

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”)

v2.9 [2023/07/17]

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	4
2	オプション	13
3	和文フォントの変更	44
4	フォントサイズ	54
5	レイアウト	60
5.1	ページレイアウト	61
6	改ページ (日本語 TeX 開発コミュニティ版のみ)	75
7	ページスタイル	77
8	文書のマークアップ	80
8.1	表題	80
8.2	章・節	85
8.3	リスト環境	98
8.4	パラメータの設定	105
8.5	フロート	106
8.6	キャプション	108
9	フォントコマンド	109

10	相互参照	112
10.1	目次の類	112
10.2	参考文献	117
10.3	索引	119
10.4	脚注	120
11	段落の頭へのグルー挿入禁止	123
12	いろいろなロゴ	127
13	amsmath との衝突の回避	127
14	初期設定	128
15	実験的コード	132
付録 A	和文ドライバの仕様 	134
付録 B	和文ドライバ：minimal 	135
B.1	準備	135
B.2	(u)pTeX 用の設定	138
B.3	pdfTeX 用の処理	143
B.4	X _Y TeX 用の処理	143
B.5	後処理 (エンジン共通)	144
付録 C	和文ドライバ：standard 	147
C.1	準備	147
C.2	和文ドライバパラメタ	148
C.3	共通処理 (1)	149
C.4	pTeX 用設定	157
C.5	pdfTeX 用設定：CJK + bxcjkjatype	162
C.6	X _Y TeX 用設定：xeCJK + zxcjkatype	164
C.7	LuaTeX 用設定：LuaTeX-ja	166
C.8	共通処理 (2)	170
付録 D	和文ドライバ：modern 	171
D.1	フォント設定	171
D.2	fixltx2e 読込	172
D.3	和文カテゴリコード	172
D.4	完了	172
付録 E	和文ドライバ：pandoc 	172
E.1	準備	172

E.2	和文ドライバパラメタ	173
E.3	dupload システム	175
E.4	lang 変数	176
E.5	geometry 変数	179
E.6	CJKmainfont 変数	179
E.7	Option clash 対策	179
E.8	レイアウト上書き禁止	180
E.9	paragraph のマーク	181
E.10	全角空白文字	181
E.11	hyperref 対策	182
E.12	Pandoc 要素に対する和文用の補正	182
E.13	ifPDFTeX スイッチ	184
E.14	完了	185
付録 F	補助パッケージ一覧	185
付録 G	補助パッケージ：bxjscompat	185
G.1	準備	185
G.2	8bit 欧文 T _E X	186
G.3	X _g T _E X	186
G.4	LuaT _E X	187
G.5	完了	189
付録 H	補助パッケージ：bxjsjkat	189
H.1	準備	189
H.2	和文カテゴリコードの設定	190
H.3	ギリシャ・キリル文字の扱い	191
H.4	初期設定	198
H.5	完了	198
付録 I	補助パッケージ：bxjspandoc	198
I.1	準備	198
I.2	パッケージオプション	199
I.3	パッケージ読込の阻止	199
I.4	fixltx2e パッケージ	200
I.5	cmap パッケージ	200
I.6	microtype パッケージ	200
I.7	Unicode 文字変換対策	200
I.8	PandoLa モジュール	202
I.9	完了	202

1 はじめに

この文書は「BXJS ドキュメントクラス」の DocStrip 形式のソースである。BXJS ドキュメントクラス（以降では「BXJS クラス」と略称する）は奥村晴彦氏および日本語 $\text{T}_{\text{E}}\text{X}$ 開発コミュニティによる「 $\text{p}_{\text{L}}\text{T}_{\text{E}}\text{X}_{2_{\epsilon}}$ 新ドキュメントクラス」（以降では「JS クラス」と呼ぶ）に改変を加えたものである。

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

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

インストール時のモジュール指定は以下のものが用意されている。

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

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

これは $\text{L}^{\text{A}}\text{T}_{\text{E}}\text{X}3$ Project の `classes.dtx` と株式会社アスキーの `jclasses.dtx` に基づいてもともと奥村晴彦により作成されたものです。現在は日本語 $\text{T}_{\text{E}}\text{X}$ 開発コミュニティにより GitHub で管理されています。

<https://github.com/texjporg/jsclasses>

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

[2016-07-13] 日本語 $\text{T}_{\text{E}}\text{X}$ 開発コミュニティによる管理に移行しました。

[2009-02-22] 田中琢爾氏による $\text{u}_{\text{P}}\text{L}^{\text{A}}\text{T}_{\text{E}}\text{X}$ 対応パッチを取り込みました。

ここでは次のドキュメントクラス（スタイルファイル）を作ります。

[2017-02-13] forum:2121 の議論を機に、jsreport クラスを新設しました。従来の jsbook の report オプションと比べると、abstract 環境の使い方および挙動がアスキーの jreport に近づきました。

```

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

```

以下では実際のコードに即して説明します。

minijs は、jsclasses に似た設定を行うパッケージです。

```

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

```

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

```

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

```

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

```

16 %</class>
17 %<*class|minijs>
18 \newif\ifjsc@needsp@tch
19 \jsc@needsp@tchfalse
20 %</class|minijs>
21 %<*class>

```

■環境検査

\jsDocClass [トークン] 文書クラスの種別。以下の定値トークンの何れかと同値： \jsArticle = bxjsarticle、\jsBook = bxjsbook、\jsReport = bxjsreport、\jsSlide = bxjsslide。

```

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

```

`\bxjs@test@engine \bxjs@test@engine\制御綴{<コード>}` : `\制御綴` の意味が同名のプリミティブである場合にのみ `<コード>` を実行する。

```

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

```

`\jsEngine` [暗黙文字トークン] エンジン (T_EX 処理系) の種別 : `j` = pT_EX 系、`x` = X_ƎT_EX、`p` = pdfT_EX、`l` = LuaT_EX、`J` = NTT jT_EX、`0` = Omega 系、`n` = 以上の何れでもない。

※ pdfT_EX と LuaT_EX については DVI モードの場合も含む。

```

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

```

現状での処理系バージョン要件は以下の通りである。

- X_ƎT_EX : 0.997 版 (2007 年) 以上

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

3.0 版での**クラス本体**の処理系バージョン要件は以下の通りである。

- T_EX : 3.0 版 [1990/03] 以上
- pT_EX : 2.0 版 [1995/03] 以上
- upT_EX : 0.10 版 [2007/07] 以上
- pdfT_EX : 1.40 版 [2007/01] 以上
- LuaT_EX : 0.60 版 [2010/04] 以上
- X_ƎT_EX : 0.9994 版 [2009/06] 以上

※ Omega と NTT jT_EX は “公式にはサポート外” の扱い (動作は何も保証されない) であるが、クラス本体では処理系の種類は敢えて検査しないことにする。

※クラス本体での要件は敢えて緩くしている。標準和文ドライバ (minimal も含む) についてまた別に要件を定めるので、実質的にはそちらの要件を満たすことが求められる。

T_EX 処理系のバージョンがサポート対象であるかを検査する。

```
41 \@tempwatrue
```

```

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

```

非サポートのバージョン場合は強制終了させる。

```

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

```

万が一「2.09 互換モード」になっていた場合は、これ以上進むと危険なので強制終了させる。

```

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

```

`\bxjs@if@format@at@least` `\bxjs@if@format@at@least{<日付>}{<真>}{<偽>}`: L^AT_EX カーネルの版が指定の日付以降であるか。

```

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

```

`\bxjs@if@package@at@least` `\bxjs@if@package@at@least{<名前>}{<日付>}{<真>}{<偽>}`: その名前のパッケージの指定の日付以降の版が読み込まれているか。そもそも読み込まれていない場合は偽になる。
 ※ 2017/04/15 版より前のカーネルの `\@ifpackagelater` は非読込の場合に実行するとエラーになることに注意。

```

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

```

`\ifjsWithupTeX` [スイッチ] エンジンが「内部漢字コードが Unicode の up_TE_X」であるか。

※つまり、`\jsEngine = j` である場合、このスイッチが真なら up_LA_TE_X、偽なら p_LA_TE_X である。2023 年 6 月に p_LA_TE_X の T_EX 処理系が「 ε -p_TE_X」から「内部漢字コードが非 Unicode の ε -up_TE_X」に変わったが、これによる影響はない。

```

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

```

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

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

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

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

※ X \TeX と Lua \TeX は ϵ - \TeX 拡張をもつ版のみがあり、NTT $\mathcal{J}\TeX$ はもたない版のみがある。その他のエンジンは両方の版がある。

```
76 \newif\ifjsWitheTeX
77 \bxjs@test@engine\epsilonTeXversion{\jsWitheTeXtrue}
```

`\ifjsInPdfMode` [スイッチ] pdf \TeX ・Lua \TeX が PDF モードで動作しているか。

```
78 \newif\ifjsInPdfMode
79 \@nameuse{jsInPdfMode\ifnum0%
80 \ifx\pdfoutput\undefined\else\the\pdfoutput\fi
81 \ifx\outputmode\undefined\else\the\outputmode\fi
82 >0 true\else false\fi}
```

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

※ 2020/02/02 版以降のカーネルには組み込まれている。

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

`\ifbxjs@brace@safe` [スイッチ] オプション中の波括弧の使用にカーネルが対応しているか。

※正確に言うと、2021/06/01 版以降のカーネルでは「未使用オプション判定」の処理で = 以降のトークン列 (key-value の value の部分) を無視するので、この部分には波括弧を含めることができる。

※`\@removeelement` と `\in@` の実装は変更されておらず、これらのマクロの第 1 引数には波括弧を含むトークン列を指定できない。

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

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

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

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

`\ifbxjs@old@hook@system` [スイッチ] \LaTeX の新しいフック管理システムが**未導入**であるか。

※カーネルの 2020/10/01 版で導入された。

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

■依存パッケージ読込

長さ値の指定で式を利用可能にするため calc を読み込む。

```
94 \RequirePackage{calc}
```

クラスオプションで key-value 形式を使用するため keyval を読み込む。

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

PDF モードの判定を L^AT_EX 公式のパッケージに任せたいので、もし「iftex の \ifpdf」が利用できるならば、jsInPdfMode スイッチをその値に一致させる。

※ iftex で \ifpdf が利用できるのは 1.0 版 [2019/10/24] から。

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

クラスの本体ではこの他に以下のパッケージが読み込まれる。

```
geometry
```

また状況によっては以下のパッケージが読み込まれる可能性がある。

```
bxwareki、jslogo、plautopatch、type1cm
```

※和文ドライバがさらにパッケージを読み込むこともある。

`\jsAtEndOfClass` このクラスの読込終了時に対するフック。(補助パッケージ中で用いられる。)

```
104 \def\jsAtEndOfClass{%
105   \expandafter\g@addto@macro\csname\bxjs@clsname.cls-h@k\endcsname}
```

互換性のための補助パッケージを読み込む。

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

■BXJS クラス特有の設定

Lua_TE_X の場合、本クラス用の Lua モジュールを用意する。

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

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

```

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

```

`\bxjs@robust@def` 無引数の頑強な命令を定義する。

```

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

```

`\bxjs@CGHN` L^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。

```

120 \bxjs@if@format@at@least{2021/11/15}{%
121 \def\bxjs@CGHN#1{#1}%
122 }{%else
123 \def\bxjs@CGHN#1{\bxjs@CGHN#a#1//}%
124 \def\bxjs@CGHN#a#1/#2/#3//{#1/#3/#2}}

```

`\bxjs@cond` `\bxjs@cond``\ifXXX……\fi`{(真)}{(偽)}

T_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。

```

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

```

TODO:2.9 `\bxjs@expanded` を定義する。

`\bxjs@cslet` `\bxjs@cslet`{(名前 1)}\制御綴 :

```

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

```

`\bxjs@csletcs` `\bxjs@csletcs`{(名前 1)}{(名前 2)} :

```

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

```

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

```

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

```

`\bxjs@ifplus` `\@ifstar` の + 版。

```

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

```

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

```

136 \def\bxjs@trim#1{\expandafter\bxjs@trim@a#1\@nil#1}
137 \def\bxjs@trim@a{\futurelet\bxjs@tmpb\bxjs@trim@b}
138 \def\bxjs@trim@b{\bxjs@cond\ifx\bxjs@tmpb\@sptoken\fi
139  {\bxjs@trim@c\bxjs@trim@a}{\bxjs@trim@d}}
140 \def\bxjs@trim@c#1 {#1}
141 \def\bxjs@trim@d#1\@nil{\bxjs@trim@e#1\@nil: \@nil\@nnil}
142 \def\bxjs@trim@e#1 \@nil#2\@nnil{\bxjs@cond\ifx\@nnil#2\@nnil\fi
143  {\bxjs@trim@f#1\@nnil}{\bxjs@trim@e#1\@nil: \@nil\@nnil}}
144 \def\bxjs@trim@f#1\@nil#2\@nnil#3{\def#3{#1}}

```

`\bxjs@set@array@from@clist` `\bxjs@set@array@from@clist{(配列名接頭辞)}{ (コンマ区切りリスト)}` : コンマ区切りの値のリストから擬似配列を生成する。

※各要素について、先頭・末尾の空白トークン群は除去される。

```

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

```

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

```

151 \let\c@bxjs@tempcnta\@tempcnta
152 \def\bxjs@gset@tempcnta{\setcounter{bxjs@tempcnta}}

```

`\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）。

```

153 \def\jsSetQHLlength#1#2{%
154  \begingroup
155  \bxjs@parse@qh{#2}%
156  \ifx\bxjs@tmpb\relax
157    \setlength\@tempdima{#2}%
158    \xdef\bxjs@g@tmpa{\the\@tempdima}%
159  \else \global\let\bxjs@g@tmpa\bxjs@tmpb
160  \fi
161  \endgroup
162  #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。

```

163 \if j\jsEngine \def\bxjs@parse@qh@units{zw,zh}
164 \else \def\bxjs@parse@qh@units{trueQ,trueH,Q,H,zw,zh}
165 \fi
166 \def\bxjs@parse@qh#1{%

```

