noexpand\RequirePackage[scale=\jsScale]{luatexja-fontspec}[2015/08/26]%
4325 }\bxjs@next

```

フォント代替の明示的定義。

```

4326 \DeclareFontShape{JY3}{mc}{m}{it}{<->ssub*mc/m/n}{-}
4327 \DeclareFontShape{JY3}{mc}{m}{sl}{<->ssub*mc/m/n}{-}
4328 \DeclareFontShape{JY3}{mc}{m}{sc}{<->ssub*mc/m/n}{-}

```

```

4329 \DeclareFontShape{JY3}{gt}{m}{it}{<->ssub*gt/m/n}{ }
4330 \DeclareFontShape{JY3}{gt}{m}{sl}{<->ssub*gt/m/n}{ }
4331 \DeclareFontShape{JY3}{mc}{bx}{it}{<->ssub*gt/m/n}{ }
4332 \DeclareFontShape{JY3}{mc}{bx}{sl}{<->ssub*gt/m/n}{ }
4333 \DeclareFontShape{JY3}{gt}{bx}{it}{<->ssub*gt/m/n}{ }
4334 \DeclareFontShape{JY3}{gt}{bx}{sl}{<->ssub*gt/m/n}{ }
4335 \DeclareFontShape{JY3}{mc}{b}{n}{<->ssub*mc/bx/n}{ }
4336 \DeclareFontShape{JY3}{mc}{b}{it}{<->ssub*mc/bx/n}{ }
4337 \DeclareFontShape{JY3}{mc}{b}{sl}{<->ssub*mc/bx/n}{ }
4338 \DeclareFontShape{JY3}{gt}{b}{n}{<->ssub*gt/bx/n}{ }
4339 \DeclareFontShape{JY3}{gt}{b}{it}{<->ssub*gt/bx/n}{ }
4340 \DeclareFontShape{JY3}{gt}{b}{sl}{<->ssub*gt/bx/n}{ }
4341 \DeclareFontShape{JT3}{mc}{m}{it}{<->ssub*mc/m/n}{ }
4342 \DeclareFontShape{JT3}{mc}{m}{sl}{<->ssub*mc/m/n}{ }
4343 \DeclareFontShape{JT3}{mc}{m}{sc}{<->ssub*mc/m/n}{ }
4344 \DeclareFontShape{JT3}{gt}{m}{it}{<->ssub*gt/m/n}{ }
4345 \DeclareFontShape{JT3}{gt}{m}{sl}{<->ssub*gt/m/n}{ }
4346 \DeclareFontShape{JT3}{mc}{bx}{it}{<->ssub*gt/m/n}{ }
4347 \DeclareFontShape{JT3}{mc}{bx}{sl}{<->ssub*gt/m/n}{ }
4348 \DeclareFontShape{JT3}{gt}{bx}{it}{<->ssub*gt/m/n}{ }
4349 \DeclareFontShape{JT3}{gt}{bx}{sl}{<->ssub*gt/m/n}{ }
4350 \DeclareFontShape{JT3}{mc}{b}{n}{<->ssub*mc/bx/n}{ }
4351 \DeclareFontShape{JT3}{mc}{b}{it}{<->ssub*mc/bx/n}{ }
4352 \DeclareFontShape{JT3}{mc}{b}{sl}{<->ssub*mc/bx/n}{ }
4353 \DeclareFontShape{JT3}{gt}{b}{n}{<->ssub*gt/bx/n}{ }
4354 \DeclareFontShape{JT3}{gt}{b}{it}{<->ssub*gt/bx/n}{ }
4355 \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。

```

4356 \bxjs@adjust@jafont{t}
4357 \ifx\bxjs@tmpa\bxjs@@noEmbed
4358   \def\bxjs@tmpa{noembed}
4359 \fi
4360 \let\bxjs@jafont@paren\@gobble
4361 \bxjs@resolve@jafont@paren\bxjs@tmpa
4362 \edef\bxjs@tmpa{\bxjs@catopt\bxjs@tmpa\bxjs@jp@font}
4363 \ifx\bxjs@tmpa\@empty
4364   \defaultjfontfeatures{ Kerning=Off }
4365   \setmainfont[BoldFont=HaranoAjiGothic-Medium.otf,JFM=ujis]{HaranoAjiMincho-Regular.otf}
4366   \setsansfont[BoldFont=HaranoAjiGothic-Medium.otf,JFM=ujis]{HaranoAjiGothic-Medium.otf}
4367 \else
4368   \edef\bxjs@next{%
4369     \noexpand\RequirePackage[\bxjs@tmpa]{luatexja-preset}%

```

```

4370 } \bxjs@next
4371 \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。

```

4372 \@ifpackagelater{luatexja}{2016/03/31}{\%else
4373 \DeclareRobustCommand\rmfamily
4374   {\not@math@alphabet\rmfamily\mathrm
4375     \romanfamily\rmdefault\kanjifamily\mcdefault\selectfont}
4376 \DeclareRobustCommand\sffamily
4377   {\not@math@alphabet\sffamily\mathsf
4378     \romanfamily\sfdefault\kanjifamily\gtdefault\selectfont}
4379 \DeclareRobustCommand\ttfamily
4380   {\not@math@alphabet\ttfamily\mathtt
4381     \romanfamily\ttdefault\kanjifamily\gtdefault\selectfont}
4382 }
4383 \long\def\jttdefault{\gtdefault}
4384 \unless\ifx\@ltj@match@familytrue\@undefined
4385   \@ltj@match@familytrue
4386 \fi
4387 \g@addto@macro\bxjs@begin@document@hook{%
4388   \reDeclareMathAlphabet{\mathrm}{\mathrm}{\mathmc}%
4389   \reDeclareMathAlphabet{\mathbf}{\mathbf}{\mathgt}%
4390   \reDeclareMathAlphabet{\mathsf}{\mathsf}{\mathgt}}%
4391 \bxjs@if@sf@default{%
4392   \renewcommand\kanjifamilydefault{\gtdefault}}

```

■和文パラメタの設定

```

4393 % 次の3つは既定値の通り
4394 \%ltjsetparameter{prebreakpenalty={`',10000}}
4395 \%ltjsetparameter{postbreakpenalty={`",10000}}
4396 \%ltjsetparameter{prebreakpenalty={`",10000}}
4397 \%ltjsetparameter{jaxspmode={`!',1}}
4398 \%ltjsetparameter{jaxspmode={`〒,2}}
4399 \%ltjsetparameter{alxspmode={`+,3}}
4400 \%ltjsetparameter{alxspmode={`%,3}}

```

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

※`\jsInhibitGlueAtParTop` は使わない。

`\ltjfakeparbegin` 現在の Lua_T_EX-ja で定義されているマクロで、段落中で段落冒頭用の処理を発動する。未定義である場合に備えて同等のものを用意する。

```

4401 \ifx\ltjfakeparbegin\@undefined
4402   \protected\def\ltjfakeparbegin{%
4403     \ifhmode
4404       \relax\directlua{%
4405         luatexja.jfmglue.create_beginpar_node()}}
4406   \fi}

```

4407 \fi

ltjs* クラスの定義と同等になるようにパッチを当てる。

```
4408 \unless\ifnum\bxjs@everyparhook=\bxjs@everyparhook@@none
4409 \begingroup
4410 \let\%\@percentchar \def\@#1{[[\detokenize{#1}]]}
4411 \@gobble\if\def\bxjs@tmpa{\@{\everypar}\fi}
4412 \ifnum\bxjs@everyparhook=\bxjs@everyparhook@@compat
4413 \@gobble\if\def\bxjs@tmpa{\@{\everypar\everyparhook}\fi}\fi
4414 \directlua{
4415   local function patchcmd(cs, code, from, to)
4416     tex.sprint(code:gsub(from:gsub("\%w", "%\\%\\%0"), "%\\%0"..to)
4417       :gsub("macro:", \@gdef..cs, 1):gsub("->", "{", 1)..")
4418   end
4419   patchcmd(\@xsect, [[\meaning\xsect]],
4420     \@{\hskip-\@tempskipa}, \@ltjfakeparbegin)
4421   patchcmd(\@item, [[\meaning@item]],
4422     \bxjs@tmpa, \@ltjfakeparbegin)}
4423 \endgroup
4424 \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。

```
4425 \ifbxjs@hyperref@enc
4426 \PassOptionsToPackage{unicode}{hyperref}
4427 \bxjs@check@hyperref@unicode{true}
4428 \fi
```

■共通命令の実装

```
4429 \protected\def\autospacing{%
4430 \ltjsetparameter{autospacing=true}}
4431 \protected\def\noautospacing{%
4432 \ltjsetparameter{autospacing=false}}
4433 \protected\def\autoxspacing{%
4434 \ltjsetparameter{autoxspacing=true}}
4435 \protected\def\noautoxspacing{%
4436 \ltjsetparameter{autoxspacing=false}}
4437 \def\bxjs@apply@kanjiskip{%
4438 \ltjsetparameter{kanjiskip={\@tempskipa}}}
4439 \def\bxjs@apply@xkanjiskip{%
4440 \ltjsetparameter{xkanjiskip={\@tempskipa}}}
```

\jachar のサブマクロの実装。

```
4441 \def\bxjs@jachar#1{%
4442 \ltjjachar`#1\relax}
```

\jathinspace の実装。

```
4443 \ifbxjs@jaspace@cmd
4444 \protected\def\jathinspace{%
```



```

4445 \hskip\ltjgetparameter{xkanjiskip}\relax}
4446 \fi

```

■和文数式ファミリー LuaTeX-j_a では和文数式ファミリーは常に有効で、既にこの時点で必要な設定は済んでいる。従って @enablejfam は常に真になる。

```

4447 \ifx f\bxjs@enablejfam
4448 \ClassWarningNoLine\bxjs@clsname
4449 {You cannot use 'enablejfam=false', since the\MessageBreak
4450 LuaTeX-ja always provides Japanese math families}
4451 \fi

```

C.8 共通処理 (2)

```

4452 \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 4453 \ifx\DeclareFixJFMCJKTextFontCommand\@undefined
4454 \DeclareJaTextFontCommand{\textmc}{\mcfamily}
4455 \DeclareJaTextFontCommand{\textgt}{\gtfamily}
4456 \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。

```

4457 \ifx\mathmc\@undefined
4458 \DeclareJaMathFontCommand{\mathmc}{\mcfamily}
4459 \DeclareJaMathFontCommand{\mathgt}{\gtfamily}
4460 \fi

```

■和文空白命令

\> 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)。

```

4461 \ifbxjs@jaspace@cmd
4462 \bxjs@protected\def\bxjs@choice@jathinspace{%
4463 \relax\ifmmode \mskip\medmuskip
4464 \else \jathinspace\ignorespaces
4465 \fi}
4466 \jsAtEndOfClass{%
4467 \ifjsWithTeX \let\>\bxjs@choice@jathinspace
4468 \else \def\>\protect\bxjs@choice@jathinspace}%
4469 \fi}
4470 \fi

```

■和文・和欧文間空白の初期値

```

4471 \setkanjiskip{0pt plus.1\jsZw minus.01\jsZw}

```

```

4472 \ifx\jsDocClass\jsSlide \setxkanjiskip{0.1em}
4473 \else \setxkanjiskip{0.25em plus 0.15em minus 0.06em}
4474 \fi

```

以上で終わり。

```

4475 %</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。

```

4476 %<*modern>
4477 \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。

```

4478 \ifnum0\if x\jsEngine1\fi\if 1\jsEngine1\fi=\z@
4479 \def\encodingdefault{T1}%
4480 \input{t1enc.def}%
4481 \fontencoding\encodingdefault\selectfont
4482 \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。

```

4483 \ifnum0\if x\jsEngine1\fi\if 1\jsEngine1\fi=\z@
4484 \renewcommand{\rmdefault}{lmr}
4485 \renewcommand{\sfdefault}{lmss}
4486 \renewcommand{\ttdefault}{lmtt}
4487 \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。

```

4488 \DeclareFontShape{OMX}{cmex}{m}{n}{%
4489   <-7.5>cmex7<7.5-8.5>cmex8%
4490   <8.5-9.5>cmex9<9.5->cmex10}{}%
4491 \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。

```

4492 \def\cmex@opt{10}

```

D.2 fixltx2e 読込

※`fixltx2e` 廃止前の L^AT_EX カーネルの場合。

```

4493 \ifx\@IncludeInRelease\@undefined
4494 \RequirePackage{fixltx2e}
4495 \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。

```

4496 \RequirePackage{bxjscjkat}

```

D.4 完了

おしまい。

```

4497 %</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

standard ドライバの設定を引き継ぐ。

```

4498 %<*pandoc>

```

```

4499 \input{bxjsja-standard.def}

```

bxjspandoc パッケージを読み込む。

```

4500 \RequirePackage{bxjspandoc}

```

ϵ -TeX ではない場合に警告を出す。

※近い将来に ϵ -TeX 拡張を必須にする予定。

```

4501 \ifjsWitheTeX\else
4502   \ClassWarningNoLine{bxjs@clsname}
4503   {!!!!!!! WARNING !!!!!!!\MessageBreak
4504     This engine does not support e-TeX extension!\MessageBreak
4505     Some feature might not work properly}
4506 \fi

```

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

```

4507 \newif\ifbxjs@bxghost@available
4508 \ifjsWitheTeX
4509   \RequirePackage{pdftexcmds}[2009/09/22]% v0.5
4510   \IfFileExists{bxghost.sty}{%
4511     \bxjs@bxghost@availabletrue
4512     \@namedef{bxjs@bgbv/79E70A0991967E27981832C84DB5DF99}{1}%v0.2.0
4513     \ifx\pdf@filemdfivesum\@undefined\else

```

```

4514 \expandafter\ifx\csname bxjs@bgbv\pdf@filemdfivesum{bxghost.sty}%
4515 \endcsname\relax\else \bxjs@bxghost@availablefalse \fi
4516 \fi
4517 }{}
4518 \fi

```

`\bxjs@endpreamble@hook` etoolbox の `\AtEndPreamble` で実行される BXJS クラス用のフック。

※ ε -TeX 以外では無効になる。(将来 pandoc の外に出す可能性あり。)

```

4519 \@onlypreamble\bxjs@endpreamble@hook
4520 \let\bxjs@endpreamble@hook\empty

```

パッケージ読込。

```

4521 \RequirePackage{iftex}[2013/04/04]% v0.2
4522 \ifjsWithTeX
4523 \RequirePackage{etoolbox}[2010/08/21]% v2.0
4524 \AtEndPreamble{\bxjs@endpreamble@hook}
4525 \RequirePackage{filehook}[2011/10/12]% v0.5d
4526 \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。

```

4527 \newif\ifbxjs@jp@fix@strong \bxjs@jp@fix@strongtrue

```

`fix-strong` オプションの処理。

```

4528 \let\bxjs@kv@fixstrong@true\bxjs@jp@fix@strongtrue
4529 \let\bxjs@kv@fixstrong@false\bxjs@jp@fix@strongfalse
4530 \define@key{bxjsPan}{fix-strong}[true]{%
4531 \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。

```

4532 \newif\ifbxjs@jp@fix@code \bxjs@jp@fix@codetrue

```

`fix-code` オプションの処理。

```

4533 \let\bxjs@kv@fixcode@true\bxjs@jp@fix@codetrue
4534 \let\bxjs@kv@fixcode@false\bxjs@jp@fix@codefalse
4535 \define@key{bxjsPan}{fix-code}[true]{%
4536 \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。

```

4537 \chardef\bxjs@jp@strong=0

```

`strong` オプションの処理。

```

4538 \def\bxjs@kv@strong@bold{\chardef\bxjs@jp@strong=0 }
4539 \def\bxjs@kv@strong@sans{\chardef\bxjs@jp@strong=1 }
4540 \def\bxjs@kv@strong@boldsans{\chardef\bxjs@jp@strong=2 }

```

```

4541 \define@key{bxjsPan}{strong}{%
4542   \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 4543 \newif\ifbxjs@jp@or@indent \bxjs@jp@or@indenttrue
\ifbxjs@jp@or@block@heading 4544 \newif\ifbxjs@jp@or@secnumdepth \bxjs@jp@or@secnumdepthtrue
4545 \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。

```

4546 \define@key{bxjsPan}{_plus}[]{}%
4547   \bxjs@jp@or@indentfalse
4548   \bxjs@jp@or@secnumdepthfalse
4549   \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。

```

4550 \let\bxjs@kv@orindent@true\bxjs@jp@or@indenttrue
4551 \let\bxjs@kv@orindent@false\bxjs@jp@or@indentfalse
4552 \define@key{bxjsPan}{or-indent}[true]{}%
4553   \bxjs@set@keyval{orindent}{#1}{}}
4554 \let\bxjs@kv@orsecnumdepth@true\bxjs@jp@or@secnumdepthtrue
4555 \let\bxjs@kv@orsecnumdepth@false\bxjs@jp@or@secnumdepthfalse
4556 \define@key{bxjsPan}{or-secnumdepth}[true]{}%
4557   \bxjs@set@keyval{orsecnumdepth}{#1}{}}
4558 \let\bxjs@kv@orblockheading@true\bxjs@jp@or@block@headingtrue
4559 \let\bxjs@kv@orblockheading@false\bxjs@jp@or@block@headingfalse
4560 \define@key{bxjsPan}{or-block-heading}[true]{}%
4561   \bxjs@set@keyval{blockheading}{#1}{}}

```

実際の `japaram` の値を適用する。

```

4562 \def\bxjs@next#1{\bxjs@safe@setkeys{bxjsPan}{#1}}
4563 \expandafter\bxjs@next\expandafter{\jsJaParam}

```

E.3 dupload システム

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。

```

4564 \@onlypreamble\bxjs@set@dupload@proc
4565 \def\bxjs@set@dupload@proc#1{%
4566   \expandafter\bxjs@set@dupload@proc@a\csname bxjs@dlp/#1\endcsname}

```

```

4567 \@onlypreamble\bxjs@set@dupload@proc@a
4568 \def\bxjs@set@dupload@proc@a#1{%
4569   \@onlypreamble#1\def#1##1}
4570 \def\bxjs@unset@dupload@proc#1{%
4571   \bxjs@cslet{bxjs@dlp/#1}\@undefined}

```

\@if@options \@if@options の再定義。

```

4572 \@onlypreamble\bxjs@org@if@options
4573 \let\bxjs@org@if@options\@if@options
4574 \@onlypreamble\bxjs@org@reset@options
4575 \let\bxjs@org@reset@options\relax
4576 \def\@if@options#1#2#3{%
4577   \let\bxjs@next\@secondoftwo
4578   \def\bxjs@tmpa{#1}\def\bxjs@tmpb{\@current}%
4579   \ifx\bxjs@tmpa\bxjs@tmpb
4580     \expandafter\ifx\csname bxjs@dlp/#2.#1\endcsname\relax\else
4581       \let\bxjs@next\@firstoftwo \fi
4582   \fi
4583   \bxjs@next\bxjs@do@dupload@proc\bxjs@org@if@options{#1}{#2}{#3}}
4584 \g@addto@macro\bxjs@begin@document@hook{%
4585   \let\@if@options\bxjs@org@if@options}
4586 \@onlypreamble\bxjs@do@dupload@proc
4587 \def\bxjs@do@dupload@proc#1#2#3{%
4588   \ifx\bxjs@org@reset@options\relax
4589     \let\bxjs@org@reset@options\@reset@options
4590   \fi
4591   \bxjs@csletcs{bxjs@next}{bxjs@dlp/#2.#1}%
4592   \def\@reset@options{%
4593     \let\@reset@options\bxjs@org@reset@options
4594     \@reset@options
4595     \bxjs@next{#3}}%
4596   \@firstoftwo}

```

E.4 lang 変数

lang=ja という言語指定が行われると、Pandoc はこれに対応していないため不完全な Babel や Polyglossia の設定を出力してしまう。これを防ぐための対策を行う。

※ Pandoc 2.12 版で lang=ja 指定に対応し、正しく L^AT_EX 側の言語名 `japanese` に変換されるようになった。しかし、日本語指定の場合は相変わらず調整処理が必要である。

\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。

```

4597 \let\bxjs@polyglossia@options\relax

```

\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。