```

167 \let\bxjs@tmpb\relax
168 \for\bxjs@tmpa:=\bxjs@parse@qh@units\do{%
169   \ifx\bxjs@tmpb\relax
170     \edef\bxjs@next{\bxjs@tmpa}{#1}}%
171   \expandafter\bxjs@parse@qh@a\csname bxjs@unit@\bxjs@tmpa\expandafter
172     \endcsname\bxjs@next
173   \fi}}
174 \def\bxjs@parse@qh@a#1#2#3{%
175   \def\bxjs@next##1#2\@nil##2\@nnil{\bxjs@parse@qh@b{##1}{##2}#1}%
176   \bxjs@next#3\@nil#2\@nil\@nnil}
177 \def\bxjs@parse@qh@b#1#2#3{%
178   \ifx\@nnil#2\@nnil\else
179     \ifx#3\relax
180       \ClassError\bxjs@clsname
181         {You cannot use '\bxjs@tmpa' here}{\@ehc}%
182       \def\bxjs@tmpb{0pt}%
183     \else
184       \@tempdimb#3\relax \@tempdimb#1\@tempdimb
185       \edef\bxjs@tmpb{\the\@tempdimb}%
186     \fi
187   \fi}

```

今の段階では Q/H だけが使用可能。

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

`\ifbxjs@after@preamble` [スイッチ] 文書本体が開始しているか。

```
189 \newif\ifbxjs@after@preamble
```

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

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

```

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

```
193 \@onlypreamble\bxjs@post@option@hook
194 \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。

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

```

`\bxjs@endpreamble@hook` BXJS クラス用の `\AtEndPreamble` フック。

※`\AtEndPreamble` が利用できない場合は無効になる。

```

197 \@onlypreamble\bxjs@endpreamble@hook
198 \let\bxjs@endpreamble@hook\@empty
199 \AtEndOfClass{%
200   \ifx\AtEndPreamble\@undefined\else
201     \AtEndPreamble{\bxjs@endpreamble@hook}%
202   \fi}

```

一時的な手続き用の制御綴。

```
203 \@onlypreamble\bxjs@tmpdo
204 \@onlypreamble\bxjs@tmpdo@a
205 \@onlypreamble\bxjs@tmpdo@b
206 \@onlypreamble\bxjs@tmpdo@c
207 \@onlypreamble\bxjs@tmpdo@d
```

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

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

2 オプション

これらのクラスは `\documentclass{jsarticle}` あるいは `\documentclass[オプション]{jsarticle}` のように呼び出します。

まず、オプションに関連するいくつかのコマンドやスイッチ（論理変数）を定義します。

`\if@restonecol` 段組のときに真になる論理変数です。

```
210 \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。

```
211 \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。

```
212 %<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。

```
213 %<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。

```
214 %<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。

```
215 \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 1m^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 1.5m^2 ですが、ISO では B1 判の辺の長さが A0 判と A1 判の辺の長さの幾何平均です。したがって ISO の B0 判は $1000\text{mm} \times 1414\text{mm}$ です。このため、 $\text{L}^{\text{A}}\text{T}_{\text{E}}\text{X} 2_{\epsilon}$ の `b5paper` は $250\text{mm} \times 176\text{mm}$ ですが、 $\text{pL}^{\text{A}}\text{T}_{\text{E}}\text{X} 2_{\epsilon}$ の `b5paper` は $257\text{mm} \times 182\text{mm}$ になっています。ここでは $\text{pL}^{\text{A}}\text{T}_{\text{E}}\text{X} 2_{\epsilon}$ にならって JIS に従いました。

デフォルトは `a4paper` です。

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

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

```

216 \@onlypreamble\bxjs@setpaper
217 \def\bxjs@setpaper#1{\def\bxjs@param@paper{#1}}
218 \newif\ifbxjs@iso@bsize
219 \DeclareOption{iso-bsize}{\bxjs@iso@bsizetrue}
220 \@onlypreamble\bxjs@setpaper@bsize
221 \def\bxjs@setpaper@bsize#1{\edef\bxjs@param@paper{%
222   b#1\ifbxjs@iso@bsize paper\else j\fi}}
223 \DeclareOption{a3paper}{\bxjs@setpaper{a3paper}}
224 \DeclareOption{a4paper}{\bxjs@setpaper{a4paper}}
225 \DeclareOption{a5paper}{\bxjs@setpaper{a5paper}}
226 \DeclareOption{a6paper}{\bxjs@setpaper{a6paper}}
227 \DeclareOption{b4paper}{\bxjs@setpaper@bsize{4}}
228 \DeclareOption{b5paper}{\bxjs@setpaper@bsize{5}}
229 \DeclareOption{b6paper}{\bxjs@setpaper@bsize{6}}
230 \DeclareOption{a4j}{\bxjs@setpaper{a4paper}}
231 \DeclareOption{a5j}{\bxjs@setpaper{a5paper}}
232 \DeclareOption{b4j}{\bxjs@setpaper{b4j}}
233 \DeclareOption{b5j}{\bxjs@setpaper{b5j}}
234 \DeclareOption{a4var}{\bxjs@setpaper{{210truemm}{283truemm}}}
235 \DeclareOption{b5var}{\bxjs@setpaper{{182truemm}{230truemm}}}
236 \DeclareOption{letterpaper}{\bxjs@setpaper{letterpaper}}
237 \DeclareOption{legalpaper}{\bxjs@setpaper{legalpaper}}
238 \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。

```

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

```

```
245 \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。

```
246 \@for\bxjs@tmpa:={0,1,2,3}\do{\edef\bxjs@next{%
247   \noexpand\DeclareOption{b\bxjs@tmpa paper}%
248   {\noexpand\bxjs@setpaper@bsize{\bxjs@tmpa}}%
249 }\bxjs@next}
```

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

```
250 \DeclareOption{a4varpaper}{\bxjs@setpaper{{210truemm}{283truemm}}}
251 \DeclareOption{b5varpaper}{\bxjs@setpaper{{182truemm}{230truemm}}}
252 \DeclareOption{screenpaper}{\bxjs@setpaper{screen}}
253 \DeclareOption{custompaper}{}
```

■横置き 用紙の縦と横の長さを入れ換えます。

```
254 \newif\if@landscape
255 \@landscapefalse
256 \DeclareOption{landscape}{\@landscapetrue}
```

■slide オプション `slide` を新設しました。

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

```
257 \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。

```
258 %<!slide>\@slidefalse
259 %<slide>\@slidetrue
```

■サイズオプション 10pt, 11pt, 12pt のほかに、8pt, 9pt, 14pt, 17pt, 21pt, 25pt, 30pt, 36pt, 43pt を追加しました。これは等比数列になるように選んだものです（従来の 20pt も残しました）。`\@ptsize` の定義が変だったのでご迷惑をおかけしましたが、標準的なドキュメントクラスと同様にポイント数から 10 を引いたものに直しました。

[2003-03-22] 14Q オプションを追加しました。

[2003-04-18] 12Q オプションを追加しました。

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

です。

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

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

```
260 \newcommand{\@ptsize}{0}
261 \def\bxjs@param@basefontsize{10pt}
262 \def\jsUnusualPtSize{-20}
```

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

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

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

```
264 \jsSetQHLlength\@tempdima{#1}%
265 \edef\bxjs@param@basefontsize{\the\@tempdima}%
266 \ifdim\@tempdima=10pt \long\def\@ptsize{0}%
267 \else\ifdim\@tempdima=10.95pt \long\def\@ptsize{1}%
268 \else\ifdim\@tempdima=12pt \long\def\@ptsize{2}%
269 \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。

```
270 \def\bxjs@setjbasefontsize#1{%
271 \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。

```
272 \newif\ifjsc@mag
273 \newif\ifjsc@mag@xreal
274 %\let\jsc@magscale\undefined

275 \DeclareOption{8pt}{\bxjs@setbasefontsize{8pt}}
276 \DeclareOption{9pt}{\bxjs@setbasefontsize{9pt}}
277 \DeclareOption{10pt}{\bxjs@setbasefontsize{10pt}}
278 \DeclareOption{11pt}{\bxjs@setbasefontsize{10.95pt}}
279 \DeclareOption{12pt}{\bxjs@setbasefontsize{12pt}}
280 \DeclareOption{14pt}{\bxjs@setbasefontsize{14.4pt}}
281 \DeclareOption{17pt}{\bxjs@setbasefontsize{17.28pt}}
282 \DeclareOption{20pt}{\bxjs@setbasefontsize{20pt}}
283 \DeclareOption{21pt}{\bxjs@setbasefontsize{20.74pt}}
284 \DeclareOption{25pt}{\bxjs@setbasefontsize{24.88pt}}
285 \DeclareOption{30pt}{\bxjs@setbasefontsize{29.86pt}}
286 \DeclareOption{36pt}{\bxjs@setbasefontsize{35.83pt}}
287 \DeclareOption{43pt}{\bxjs@setbasefontsize{43pt}}
288 \DeclareOption{12Q}{\bxjs@setjbasefontsize{3mm}}
```



```

289 \DeclareOption{14Q}{\bxjs@setjbasefontsize{3.5mm}}
290 \DeclareOption{10ptj}{\bxjs@setjbasefontsize{10pt}}
291 \DeclareOption{10.5ptj}{\bxjs@setjbasefontsize{10.5pt}}
292 \DeclareOption{11ptj}{\bxjs@setjbasefontsize{11pt}}
293 \DeclareOption{12ptj}{\bxjs@setjbasefontsize{12pt}}

```

JS クラス互換の magstyle 設定オプション。

```

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

取りあえず、pT_εX 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。

```

297 \if j\jsEngine
298 \hour\time \divide\hour by 60\relax
299 \@tempcnta\hour \multiply\@tempcnta 60\relax
300 \minute\time \advance\minute-\@tempcnta
301 \DeclareOption{tombow}{%
302   \tombowtrue \tombowdatetrue
303   \setlength{\@tombowwidth}{.1\p@}%
304   \@bannertoken{%
305     \jobname\space(\number\year-\two@digits\month-\two@digits\day
306     \space\two@digits\hour:\two@digits\minute)}%
307   \maketombowbox}
308 \DeclareOption{tombo}{%
309   \tombowtrue \tombowdatefalse
310   \setlength{\@tombowwidth}{.1\p@}%
311   \maketombowbox}
312 \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。

```

313 \if j\jsEngine
314 \DeclareOption{mentuke}{%
315   \tombowtrue \tombowdatefalse
316   \setlength{\@tombowwidth}{\z@}%
317   \maketombowbox}
318 \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。

```
319 \DeclareOption{oneside}{\@twosidefalse \@mparswitchfalse}
320 \DeclareOption{twoside}{\@twosidetrue \@mparswitchtrue}
321 \DeclareOption{vartwoside}{\@twosidetrue \@mparswitchfalse}
```

■二段組 `twocolumn` で二段組になります。

```
322 \DeclareOption{onecolumn}{\@twocolumnfalse}
323 \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。

```
324 \DeclareOption{titlepage}{\@titlepagetrue}
325 \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。

```
326 %<book|report>\DeclareOption{openright}{\@openrighttrue\@openleftfalse}
327 %<book|report>\DeclareOption{openleft}{\@openlefttrue\@openrightfalse}
328 %<book|report>\DeclareOption{openany}{\@openrightfalse\@openleftfalse}
```

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

`eqnarray (env.)` \LaTeX の `eqnarray` 環境では `&` でできるアキが大きすぎるようですので、少し小さくします。また、中央の要素も `\displaystyle` にします。

[2022-09-13] \LaTeX 2_ε 2021-11-15 (l_tmath.dtx 2021/10/14 v1.2j) で `\@currentcounter` が追加されましたので、追随します。

```
329 \def\eqnarray{%
330   \stepcounter{equation}%
331   \def\@currentlabel{\p@equation\theequation}%
332   \def\@currentcounter{equation}%
333   \global\@eqnswtrue
334   \m@th
335   \global\@eqcnt\z@
336   \tabskip\@centering
337   \let\\\@eqnocr
338   $$\everycr{\halign to\displaywidth\bgroup
339     \hskip\@centering$\displaystyle\tabskip\z@skip{##}$\@eqnse1
340     &\global\@eqcnt\@ne \hfil$\displaystyle{\}{##{}}$\hfil
341     &\global\@eqcnt\tw@ $\displaystyle{##}$\hfil\tabskip\@centering
342     &\global\@eqcnt\thr@@ \hb@xt@\z@\bgroup\hss##\egroup
```

```

343     \tabskip\z@skip
344     \cr}

```

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

[2022-09-13] L^AT_EX 2_ε 2021-11-15 (l^AT_EX 2021/10/14 v1.2j) で \@currentcounter
が追加されましたので、追従します。

```

345 \DeclareOption{leqno}{\input{leqno.clo}}
346 \DeclareOption{fleqn}{\input{fleqn.clo}}%
347 % fleqn 用の eqnarray 環境の再定義
348 \def\eqnarray{%
349     \stepcounter{equation}%
350     \def\@currentlabel{\p@equation\theequation}%
351     \def\@currentcounter{equation}%
352     \global\@eqnswtrue\m@th
353     \global\@eqcnt\z@
354     \tabskip\mathindent
355     \let\@=\@eqncr
356     \setlength\abovedisplayskip{\topsep}%
357     \ifvmode
358         \addtolength\abovedisplayskip{\partopsep}%
359     \fi
360     \addtolength\abovedisplayskip{\parskip}%
361     \setlength\belowdisplayskip{\abovedisplayskip}%
362     \setlength\belowdisplayskip{\abovedisplayskip}%
363     \setlength\belowdisplayskip{\abovedisplayskip}%
364     $$\everycr{}\halign to\linewidth% $$
365     \bgroup
366     \hskip\@centering$\displaystyle\tabskip\z@skip{##}$\@eqnset
367     &\global\@eqcnt\@ne \hfil$\displaystyle{##}$\hfil
368     &\global\@eqcnt\tw@
369     $\displaystyle{##}$\hfil \tabskip\@centering
370     &\global\@eqcnt\thr@@ \hb@xt@\z@\bgroup\hss##\egroup
371     \tabskip\z@skip\cr
372     }}

```

■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。

```

373 % \DeclareOption{openbib}{%
374 %     \AtEndOfPackage{%
375 %         \renewcommand\@openbib@code{%
376 %             \advance\leftmargin\bibindent
377 %             \itemindent -\bibindent
378 %             \listparindent \itemindent
379 %             \parsep \z@}%
380 %         \renewcommand\newblock{\par}}}

```

■数式フォントとして和文フォントを登録しないオプション 数式中では 16 通りのフォントしか使えません。AMSFonTS や mathpmtx 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：

```
381 %\let\bxjs@enablejfam\@undefined
```

enablejfam オプションの処理。

```
382 \def\bxjs@kv@enablejfam@true{\let\bxjs@enablejfam=t}
383 \def\bxjs@kv@enablejfam@false{\let\bxjs@enablejfam=f}
384 \def\bxjs@kv@enablejfam@default{\let\bxjs@enablejfam\@undefined}
385 \define@key{bxjs}{enablejfam}[true]{%
386 \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。

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

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

■ドラフト draft で overfull box の起きた行末に 5pt の罫線を引きます。

[2016-07-13] \ifdraft を定義するのをやめました。

\ifjsDraft [スイッチ] draft オプションが指定されているか。

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