```

4598 \let\bxjs@babel@options\relax

```

■Polyglossia について つまり Xe_{La}TeX および Lua_{La}TeX（古い Pandoc で）の場合。

※この場合 etoolbox が使用可能になっている。

```
4599 \ifnum0\if x\jsEngine1\fi\if l\jsEngine1\fi>0
```

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

```
4600 \pandocSkipLoadPackage{polyglossia}
4601 \bxjs@set@dupload@proc{polyglossia.sty}{%
4602   \bxjs@unset@dupload@proc{polyglossia.sty}%
4603   \ClassWarning\bxjs@clsname
4604     {Package polyglossia is requested}%
4605   \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。

※先に \setmainlanguage 以外が実行された場合はエラーになる。

```
4606   \newcommand*\setmainlanguage[2] []{%
```

もし、言語名が空の \setmainlanguage{} が実行された場合は、lang=ja が指定されたと見なす。言語名が japanese だった場合も同様。

```
4607     \ifboolexpr{test{\ifblank{##2}}or test{\ifstrequal{##2}{japanese}}}{%
4608       \ClassWarning\bxjs@clsname
4609         {Main language is 'japanese', thus fallback\MessageBreak
4610           definitions will be employed}%
4611       \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。

```
4612   }{%else
4613     \ClassWarning\bxjs@clsname
4614       {Main language is '##2',\MessageBreak
4615         thus polyglossia will be loaded}%
4616     \csundef{ver@polyglossia.sty}%
4617     \edef\bxjs@next{%
4618       \noexpand\RequirePackage[\bxjs@polyglossia@options]{polyglossia}[]%
4619     }\bxjs@next
4620     \setmainlanguage[##1]{##2}%
4621   }}
```

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、読込済マークを外す。

```
4622 \g@addto@macro\bxjs@endpreamble@hook{%
4623   \ifx\bxjs@polyglossia@options\relax
4624     \csundef{ver@polyglossia.sty}%
4625   \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。その時点でダミーの \setotherlanguage(s) を定義する。

※現在では Polyglossia の日本語用の定義ファイル (gloss-japanese.ldf) が存在するので、本来なら普通に処理できるはずであるが、現状の定義ファイルはアレなので回避したい。

```

4626 \onlypreamble\bxjs@pandoc@polyglossia@ja
4627 \def\bxjs@pandoc@polyglossia@ja{%
4628   \renewcommand*\setmainlanguage[2] [] {}%
4629   \newcommand*\setotherlanguage[2] [] {}%
4630   \ifblank{##2}{}{}%else
4631     \cslet{##2}\@empty \cslet{end##2}\@empty
4632     \cslet{text##2}\@firstofone}%
4633   \newcommand*\setotherlanguages[2] [] {}%
4634   \@for\bxjs@tmpa:={##2}\do{%
4635     \setotherlangauge{\bxjs@tmpa}}}%

```

Polyglossia の読込済マークは外れるようにしておく。

```

4636 \let\bxjs@polyglossia@options\relax}%
4637 \fi

```

■Babel について Xe_{La}TeX 以外の場合。

※ Pandoc 2.15 版から、テンプレートで用いられる多言語パッケージが Babel に統一された。(Lua_{TeX} は 2.6 版で Polyglossia から Babel に変更されている。)

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

```

4638 \pandocSkipLoadPackage{babel}
4639 \bxjs@set@dupload@proc{babel.sty}{%
4640   \bxjs@unset@dupload@proc{babel.sty}%
4641   \ClassWarning\bxjs@clsname
4642     {Package babel is requested}}%

```

パッケージオプションに言語名が空の main= があるかを調べる。ある場合は lang=ja 対策を実行する。

※\bxjs@babel@options には main= を除いたオプション列を格納する。

```

4643 \@tempswafalse \let\bxjs@babel@options\@empty
4644 \def\bxjs@tmpb{main=}%
4645 \def\bxjs@next{main=japanese}%
4646 \@for\bxjs@tmpa:=#1\do{%
4647   \ifx\bxjs@tmpa\bxjs@tmpb \@tempswattrue
4648   \else\ifx\bxjs@tmpa\bxjs@next \@tempswattrue
4649   \else \edef\bxjs@babel@options{\bxjs@babel@options,\bxjs@tmpa}%
4650   \fi\fi}%
4651 \if@tempswa
4652   \ClassWarning\bxjs@clsname
4653     {Main language is 'japanese', thus fallback\MessageBreak
4654       definitions will be employed}%
4655   \bxjs@pandoc@babel@ja

```

ない場合は、本来の babel の処理を実行する。

```

4656 \else
4657   \ClassWarning\bxjs@clsname

```



```

4658     {Main language is not 'japanese',\MessageBreak
4659     thus babel will be loaded}%
4660     \bxjs@cslet{ver@babel.sty}\@undefined
4661     \RequirePackage[#1]{babel}[]%
4662     \fi}

```

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。

```

4663 \g@addto@macro\bxjs@endpreamble@hook{%
4664     \ifx\bxjs@babel@options\relax
4665     \bxjs@cslet{ver@babel.sty}\@undefined
4666     \fi}

```

`\bxjs@pandoc@babel@ja` Pandoc 側で `lang=ja` が指定されていた場合の処理。

```

4667 \@onlypreamble\bxjs@pandoc@babel@ja
4668 \def\bxjs@pandoc@babel@ja{%
4669     \bxjs@cslet{ver@babel.sty}\@undefined
4670     \edef\bxjs@next{%
4671         \noexpand\RequirePackage[\bxjs@babel@options,english]{babel}[]%
4672     }\bxjs@next
4673     \if j\jsEngine
4674         \RequirePackage[main=japanese]{pxbabel}[]%
4675     \else
4676         \RequirePackage{bxorigcapt}[]%
4677     \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。

```

4678 \bxjs@set@dupload@proc{geometry.sty}{%
4679     \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。

```

4680 \if l\jsEngine
4681     \pandocSkipLoadPackage{xeCJK}
4682     \providecommand*\setCJKmainfont{\setmainfont}
4683 \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。

```
4684 \if x\jsEngine
4685   \expandafter\g@addto@macro\csname opt@xeCJK.sty\endcsname{%
4686     ,space}
4687 \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。

```
4688 \ifjsWithTeX
4689 \onlypreamble\bxjs@info@or@ban
4690 \def\bxjs@info@or@ban#1{%
4691   \PackageInfo\bxjs@clsname
4692   {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。

```
4693 \unless\ifbxjs@jp@or@indent
4694   \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。

```
4695   \IfFileExists{parskip.sty}{%
4696     \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。

```
4697 }{%else
4698   \eappto\bxjs@endpreamble@hook{%
4699     \parindent=\the\parindent\relax
4700     \parskip=\the\parskip\relax}}
4701 \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。

```
4702 \ifbxjs@jp@or@secnumdepth\else
4703   \bxjs@info@or@ban{secnumdepth}
4704   \eappto\bxjs@endpreamble@hook{%
4705     \c@secnumdepth=\the\c@secnumdepth\relax}
4706 \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。

```
4707 \ifbxjs@jp@or@block@heading\else
4708   \let\bxjs@frozen@paragraph\paragraph
4709   \let\bxjs@frozen@subparagraph\subparagraph
```

```

4710 \bxjs@info@or@ban{block-heading}
4711 \appto\bxjs@endpreamble@hook{%
4712 \let\oldparagraph\@undefined
4713 \let\paragraph\bxjs@frozen@paragraph
4714 \let\subparagraph\bxjs@frozen@subparagraph}
4715 \fi

    以上。
4716 \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 がプレアンプルで行う再定義の結果を調べるため、begin-document フックを利用する。

```

4717 \g@addto@macro\bxjs@begin@document@hook{%
4718 \@tempswafalse

```

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

```

4719 \ifx\oldparagraph\@undefined\else
4720 \@tempswatrue
4721 \fi

```

`\paragraph` が番号付きの場合は、マーク変更が必要である。

```

4722 \ifnum\c@secnumdepth>3
4723 \@tempswatrue
4724 \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。

```

4725 \if@tempswa\ifx\jsParagraphMark\bxjs@org@paragraph@mark
4726 \let\jsParagraphMark\@empty
4727 \fi\fi}

```

E.10 全角空白文字

L^AT_EX 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^AT_EX 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。

```
4728 \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。

```
4729 \ifnum0\if x\jsEngine1\fi\if 1\jsEngine1\fi>\z@
4730 \catcode"3000=\active
4731 \begingroup \catcode`\!=7
4732 \protected\gdef!!!3000{\pandocZWSpace}
4733 \endgroup
4734 \else\ifx\DeclareUnicodeCharacter\@undefined\else
4735 \DeclareUnicodeCharacter{3000}{\bxjs@zsp@char}
4736 \bxjs@protected\def\bxjs@zsp@char{\pandocZWSpace}
4737 \fi\fi
```

E.11 hyperref 対策

`hyperref` の `unicode` オプションの固定を行う。

```
4738 \if j\jsEngine
4739 \bxjs@fix@hyperref@unicode{false}
4740 \else
4741 \bxjs@fix@hyperref@unicode{true}
4742 \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 $\text{L}^{\text{A}}\text{T}_{\text{E}}\text{X}$ 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 (うわあ)。

```
4743 \ifbxjs@jp@fix@strong\ifbxjs@jp@strong@cmd
4744 \let\orgtextbf\textbf
4745 \DeclareRobustCommand\pandocTextbf[1]{%
4746 \begingroup
4747 \let\textbf\orgtextbf
4748 \strong{#1}%
4749 \endgroup}%
4750 \g@addto@macro\bxjs@begin@document@hook{%
4751 \let\textbf\pandocTextbf}
4752 \fi\fi
```

`\strong` の書体を設定する。

```
4753 \jsAtEndOfClass{%
4754 \ifx\strongfontdeclare\@undefined\else
4755 \ifcase\bxjs@jp@strong
4756 \or \strongfontdeclare{\sffamily}%
4757 \or \strongfontdeclare{\sffamily\bfseries}%
4758 \fi
4759 \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^AT_EX 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。

4760 `\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。

```

4761 \ifbxjs@bxghost@available
4762   \RequirePackage[verb]{bxghost}[2020/01/31]% v0.3.0
4763   \let\bxjs@eghostguarded\eghostguarded
4764   \else
4765   \chardef\bxjs@eghost@c=23
4766   \ifx j\jsEngine \xspcode\bxjs@eghost@c=3
4767   \else\ifx l\jsEngine \ltjsetParameter{alxspmode={\bxjs@eghost@c,3}}
4768   \else\ifx x\jsEngine %no-op
4769   \else \let\bxjs@eghost@c\@undefined
4770   \fi\fi\fi
4771   \ifx\bxjs@eghost@c\@undefined\else
4772   \font\bxjs@eghost@f=ec-lmr10 at 1.23456pt
4773   \def\bxjs@pan@eghost{\bgroup\bxjs@eghost@f\bxjs@eghost@c\egroup}
4774   \def\bxjs@eghostguarded#1{%
4775     \bxjs@pan@eghost\@null#1\@null\bxjs@pan@eghost}
4776   \fi
4777   \fi
4778   \ifx\bxjs@eghostguarded\@undefined\else
4779   \let\orgtexttt\texttt
4780   \DeclareRobustCommand\pandocTexttt[1]{%
4781     \ifmmode \nfss@text{\ttfamily #1}%
4782     \else
4783     \ifvmode \leavevmode \fi
4784     \bxjs@eghostguarded{\begingroup\ttfamily#1\endgroup}%
4785     \fi}
4786   \g@addto@macro\bxjs@begin@document@hook{%
4787     \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。

```

4788 \ifbxjs@bxghost@available\else
4789 \expandafter\def\expandafter\verb\expandafter{%
4790 \expandafter\bxjs@pan@eghost\verb}
4791 \g@addto@macro\verb\egroup{\bxjs@pan@eghost}
4792 \fi
4793 \fi
4794 \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。

```
4795 \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。
```

```
4796 \def\bxjs@check@frontier{%
4797   \expandafter\bxjs@check@frontier@a\@currnamestack\noindent...\@nil}
4798 \def\bxjs@check@frontier@a#1#2#3#4#5\@nil#6{%
4799   \ifx\noindent#4#6\fi}
```

```
\bxjs@unforge@ifPDFTeX \ifPDFTeX を偽（正しい値）にする。
```

```
4800 \@onlypreamble\bxjs@unforge@ifPDFTeX
4801 \def\bxjs@unforge@ifPDFTeX{\global\bxjs@csletcs{ifPDFTeX}{iffalse}}
```

```
\bxjs@forge@ifPDFTeX \ifPDFTeX を真（偽装した値）にする。
```

```
4802 \@onlypreamble\bxjs@forge@ifPDFTeX
4803 \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。
```

```
4804 \def\bxjs@unload@forge@ifPDFTeX{%
4805   \bxjs@unforge@ifPDFTeX
4806   \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。

```
4807 \jsAtEndOfClass{\bxjs@forge@ifPDFTeX}
4808 \ifjsWithTeX
4809   \AtBeginOfEveryFile{\bxjs@check@frontier\bxjs@unforge@ifPDFTeX}%
4810   \AtEndOfEveryFile{\bxjs@check@frontier\bxjs@forge@ifPDFTeX}%
4811   \g@addto@macro\bxjs@endpreamble\hook{\bxjs@unload@forge@ifPDFTeX}
4812 \else
4813   \g@addto@macro\bxjs@begin@document\hook{\bxjs@unload@forge@ifPDFTeX}
4814 \fi
4815 \fi
```

E.14 完了

おしまい。

```
4816 %</pandoc>
```

和文ドライバ実装はここまで。

```
4817 %</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 : ムニャムニャムニャ。
- 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。

```
4818 %<*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。

G.1 準備

```
4819 %<*compat>
```

```
4820 \def\bxac@pkgname{bxjscompat}
```

`\bxjx@engine` エンジンの種別。

```
4821 \let\bxac@engine=n
```

```
4822 \def\bxac@do#1#2{%
```

```
4823   \edef\bxac@tmpa{\string#1}%
```

```
4824   \edef\bxac@tmpb{\meaning#1}%
```

```
4825   \ifx\bxac@tmpa\bxac@tmpb #2\fi}
```

```
4826 \bxac@do\XeTeXversion{\let\bxac@engine=x}
```

```
4827 \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。

```
4828 \ifx\jsAtEndOfClass\undefined
```

```
4829   \let\bxac@delayed@if@bxjs\@firstofone
```

```
4830 \else \let\bxac@delayed@if@bxjs\jsAtEndOfClass
```

```
4831 \fi
```

`\ImposeOldLuaTeXBehavior` `\ImposeOldLuaTeXBehavior` は 0.85 版以降の LuaTeX を一時的に pdfTeX と互換である `\RevokeOldLuaTeXBehavior` ように見せかける。`\RevokeOldLuaTeXBehavior` で元に戻すことができる。

※エンジンが Lua_T_EX 以外の場合は何もしない。

```
4832 \newif\ifbxac@in@old@behavior
4833 \let\ImposeOldLuaTeXBehavior\relax
4834 \let\RevokeOldLuaTeXBehavior\relax
```

G.2 Xe_TE_X 部分

```
4835 \ifx x\bxac@engine
```

■文字クラスの設定 Xe_TE_X の文字クラス (`\XeTeXcharclass`) の Unicode 規定に基づく設定は、初期の版ではフォーマットに組み込まれていたが、2016/02/01 以降の L^AT_EX 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、Xe_TE_X に「文字間トークン挿入」の機能が導入されたのは 0.997 版 (2007 年頃) からのようだ。

ただし xeCJK が読込済ならば (そちらが適切に設定しているはずなので) 何もしない。

```
4836 \ifx\XeTeXcharclass\undefined\else
4837 \bxac@delayed@if@bxjs{%
4838   \@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。

```
4839   \ifx\Xe@alloc@intercharclass\undefined\else
4840     \ifnum\Xe@alloc@intercharclass=\z@
4841       \PackageInfo\bxac@pkgname
4842       {Setting up interchar class for CJK...\@gobble}%
4843       \InputIfFileExists{load-unicode-xetex-classes.tex}{-}%
4844       \Xe@alloc@intercharclass=3
4845     }{-}\else
4846       \PackageWarning\bxac@pkgname
4847       {Cannot find file 'load-unicode-xetex-classes.tex'%
4848        \@gobble}%
4849     }%
4850   \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。

```
4851   \ifnum\XeTeXcharclass"3041=\z@
4852     \PackageInfo\bxac@pkgname
4853     {Adjusting interchar class for CJK...\@gobble}%
4854     \@for\bxac@tmpb:={%
4855       3041,3043,3045,3047,3049,3063,3083,3085,3087,308E,%
4856       3095,3096,30A1,30A3,30A5,30A7,30A9,30C3,30E3,30E5,%
4857       30E7,30EE,30F5,30F6,30FC,31F0,31F1,31F2,31F3,31F4,%
4858       31F5,31F6,31F7,31F8,31F9,31FA,31FB,31FC,31FD,31FE,%
4859       31FF%
4860     }\do{\XeTeXcharclass"\bxac@tmpb=\@ne}%
4861   \fi
4862 }%
4863 }
```


4864 \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。

```
4865 \chardef\bxac@tmpb=11
4866 \def\bxac@do#1#2{%
4867   \@tempcnta=#1\relax
4868   \unless\ifnum\catcode\@tempcnta=\bxac@tmpb
4869     \chardef\bxac@tmpa=#2\relax
4870     \@whilenum{\@tempcnta<\bxac@tmpa}\do{%
4871       \catcode\@tempcnta\bxac@tmpb \advance\@tempcnta\@ne}%
4872   \fi}
4873 \bxac@do{"4E00}{ "9FCD}
```

以上。

4874 \fi

G.3 LuaTeX 部分

4875 \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。

```
4876 %\unless\ifnum\luatexversion<80 \ifnum\luatexversion<85
4877 % \chardef\pdfTeXversion=200
4878 % \def\pdfTeXrevision{0}
4879 % \let\pdfTeXbanner\luatexbanner
4880 %\fi\fi
```

\ImposeOldLuaTeXBehavior 0.85 版以降であるかを検査する。

```
\RevokeOldLuaTeXBehavior 4881 \begingroup\expandafter\expandafter\expandafter\endgroup
4882 \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。

```
4883 \def\bxac@ob@list{%
4884   \do{\let}\pdfoutput{\outputmode}%
4885   \do{\let}\pdfpagewidth{\pagewidth}%
4886   \do{\let}\pdfpageheight{\pageheight}%
4887   \do{\protected\edef}\pdfhorigin{{\pdfvariable horigin}}%
4888   \do{\protected\edef}\pdfvorigin{{\pdfvariable vorigin}}%
4889 \def\bxac@ob@do#1#2{\begingroup
4890   \expandafter\bxac@ob@do@a\csname bxac@string#2\endcsname{#1}#2}
4891 \def\bxac@ob@do@a#1#2#3#4{\endgroup
4892   \ifbxac@in@old@behavior \let#1#3\relax #2#3#4\relax
4893   \else \let#3#1\relax \let#1\@undefined
4894   \fi}
4895 \protected\def\ImposeOldLuaTeXBehavior{%
4896   \unless\ifbxac@in@old@behavior
4897     \bxac@in@old@behaviortrue
4898     \let\do\bxac@ob@do \bxac@ob@list
```

```

4899 \fi}
4900 \protected\def\RevokeOldLuaTeXBehavior{%
4901 \ifbxac@in@old@behavior
4902 \bxac@in@old@behaviorfalse
4903 \let\do\bxac@ob@do \bxac@ob@list
4904 \fi}
4905 \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。

```

4906 \directlua{
4907   local function range(cs, ce, cc, ff)
4908     if ff or not tex.getcatcode(cs) == cc then
4909       local setcc = tex.setcatcode
4910       for c = cs, ce do setcc(c, cc) end
4911     end
4912   end
4913   range(0x3400, 0x4DB5, 11, false)
4914   \ifnum\luatexversion>64
4915     range(0x4DB5, 0x4DBF, 11, true)
4916     range(0x4E00, 0x9FCC, 11, false)
4917     range(0x9FCD, 0x9FFF, 11, true)
4918     range(0xAC00, 0xD7A3, 11, false)
4919     range(0x20000, 0x2A6D6, 11, false)
4920     range(0x2A6D7, 0x2A6FF, 11, true)
4921     range(0x2A700, 0x2B734, 11, false)
4922     range(0x2B735, 0x2B73F, 11, true)
4923     range(0x2B740, 0x2B81D, 11, false)
4924     range(0x2B81E, 0x2B81F, 11, true)
4925     range(0x2B820, 0x2CEA1, 11, false)
4926     range(0x2CEA2, 0x2FFFD, 11, true)
4927   \fi
4928 }