```
388 \newif\ifjsDraft
389 \DeclareOption{draft}{\jsDrafttrue \overfullrule=5pt }
390 \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, dviptdpmx など出力出来るようになる) が使えます。

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

BXJS クラスではここは和文ドライバの管轄。

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

BXJS クラスでは geometry パッケージがこの処理を行う。

`\ifbxjs@papersize` [スイッチ] papersize スペシャルを出力するか。既定で有効であるが、nopapersize オプションで無効にできる。

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

```
391 \newif\ifbxjs@papersize
392 \bxjs@papersizetrue
393 \DeclareOption{nopapersize}{\bxjs@papersizefalse}
394 \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。

```
395 \newif\if@english
396 \@englishfalse
397 \DeclareOption{english}{\@englishtrue}
```

■jsbook を jsreport もどきに オプション report を新設しました。

[2017-02-13] 従来は「jsreport 相当」を jsbook の report オプションで提供していましたが、新しく jsreport クラスも作りました。どちらでもお好きな方を使ってください。

BXJS では当初から bxjsreport クラスが用意されている。

■jslogo パッケージの読み込み 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。

```

398 \newif\if@jslogo \@jslogofalse
399 \DeclareOption{jslogo}{\@jslogotrue}
400 \DeclareOption{nojslogo}{\@jslogofalse}

```

■複合設定オプション

TODO:3.x `\bxjs@invscale` を書く場所を決める。(JS クラスと同じにはできなそう。)

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

※局所化の `\begingroup~\endgroup` について、以前は `\group~\egroup` を使っていたが、これだと数式モード中では空のサブ数式を生み出してしまうため修正した。

※元の長さが 128 pt 以上の場合でも動作するように修正した。

```

401 \mathchardef\bxjs@isc@ll=128
402 \mathchardef\bxjs@isc@s1=259
403 \def\bxjs@isc@s1h{65539 }
404 \def\bxjs@invscale#1#2{%
405   \begingroup \@tempdima=#1\relax \@tempdimb#2\p@\relax
406     \ifdim\@tempdima<\bxjs@isc@ll\p@
407       \@tempcnta\@tempdima \multiply\@tempcnta\@cclvi
408       \divide\@tempcnta\@tempdimb \multiply\@tempcnta\@cclvi
409     \else
410       \@tempcnta\@tempdima \divide\@tempcnta\@tempdimb
411       \multiply\@tempcnta\p@ \let\bxjs@isc@s1\bxjs@isc@s1h
412     \fi
413     \@tempcntb\p@ \divide\@tempcntb\@tempdimb
414     \advance\@tempcnta-\@tempcntb \advance\@tempcnta-\tw@
415     \@tempdimb\@tempcnta\@ne
416     \advance\@tempcnta\@tempcntb \advance\@tempcnta\@tempcntb
417     \advance\@tempcnta\bxjs@isc@s1 \@tempdimc\@tempcnta\@ne
418     \@whiledim\@tempdimb<\@tempdimc\do{%
419       \@tempcntb\@tempdimb \advance\@tempcntb\@tempdimc
420       \advance\@tempcntb\@ne \divide\@tempcntb\tw@
421     \ifdim #2\@tempcntb>\@tempdima
422       \advance\@tempcntb\m@ne \@tempdimc=\@tempcntb\@ne
423     \else \@tempdimb=\@tempcntb\@ne \fi}%
424     \xdef\bxjs@gtmpa{\the\@tempdimb}%
425   \endgroup #1=\bxjs@gtmpa\relax}

```

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

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

```

426 \DeclareOption{pandoc}{%
427   \bxjs@apply@pandoc@opt}
428 \@onlypreamble\bxjs@apply@pandoc@opt

```

```
429 \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。

```
430 \g@addto@macro\bxjs@post@option@hook{%
431   \bxjs@oldfontcommandstrue
432   \setkeys{bxjs}{ja=pandoc}%
433   \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）。

```
434 \ifx\bxjs@driver@opt\@undefined
435   \def\bxjs@driver@opt{dvipdfmx}%
436   \bxjs@dvi@opttrue
437 \fi
438 \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。

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

■エンジン・ドライバオプション

`\bxjs@engine@given` [暗黙文字トークン] オプションで明示されたエンジンの種別。

```
443 %\let\bxjs@engine@given\@undefined
```

`\bxjs@engine@opt` 明示されたエンジンのオプション名。

```
444 %\let\bxjs@engine@opt\@undefined
```

エンジン明示指定のオプションの処理。

※ 0.9pre 版の暫定仕様と異なり、エンジン名は `...latex` に限定する。`xetex` や `pdftex` は一般的な \LaTeX の慣習に従って「ドライバの指定」とみなすべきだから。

```
445 \DeclareOption{autodetect-engine}{%
446   \let\bxjs@engine@given=*\%
447 \DeclareOption{latex}{%
448   \def\bxjs@engine@opt{latex}%
449   \let\bxjs@engine@given=n}
450 \DeclareOption{platex}{%
451   \def\bxjs@engine@opt{platex}%
452   \let\bxjs@engine@given=j}
453 \DeclareOption{uplatex}{%
454   \def\bxjs@engine@opt{uplatex}%
455   \let\bxjs@engine@given=u}
456 \DeclareOption{xelatex}{%
```

```

457 \def\bxjs@engine@opt{xelatex}%
458 \let\bxjs@engine@given=x}
459 \DeclareOption{pdflatex}{%
460 \def\bxjs@engine@opt{pdflatex}%
461 \let\bxjs@engine@given=p}
462 \DeclareOption{lualatex}{%
463 \def\bxjs@engine@opt{lualatex}%
464 \let\bxjs@engine@given=l}
465 \DeclareOption{platex-ng}{%
466 \def\bxjs@engine@opt{platex-ng}%
467 \let\bxjs@engine@given=g}
468 \DeclareOption{platex-ng*}{%
469 \def\bxjs@engine@opt{platex-ng*}%
470 \let\bxjs@platexng@nodrv=t%
471 \let\bxjs@engine@given=g}

```

`\bxjs@driver@given` [暗黙文字トークン] オプションで明示されたドライバの種別。

```

472 %\let\bxjs@driver@given\@undefined
473 \let\bxjs@driver@@dvimode=0
474 \let\bxjs@driver@@dvipdfmx=1
475 \let\bxjs@driver@@pdfmode=2
476 \let\bxjs@driver@@xetex=3
477 \let\bxjs@driver@@dvips=4
478 \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。

```

479 %\let\bxjs@driver@opt\@undefined

```

※ `class-nodvidriver` は BXJS クラスの仕様上は `nodvidriver` と完全に等価であるが、「グローバルオプションに何かがあるか」の点で異なる。

```

480 \DeclareOption{dvips}{%
481 \def\bxjs@driver@opt{dvips}%
482 \let\bxjs@driver@given\bxjs@driver@@dvips}
483 \DeclareOption{dviout}{%
484 \def\bxjs@driver@opt{dviout}%
485 \let\bxjs@driver@given\bxjs@driver@@dvimode}
486 \DeclareOption{xdvi}{%
487 \def\bxjs@driver@opt{xdvi}%
488 \let\bxjs@driver@given\bxjs@driver@@dvimode}
489 \DeclareOption{dvipdfmx}{%
490 \def\bxjs@driver@opt{dvipdfmx}%
491 \let\bxjs@driver@given\bxjs@driver@@dvipdfmx}
492 \DeclareOption{nodvidriver}{%
493 \def\bxjs@driver@opt{nodvidriver}%
494 \let\bxjs@driver@given\bxjs@driver@@none}
495 \DeclareOption{class-nodvidriver}{%
496 \def\bxjs@driver@opt{class-nodvidriver}%
497 \let\bxjs@driver@given\bxjs@driver@@none}
498 \DeclareOption{pdftex}{%

```



```

499 \def\bxjs@driver@opt{pdfTeX}%
500 \let\bxjs@driver@given\bxjs@driver@pdfmode}
501 \DeclareOption{luatex}{%
502 \def\bxjs@driver@opt{luatex}%
503 \let\bxjs@driver@given\bxjs@driver@pdfmode}
504 \DeclareOption{xetex}{%
505 \def\bxjs@driver@opt{xetex}%
506 \let\bxjs@driver@given\bxjs@driver@xetex}

dvipdfmx-if-dvi は 2.0 版より非推奨となった。
507 \DeclareOption{dvipdfmx-if-dvi}{\bxjs@depre@opt@do{dvipdfmx-if-dvi}{dvi=dvipdfmx}}

```

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

`\bxjs@depre@opt` 非推奨のオプションについて警告を出す。

```

\bxjs@depre@opt@do 508 \@onlypreamble\bxjs@depre@opt
509 \def\bxjs@depre@opt#1#2{%
510 \ClassWarningNoLine\bxjs@clsname
511 {The old option '#1' is DEPRECATED\MessageBreak
512 and may be abolished in future!\MessageBreak
513 You should instead write:\MessageBreak
514 \space\space #2}}
515 \@onlypreamble\bxjs@depre@opt@do
516 \def\bxjs@depre@opt@do#1#2{%
517 \bxjs@depre@opt{#1}{#2}%
518 \setkeys{bxjs}{#2}}

```

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

※ 2.0 版より、既定値を常に真とする。

```

519 \newif\ifbxjs@bigcode \bxjs@bigcodetrue

nobigcode / bigcode オプションの定義。
520 \DeclareOption{nobigcode}{%
521 \bxjs@bigcodefalse}
522 \DeclareOption{bigcode}{%
523 \bxjs@bigcodetrue}

```

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

```

524 \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。

```

525 \DeclareOption{nooldfontcommands}{%
526 \bxjs@oldfontcommandsfalse}

```

```
527 \DeclareOption{oldfontcommands}{%
528   \bxjs@oldfontcommandstrue}
```

■無効および廃止されたオプション ☹

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

```
529 \ifbxjs@brace@safe
530   \@onlypreamble\bxjs@register@badopt
531   \def\bxjs@register@badopt#1{%
532     \expandafter\@onlypreamble\csname bxjs@badopt/#1\endcsname
533     \@namedef{bxjs@badopt/#1}}
534   \g@addto@macro\bxjs@begin@document@hook{%
535     \@for\bxjs@tmpa:=\@unusedoptionlist\do{%
536       \@nameuse{bxjs@badopt/\bxjs@tmpa}}}
537 \fi
```

`\bxjs@invalid@opt` 無効オプションを宣言する。そのオプションが指定された場合、それがグローバルオプションとして他のパッケージによって使用されていなければ、文書本体開始時にエラーを出す。※古いカーネルでは未使用検査ができないため、その場で警告を出す。

```
538 \@onlypreamble\bxjs@invalid@opt
539 \ifbxjs@brace@safe
540   \def\bxjs@invalid@opt#1#2{%
541     \bxjs@register@badopt{#1}{\ClassError\bxjs@clsname{#2}\@ehc}}
542 \else
543   \def\bxjs@invalid@opt#1#2{%
544     \DeclareOption{#1}{\ClassWarningNoLine\bxjs@clsname{#2}}}
545 \fi
```

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

※ `ltjclasses` クラスでも警告を出している。

```
546 \bxjs@invalid@opt{winjis}{%
547   This class does not support 'winjis' option}
548 \bxjs@invalid@opt{mingoth}{%
549   This class does not support 'mingoth' option}
550 \bxjs@invalid@opt{jis}{%
551   This class does not support 'jis' option}
552 \if j\jsEngine\else
553 \bxjs@invalid@opt{tombo}{%
554   Option 'tombo' can be used only on (u)pLaTeX}
555 \bxjs@invalid@opt{tombow}{%
556   Option 'tombow' can be used only on (u)pLaTeX}
557 \bxjs@invalid@opt{mentuke}{%
558   Option 'mentuke' can be used only on (u)pLaTeX}
559 \fi
```

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

```

560 \DeclareOption*{%
561   \bxjs@check@ja@prefix \ifx\bxjs@next\relax
562   \def\bxjs@next{\bxjs@cls@setkeys{bxjs}}%
563   \expandafter\bxjs@next\expandafter{\CurrentOption}%
564   \else

```

オプションが ja:XXX という形式である場合は japaram={XXX} に振り替える。

```

565   \edef\bxjs@next{%
566     \noexpand\setkeys{bxjs}{japaram={\bxjs@next}}%
567   }\bxjs@next
568 \fi}

```

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

```

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

```

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

```

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

```

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

```

578 \def\bxjs@cls@setkeys#1#2{%
579   \let\bxjs@save@KV@errx\KV@errx
580   \def\KV@errx##1{\OptionNotUsed}%
581   \setkeys{#1}{#2}%
582   \let\KV@errx\bxjs@save@KV@errx}
583 \ifbxjs@brace@safe\else
584   \let\bxjs@cls@setkeys\bxjs@safe@setkeys
585 \fi

```

`\bxjs@declare@enum@option` `\bxjs@declare@enum@option{<オプション名>}{<enum 名>}{<初期値>}`

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

```

586 \@onlypreamble\bxjs@declare@enum@option
587 \def\bxjs@declare@enum@option#1#2#3{%
588   \bxjs@csletcs{bxjs@#2}{bxjs@#2@#3}%
589   \define@key{bxjs}{#1}{%
590     \@ifundefined{bxjs@#2@##1}{%
591       \bxjs@error@keyval{#1}{##1}%
592     }\bxjs@csletcs{bxjs@#2}{bxjs@#2@##1}}}}

```

`\bxjs@declare@bool@option` `\bxjs@declare@bool@option{<オプション名>}{<スイッチ名>}{<初期値>}`
 “<オプション名>=<真偽値>” のオプション指定に対して、`\if[bxjs@<スイッチ名>]` を設定する、という動作を規定する。

```
593 \onlypreamble\bxjs@declare@bool@option
594 \def\bxjs@declare@bool@option#1#2#3{%
595   \csname newif\expandafter\endcsname\csname ifbxjs@#2\endcsname
596   \@nameuse{bxjs@#2#3}%
597   \define@key{bxjs}{#1}[true]{%
598     \ifundefined{bxjs@#2##1}{%
599       \bxjs@error@keyval{#1}{##1}%
600     }{\@nameuse{bxjs@#2##1}}}
```

`\bxjs@set@keyval` `\bxjs@set@keyval{<key>}{<value>}{<error>}`
`\bxjs@kv@<key>@<value>` が定義済ならそれを実行し、未定義ならエラーを出す。

```
601 \def\bxjs@set@keyval#1#2#3{%
602   \bxjs@csletcs{bxjs@next}{bxjs@kv@#1@#2}%
603   \ifx\bxjs@next\relax
604     \bxjs@error@keyval{#1}{#2}%
605     #3%
606   \else \bxjs@next
607   \fi}
608 \onlypreamble\bxjs@error@keyval
609 \def\bxjs@error@keyval#1#2{%
610   \ClassError\bxjs@clsname
611   {Invalid value '#2' for option #1}\@ehc}
```