```

以上。

```

4929 \fi

```

G.4 完了

おしまい。

```

4930 %</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

```

4931 %<*cjkcat>
4932 \def\bxjx@pkgname{bxjscjkcat}
4933 \newcount\bxjx@canta
4934 \@onlypreamble\bxjx@tmpdo
4935 \@onlypreamble\bxjx@tmpdo@a
4936 \@onlypreamble\bxjx@tmpdo@b

```

`\bxjx@engine` エンジンの種別。

```

4937 \let\bxjx@engine=n
4938 \def\bxjx@tmpdo#1#2{%
4939   \edef\bxjx@tmpa{\string#1}%
4940   \edef\bxjx@tmpb{\meaning#1}%
4941   \ifx\bxjx@tmpa\bxjx@tmpb #2\fi}
4942 \bxjx@tmpdo\kanjiskip{\let\bxjx@engine=j}
4943 \bxjx@tmpdo\enablecjktoken{\let\bxjx@engine=u}
4944 \bxjx@tmpdo\XeTeXversion{\let\bxjx@engine=x}
4945 \bxjx@tmpdo\pdfTeXversion{\let\bxjx@engine=p}
4946 \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。

```

4947 \def\bxjx@tmpdo#1#2{%
4948   \if#1\bxjx@engine
4949     \@ifpackageloaded{#2}{-}{%else
4950       \PackageError\bxjx@pkgname
4951         {Package '#2' must be loaded}%
4952         {Package loading is aborted.\MessageBreak\@ehc}%
4953       \endinput}
4954   \fi}
4955 \bxjx@tmpdo{p}{bxcjkatype}
4956 \bxjx@tmpdo{x}{xeCJK}
4957 \bxjx@tmpdo{l}{luaTeXja}

```

古い L^AT_EX の場合、`\TextOrMath` は `fixltx2e` パッケージで提供される。

```

4958 \ifx\TextOrMath\@undefined
4959   \RequirePackage{fixltx2e}
4960 \fi

```

H.2 和文カテゴリコードの設定

upL^AT_EX の場合、和文カテゴリコードの設定を LuaT_EX-ja と（ほぼ）等価なものに変更
 する。

※ LuaT_EX-ja との相違点：A830、A960、1B000。

```

4961 \if u\bxjx@engine
4962 \@for\bxjx@tmpa:={%
4963 0080,0100,0180,0250,02B0,0300,0500,0530,0590,0600,%
4964 0700,0750,0780,07C0,0800,0840,0860,08A0,0900,0980,%
4965 0A00,0A80,0B00,0B80,0C00,0C80,0D00,0D80,0E00,0E80,%

```

```

4966 0F00,1000,10A0,1200,1380,13A0,1400,1680,16A0,1700,%
4967 1720,1740,1760,1780,1800,18B0,1900,1950,1980,19E0,%
4968 1A00,1A20,1AB0,1B00,1B80,1BC0,1C00,1C50,1C80,1CC0,%
4969 1CD0,1D00,1D80,1DC0,1E00,1F00,2440,27C0,27F0,2800,%
4970 2A00,2C00,2C60,2C80,2D00,2D30,2D80,2DE0,2E00,4DC0,%
4971 A4D0,A500,A640,A6A0,A700,A720,A800,A830,A840,A880,%
4972 A8E0,A900,A930,A980,A9E0,AA00,AA60,AA80,AAE0,AB00,%
4973 AB30,AB70,ABC0,DB80,DC00,E000,FB00,FB50,FE00,%
4974 FE70,FFF0,%
4975 10000,10080,10100,10140,10190,101D0,10280,102A0,%
4976 102E0,10300,10330,10350,10380,103A0,10400,10450,%
4977 10480,104B0,10500,10530,10600,10800,10840,10860,%
4978 10880,108E0,10900,10920,10980,109A0,10A00,10A60,%
4979 10A80,10AC0,10B00,10B40,10B60,10B80,10C00,10C80,%
4980 10E60,11000,11080,110D0,11100,11150,11180,111E0,%
4981 11200,11280,112B0,11300,11400,11480,11580,11600,%
4982 11660,11680,11700,118A0,11A00,11A50,11AC0,11C00,%
4983 11C70,11D00,12000,12400,12480,13000,14400,16800,%
4984 16A40,16AD0,16B00,16F00,1BC00,1BCA0,1D000,1D100,%
4985 1D200,1D300,1D360,1D400,1D800,1E000,1E800,1E900,%
4986 1EE00,1F000,1F030,1F0A0,1F300,1F600,1F650,1F680,%
4987 1F700,1F780,1F800,1F900,E0000,E0100,F0000,100000,%
4988 00C0%
4989 }\do{%
4990 \@tempcnta="\bxjx@tmpa\relax
4991 \@tempcntb\@tempcnta \advance\@tempcntb\m@ne
4992 \chardef\bxjx@tmpb\kcatcode\@tempcntb
4993 \kcatcode\@tempcnta=15 \kcatcode\@tempcntb\bxjx@tmpb}
4994 \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。

```

4995 \@onlypreamble\bxjx@grkcyr@list
4996 \def\bxjx@grkcyr@list{%
4997 \do{0391}{LGR}{\textAlpha}{A}%           % GR. C. L. ALPHA
4998 \do{0392}{LGR}{\textBeta}{B}%           % GR. C. L. BETA
4999 \do{0393}{LGR}{\textGamma}{\Gamma}%     % GR. C. L. GAMMA

```

5000	$\backslash\mathrm{do}\{0394\}\{\mathrm{LGR}\}\{\backslash\mathrm{textDelta}\}\{\backslash\Delta\}\%$	% GR. C. L. DELTA
5001	$\backslash\mathrm{do}\{0395\}\{\mathrm{LGR}\}\{\backslash\mathrm{textEpsilon}\}\{\backslash\mathrm{E}\}\%$	% GR. C. L. EPSILON
5002	$\backslash\mathrm{do}\{0396\}\{\mathrm{LGR}\}\{\backslash\mathrm{textZeta}\}\{\backslash\mathrm{Z}\}\%$	% GR. C. L. ZETA
5003	$\backslash\mathrm{do}\{0397\}\{\mathrm{LGR}\}\{\backslash\mathrm{textEta}\}\{\backslash\mathrm{H}\}\%$	% GR. C. L. ETA
5004	$\backslash\mathrm{do}\{0398\}\{\mathrm{LGR}\}\{\backslash\mathrm{textTheta}\}\{\backslash\mathrm{Theta}\}\%$	% GR. C. L. THETA
5005	$\backslash\mathrm{do}\{0399\}\{\mathrm{LGR}\}\{\backslash\mathrm{textIota}\}\{\backslash\mathrm{I}\}\%$	% GR. C. L. IOTA
5006	$\backslash\mathrm{do}\{039A\}\{\mathrm{LGR}\}\{\backslash\mathrm{textKappa}\}\{\backslash\mathrm{K}\}\%$	% GR. C. L. KAPPA
5007	$\backslash\mathrm{do}\{039B\}\{\mathrm{LGR}\}\{\backslash\mathrm{textLambda}\}\{\backslash\mathrm{Lambda}\}\%$	% GR. C. L. LAMDA
5008	$\backslash\mathrm{do}\{039C\}\{\mathrm{LGR}\}\{\backslash\mathrm{textMu}\}\{\backslash\mathrm{M}\}\%$	% GR. C. L. MU
5009	$\backslash\mathrm{do}\{039D\}\{\mathrm{LGR}\}\{\backslash\mathrm{textNu}\}\{\backslash\mathrm{N}\}\%$	% GR. C. L. NU
5010	$\backslash\mathrm{do}\{039E\}\{\mathrm{LGR}\}\{\backslash\mathrm{textXi}\}\{\backslash\mathrm{Xi}\}\%$	% GR. C. L. XI
5011	$\backslash\mathrm{do}\{039F\}\{\mathrm{LGR}\}\{\backslash\mathrm{textOmicron}\}\{\backslash\mathrm{O}\}\%$	% GR. C. L. OMICRON
5012	$\backslash\mathrm{do}\{03A0\}\{\mathrm{LGR}\}\{\backslash\mathrm{textPi}\}\{\backslash\mathrm{Pi}\}\%$	% GR. C. L. PI
5013	$\backslash\mathrm{do}\{03A1\}\{\mathrm{LGR}\}\{\backslash\mathrm{textRho}\}\{\backslash\mathrm{P}\}\%$	% GR. C. L. RHO
5014	$\backslash\mathrm{do}\{03A3\}\{\mathrm{LGR}\}\{\backslash\mathrm{textSigma}\}\{\backslash\mathrm{Sigma}\}\%$	% GR. C. L. SIGMA
5015	$\backslash\mathrm{do}\{03A4\}\{\mathrm{LGR}\}\{\backslash\mathrm{textTau}\}\{\backslash\mathrm{T}\}\%$	% GR. C. L. TAU
5016	$\backslash\mathrm{do}\{03A5\}\{\mathrm{LGR}\}\{\backslash\mathrm{textUpsilon}\}\{\backslash\mathrm{Upsilon}\}\%$	% GR. C. L. UPSILON
5017	$\backslash\mathrm{do}\{03A6\}\{\mathrm{LGR}\}\{\backslash\mathrm{textPhi}\}\{\backslash\mathrm{Phi}\}\%$	% GR. C. L. PHI
5018	$\backslash\mathrm{do}\{03A7\}\{\mathrm{LGR}\}\{\backslash\mathrm{textChi}\}\{\backslash\mathrm{X}\}\%$	% GR. C. L. CHI
5019	$\backslash\mathrm{do}\{03A8\}\{\mathrm{LGR}\}\{\backslash\mathrm{textPsi}\}\{\backslash\mathrm{Psi}\}\%$	% GR. C. L. PSI
5020	$\backslash\mathrm{do}\{03A9\}\{\mathrm{LGR}\}\{\backslash\mathrm{textOmega}\}\{\backslash\mathrm{Omega}\}\%$	% GR. C. L. OMEGA
5021	$\backslash\mathrm{do}\{03B1\}\{\mathrm{LGR}\}\{\backslash\mathrm{textalpha}\}\{\backslash\mathrm{alpha}\}\%$	% GR. S. L. ALPHA
5022	$\backslash\mathrm{do}\{03B2\}\{\mathrm{LGR}\}\{\backslash\mathrm{textbeta}\}\{\backslash\mathrm{beta}\}\%$	% GR. S. L. BETA
5023	$\backslash\mathrm{do}\{03B3\}\{\mathrm{LGR}\}\{\backslash\mathrm{textgamma}\}\{\backslash\mathrm{gamma}\}\%$	% GR. S. L. GAMMA
5024	$\backslash\mathrm{do}\{03B4\}\{\mathrm{LGR}\}\{\backslash\mathrm{textdelta}\}\{\backslash\mathrm{delta}\}\%$	% GR. S. L. DELTA
5025	$\backslash\mathrm{do}\{03B5\}\{\mathrm{LGR}\}\{\backslash\mathrm{textepsilon}\}\{\backslash\mathrm{epsilon}\}\%$	% GR. S. L. EPSILON
5026	$\backslash\mathrm{do}\{03B6\}\{\mathrm{LGR}\}\{\backslash\mathrm{textzeta}\}\{\backslash\mathrm{zeta}\}\%$	% GR. S. L. ZETA
5027	$\backslash\mathrm{do}\{03B7\}\{\mathrm{LGR}\}\{\backslash\mathrm{texteta}\}\{\backslash\mathrm{eta}\}\%$	% GR. S. L. ETA
5028	$\backslash\mathrm{do}\{03B8\}\{\mathrm{LGR}\}\{\backslash\mathrm{texttheta}\}\{\backslash\mathrm{theta}\}\%$	% GR. S. L. THETA
5029	$\backslash\mathrm{do}\{03B9\}\{\mathrm{LGR}\}\{\backslash\mathrm{textiota}\}\{\backslash\mathrm{iota}\}\%$	% GR. S. L. IOTA
5030	$\backslash\mathrm{do}\{03BA\}\{\mathrm{LGR}\}\{\backslash\mathrm{textkappa}\}\{\backslash\mathrm{kappa}\}\%$	% GR. S. L. KAPPA
5031	$\backslash\mathrm{do}\{03BB\}\{\mathrm{LGR}\}\{\backslash\mathrm{textlambda}\}\{\backslash\mathrm{lambda}\}\%$	% GR. S. L. LAMDA
5032	$\backslash\mathrm{do}\{03BC\}\{\mathrm{LGR}\}\{\backslash\mathrm{textmu}\}\{\backslash\mathrm{mu}\}\%$	% GR. S. L. MU
5033	$\backslash\mathrm{do}\{03BD\}\{\mathrm{LGR}\}\{\backslash\mathrm{textnu}\}\{\backslash\mathrm{nu}\}\%$	% GR. S. L. NU
5034	$\backslash\mathrm{do}\{03BE\}\{\mathrm{LGR}\}\{\backslash\mathrm{textxi}\}\{\backslash\mathrm{xi}\}\%$	% GR. S. L. XI
5035	$\backslash\mathrm{do}\{03BF\}\{\mathrm{LGR}\}\{\backslash\mathrm{textomicron}\}\{\backslash\mathrm{o}\}\%$	% GR. S. L. OMICRON
5036	$\backslash\mathrm{do}\{03C0\}\{\mathrm{LGR}\}\{\backslash\mathrm{textpi}\}\{\backslash\mathrm{pi}\}\%$	% GR. S. L. PI
5037	$\backslash\mathrm{do}\{03C1\}\{\mathrm{LGR}\}\{\backslash\mathrm{textrho}\}\{\backslash\mathrm{rho}\}\%$	% GR. S. L. RHO
5038	$\backslash\mathrm{do}\{03C2\}\{\mathrm{LGR}\}\{\backslash\mathrm{textvarsigma}\}\{\backslash\mathrm{varsigma}\}\%$	% GR. S. L. FINAL SIGMA
5039	$\backslash\mathrm{do}\{03C3\}\{\mathrm{LGR}\}\{\backslash\mathrm{textsigma}\}\{\backslash\mathrm{sigma}\}\%$	% GR. S. L. SIGMA
5040	$\backslash\mathrm{do}\{03C4\}\{\mathrm{LGR}\}\{\backslash\mathrm{texttau}\}\{\backslash\mathrm{tau}\}\%$	% GR. S. L. TAU
5041	$\backslash\mathrm{do}\{03C5\}\{\mathrm{LGR}\}\{\backslash\mathrm{textupsilon}\}\{\backslash\mathrm{upsilon}\}\%$	% GR. S. L. UPSILON
5042	$\backslash\mathrm{do}\{03C6\}\{\mathrm{LGR}\}\{\backslash\mathrm{textphi}\}\{\backslash\mathrm{phi}\}\%$	% GR. S. L. PHI
5043	$\backslash\mathrm{do}\{03C7\}\{\mathrm{LGR}\}\{\backslash\mathrm{textchi}\}\{\backslash\mathrm{chi}\}\%$	% GR. S. L. CHI
5044	$\backslash\mathrm{do}\{03C8\}\{\mathrm{LGR}\}\{\backslash\mathrm{textpsi}\}\{\backslash\mathrm{psi}\}\%$	% GR. S. L. PSI
5045	$\backslash\mathrm{do}\{03C9\}\{\mathrm{LGR}\}\{\backslash\mathrm{textomega}\}\{\backslash\mathrm{omega}\}\%$	% GR. S. L. OMEGA
5046	$\backslash\mathrm{do}\{0401\}\{\mathrm{T2A}\}\{\backslash\mathrm{CYRYO}\}\{\backslash\}\%$	% CY. C. L. IO
5047	$\backslash\mathrm{do}\{0410\}\{\mathrm{T2A}\}\{\backslash\mathrm{CYRA}\}\{\backslash\}\%$	% CY. C. L. A
5048	$\backslash\mathrm{do}\{0411\}\{\mathrm{T2A}\}\{\backslash\mathrm{CYRB}\}\{\backslash\}\%$	% CY. C. L. BE

5049 \do{0412}{T2A}{\CYRV}{}%	% CY. C. L. VE
5050 \do{0413}{T2A}{\CYRG}{}%	% CY. C. L. GHE
5051 \do{0414}{T2A}{\CYRD}{}%	% CY. C. L. DE
5052 \do{0415}{T2A}{\CYRE}{}%	% CY. C. L. IE
5053 \do{0416}{T2A}{\CYRZH}{}%	% CY. C. L. ZHE
5054 \do{0417}{T2A}{\CYRZ}{}%	% CY. C. L. ZE
5055 \do{0418}{T2A}{\CYRI}{}%	% CY. C. L. I
5056 \do{0419}{T2A}{\CYRISHRT}{}%	% CY. C. L. SHORT I
5057 \do{041A}{T2A}{\CYRK}{}%	% CY. C. L. KA
5058 \do{041B}{T2A}{\CYRL}{}%	% CY. C. L. EL
5059 \do{041C}{T2A}{\CYRM}{}%	% CY. C. L. EM
5060 \do{041D}{T2A}{\CYRN}{}%	% CY. C. L. EN
5061 \do{041E}{T2A}{\CYRO}{}%	% CY. C. L. O
5062 \do{041F}{T2A}{\CYRP}{}%	% CY. C. L. PE
5063 \do{0420}{T2A}{\CYRR}{}%	% CY. C. L. ER
5064 \do{0421}{T2A}{\CYRS}{}%	% CY. C. L. ES
5065 \do{0422}{T2A}{\CYRT}{}%	% CY. C. L. TE
5066 \do{0423}{T2A}{\CYRU}{}%	% CY. C. L. U
5067 \do{0424}{T2A}{\CYRF}{}%	% CY. C. L. EF
5068 \do{0425}{T2A}{\CYRH}{}%	% CY. C. L. HA
5069 \do{0426}{T2A}{\CYRC}{}%	% CY. C. L. TSE
5070 \do{0427}{T2A}{\CYRCH}{}%	% CY. C. L. CHE
5071 \do{0428}{T2A}{\CYRSH}{}%	% CY. C. L. SHA
5072 \do{0429}{T2A}{\CYRSHCH}{}%	% CY. C. L. SHCHA
5073 \do{042A}{T2A}{\CYRHRDSN}{}%	% CY. C. L. HARD SIGN
5074 \do{042B}{T2A}{\CYRERY}{}%	% CY. C. L. YERU
5075 \do{042C}{T2A}{\CYRSFTSN}{}%	% CY. C. L. SOFT SIGN
5076 \do{042D}{T2A}{\CYREREV}{}%	% CY. C. L. E
5077 \do{042E}{T2A}{\CYRYU}{}%	% CY. C. L. YU
5078 \do{042F}{T2A}{\CYRYA}{}%	% CY. C. L. YA
5079 \do{0430}{T2A}{\cyra}{}%	% CY. S. L. A
5080 \do{0431}{T2A}{\cyrb}{}%	% CY. S. L. BE
5081 \do{0432}{T2A}{\cyrv}{}%	% CY. S. L. VE
5082 \do{0433}{T2A}{\cyrg}{}%	% CY. S. L. GHE
5083 \do{0434}{T2A}{\cyrd}{}%	% CY. S. L. DE
5084 \do{0435}{T2A}{\cyre}{}%	% CY. S. L. IE
5085 \do{0436}{T2A}{\cyrzh}{}%	% CY. S. L. ZHE
5086 \do{0437}{T2A}{\cyrz}{}%	% CY. S. L. ZE
5087 \do{0438}{T2A}{\cyri}{}%	% CY. S. L. I
5088 \do{0439}{T2A}{\cyrishrt}{}%	% CY. S. L. SHORT I
5089 \do{043A}{T2A}{\cyrk}{}%	% CY. S. L. KA
5090 \do{043B}{T2A}{\cyrl}{}%	% CY. S. L. EL
5091 \do{043C}{T2A}{\cyrm}{}%	% CY. S. L. EM
5092 \do{043D}{T2A}{\cyrn}{}%	% CY. S. L. EN
5093 \do{043E}{T2A}{\cyro}{}%	% CY. S. L. O
5094 \do{043F}{T2A}{\cyrp}{}%	% CY. S. L. PE
5095 \do{0440}{T2A}{\cyrr}{}%	% CY. S. L. ER
5096 \do{0441}{T2A}{\cyrs}{}%	% CY. S. L. ES
5097 \do{0442}{T2A}{\cyrt}{}%	% CY. S. L. TE