`\jsScale` [実数値マクロ] 和文スケール値。

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

`\bxjs@base@opt` 明示された base オプションの値。

```
613 %\let\bxjs@base@opt\@undefined
```

base オプションの処理。

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

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

```
618 %\let\bxjs@jbase@opt\@undefined
```

jbase オプションの処理。

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

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

```
621 %\let\bxjs@scale@opt\@undefined
```

scale オプションの処理。

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

noscale オプションの処理。

TODO:3.0 noscale は廃止の予定。

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

`\bxjs@param@mag` mag オプションの値。

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

mag オプションの処理。

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

paper オプションの処理。

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

`\bxjs@jadriver` 和文ドライバの名前。

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

`\bxjs@jadriver@opt` 明示された和文ドライバの名前。

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

ja オプションの処理。

※ jadriver は 0.9 版で用いられた旧称。

TODO:3.0 jadriver は廃止の予定。

※単なる ja という指定は無視される (Pandoc 対策)。

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

`\jsJaFont` 和文フォント設定の名前。

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

jafont オプションの処理。

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

`\jsJaParam` 和文ドライバパラメタの文字列。

```
638 \let\jsJaParam\@empty
```

japaram オプションの処理。

```
639 \define@key{bxjs}{japaram}{%
640   \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。

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

```

642 \ExecuteOptions{pandoc}%
643 \edef\jsJaParam{\bxjs@catopt\jsJaParam{#1}}
644 \define@key{bxjs}{pandoc}[] {%
645 \ExecuteOptions{pandoc}%
646 \edef\jsJaParam{\bxjs@catopt\jsJaParam{#1}}

```

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

```

647 \let\bxjs@magstyle@@mag=m
648 \let\bxjs@magstyle@@real=r
649 \let\bxjs@magstyle@@xreal=x

```

(新しい素敵なお名前。)

※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。

```

650 \let\bxjs@magstyle@@usemag\bxjs@magstyle@@mag
651 \let\bxjs@magstyle@@nomag\bxjs@magstyle@@real
652 \bxjs@csllet{bxjs@magstyle@@nomag*}\bxjs@magstyle@@xreal

```

`\bxjs@magstyle@@default` は既定の値を表す。

```

653 \let\bxjs@magstyle@@default\bxjs@magstyle@@usemag
654 \ifx l\jsEngine \ifnum\luatexversion>86
655 \let\bxjs@magstyle@@default\bxjs@magstyle@@xreal
656 \fi\fi
657 \ifjsWithpTeXng
658 \let\bxjs@magstyle@@default\bxjs@magstyle@@xreal
659 \fi
660 \let\bxjs@magstyle\bxjs@magstyle@@default

```

magstyle オプションの処理。

```

661 \define@key{bxjs}{magstyle}{%
662 \bxjs@cslletcs{bxjs@magstyle}{bxjs@magstyle@@#1}%
663 \ifx\bxjs@magstyle\relax
664 \bxjs@error@keyval{magstyle}{#1}%
665 \let\bxjs@magstyle\bxjs@magstyle@@default
666 \fi}

```

`\bxjs@geometry geometry` オプションの指定値。

```

667 \let\bxjs@geometry@@class=c
668 \let\bxjs@geometry@@user=u
669 \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。

```

670 \bxjs@declare@bool@option{fancyhdr}{fancyhdr}{true}

```

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

```

671 \newif\ifbxjs@dvi@opt

```

DVI モードのドライバとドライバ種別との対応。

```

672 \let\bxjs@dvidriver@@dvipdfmx=\bxjs@driver@@dvipdfmx

```

```

673 \let\bxjs@dvidriver@@dvips=\bxjs@driver@@dvips
674 \let\bxjs@dvidriver@@dviout=\bxjs@driver@@dvimode
675 \let\bxjs@dvidriver@@xdvi=\bxjs@driver@@dvimode
676 \let\bxjs@dvidriver@@nodvidriver=\bxjs@driver@@none
677 \bxjs@cslet{bxjs@dvidriver@@class-nodvidriver}\bxjs@driver@@none

```

dvi オプションの処理。

```

678 \define@key{bxjs}{dvi}{%
679   \bxjs@csletcs{bxjs@tmpa}{bxjs@dvidriver@@#1}%
680   \ifx\bxjs@tmpa\relax
681     \bxjs@error@keyval{dvi}{#1}%
682   \else

```

\bxjs@driver@given を未定義にしていることに注意。

```

683   \def\bxjs@driver@opt{#1}%
684   \let\bxjs@driver@given\@undefined
685   \bxjs@dvi@opttrue
686 \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。

```

687 \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。

```

688 \newif\ifbxjs@force@chapterabstract
689 %<book>\bxjs@force@chapterabstracttrue

```

layout オプションの処理。

```

690 \@namedef{bxjs@kv@layout@v1}{%
691 %<book>\bxjs@layout@buggyhmargintrue
692 %<report>\bxjs@force@chapterabstracttrue
693 }
694 \@namedef{bxjs@kv@layout@v2}{%
695 %<book>\bxjs@layout@buggyhmarginfalse
696 %<report>\bxjs@force@chapterabstractfalse
697 }
698 \define@key{bxjs}{layout}{%
699   \bxjs@set@keyval{layout}{#1}{}}

```

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

```

700 %\let\bxjs@textwidth@limit@opt\@undefined
701 \define@key{bxjs}{textwidth-limit}{%
702   \bxjs@depre@opt{textwidth-limit}{textwidth=#1zw}%
703   \edef\bxjs@textwidth@limit@opt{#1}}

```

\bxjs@textwidth@opt textwidth の指定値。

```

704 %\let\bxjs@textwidth@opt\@undefined
705 \define@key{bxjs}{textwidth}{\edef\bxjs@textwidth@opt{#1}}
706 \define@key{bxjs}{line_length}{\setkeys{bxjs}{textwidth=#1}}

```

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

```
707 %\let\bxjs@number@of@lines@opt\@undefined
708 \define@key{bxjs}{number-of-lines}{\edef\bxjs@number@of@lines@opt{#1}}
709 \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。

```
710 %\let\bxjs@paragraph@mark\@undefined
711 \define@key{bxjs}{paragraph-mark}{%
712 \edef\bxjs@paragraph@mark{#1}}
```

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

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

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

```
714 \bxjs@declare@bool@option{jaspace-cmd}{jaspace@cmd}{true}
715 \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。

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

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

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

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

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

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

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

`\ifbxjs@usezw` [スイッチ] `use-zw` の指定値。

TODO:3.0 `zw/nozw` は廃止の予定。

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

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

TODO:3.0 `js/nojs` は廃止の予定。

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


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

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

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

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

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

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

■ オプションの実行

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

※クラスに `\DeclareOption*` があり `\OptionNotUsed` を使っていない場合は `\@unusedoptions` は常に空のままであることを利用している。

```
742 \ifbxjs@brace@safe\else
743 \let\bxjs@org@removeelement\@removeelement
744 \def\@removeelement#1#2#3{%
745   \def\reserved@a{#2}%
746   \ifx\reserved@a\@empty \let#3\@empty
747   \else \bxjs@org@removeelement{#1}{#2}{#3}%
748   \fi}
749 \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。

```
750 %<article>\ExecuteOptions{a4paper,oneside,onecolumn,notitlepage,final}
751 %<book>\ExecuteOptions{a4paper,twoside,onecolumn,titlepage,openright,final}
752 %<report>\ExecuteOptions{a4paper,oneside,onecolumn,titlepage,openany,final}
753 %<slide>\ExecuteOptions{36pt,a4paper,landscape,oneside,onecolumn,titlepage,final}
754 \ProcessOptions\relax
755 \bxjs@post@option@hook
```

後処理

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

```
756 \if@slide
757 \def\maybeblue{\@ifundefined{ver@color.sty}{\color{blue}}
758 \fi
759 %<*jsclasses>
760 \if@landscape
761 \setlength\@tempdima {\paperheight}
762 \setlength\paperheight{\paperwidth}
763 \setlength\paperwidth {\@tempdima}
764 \fi
765 %</jsclasses>
```

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

TODO:3.0 2021/06/01 版以降のカーネルについてこの処理を廃止する。(仕様変更に準じる扱いとする。)

```
766 \def\bxjs@tmpdo{%
767 \def\bxjs@tmpa{\@gobble}%
768 \expandafter\bxjs@tmpdo@a\@classoptionslist,\@nil,%
769 \let\@classoptionslist\bxjs@tmpa}
770 \def\bxjs@tmpdo@a#1,{%
771 \ifx\@nil#1\relax\else
772 \bxjs@tmpdo@b#1{\@nil
773 \if@tempswa \edef\bxjs@tmpa{\bxjs@tmpa,#1}\fi
774 \expandafter\bxjs@tmpdo@a
775 \fi}
776 \def\bxjs@tmpdo@b#1#\bxjs@tmpdo@c}
777 \def\bxjs@tmpdo@c#1\@nil{%
778 \ifx\@nil#1\@nil \@tempwatrue \else \@tempwafalse \fi}
779 \bxjs@tmpdo
```

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

TODO:3.0 noscale オプションは廃止予定。

```
780 \@expandtwoargs\@removeelement
781 {papersize}\@classoptionslist\@classoptionslist
782 \@expandtwoargs\@removeelement
783 {10pt}\@classoptionslist\@classoptionslist
784 \@expandtwoargs\@removeelement
785 {noscale}\@classoptionslist\@classoptionslist
```

■**使用エンジンの検査・自動判定** デフォルトで現在使われているエンジンが pL^AT_EX か upL^AT_EX かを判定します。ユーザによって platex オプションまたは uplatex オプションが明示的に指定されている場合は、実際に使われているエンジンと一致しているかを検査

し、一致しない場合はエラーメッセージを表示します。

[2016-11-09] pL^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 `otf` パッケージにも `uplatex` オプションが渡るようにします。

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

正規化前の和文ドライバの値を `\bxjs@jadriver` に設定する。

```
786 \ifx\bxjs@jadriver@opt\undefined\else
787   \let\bxjs@jadriver\bxjs@jadriver@opt
788 \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。

```
789 \let\bxjs@tmpb\jsEngine
790 \ifx j\bxjs@tmpb\ifjsWithpTeXng
791   \let\bxjs@tmpb=g
792 \fi\fi
793 \ifx j\bxjs@tmpb\ifjsWithupTeX
794   \let\bxjs@tmpb=u
795 \fi\fi
796 \ifx p\bxjs@tmpb\ifjsInPdfMode\else
797   \let\bxjs@tmpb=n
798 \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。)

```
799 \ifx *\bxjs@engine@given
800   \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。

```
801 \ifx j\bxjs@engine@given
802   \g@addto@macro\@classoptionslist{,platex}
803 \else\ifx u\bxjs@engine@given
804   \g@addto@macro\@classoptionslist{,uplatex}
805 \fi\fi
806 \fi
807 \ifx\bxjs@engine@given\undefined\else
808   \ifx\bxjs@engine@given\bxjs@tmpb\else
809     \ClassError\bxjs@clsname
810       {Option '\bxjs@engine@opt' used on wrong engine}\@ehc
811   \fi
812 \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。

```

813 \ifjsWithpTeXng
814 \g@addto@macro\@classoptionslist{,uplaxex}
815 \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。

```

816 \@tempwattrue
817 \ifx \bxjs@driver@given\@undefined\else
818 \ifjsInPdfMode
819 \ifx\bxjs@driver@given\bxjs@driver@@pdfmode\else
820 \@tempwafalse
821 \fi
822 \else\ifx x\jsEngine
823 \ifx\bxjs@driver@given\bxjs@driver@@xetex\else
824 \@tempwafalse
825 \fi
826 \else
827 \ifx\bxjs@driver@given\bxjs@driver@@pdfmode
828 \@tempwafalse
829 \else\ifx\bxjs@driver@given\bxjs@driver@@xetex
830 \@tempwafalse
831 \fi\fi
832 \ifjsWithpTeXng\ifx\bxjs@driver@given\bxjs@driver@@dvipdfmx\else
833 \@tempwafalse
834 \fi\fi
835 \fi\fi
836 \fi
837 \if@tempwa\else
838 \ClassError\bxjs@clsname
839 {Option '\bxjs@driver@opt' used on wrong engine}\@ehc
840 \fi

```

DVI 出力のエンジンである場合の追加処理。

```

841 \ifjsInPdfMode \@tempwafalse
842 \else\ifx x\jsEngine \@tempwafalse
843 \else\ifjsWithpTeXng \@tempwafalse
844 \else \@tempwattrue
845 \fi\fi\fi
846 \if@tempwa

```

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)。

```

847 \ifx\bxjs@driver@opt\@undefined
848 \if \ifbxjs@explIII T\else\ifx\bxjs@jadriver@opt\@undefined F\else T\fi\fi T%
849 \ClassWarningNoLine\bxjs@clsname
850 {A driver option is MISSING!!\MessageBreak
851 You should properly specify one of the valid\MessageBreak
852 driver options according to the DVI driver\MessageBreak

```

```

853         that is in use:\MessageBreak
854         \@spaces dvips, dvipdfmx, dviout, xdvi,\MessageBreak
855         \@spaces nodvidriver}
856     \fi
857 \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。)

```

858 \ifbxjs@dvi@opt
859     \edef\bxjs@next{%
860         \let\noexpand\bxjs@driver@given
861         \csname bxjs@dvidriver@@\bxjs@driver@opt\endcsname
862         \noexpand\g@addto@macro\noexpand\@classoptionslist
863         {,\bxjs@driver@opt}%
864     }\bxjs@next
865 \fi
866 \fi

```

エンジンが pTeX-ng の場合、グローバルオプションに dvipdfmx を追加する。ただし、エンジンオプションが platex-ng* (*付) の場合、および既に dvipdfmx が指定されている場合を除く。

```

867 \ifjsWithpTeXng
868     \ifx\bxjs@driver@given\bxjs@driver@@dvipdfmx
869         \let\bxjs@platexng@nodrv\undefined
870     \else\ifx t\bxjs@platexng@nodrv\else
871         \g@addto@macro\@classoptionslist{,dvipdfmx}
872     \fi\fi
873 \fi

```

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

```

874 \ifx\bxjs@driver@given\bxjs@driver@@none
875     \bxjs@papersizefalse
876 \fi

```

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