```

5098 \do{0443}{T2A}{\cyru}{}% % CY. S. L. U
5099 \do{0444}{T2A}{\cyrf}{}% % CY. S. L. EF
5100 \do{0445}{T2A}{\cyrh}{}% % CY. S. L. HA
5101 \do{0446}{T2A}{\cyrc}{}% % CY. S. L. TSE
5102 \do{0447}{T2A}{\cyrch}{}% % CY. S. L. CHE
5103 \do{0448}{T2A}{\cyrrh}{}% % CY. S. L. SHA
5104 \do{0449}{T2A}{\cyrrhch}{}% % CY. S. L. SHCHA
5105 \do{044A}{T2A}{\cyrrhdsn}{}% % CY. S. L. HARD SIGN
5106 \do{044B}{T2A}{\cyrrery}{}% % CY. S. L. YERU
5107 \do{044C}{T2A}{\cyrsftsn}{}% % CY. S. L. SOFT SIGN
5108 \do{044D}{T2A}{\cyrrerev}{}% % CY. S. L. E
5109 \do{044E}{T2A}{\cyryu}{}% % CY. S. L. YU
5110 \do{044F}{T2A}{\cyrya}{}% % CY. S. L. YA
5111 \do{0451}{T2A}{\cyryo}{}% % CY. S. L. IO
5112 \do{00A7}{TS1}{\textsection}{\mathsection}% SECTION SYMBOL
5113 \do{00A8}{TS1}{\textasciidieresis}{}% % DIAERESIS
5114 \do{00B0}{TS1}{\textdegree}{\mathdegree}% % DEGREE SIGN
5115 \do{00B1}{TS1}{\textpm}{\pm}% % PLUS-MINUS SIGN
5116 \do{00B4}{TS1}{\textasciicute}{}% % ACUTE ACCENT
5117 \do{00B6}{TS1}{\textparagraph}{\mathparagraph}% PILCROW SIGN
5118 \do{00D7}{TS1}{\texttimes}{\times}% % MULTIPLICATION SIGN
5119 \do{00F7}{TS1}{\textdiv}{\div}% % DIVISION SIGN
5120 }

```

`\mathdegree` 面倒なので補っておく。

```
5121 \providecommand*\mathdegree{{}^\circ}
```

`\ifbxjx@gcc@cjkl` [スイッチ]「特定 CJK 曖昧文字」を和文扱いにするか。

```
5122 \newif\ifbxjx@gcc@cjkl
```

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

```
5123 \newcommand*\greekasCJK{%
```

```
5124 \bxjx@gcc@cjkltrue}
```

`\nogreekasCJK` [公開命令]「特定 CJK 曖昧文字」を欧文扱いにする。

```
5125 \newcommand*\nogreekasCJK{%
```

```
5126 \bxjx@gcc@cjklfalse}
```

`\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。

```
5127 \def\bxjx@tmpdo#1\relax{%
```

```
5128 \def\bxjx@fake@grk##1##2{%
```

```
5129 \expandafter\bxjx@fake@grk@a\meaning##2#1\@nil{##1}{##2}}%
```

```

5130 \def\bxjx@fake@grk@a##1#1##2\@nil##3##4{%
5131 \ifx\\##1\\%
5132 \bxjx@canta##4\divide\bxjx@canta\@cclvi
5133 \multiply\bxjx@canta\@cclvi \advance\bxjx@canta`##3\relax
5134 \mathchar\bxjx@canta
5135 \else ##3\fi}
5136 }\expandafter\bxjx@tmpdo\string\mathchar\relax

```

■pdfTeX・upTeX の場合

```

5137 \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^AT_EX においても、必ず「`inputenc` が明示的に読み込まれた」状態になる。

```

5138 \@ifpackageloaded{inputenc}{\fi}{%else
5139 \RequirePackage[utf8]{inputenc}}
5140 \def\bxjx@tmpa{utf8}
5141 \ifx\bxjx@tmpa\inputencdoingname
5142 \PackageWarningNoLine\bxjx@pkgnam
5143 {Input encoding changed to utf8}%
5144 \inputencoding{utf8}%
5145 \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。

```

5146 \if u\bxjx@engine
5147 \kcatcode"0370=15
5148 \kcatcode"0400=15
5149 \kcatcode"0500=15
5150 \fi

```

各文字について `\DeclareUnicodeCharacter` を実行する。

```

5151 \def\bxjx@tmpdo#1{%
5152 \@tempcnta="#1\relax
5153 \expandafter\bxjx@tmpdo@a\csname bxjx@KC/\the\@tempcnta\endcsname{#1}}
5154 \def\bxjx@tmpdo@a#1#2#3#4#5{%

```

引数 = `\[bxjx@KC/⟨符号値⟩]{⟨符号値⟩}{⟨fontenc⟩}{⟨LICR⟩}{⟨数式 LICR⟩}`

“数式中の動作”を決定する。⟨数式 LICR⟩ が空（数式非対応）なら警告を出す。

```

5155 \ifx\\#5\\%
5156 \def\bxjx@tmpa{\@inmathwarn#4}%

```

⟨数式 LICR⟩ が英字である場合は `\bxjx@fake@grk` で出力する。大文字なら `\Pi`、小文字なら `\pi` を基準文字にする。

```

5157 \else\ifcat A\noexpand#5%

```



```

5158 \edef\bxjx@tmpa{\noexpand\bxjx@fake@grk{#5}%
5159 {\ifnum\uccode`#5=`#5\noexpand\Pi\else\noexpand\pi\fi}}%

```

それ以外は〈数式 LICR〉をそのまま実行する。

```

5160 \else \def\bxjx@tmpa{#5}%
5161 \fi\fi
5162 \def\bxjx@tmpb{\bxjx@tmpdo@b{#1}{#2}{#3}{#4}}%
5163 \expandafter\bxjx@tmpb\expandafter{\bxjx@tmpa}

```

以降はエンジン種別で分岐する。upTeX の場合。

```

5164 \if u\bxjx@engine
5165 \def\bxjx@tmpdo@b#1#2#3#4#5{%

```

引数 = $\backslash\text{bxjx@KC}/\langle\text{符号値}\rangle\{\langle\text{符号値}\rangle\}\{\langle\text{fontenc}\rangle\}\{\langle\text{LICR}\rangle\}\{\langle\text{数式中の動作}\rangle\}$

当該の Unicode 文字の動作は「テキストでは〈LICR〉、数式では〈数式中の動作〉」となる。

LICR は現在エンコーディングで有効な定義がある場合はそれが実行されるはずである。(つまり、現在が LGR である場合はギリシャ文字は常に欧文扱いになる。) それ以外の場合は LICR を $\backslash\text{bxjx@ja@or@not}$ に帰着させる。この際に、和文用の定義として当該の `kchardef` を使用し、その制御綴として $\backslash\text{bxjx@KC}/\dots$ を流用している。

```

5166 \kchardef#1=\@tempcnta
5167 \DeclareTextCommandDefault{#4}{\bxjx@ja@or@not{#1}{#3}{#4}}%
5168 \DeclareUnicodeCharacter{#2}{\TextOrMath{#4}{#5}}

```

pdfTeX の場合も処理はほとんど同じ。ただし、和文用の定義として $\backslash\text{UTF}\{\langle\text{符号値}\rangle\}$ を使う ($\backslash\text{UTF}$ は `bxCJKatype` の命令)。 $\backslash\text{bxjx@KC}/\dots$ は使わないが定義済にする必要がある。

```

5169 \else\if p\bxjx@engine
5170 \def\bxjx@tmpdo@b#1#2#3#4#5{%
5171 \mathchardef#1=\@tempcnta
5172 \DeclareTextCommandDefault{#4}{\bxjx@ja@or@not{\UTF{#2}}{#3}{#4}}%
5173 \DeclareUnicodeCharacter{#2}{\TextOrMath{#4}{#5}}
5174 \fi\fi

```

以上の処理を「特定 CJK 曖昧文字」の各々に適用する。

```

5175 \let\do\bxjx@tmpdo \bxjx@grkcyr@list

```

$\backslash\text{bxjx@DeclareUnicodeCharacter}$ $\backslash\text{bxjx@DeclareUnicodeCharacter}$ 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。

```

5176 \@onlypreamble\bxjx@org@DeclareUnicodeCharacter
5177 \let\bxjx@org@DeclareUnicodeCharacter\DeclareUnicodeCharacter
5178 \@onlypreamble\bxjx@DeclareUnicodeCharacter
5179 \def\bxjx@DeclareUnicodeCharacter#1#2{%
5180 \count@=#1\relax
5181 \expandafter\ifx\csname bxjx@KC/\the\count@\endcsname\relax
5182 \bxjx@org@DeclareUnicodeCharacter{#1}{#2}%
5183 \else
5184 \wlog{ \space\space skipped defining Unicode char U+#1}%
5185 \fi}

```

$\backslash\text{bxjx@ja@or@not}$ $\backslash\text{bxjx@ja@or@not}\{\langle\text{和文用定義}\rangle\}\{\langle\text{対象 fontenc}\rangle\}\{\langle\text{LICR}\rangle\}$: $\backslash\text{[no]greekasCJK}$ の状態

に応じて和文または欧文で文字を出力する。

```
5186 \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。

```
5187 \ifbxjx@gcc@cj k #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。