```

877 \@ifpackageloaded{plautopatch}{%
878     \bxjs@plautopatchtrue
879 }{}

```

標準の和文ドライバの名前の定数。

```

880 \def\bxjs@@minimal{minimal}
881 \def\bxjs@@standard{standard}
882 \def\bxjs@@pandoc{pandoc}
883 \def\bxjs@@modern{modern}

```

\bxjs@jadriver の正規化。値が未指定の場合は minimal に変える。ただしエンジンが (u)pTeX である場合は standard に変える。

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

```

884 \ifx\bxjs@jadriver\relax
885 \ifx j\jsEngine
886 \let\bxjs@jadriver\bxjs@@standard
887 \else
888 \ClassWarningNoLine\bxjs@clsname
889 {The option 'ja' is MISSING!!\MessageBreak
890 So 'ja=minimal' is assumed as fallback, but\MessageBreak
891 such implicit setting is now DEPRECATED!\MessageBreak
892 You should write 'ja=minimal' explicitly,\MessageBreak
893 if it is intended}
894 \let\bxjs@jadriver\bxjs@@minimal
895 \fi
896 \fi

```

plautopatch が真の場合はここで plautopatch を読み込む。

※この時点で既に読み込まれているパッケージは、calc、keyval、iftex。

※ Pandoc モードでは plautopatch の既定値を真とする。

```

897 \ifx\bxjs@jadriver\bxjs@@pandoc \ifx\bxjs@plautopatch@given\@undefined
898 \ifjsWithTeX
899 \bxjs@plautopatchtrue
900 \fi\fi\fi
901 \ifx j\jsEngine \ifbxjs@plautopatch
902 \RequirePackage{plautopatch}[2018/08/22]%v0.3
903 \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。

```

904 \ifx\bxjs@jadriver@opt\@undefined\else
905 \ifx\bxjs@engine@given\@undefined
906 \ClassError\bxjs@clsname
907 {An engine option must be explicitly given}%
908 {When you use a Japanese-driver you must specify a correct\MessageBreak
909 engine option.\MessageBreak\@ehc}
910 \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。)

```

911 \ifx\bxjs@magstyle@@default\bxjs@magstyle@@mag\else
912 \ifx\bxjs@magstyle\bxjs@magstyle@@mag
913 \let\bxjs@magstyle\bxjs@magstyle@@default
914 \ClassError\bxjs@clsname
915 {The engine does not support 'magstyle=usemag'}%
916 {LuaTeX v0.87 or later no longer supports the "mag" feature of TeX.\MessageBreak
917 The default value 'nomag*' is used instead.\MessageBreak \@ehc}
918 \fi
919 \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の実施されていることに注意。`jbases` 非指定の場合はこのままでよい。

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

`jbases` 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。

```
922 \jsSetQHLlength\@tempdima{\bxjs@jbase@opt}%
923 \bxjs@invscale\@tempdima\jsScale
924 \bxjs@setbasefontsize{\@tempdima}%
925 \else
```

`jbases` 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。

```
926 \ifx\bxjs@scale@opt\@undefined\else
927 \ClassWarningNoLine\bxjs@clsname
928 {Redundant 'scale' option is ignored}%
929 \fi
930 \jsSetQHLlength\@tempdima{\bxjs@jbase@opt}%
931 \@tempdimb=\bxjs@param@basefontsize\relax
932 \edef\jsScale{\strip@pt\@tempdimb}%
933 \bxjs@invscale\@tempdima\jsScale
934 \edef\jsScale{\strip@pt\@tempdima}%
935 \fi
936 \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。

```
937 \let\Cjascale\jsScale
```

`disguise-js=true` 指定時は、`jsarticle` (または `jsbook`) クラスを読込済のように振舞う。※「2つのクラスを読み込んだ状態」は `\LoadClass` を使用した場合に出現するので、別に異常ではない。

```
938 \ifbxjs@disguise@js
939 %<book|report>\def\bxjs@js@clsname{jsbook}
940 %<!book&!report>\def\bxjs@js@clsname{jsarticle}
941 \namedef{ver@\bxjs@js@clsname.cls}{2001/01/01 (bxjs)}
942 \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。

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

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

```
944 \ifbxjs@oldfontcommands
945 \AtEndOfClass{\allowoldfontcommands}
946 \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)。

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

BXJS では出力用紙サイズ記録は geometry パッケージが行う。

また、JS クラスと異なり、\stockwidth、\stockheight は常に定義される。

```
947 \ifx\stockwidth\undefined\newdimen\stockwidth\fi
948 \ifx\stockheight\undefined\newdimen\stockheight\fi
949 \begingroup\expandafter\expandafter\expandafter\endgroup
950 \expandafter\ifx\csname iftombow\expandafter\endcsname\csname iftrue\endcsname
951   \setlength{\stockwidth}{\paperwidth}
952   \setlength{\stockheight}{\paperheight}
953   \advance \stockwidth 2in
954   \advance \stockheight 2in
955 \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。

```
956 %<slide>\def\n@baseline{13}%
957 %<!slide>\ifdim\bxjs@param@basefontsize<10pt \def\n@baseline{15}%
958 %<!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。

```
959 \ifx\bxjs@magstyle\bxjs@magstyle@@mag
960   \jsc@magtrue
961 \else\ifx\bxjs@magstyle\bxjs@magstyle@@xreal
962   \jsc@mag@xrealtrue
963 \fi\fi
```

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

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

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

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

```
964 \ifx\bxjs@param@mag\relax
965   \@tempdima=\bxjs@param@basefontsize
966   \advance\@tempdima.001pt \multiply\@tempdima25
967   \divide\@tempdima16384\relax \@tempcmta\@tempdima\relax
968   \edef\bxjs@param@mag{\the\@tempcmta}
969 \else
970 % mag 値が直接指定された場合
971   \bxjs@gset@tempcmta{\bxjs@param@mag}
972   \ifnum\@tempcmta<\z@ \@tempcmta=\z@ \fi
973 % 有効な mag 値の範囲は 1--32768
974   \edef\bxjs@param@mag{\the\@tempcmta}
975   \advance\@tempcmta100000
976   \def\bxjs@tmpa#1#2#3#4#5\@nil{\@tempdima=#2#3#4.#5\p@}
977   \expandafter\bxjs@tmpa\the\@tempcmta\@nil
978   \edef\bxjs@param@basefontsize{\the\@tempdima}
979 \fi
980 \@tempcmta\bxjs@param@mag \advance\@tempcmta100000
981 \def\bxjs@tmpa#1#2#3#4\@nil{\@tempdima=#2#3.#4\p@}
982 \expandafter\bxjs@tmpa\the\@tempcmta\@nil
983 \edef\jsc@magscale{\strip@pt\@tempdima}
984 \let\jsBaseFontSize\bxjs@param@basefontsize
```

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

※ 2.9 版において `\p@?` 表記を廃止。

```
985 \newdimen\jsc@mpt
986 \newdimen\jsc@mmm
987 \ifjsc@mag
988   \jsc@mpt=1\p@
989   \jsc@mmm=1mm
990 \else
991   \jsc@mpt=\jsc@magscale\p@
992   \jsc@mmm=\jsc@magscale mm
993 \fi
```

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

`use-zw` が真の時は `\zw` を `\jsZw` と同義にする。

```
994 \newdimen\jsZw
995 \jsZw=10\jsc@mpt \jsZw=\jsScale\jsZw
996 \ifbxjs@usezw
997   \providecommand*\zw{\jsZw}
998 \fi
```

`\zwspace` 全角幅の水平空き。

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

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

```
1000 \ifjsc@mag@xreal
1001   \RequirePackage{type1cm}
1002   \let\jsc@invscale\bxjs@invscale



---


1003 \ifbxjs@TUenc
1004   \expandafter\let\csname TU/lmr/m/n/10\endcsname\relax
1005 \else
1006   \expandafter\let\csname OT1/cmr/m/n/10\endcsname\relax
1007 \fi
1008 \expandafter\let\csname OMX/cmex/m/n/10\endcsname\relax
1009 \let\jsc@get@external@font\get@external@font
1010 \def\get@external@font{%
1011   \jsc@preadjust@extract@font
1012   \jsc@get@external@font}
1013 \def\jsc@fstrunc#1{%
1014   \edef\jsc@tmpa{\strip@pt#1}%
1015   \expandafter\jsc@fstrunc@a\jsc@tmpa.****\@nil}
1016 \def\jsc@fstrunc@a#1.#2#3#4#5#6\@nil{%
1017   \if#5*\else
1018     \edef\jsc@tmpa{#1%
1019       \ifnum#2#3>\z@ .#2\ifnum#3>\z@ #3\fi\fi}%
1020   \fi}
1021 \def\jsc@preadjust@extract@font{%
1022   \let\jsc@req@size\fontsize
```

```

1023 \dimen@f@size\p@ \jsc@invscale\dimen@\jsc@magscale
1024 \advance\dimen@.005pt\relax \jsc@fstrunc\dimen@
1025 \let\jsc@ref@size\jsc@tmpa
1026 \let\f@size\jsc@ref@size}
1027 \def\execute@size@function#1{%
1028 \let\jsc@cref@size\f@size
1029 \let\f@size\jsc@req@size
1030 \csname s@fct@#1\endcsname}
1031 \let\jsc@DeclareErrorFont\DeclareErrorFont
1032 \def\DeclareErrorFont#1#2#3#4#5{%
1033 \@tempdimc#5\p@ \@tempdimc\jsc@magscale\@tempdimc
1034 \edef\jsc@tmpa{#{1}-#{2}-#{3}-#{4}-{\strip@pt\@tempdimc}}
1035 \expandafter\jsc@DeclareErrorFont\jsc@tmpa}
1036 \def\gen@sfcnt{%
1037 \edef\mandatory@arg{\mandatory@arg\jsc@cref@size}%
1038 \empty@sfcnt}
1039 \def\genb@sfcnt{%
1040 \edef\mandatory@arg{%
1041 \mandatory@arg\expandafter\genb@x\jsc@cref@size..\@@}%
1042 \empty@sfcnt}
1043 \ifxjs@TUenc\else
1044 \DeclareErrorFont{OT1}{cmr}{m}{n}{10}
1045 \fi
1046 \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 1047 \def\jsc@smallskip{\vspace\jsc@smallskipamount}
\jsc@bigskip 1048 %\def\jsc@medskip{\vspace\jsc@medskipamount}
1049 %\def\jsc@bigskip{\vspace\jsc@bigskipamount}

```

```

\jsc@smallskipamount
\jsc@medskipamount 1050 \newskip\jsc@smallskipamount
1051 \jsc@smallskipamount=3\jsc@mpt plus 1\jsc@mpt minus 1\jsc@mpt
\jsc@bigskipamount 1052 %\newskip\jsc@medskipamount
1053 %\jsc@medskipamount =6\jsc@mpt plus 2\jsc@mpt minus 2\jsc@mpt
1054 %\newskip\jsc@bigskipamount
1055 %\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。

```
1056 % \ifpapersize
1057 %   \setlength{\@tempdima}{\paperwidth}
1058 %   \setlength{\@tempdimb}{\paperheight}
1059 %   \iftombow
1060 %     \advance \@tempdima 2truein
1061 %     \advance \@tempdimb 2truein
1062 %   \fi
1063 %   \AtBeginDvi{\special{pdf: pagesize width \the\@tempdima\space height \the\@tempdimb}}
1064 % \fi
```

3 和文フォントの変更

和文フォントの設定は和文ドライバの管轄。

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

```
1065 %</class>
1066 %<*jsclasses>
1067 %<*class>
```

JIS の 1 ポイントは 0.3514mm (約 1/72.28 インチ)、PostScript の 1 ポイントは 1/72 インチですが、 $\text{T}_{\text{E}}\text{X}$ では 1/72.27 インチを 1pt (ポイント)、1/72 インチを 1bp (ビッグポイント) と表します。QuarkXPress などの DTP ソフトは標準で 1/72 インチを 1 ポイントとしますが、以下ではすべて 1/72.27 インチを 1pt としています。1 インチは定義により 25.4mm です。

さらにややこしいことに、 $\text{pT}_{\text{E}}\text{X}$ (アスキーが日本語化した $\text{T}_{\text{E}}\text{X}$) の公称 10 ポイントの和文フォント (`min10` など) は、実寸 (標準の字送り量) が 9.62216pt です。これは 3.3818mm、写研の写植機の単位では 13.527 級、PostScript の単位では 9.5862 ポイントになります。`jis` フォントなどもこの値を踏襲しています。

この公称 10 ポイントのフォントを、ここでは 13 級に縮小して使うことにします。そのためには、 $13/13.527 = 0.961$ 倍すればいいことになります (`min10` や `jis` の場合)。9.62216 ポイントの和文フォントをさらに 0.961 倍したことにより、約 9.25 ポイント、DTP で使う単位 (1/72 インチ) では 9.21 ポイントということになり、公称 10 ポイントといっても実は 9 ポイント強になります。

[2018-02-04] 上記のとおり「クラスファイルが意図する和文スケール値 (1zw ÷ 要求サイズ)」を表す実数値マクロ `\Cjascale` を定義します。このマクロが定義されている場合、OTF パッケージ (2018/02/01 以降のバージョン) はこれに従います。`jsarticle`、`jsbook`、`jsreport` では、 $9.62216 \text{ pt} * 0.961 / 10 \text{ pt} = 0.924690$ です。

```

1068 %</class>
1069 %<*minijs>
1070 %% min/goth -> jis/jisg (for pLaTeX only)
1071 \ifnum\jis"2121="3000 \else
1072 \@for\@tempa:=5,6,7,8,9,10,10.95,12,14.4,17.28,20.74,24.88\do{%
1073   \expandafter\let\csname JY1/mc/m/n/\@tempa\endcsname\relax
1074   \expandafter\let\csname JY1/gt/m/n/\@tempa\endcsname\relax
1075   \expandafter\let\csname JT1/mc/m/n/\@tempa\endcsname\relax
1076   \expandafter\let\csname JT1/gt/m/n/\@tempa\endcsname\relax
1077 }
1078 \def\Cjascale{0.924690}
1079 \DeclareFontShape{JY1}{mc}{m}{n}{<-> s * [0.961] jis}{}
1080 \DeclareFontShape{JY1}{gt}{m}{n}{<-> s * [0.961] jisg}{}
1081 \DeclareFontShape{JT1}{mc}{m}{n}{<-> s * [0.961] tmin10}{}
1082 \DeclareFontShape{JT1}{gt}{m}{n}{<-> s * [0.961] tgoth10}{}
1083 \fi
1084 %</minijs>
1085 %<*class>
1086 %<!*jspf>
1087 \def\Cjascale{0.924690}
1088 \ifmingoth
1089   \DeclareFontShape{\jsc@JYn}{mc}{m}{n}{<-> s * [0.961] \jsc@pfx@ min10}{}
1090   \DeclareFontShape{\jsc@JYn}{gt}{m}{n}{<-> s * [0.961] \jsc@pfx@ goth10}{}
1091   \DeclareFontShape{\jsc@JTn}{mc}{m}{n}{<-> s * [0.961] \jsc@pfx@ tmin10}{}
1092   \DeclareFontShape{\jsc@JTn}{gt}{m}{n}{<-> s * [0.961] \jsc@pfx@ tgoth10}{}
1093 \else
1094   \ifjisfont
1095     \DeclareFontShape{\jsc@JYn}{mc}{m}{n}{<-> s * [0.961] \jsc@pfx@ jis}{}
1096     \DeclareFontShape{\jsc@JYn}{gt}{m}{n}{<-> s * [0.961] \jsc@pfx@ jisg}{}
1097     \DeclareFontShape{\jsc@JTn}{mc}{m}{n}{<-> s * [0.961] \jsc@pfx@ tmin10}{}
1098     \DeclareFontShape{\jsc@JTn}{gt}{m}{n}{<-> s * [0.961] \jsc@pfx@ tgoth10}{}
1099   \else
1100     \if@jsc@uplatex
1101       \DeclareFontShape{JY2}{mc}{m}{n}{<-> s * [0.924690] upjisr-h}{}
1102       \DeclareFontShape{JY2}{gt}{m}{n}{<-> s * [0.924690] upjisg-h}{}
1103       \DeclareFontShape{JT2}{mc}{m}{n}{<-> s * [0.924690] upjisr-v}{}
1104       \DeclareFontShape{JT2}{gt}{m}{n}{<-> s * [0.924690] upjisg-v}{}
1105     \else
1106       \DeclareFontShape{\jsc@JYn}{mc}{m}{n}{<-> s * [0.961] \jsc@pfx@ jis}{}
1107       \DeclareFontShape{\jsc@JYn}{gt}{m}{n}{<-> s * [0.961] \jsc@pfx@ jisg}{}
1108       \DeclareFontShape{\jsc@JTn}{mc}{m}{n}{<-> s * [0.961] \jsc@pfx@ tmin10}{}
1109       \DeclareFontShape{\jsc@JTn}{gt}{m}{n}{<-> s * [0.961] \jsc@pfx@ tgoth10}{}
1110     \fi
1111   \fi
1112 \fi
1113 %</!*jspf>

```

某学会誌では、和文フォントを PostScript の 9 ポイントにするために、 $9/(9.62216 * 72/72.27) = 0.93885$ 倍します。

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

```
1114 %<*jspf>
1115 \def\Cjascale{0.903375}
1116 \ifmingoth
1117 \DeclareFontShape{\jsc@JYn}{mc}{m}{n}{<-> s * [0.93885] \jsc@pfx@ min10}{}
1118 \DeclareFontShape{\jsc@JYn}{gt}{m}{n}{<-> s * [0.93885] \jsc@pfx@ goth10}{}
1119 \DeclareFontShape{\jsc@JTn}{mc}{m}{n}{<-> s * [0.93885] \jsc@pfx@ tmin10}{}
1120 \DeclareFontShape{\jsc@JTn}{gt}{m}{n}{<-> s * [0.93885] \jsc@pfx@ tgoth10}{}
1121 \else
1122 \ifjisfont
1123 \DeclareFontShape{\jsc@JYn}{mc}{m}{n}{<-> s * [0.93885] \jsc@pfx@ jis}{}
1124 \DeclareFontShape{\jsc@JYn}{gt}{m}{n}{<-> s * [0.93885] \jsc@pfx@ jisg}{}
1125 \DeclareFontShape{\jsc@JTn}{mc}{m}{n}{<-> s * [0.93885] \jsc@pfx@ tmin10}{}
1126 \DeclareFontShape{\jsc@JTn}{gt}{m}{n}{<-> s * [0.93885] \jsc@pfx@ tgoth10}{}
1127 \else
1128 \if@jsc@uplatex
1129 \DeclareFontShape{JY2}{mc}{m}{n}{<-> s * [0.903375] upjisr-h}{}
1130 \DeclareFontShape{JY2}{gt}{m}{n}{<-> s * [0.903375] upjisg-h}{}
1131 \DeclareFontShape{JT2}{mc}{m}{n}{<-> s * [0.903375] upjisr-v}{}
1132 \DeclareFontShape{JT2}{gt}{m}{n}{<-> s * [0.903375] upjisg-v}{}
1133 \else
1134 \DeclareFontShape{\jsc@JYn}{mc}{m}{n}{<-> s * [0.93885] \jsc@pfx@ jis}{}
1135 \DeclareFontShape{\jsc@JYn}{gt}{m}{n}{<-> s * [0.93885] \jsc@pfx@ jisg}{}
1136 \DeclareFontShape{\jsc@JTn}{mc}{m}{n}{<-> s * [0.93885] \jsc@pfx@ tmin10}{}
1137 \DeclareFontShape{\jsc@JTn}{gt}{m}{n}{<-> s * [0.93885] \jsc@pfx@ tgoth10}{}
1138 \fi
1139 \fi
1140 \fi
1141 %</jspf>
```

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

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

[2004-11-03] \rmfamily も和文対応にしました。

```
1142 % \DeclareFontShape{\jsc@JYn}{mc}{bx}{n}{<->ssub*gt/m/n}{} % in \jsc@JYnmc
1143 % \DeclareFontShape{\jsc@JYn}{gt}{bx}{n}{<->ssub*gt/m/n}{} % in \jsc@JYngt
1144 \DeclareFontShape{\jsc@JYn}{mc}{m}{it}{<->ssub*mc/m/n}{}
1145 \DeclareFontShape{\jsc@JYn}{mc}{m}{sl}{<->ssub*mc/m/n}{}
1146 \DeclareFontShape{\jsc@JYn}{mc}{m}{sc}{<->ssub*mc/m/n}{}
1147 \DeclareFontShape{\jsc@JYn}{gt}{m}{it}{<->ssub*gt/m/n}{}
1148 \DeclareFontShape{\jsc@JYn}{gt}{m}{sl}{<->ssub*gt/m/n}{}
1149 \DeclareFontShape{\jsc@JYn}{mc}{bx}{it}{<->ssub*gt/m/n}{}
```

```

1150 \DeclareFontShape{\jsc@JYn}{mc}{bx}{sl}{<->ssub*gt/m/n}{}
1151 % \DeclareFontShape{\jsc@JTn}{mc}{bx}{n}{<->ssub*gt/m/n}{} % in \jsc@JTnmc
1152 % \DeclareFontShape{\jsc@JTn}{gt}{bx}{n}{<->ssub*gt/m/n}{} % in \jsc@JTngt
1153 \DeclareFontShape{\jsc@JTn}{mc}{m}{it}{<->ssub*mc/m/n}{}
1154 \DeclareFontShape{\jsc@JTn}{mc}{m}{sl}{<->ssub*mc/m/n}{}
1155 \DeclareFontShape{\jsc@JTn}{mc}{m}{sc}{<->ssub*mc/m/n}{}
1156 \DeclareFontShape{\jsc@JTn}{gt}{m}{it}{<->ssub*gt/m/n}{}
1157 \DeclareFontShape{\jsc@JTn}{gt}{m}{sl}{<->ssub*gt/m/n}{}
1158 \DeclareFontShape{\jsc@JTn}{mc}{bx}{it}{<->ssub*gt/m/n}{}
1159 \DeclareFontShape{\jsc@JTn}{mc}{bx}{sl}{<->ssub*gt/m/n}{}

```

[2020-02-02] L^AT_EX 2_ε 2020-02-02 で NFSS が拡張され、それに伴いオリジナルの `\rmfamily` などの定義が変化しました。`\DeclareRobustCommand` で直接定義すると、これを上書きして NFSS の拡張部分を壊してしまいますので、新たに提供されたフックにコードを挿入します。従来のコードも L^AT_EX 2_ε 2019-10-01 以前のために残してありますが、`mweights` パッケージ対策も施しました (forum:2763)。

[2020-10-04] L^AT_EX 2_ε 2020-10-01 では `\AddToHook` を利用します。

```

1160 %</class>
1161 %<*class|minijs>
1162 %% ad-hoc "relation font"
1163 \@ifl@t@r\fmtversion{2020/10/01}
1164     {\jsc@needsp@tchfalse}{\jsc@needsp@tchtrue}
1165 \ifjsc@needsp@tch          % --- for 2020-02-02 or older BEGIN
1166 \ifx\@rmfamilyhook\@undefined % old
1167 \DeclareRobustCommand\rmfamily
1168     {\not@math@alphabet\rmfamily\mathrm
1169     \romanfamily\rmdefault\kanjifamily\mcdefault\selectfont}
1170 \DeclareRobustCommand\sffamily
1171     {\not@math@alphabet\sffamily\mathsf
1172     \romanfamily\sfdefault\kanjifamily\gtdefault\selectfont}
1173 \DeclareRobustCommand\ttfamily
1174     {\not@math@alphabet\ttfamily\mathtt
1175     \romanfamily\ttdefault\kanjifamily\gtdefault\selectfont}
1176 \AtBeginDocument{%
1177   \ifx@mweights@init\@undefined\else % mweights.sty is loaded
1178     % my definitions above should have been overwritten, recover it!
1179     % \selectfont is executed twice but I don't care about speed...
1180     \expandafter@g@addto@macro\csname rmfamily \endcsname
1181       {\kanjifamily\mcdefault\selectfont}%
1182     \expandafter@g@addto@macro\csname sffamily \endcsname
1183       {\kanjifamily\gtdefault\selectfont}%
1184     \expandafter@g@addto@macro\csname ttfamily \endcsname
1185       {\kanjifamily\gtdefault\selectfont}%
1186   \fi}
1187 \else          % 2020-02-02
1188 \g@addto@macro\@rmfamilyhook
1189   {\prepare@family@series@update@kanji{mc}\mcdefault}
1190 \g@addto@macro\@sffamilyhook

```

```

1191 {\prepare@family@series@update@kanji{gt}}\gtdefault}
1192 \g@addto@macro\@ttfamilyhook
1193 {\prepare@family@series@update@kanji{gt}}\gtdefault}
1194 \fi
1195 \else % --- for 2020-02-02 or older END & for 2020-10-01 BEGIN
1196 \AddToHook{rmfamily}%
1197 {\prepare@family@series@update@kanji{mc}}\mcdefault}
1198 \AddToHook{sffamily}%
1199 {\prepare@family@series@update@kanji{gt}}\gtdefault}
1200 \AddToHook{ttfamily}%
1201 {\prepare@family@series@update@kanji{gt}}\gtdefault}
1202 \fi % --- for 2020-10-01 END
1203 %</class|minijs>
1204 %<*class>

```

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

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

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

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

```

1205 \ifx\DeclareFixJFMCJKTextFontCommand\@undefined
1206 \DeclareRobustCommand\textmc[1]{%
1207   \relax\ifmmode \expandafter\nfss@text \fi{\mcfamily #1}}
1208 \DeclareRobustCommand\textgt[1]{%
1209   \relax\ifmmode \expandafter\nfss@text \fi{\gtfamily #1}}
1210 \fi

```

新クラスでも `disablejfam` オプションを与えなければ数式内で日本語が使えるようにしました。

さらに 2005/12/01 版の LaTeX に対応した pLaTeX に対応しました (Thanks: ymt さん)。

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

```

1211 \def\reDeclareMathAlphabet#1#2#3{%
1212   \edef\@tempa{\expandafter\@gobble\string#2}%
1213   \edef\@tempb{\expandafter\@gobble\string#3}%

```



```

1214 \edef\@tempc{\string @\expandafter@gobbletwo\string#2}%
1215 \ifx\@tempc\@tempa%
1216   \edef\@tempa{\expandafter@gobbletwo\string#2}%
1217   \edef\@tempb{\expandafter@gobbletwo\string#3}%
1218   \fi
1219   \begingroup
1220     \let\protect\noexpand
1221     \def\@tempaa{\relax}%
1222     \expandafter\ifx\csname RDMAorg@\@tempa\endcsname\relax
1223       \edef\@tempaa{\expandafter\def\expandafter\noexpand%
1224         \csname RDMAorg@\@tempa\endcsname{%
1225           \expandafter\noexpand\csname\@tempa\endcsname}}%
1226       \fi
1227       \def\@tempbb{\relax}%
1228       \expandafter\ifx\csname RDMAorg@\@tempb\endcsname\relax
1229         \edef\@tempbb{\expandafter\def\expandafter\noexpand%
1230           \csname RDMAorg@\@tempb\endcsname{%
1231             \expandafter\noexpand\csname\@tempb\endcsname}}%
1232         \fi
1233         \edef\@tempc{\@tempaa\@tempbb}%
1234     \expandafter\endgroup\@tempc%
1235     \edef#1{\noexpand\protect\expandafter\noexpand\csname%
1236       \expandafter@gobble\string#1\space\space\endcsname}%
1237     \expandafter\edef\csname\expandafter@gobble\string#1\space\space\endcsname%
1238       {\noexpand\DualLang@mathalph@bet%
1239         {\expandafter\noexpand\csname RDMAorg@\@tempa\endcsname}%
1240         {\expandafter\noexpand\csname RDMAorg@\@tempb\endcsname}%
1241       }%
1242   }
1243 \onlypreamble\reDeclareMathAlphabet
1244 \def\DualLang@mathalph@bet#1#2{%
1245   \relax\ifmmode
1246     \ifx\math@bgroup\bgroup%      2e normal style (\mathrm{...})
1247       \bgroup\let\DualLang@Mfontsw\DLMfontsw@standard
1248     \else
1249       \ifx\math@bgroup\relax%     2e two letter style (\rm->\mathrm)
1250         \let\DualLang@Mfontsw\DLMfontsw@oldstyle
1251       \else
1252         \ifx\math@bgroup\@empty% 2.09 oldfont style ({\mathrm ...})
1253           \let\DualLang@Mfontsw\DLMfontsw@oldfont
1254         \else%                      panic! assume 2e normal style
1255           \bgroup\let\DualLang@Mfontsw\DLMfontsw@standard
1256         \fi
1257       \fi
1258     \fi
1259   \else
1260     \let\DualLang@Mfontsw\@firstoftwo
1261   \fi
1262   \DualLang@Mfontsw{#1}{#2}%