```
5188 \else\expandafter\ifx\csname T@#2\endcsname\relax #1%
```

```
5189 \else \UseTextSymbol{#2}{#3}%
```

```
5190 \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。

```
5191 \begingroup
```

```
5192 \toks@\expandafter{\DeclareFontEncoding@{#1}{#2}{#3}}
```

```
5193 \xdef\next{\def\noexpand\DeclareFontEncoding@##1##2##3{%
```

```
5194 \noexpand\bxjx@swap@DUC@cmd
```

```
5195 \the\toks@
```

```
5196 \noexpand\bxjx@swap@DUC@cmd}}}
```

```
5197 \endgroup\next
```

```
5198 \def\bxjx@swap@DUC@cmd{%
```

```
5199 \let\bxjx@tmpa\DeclareUnicodeCharacter
```

```
5200 \let\DeclareUnicodeCharacter\bxjx@DeclareUnicodeCharacter
```

```
5201 \let\bxjx@DeclareUnicodeCharacter\bxjx@tmpa
```

```
5202 \let\bxjx@tmpa\relax}
```

以上。

■X_YTeX・LuaTeX の場合

```
5203 \else\ifnum0\if x\bxjx@engine1\fi\if 1\bxjx@engine1\fi>0
```

各文字について、数式中の動作を定義する。

```
5204 \def\bxjx@tmpdo#1{%
```

```
5205 \bxjx@cmta="#1\relax
```

```
5206 \begingroup
```

```
5207 \lccode`~=\bxjx@cmta
```

```
5208 \lowercase{\endgroup
```

```
5209 \bxjx@tmpdo@a{~}}{#1}}
```

```
5210 \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、up¹LaTeX 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。

```
5211 \ifx\\#5\\\let\bxjx@tmpa\relax
```

```
5212 \else\ifcat A\noexpand#5%
```

```
5213 \edef\bxjx@tmpa{\noexpand\bxjx@fake@grk{#5}%
```

```
5214 {\ifnum\uccode`#5=#5\noexpand\Pi\else\noexpand\pi\fi}}%
```

```
5215 \else \def\bxjx@tmpa{#5}%
```

```
5216 \fi\fi
```

```

5217 \ifx\bxjx@tmpa\relax\else
5218 \mathcode\bxjx@cmta"8000 \let#1\bxjx@tmpa
5219 \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。

```

5220 \mathchardef\bxjx@tmpa="119
5221 \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_T_EX の場合は、Lua_T_EX-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。

```

5222 \if 1\bxjx@engine
5223 \protected\def\greekasCJK{%
5224 \bxjx@gcc@cjctrue
5225 \ltjsetparameter{jacharrange={+2, +8}}}
5226 \protected\def\nogreekasCJK{%
5227 \bxjx@gcc@cjcfalse
5228 \ltjsetparameter{jacharrange={-2, -8}}}
5229 \fi

```

X_ƎT_EX の場合、`xeCJK` は X_ƎT_EX 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。

```

5230 \if x\bxjx@engine
5231 \let\bxjx@gcc@cjkl@list\@empty
5232 \def\do#1#2#3#4{%
5233 \edef\bxjx@gcc@cjkl@list{\bxjx@gcc@cjkl@list
5234 \noexpand\XeTeXcharclass"#1\bxjx@cmta}}
5235 \bxjx@grkcyr@list
5236 \protected\def\greekasCJK{%
5237 \bxjx@gcc@cjctrue
5238 \bxjx@cmta=\@ne \bxjx@gcc@cjkl@list}
5239 \protected\def\nogreekasCJK{%
5240 \bxjx@gcc@cjcfalse
5241 \bxjx@cmta=\z@ \bxjx@gcc@cjkl@list}
5242 \fi

```

以上。

```

5243 \fi\fi

```

H.4 初期設定

「特定 CJK 曖昧文字」を欧文扱いにする。

```

5244 \nogreekasCJK

```

H.5 完了

おしまい。

```
5245 %</cjkat>
```

付録 I 補助パッケージ：bxjspandoc 🍷

Pandoc の L^AT_EX 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_EX 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

```
5246 %<*ancpandoc>
```

```
5247 %% このファイルは日本語文字を含みます.
```

```
5248 \def\bxjsp@pkgname{bxjscjkat}
```

\bxjsp@engine エンジンの種別。

```
5249 \let\bxjsp@engine=n
```

```
5250 \@onlypreamble\bxjsp@do
```

```
5251 \def\bxjsp@do#1#2{%
```

```
5252   \edef\bxjsp@tmpa{\string#1}%
```

```
5253   \edef\bxjsp@tmpb{\meaning#1}%
```

```
5254   \ifx\bxjsp@tmpa\bxjsp@tmpb #2\fi}
```

```
5255 \bxjsp@do\kanjiskip{\let\bxjsp@engine=j}
```

```
5256 \bxjsp@do\XeTeXversion{\let\bxjsp@engine=x}
```

```
5257 \bxjsp@do\pdftexversion{\let\bxjsp@engine=p}
```

```
5258 \bxjsp@do\luatexversion{\let\bxjsp@engine=l}
```

\bxjsp@begin@document@hook 文書本体開始時フック。

```
5259 \@onlypreamble\bxjsp@begin@document@hook
```

```
5260 \let\bxjsp@begin@document@hook\@empty
```

```
5261 \AtBeginDocument{\bxjsp@begin@document@hook}
```

\ifbxjsp@babel@used [スイッチ] Babel が読み込まれたか。

```
5262 \newif\ifbxjsp@babel@used
```

```
5263 \g@addto@macro\bxjsp@begin@document@hook{%
```

```
5264   \@ifpackageloaded{babel}{\bxjsp@babel@usedtrue}{}}
```

I.2 パッケージオプション

english オプションが指定されている場合、\ldots の調整を抑止する。

※つまり、「グローバルの `english` オプション」が指定されている場合も抑止の対象になる。
 BXJS クラスの英語モードを想定しているが、それ以外の場合でも、一般的な L^AT_EX 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。

```
5265 \newif\ifbxjsp@english
5266 \DeclareOption{english}{\bxjsp@englishttrue}

      オプション定義はおしまい。

5267 \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。

```
5268 \@onlypreamble\pandocSkipLoadFile
5269 \newcommand*\pandocSkipLoadFile[1]{%
5270   \expandafter\bxjsp@skip@load@file@a\csname ver@#1\endcsname{#1}}
5271 \def\bxjsp@skip@load@file@a#1#2{%
5272   \ifx#1\relax
5273     \def#1{2001/01/01}%
5274     \PackageInfo{bxjsp@pkgname
5275       {File '#2' marked as loaded\@gobble}}%
5276   \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。

```
5277 \@onlypreamble\pandocSkipLoadPackage
5278 \newcommand*\pandocSkipLoadPackage[1]{%
5279   \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^AT_EX
 ではこれで警告が出る。これを抑止する。

L^AT_EX カーネルが新しい場合は `fixltx2e` を読込済にする。

```
5280 \ifx\@IncludeInRelease\@undefined\else
5281   \pandocSkipLoadPackage{fixltx2e}
5282 \fi
```

1.5 cmap パッケージ

エンジンが (u)pL^AT_EX 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。)

```
5283 \if j\bxjsp@engine
5284   \pandocSkipLoadPackage{cmap}
5285 \fi
```

I.6 microtype パッケージ

警告が多すぎなので消す。

```
5286 \if j\bxjsp@engine \else
5287   \PassOptionsToPackage{verbose=silent}{microtype}
5288 \fi
```

エンジンが (u)pL^AT_EX 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。

```
5289 \if j\bxjsp@engine
5290   \pandocSkipLoadPackage{microtype}
5291   \newcommand*\UseMicrotypeSet[2][]{ }
5292 \fi
```

I.7 Unicode 文字変換対策

Pandoc で L^AT_EX 形式に書き出す場合は、元データ中の一部の Unicode 文字を「L^AT_EX 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^AT_EX では「L^AT_EX 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。

```
5293 \DeclareRobustCommand{\pandocLdots}{%
5294   \let\bxjsp@do\bxjsp@ja@ellipsis
5295   \ifmmode \let\bxjsp@do\bxjsp@org@ldots
5296   \else\ifbxjsp@babel@used
5297     \expandafter\ifx\csname bxjsp@ld/\language\endcsname\relax
5298     \let\bxjsp@do\bxjsp@org@ldots \fi
5299   \fi\fi \bxjsp@do}
5300 \@namedef{bxjsp@ld/japanese}{1}
5301 \def\bxjsp@ja@ellipsis{…}
5302 \let\bxjsp@org@ldots\ldots
```

`\ldots` の実装を `\pandocLdots` に置き換える。

```

5303 \g@addto@macro\bxjsp@begin@document@hook{%
5304   \let\bxjsp@org@ldots\ldots

```

もしここで `\newcommand\pandocLdots{\ldots}` という定義である場合は置き換えない。

```

5305   \long\def\bxjsp@tmpa{\ldots}%
5306   \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。

```

5307     \ifnum0\ifbxjsp@english\ifbxjsp@babel@used\else1\fi\fi=0
5308     \let\ldots\pandocLdots
5309   \fi
5310 \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 $X_{\text{T}}\text{E}_X \cdot \text{LuaT}_{\text{E}}X$ 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、非英字扱いにしておく。

※ Pandoc は「Unicode 7.0 で GC が Letter」な文字を英字と判定している。

```

5311 \chardef\bxjsp@cc@other=12
5312 \@onlypreamble\bxjsp@makeother@range
5313 \def\bxjsp@makeother@range#1#2{%
5314   \@tempcnta"#1\relax \@tempcntb"#2\relax
5315   \loop\ifnum\@tempcnta<\@tempcntb
5316     \catcode\@tempcnta\bxjsp@cc@other
5317     \advance\@tempcnta\@ne
5318   \repeat}
5319 \ifnum0\if x\bxjsp@engine1\fi\if 1\bxjsp@engine1\fi>0
5320   \catcode"1F23B=\bxjsp@cc@other
5321   \bxjsp@makeother@range{9FCD}{A000}
5322   \bxjsp@makeother@range{1B002}{1B170}
5323   \bxjsp@makeother@range{2B820}{2EBF0}
5324 \fi

```

I.8 PandoLa モジュール

インストール済であれば読み込む。

```

5325 \IfFileExists{bxpandola.sty}{%
5326   \RequirePackage{bxpandola}\relax
5327   \PackageInfo\bxjsp@pkgname
5328   {PandoLa module is loaded\@gobble}
5329 }{}

```

I.9 完了

おしまい。

```

5330 %</ancpandoc>

```

補助パッケージ実装はここまで。

5331 %</anc>