```

```

1263 }
1264 \def\DLMfontsw@standard#1#2#3{#1{#2{#3}}\egroup}
1265 \def\DLMfontsw@oldstyle#1#2{#1\relax\@fontswitch\relax{#2}}
1266 \def\DLMfontsw@oldfont#1#2{#1\relax#2\relax}
1267 \if@enablejfam
1268 \DeclareSymbolFont{mincho}{\jsc@JYn}{mc}{m}{n}
1269 \DeclareSymbolFontAlphabet{\mathmc}{mincho}
1270 \SetSymbolFont{mincho}{bold}{\jsc@JYn}{gt}{m}{n}
1271 \jfam\symmincho
1272 \DeclareMathAlphabet{\mathgt}{\jsc@JYn}{gt}{m}{n}
1273 \AtBeginDocument{%
1274 \reDeclareMathAlphabet{\mathrm}{\@mathrm}{\@mathmc}
1275 \reDeclareMathAlphabet{\mathbf}{\@mathbf}{\@mathgt}}
1276 \fi

```

`\textsterling` これは `\pounds` 命令で実際に呼び出される文字です。従来からの OT1 エンコーディングでは `\$` のイタリック体が `\pounds` なので `cmti` が使われていましたが、1994 年春からは `cmu` (upright italic, 直立イタリック体) に変わりました。しかし `cmu` はその性格からして実験的なものであり、`\pounds` 以外で使われるとは思えないので、ここでは `cmti` に戻してしまいます。

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

```
1277 % \DeclareTextCommand{\textsterling}{OT1}{\itshape\char`$}
```

禁則パラメータも若干修正します。

アスキーの `kinsoku.dtx` では次の三つが 5000 に設定されています。これを 10000 に再設定します。

```

1278 \prebreakpenalty\jis"2147=10000      % 5000  '
1279 \postbreakpenalty\jis"2148=10000     % 5000  "
1280 \prebreakpenalty\jis"2149=10000     % 5000  "

```

「`TEX!`」「`〒515`」の記号と数字の間に四分アキが入らないようにします。

```

1281 \inhibitxspcode`!=1
1282 \inhibitxspcode`〒=2

```

以前の版では、たとえば「ベース名. 拡張子」のように和文文字で書いたとき、ピリオドの後に四分アキが入らないようにするために

```
1283 % \xspcode`. =0
```

のようにしていました。ただ、「Foo Inc. は……」のように書いたときにもスペースが入らなくなるので、ちょっとまずい修正だったかもしれません。元に戻しました。

とりあえず「ベース名.`\mbox{}`拡張子」と書いてください。

「C や C++ では……」と書くと、C++ の直後に四分アキが入らないのでバランスが悪くなります。四分アキが入るようにしました。% の両側も同じです。

```

1284 \xspcode`+=3
1285 \xspcode`\%=3

```

これ以外に T1 エンコーディングで 80~ff の文字もすべて欧文文字ですので、両側の和文文字との間にスペースが入らなければなりません。

1286 \xspcode`^^80=3
1287 \xspcode`^^81=3
1288 \xspcode`^^82=3
1289 \xspcode`^^83=3
1290 \xspcode`^^84=3
1291 \xspcode`^^85=3
1292 \xspcode`^^86=3
1293 \xspcode`^^87=3
1294 \xspcode`^^88=3
1295 \xspcode`^^89=3
1296 \xspcode`^^8a=3
1297 \xspcode`^^8b=3
1298 \xspcode`^^8c=3
1299 \xspcode`^^8d=3
1300 \xspcode`^^8e=3
1301 \xspcode`^^8f=3
1302 \xspcode`^^90=3
1303 \xspcode`^^91=3
1304 \xspcode`^^92=3
1305 \xspcode`^^93=3
1306 \xspcode`^^94=3
1307 \xspcode`^^95=3
1308 \xspcode`^^96=3
1309 \xspcode`^^97=3
1310 \xspcode`^^98=3
1311 \xspcode`^^99=3
1312 \xspcode`^^9a=3
1313 \xspcode`^^9b=3
1314 \xspcode`^^9c=3
1315 \xspcode`^^9d=3
1316 \xspcode`^^9e=3
1317 \xspcode`^^9f=3
1318 \xspcode`^^a0=3
1319 \xspcode`^^a1=3
1320 \xspcode`^^a2=3
1321 \xspcode`^^a3=3
1322 \xspcode`^^a4=3
1323 \xspcode`^^a5=3
1324 \xspcode`^^a6=3
1325 \xspcode`^^a7=3
1326 \xspcode`^^a8=3
1327 \xspcode`^^a9=3
1328 \xspcode`^^aa=3
1329 \xspcode`^^ab=3
1330 \xspcode`^^ac=3
1331 \xspcode`^^ad=3

1332 \xspcode`^^ae=3
1333 \xspcode`^^af=3
1334 \xspcode`^^b0=3
1335 \xspcode`^^b1=3
1336 \xspcode`^^b2=3
1337 \xspcode`^^b3=3
1338 \xspcode`^^b4=3
1339 \xspcode`^^b5=3
1340 \xspcode`^^b6=3
1341 \xspcode`^^b7=3
1342 \xspcode`^^b8=3
1343 \xspcode`^^b9=3
1344 \xspcode`^^ba=3
1345 \xspcode`^^bb=3
1346 \xspcode`^^bc=3
1347 \xspcode`^^bd=3
1348 \xspcode`^^be=3
1349 \xspcode`^^bf=3
1350 \xspcode`^^c0=3
1351 \xspcode`^^c1=3
1352 \xspcode`^^c2=3
1353 \xspcode`^^c3=3
1354 \xspcode`^^c4=3
1355 \xspcode`^^c5=3
1356 \xspcode`^^c6=3
1357 \xspcode`^^c7=3
1358 \xspcode`^^c8=3
1359 \xspcode`^^c9=3
1360 \xspcode`^^ca=3
1361 \xspcode`^^cb=3
1362 \xspcode`^^cc=3
1363 \xspcode`^^cd=3
1364 \xspcode`^^ce=3
1365 \xspcode`^^cf=3
1366 \xspcode`^^d0=3
1367 \xspcode`^^d1=3
1368 \xspcode`^^d2=3
1369 \xspcode`^^d3=3
1370 \xspcode`^^d4=3
1371 \xspcode`^^d5=3
1372 \xspcode`^^d6=3
1373 \xspcode`^^d7=3
1374 \xspcode`^^d8=3
1375 \xspcode`^^d9=3
1376 \xspcode`^^da=3
1377 \xspcode`^^db=3
1378 \xspcode`^^dc=3
1379 \xspcode`^^dd=3
1380 \xspcode`^^de=3

```

1381 \xspcode^^df=3
1382 \xspcode^^e0=3
1383 \xspcode^^e1=3
1384 \xspcode^^e2=3
1385 \xspcode^^e3=3
1386 \xspcode^^e4=3
1387 \xspcode^^e5=3
1388 \xspcode^^e6=3
1389 \xspcode^^e7=3
1390 \xspcode^^e8=3
1391 \xspcode^^e9=3
1392 \xspcode^^ea=3
1393 \xspcode^^eb=3
1394 \xspcode^^ec=3
1395 \xspcode^^ed=3
1396 \xspcode^^ee=3
1397 \xspcode^^ef=3
1398 \xspcode^^f0=3
1399 \xspcode^^f1=3
1400 \xspcode^^f2=3
1401 \xspcode^^f3=3
1402 \xspcode^^f4=3
1403 \xspcode^^f5=3
1404 \xspcode^^f6=3
1405 \xspcode^^f7=3
1406 \xspcode^^f8=3
1407 \xspcode^^f9=3
1408 \xspcode^^fa=3
1409 \xspcode^^fb=3
1410 \xspcode^^fc=3
1411 \xspcode^^fd=3
1412 \xspcode^^fe=3
1413 \xspcode^^ff=3

1414 %</class>
1415 %</jsclasses>
1416 %<*class>

```

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

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

BXJS クラスでの変更点：

- `fix-at-cmd` オプションが偽の場合は再定義しない。
- 固定の 3000 でなく実際のピリオドの `scode` 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。

```

1417 \chardef\bxjs@periodchar=`\ .
1418 \bxjs@protected\def\bxjs@SE{%
1419   \ifnum\spacefactor<\@m \spacefactor\@m
1420   \else \spacefactor\sfcode\bxjs@periodchar
1421   \fi}
1422 \ifbxjs@fix@at@cmd
1423   \def\@{\bxjs@SE{}}
1424 \fi

```

4 フォントサイズ

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

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

のようにして行います。これは

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

という意味です。ただし、処理を速くするため、以下では 10 と同義の L^AT_EX の内部命令 `\@xpt` を使っています。この `\@xpt` の類は次のものがあり、L^AT_EX 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>\@xipt</code>	12	<code>\@xivpt</code>	14.4

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

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

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

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

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

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

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

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

TODO:3.0 新しい L^AT_EX カーネルで `selectfont` フックを利用する。

```

1425 %\let\bxjs@prev@set@fontsize\@undefined
1426 \@onlypreamble\bxjs@patch@set@fontsize
1427 \def\bxjs@patch@set@fontsize{%
1428   \ifx\bxjs@prev@set@fontsize\set@fontsize\else
1429     \def\bxjs@tmpa{\def\set@fontsize####1####2####3}%
1430 \expandafter\bxjs@tmpa\expandafter{%
1431   \set@fontsize{##1}{##2}{##3}%
1432 % 末尾にコードを追加
1433   \expandafter\def\expandafter\size@update\expandafter{%
1434     \size@update
1435     \jsFontSizeChanged}%
1436 }
1437   \let\bxjs@prev@set@fontsize\set@fontsize
1438   \fi}

```

この場とパッケージ末尾で作動させる。

```

1439 \bxjs@patch@set@fontsize
1440 \AtEndOfClass{\bxjs@patch@set@fontsize}

```

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

```

1441 \newcommand*\jsFontSizeChanged{%
1442   \jsZw=\f@size\p@
1443   \jsZw=\jsScale \jsZw
1444   \ifdim\parindent>\z@
1445     \if@english \parindent=1em
1446     \else       \parindent=1\jsZw
1447   \fi
1448   \fi\relax
1449   \jsResetDimen}

```

`\jsResetDimen` ユーザ定義用のフック。

```

1450 \providecommand*\jsResetDimen{}

```

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

```

1451 \ifjsc@mag
1452 \let\jsc@setfontsize\setfontsize
1453 \else
1454 \def\jsc@setfontsize#1#2#3{%
1455   \@setfontsize#1{#2\jsc@mpt}{#3\jsc@mpt}}
1456 % microtype 対策
1457 \ifjsWitheTeX\if j\jsEngine\else
1458   \def\jsc@setfontsize#1#2#3{%
1459     \edef\bxjs@sfs@next{%
1460       \unexpanded{\@setfontsize#1}%
1461       {\the\dimexpr#2\jsc@mpt\relax}{\the\dimexpr#3\jsc@mpt\relax}%
1462     }\bxjs@sfs@next}
1463 \fi\fi
1464 \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。

```

1465 \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、しっぽ愛好家さんが次の修正を教えてくださいました。

[2008-02-18] `english` オプションで最初の段落のインデントをしないようにしました。

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

別行立て数式に入るときに `\narrowbaselines` が呼ばれるが、このコードでは「数式中で `\normalsize` などのサイズ命令 (`\@currsiz` 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。

```

1466 \newif\ifnarrowbaselines
1467 \if@english
1468   \narrowbaselinestrue
1469 \fi
1470 \def\narrowbaselines{%
1471   \narrowbaselinestrue
1472   \skip0=\abovedisplayskip
1473   \skip2=\abovedisplayshortskip
1474   \skip4=\belowdisplayskip

```



```

1475 \skip6=\belowdisplayshortskip
1476 % 一時的に警告を無効化する
1477 \let\bxjs@save@nomath\@nomath
1478 \let\@nomath\@gobble
1479 \@currsize\selectfont
1480 \let\@nomath\bxjs@save@nomath
1481 \abovedisplayskip=\skip0
1482 \abovedisplayshortskip=\skip2
1483 \belowdisplayskip=\skip4
1484 \belowdisplayshortskip=\skip6\relax}
1485 \def\widebaselines{\narrowbaselinesfalse\@currsize\selectfont}

```

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

`\bxjs@if@narrowbaselines` スイッチ `narrowbaselines` を L^AT_EX 式条件文にしたもの。

```

1486 \def\bxjs@if@narrowbaselines{%
1487   \ifnarrowbaselines\expandafter\@firstoftwo
1488   \else \expandafter\@secondoftwo
1489   \fi
1490 }

```

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

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

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

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

```

1491 \renewcommand{\normalsize}{%
1492   \bxjs@if@narrowbaselines{%
1493     \jsc@setfontsize\normalsize\@xpt\@xiipt
1494   }{%else
1495     \jsc@setfontsize\normalsize\@xpt{\n@baseline}%
1496   }%

```

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。

```
1497 \abovedisplayskip 11\jsc@mpt \@plus3\jsc@mpt \@minus4\jsc@mpt
1498 \abovedisplayshortskip \z@ \@plus3\jsc@mpt
1499 \belowdisplayskip 9\jsc@mpt \@plus3\jsc@mpt \@minus4\jsc@mpt
1500 \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。

```
1501 \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。

```
1502 %</class>
1503 %<*class|minijs>
1504 %% initialize
1505 \normalsize
1506 %</class|minijs>
1507 %<*class>
```

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

`\Chs`

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

```
1508 \ifx\Cht\undefined \newdimen\Cht \fi
1509 \ifx\Cdp\undefined \newdimen\Cdp \fi
1510 \ifx\Cwd\undefined \newdimen\Cwd \fi
1511 \ifx\Cvs\undefined \newdimen\Cvs \fi
1512 \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。

```
1513 \setlength\Cht{0.88\jsZw}
1514 \setlength\Cdp{0.12\jsZw}
1515 \setlength\Cwd{1\jsZw}
1516 \setlength\Cvs{\baselineskip}
1517 \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。

```
1518 \newcommand{\small}{%
1519 \bxjs@if@narrowbaselines{%
1520 %<!kiyou> \jsc@setfontsize\small\@ixpt{11}%
```

```

1521 %<kiyou> \jsc@setfontsize\small{8.8888}{11}%
1522 }{%else
1523 %<!kiyou> \jsc@setfontsize\small\@ixpt{13}%
1524 %<kiyou> \jsc@setfontsize\small{8.8888}{13.2418}%
1525 }%
1526 \abovedisplayskip 9\jsc@empt \@plus3\jsc@empt \@minus4\jsc@empt
1527 \abovedisplayshortskip \z@ \@plus3\jsc@empt
1528 \belowdisplayskip \abovedisplayskip
1529 \belowdisplayshortskip \belowdisplayskip
1530 \def\@listi{\leftmargin\leftmargini
1531 \topsep \z@
1532 \parsep \z@
1533 \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。

```

1534 \newcommand{\footnotesize}{%
1535 \bxjs@if@narrowbaselines{%
1536 %<!kiyou> \jsc@setfontsize\footnotesize\@viipt{9.5}%
1537 %<kiyou> \jsc@setfontsize\footnotesize{8.8888}{11}%
1538 }{%else
1539 %<!kiyou> \jsc@setfontsize\footnotesize\@viipt{11}%
1540 %<kiyou> \jsc@setfontsize\footnotesize{8.8888}{13.2418}%
1541 }%
1542 \abovedisplayskip 6\jsc@empt \@plus2\jsc@empt \@minus3\jsc@empt
1543 \abovedisplayshortskip \z@ \@plus2\jsc@empt
1544 \belowdisplayskip \abovedisplayskip
1545 \belowdisplayshortskip \belowdisplayskip
1546 \def\@listi{\leftmargin\leftmargini
1547 \topsep \z@
1548 \parsep \z@
1549 \itemsep \parsep}}

```

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

`\LARGE` [2004-11-03] `\HUGE` を追加。

```

\huge 1550 \newcommand{\scriptsize}{\jsc@setfontsize\scriptsize\@viipt\@viipt}
1551 \newcommand{\tiny}{\jsc@setfontsize\tiny\@vpt\@vpt}
\Huge 1552 \if@twocolumn
\HUGE 1553 %<!kiyou> \newcommand{\large}{\jsc@setfontsize\large\@xipt{\n@baseline}}
1554 %<kiyou> \newcommand{\large}{\jsc@setfontsize\large{11.111}{\n@baseline}}
1555 \else
1556 %<!kiyou> \newcommand{\large}{\jsc@setfontsize\large\@xipt{17}}
1557 %<kiyou> \newcommand{\large}{\jsc@setfontsize\large{11.111}{17}}
1558 \fi
1559 %<!kiyou>\newcommand{\Large}{\jsc@setfontsize\Large\@xivpt{21}}

```

```

1560 %<kiyou>\newcommand{\Large}{\jsc@setfontsize\Large{12.222}{21}}
1561 \newcommand{\LARGE}{\jsc@setfontsize\LARGE\@xvipt{25}}
1562 \newcommand{\huge}{\jsc@setfontsize\huge\@xxpt{28}}
1563 \newcommand{\Huge}{\jsc@setfontsize\Huge\@xxvpt{33}}
1564 \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。

```

1565 \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。とりえず `amsmath` の `equation` 関係は `okumacro` のほうで逃げていますが、もっとうまい逃げ道があれば教えてください。

見出し用のフォントは `\bfseries` 固定ではなく、`\headfont` という命令で定めることにします。これは太ゴシックが使えるときは `\sffamily \bfseries` でいいと思いますが、通常の中ゴシックでは単に `\sffamily` だけのほうがよさそうです。『`LaTeX 2ε` 美文書作成入門』(1997年)では `\sffamily \fontseries{sbc}` として新ゴ M と合わせましたが、`\fontseries{sbc}` はちょっと幅が狭いように感じました。

```

1566 % \newcommand{\headfont}{\bfseries}
1567 \newcommand{\headfont}{\sffamily}
1568 % \newcommand{\headfont}{\sffamily\fontseries{sbc}\selectfont}

```

5 レイアウト

■二段組

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

```

1569 %<!kiyou>\setlength\columnsep{2\Cwd}
1570 %<kiyou>\setlength\columnsep{28truebp}
1571 \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 1572 \setlength\lineskip{1\jsc@mpt}
\normallineskip 1573 \setlength\normallineskip{1\jsc@mpt}
\normallineskiplimit 1574 \setlength\lineskiplimit{1\jsc@mpt}
1575 \setlength\normallineskiplimit{1\jsc@mpt}

```

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

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

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

```
1576 \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。

```
1577 \setlength{\parskip}{\z@}
1578 \if@slide
1579 \setlength{\parindent}{0\p@}
1580 \else
1581 \setlength{\parindent}{1\Cwd}
1582 \fi
```

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

```
\@highpenalty 1583 \@lowpenalty 51
1584 \@medpenalty 151
1585 \@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。

```
1586 % \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。

```
1587 % \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。

```
1588 \onlypreamble\bxjs@bd@pre@geometry@hook
1589 \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。

```
1590 \ifjsc@mag
1591 \mag=\bxjs@param@mag
1592 \fi
1593 \setlength{\topskip}{10\jsc@empt}
```

`\jsSetQHLlength` のための和文単位の定義。

```
1594 \def\bxjs@unit@trueQ{0.25trueem}\let\bxjs@unit@trueH\bxjs@unit@trueQ
1595 \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。

```
1596 \@tempwafalse
1597 \def\bxjs@tmpdo{\@ifnextchar\bgroup\bxjs@tmpdo@a\remove@to@nnil}
1598 \def\bxjs@tmpdo@a#1{\edef\bxjs@tmpa{#1}%
1599   \@ifnextchar\bgroup\bxjs@tmpdo@b\remove@to@nnil}
1600 \def\bxjs@tmpdo@b#1{\edef\bxjs@tmpa{\bxjs@tmpa,#1}%
1601   \@ifnextchar\@nnil\bxjs@tmpdo@c\remove@to@nnil}
1602 \def\bxjs@tmpdo@c\@nnil{\@tempwatrue
1603   \edef\bxjs@param@paper{papersize={\bxjs@tmpa}}}
1604 \expandafter\bxjs@tmpdo\bxjs@param@paper\@nnil
```

`W,H` の形式について。

```
1605 \if@tempwa\else
1606   \def\bxjs@tmpa{\@nil,\@nil}
1607   \def\bxjs@tmpdo#1,#2,#3\@nnil{%
1608     \def\bxjs@tmpb{#3}\ifx\bxjs@tmpa\bxjs@tmpb
1609       \@tempwatrue\edef\bxjs@param@paper{papersize={#1,#2}}\fi}
1610   \expandafter\bxjs@tmpdo\bxjs@param@paper,\@nil,\@nil\@nnil
1611 \fi
```

`W*H` の形式について。

```
1612 \if@tempwa\else
1613   \def\bxjs@tmpa{\@nil*\@nil}
1614   \def\bxjs@tmpdo#1*#2*#3\@nnil{%
1615     \def\bxjs@tmpb{#3}\ifx\bxjs@tmpa\bxjs@tmpb
1616       \@tempwatrue\edef\bxjs@param@paper{papersize={#1,#2}}\fi}
1617   \expandafter\bxjs@tmpdo\bxjs@param@paper*\@nil*\@nil\@nnil
1618 \fi
```

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

```
1619 \edef\bxjs@layout@paper{%
1620   \ifjsc@mag truedimen,\fi
1621   \if@landscape landscape,\fi
1622   \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。

```
1623 %<*article|report>
1624 \def\bxjs@layout@base{%
1625   headheight=\topskip,footskip=0.03367\paperheight,%
1626   headsep=\footskip-\topskip,includeheadfoot,%
1627 }
1628 \edef\bxjs@layout{\bxjs@layout@base
1629   hscale=0.76,hmarginratio=1:1,%
1630   vscale=0.83,vmarginratio=1:1,%
```

```

1631 }
1632 %</article|report>
1633 %<*book>
1634 \def\bxjs@layout@base{%
1635   headheight=\topskip,headsep=6\jsc@mmm,nofoot,includeheadfoot,%
1636 }
1637 \ifbxjs@layout@buggyhmargin      %---
1638 % アレ
1639 \edef\bxjs@layout{\bxjs@layout@base
1640   hmargin=36\jsc@mmm,hmarginratio=1:1,%
1641   vscale=0.83,vmarginratio=1:1,%
1642 }
1643 \else                               %---
1644 % 非アレ
1645 \edef\bxjs@layout{\bxjs@layout@base
1646   hmargin=18\jsc@mmm,%
1647   vscale=0.83,vmarginratio=1:1,%
1648 }
1649 \fi                                  %---
1650 %</book>
1651 %<*slide>
1652 \def\bxjs@layout@base{%
1653   noheadfoot,%
1654 }
1655 \edef\bxjs@layout{\bxjs@layout@base
1656   hscale=0.9,hmarginratio=1:1,%
1657   vscale=0.944,vmarginratio=1:1,%
1658 }
1659 %</slide>

```

textwidth オプションの設定を反映する。

```

1660 %<!*book>
1661 \ifx\bxjs@textwidth@opt\undefined\else
1662   \jsSetQHLength\@tempdima{\bxjs@textwidth@opt}
1663   \edef\bxjs@layout{\bxjs@layout width=\the\@tempdima,}
1664 \fi
1665 %</!book>
1666 \ifx\bxjs@number@of@lines@opt\undefined\else
1667   \bxjs@gset@tempcnta{\bxjs@number@of@lines@opt}
1668   \edef\bxjs@layout{\bxjs@layout lines=\the\@tempcnta,}
1669 \fi

```

\fullwidth [寸法レジスタ] ヘッダ・フッタ領域の横幅。

```
1670 \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。

```
1671 %<*book>
```

```

1672 \newcommand\jsTextWidthLimit{40}
1673 \@tempdima=\jsTextWidthLimit\Cwd
1674 \ifx\bxjs@textwidth@limit@opt\undefined\else
1675   \bxjs@gset@tempcnta{\bxjs@textwidth@limit@opt}
1676   \@tempdima=\@tempcnta\Cwd
1677 \fi
1678 \ifx\bxjs@textwidth@opt\undefined\else
1679   \jsSetQHLength\@tempdima{\bxjs@textwidth@opt}
1680 \fi
1681 \edef\bxjs@textwidth@limit{\the\@tempdima}
1682 \ifdim\@tempdima=\jsTextWidthLimit\Cwd\else
1683   \bxjs@invscale\@tempdima{\strip@pt\Cwd}
1684   \long\edef\jsTextWidthLimit{\strip@pt\@tempdima}
1685 \fi
1686 %</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。

```

1687 \def\bxjs@preproc@layout{%
1688   \edef\bxjs@save@ht@strutbox{\the\ht\strutbox}\ht\strutbox=10\jsc@empt}

```

\bxjs@postproc@layout geometry の後処理。

```

1689 \def\bxjs@postproc@layout{%

```

geometry のドライバを再設定する。

```

1690   \ifx\bxjs@geometry@driver\relax\else
1691     \let\Gm@driver\bxjs@geometry@driver
1692   \fi

```

\ht\strutbox の値を元に戻す。

```

1693   \ht\strutbox=\bxjs@save@ht@strutbox\relax

```

\textwidth の値を補正する。

```

1694   \ifbxjs@whole@zw@lines
1695     \@tempdimb=\textwidth
1696     \if@twocolumn \@tempdima=2\Cwd \else \@tempdima=1\Cwd \fi
1697     \advance\textwidth.005pt\relax
1698     \divide\textwidth\@tempdima \multiply\textwidth\@tempdima
1699     \advance\@tempdimb-\textwidth
1700     \advance\oddsidemargin 0.5\@tempdimb
1701     \advance\evensidemargin 0.5\@tempdimb
1702   \fi
1703   \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。

```

1704 %<*book>

```



```

1705 \@tempdima=\bxjs@textwidth@limit\relax
1706 \ifbxjs@whole@zw@lines
1707   \advance\@tempdima.005pt\relax
1708   \divide\@tempdima\Cwd \multiply\@tempdima\Cwd
1709 \fi
1710 \ifdim\textwidth>\@tempdima
1711   \textwidth=\@tempdima
1712   \addtolength\evensidemargin{\fullwidth-\textwidth}
1713 \fi
1714 %</book>

```

`\textheight` 関連の調整。

```

1715 \@tempdimb=\textheight
1716 \advance\textheight-\topskip
1717 \advance\textheight.005pt\relax
1718 \divide\textheight\baselineskip \multiply\textheight\baselineskip
1719 \advance\textheight\topskip
1720 \advance\@tempdimb-\textheight
1721 \advance\topmargin0.5\@tempdimb

```

`\headheight` 関連の調整。

```

1722 \@tempdima=\topskip
1723 \advance\headheight\@tempdima
1724 \advance\topmargin-\@tempdima

```

`marginpar` 関連の調整。

```

1725 \setlength\marginparsep{\columnsep}
1726 \setlength\marginparpush{\baselineskip}
1727 \setlength\marginparwidth{\paperwidth-\oddsidemargin-1truein%
1728   -\textwidth-10\jsc@mmm-\marginparsep}
1729 \ifbxjs@whole@zw@lines
1730   \divide\marginparwidth\Cwd \multiply\marginparwidth\Cwd
1731 \fi

```

連動する変数。

```

1732 \maxdepth=.5\topskip
1733 \stockwidth=\paperwidth
1734 \stockheight=\paperheight
1735 }

```

`\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。

```

1736 \edef\jsGeometryOptions{%
1737   \bxjs@layout@paper,\bxjs@layout}

```

■ `geometry` パッケージを読み込む

`\js@apply@bd@pre@geometry@hook` `geometry` パッケージの `begin-document` フックの処理に割り込む。

※ L^AT_EX のフックシステムがある場合はムニャムニャ。

```

1738 \def\bxjs@geometry@name{geometry}
1739 \ifbxjs@old@hook@system
1740 \let\bxjs@apply@bd@pre@geometry@hook\AtBeginDocument
1741 \else
1742 \def\bxjs@apply@bd@pre@geometry@hook{%
1743 \AddToHook{begindocument}{\bxjs@geometry@name}}
1744 \fi

```

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

```

1745 \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。

```

1746 \ifbxjs@papersize
1747 \ifx\bxjs@driver@given\bxjs@driver@@dvipdfmx
1748 \PassOptionsToPackage{dvipdfm}{geometry}
1749 \else\ifx\bxjs@driver@given\bxjs@driver@@dvimode
1750 \PassOptionsToPackage{dvipdfm}{geometry}
1751 \fi\fi
1752 \let\bxPapersizeSpecialDone=t
1753 \else
1754 \PassOptionsToPackage{driver=none}{geometry}
1755 \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。

```

1756 \bxjs@apply@bd@pre@geometry@hook{%
1757 \bxjs@bd@pre@geometry@hook
1758 \@nameuse{ImposeOldLuaTeXBehavior}}
1759 \bxjs@preproc@layout
1760 \edef\bxjs@next{%
1761 \noexpand\RequirePackage[\bxjs@layout@paper,\bxjs@layout]{geometry}%
1762 }\bxjs@next
1763 \bxjs@apply@bd@pre@geometry@hook{\@nameuse{RevokeOldLuaTeXBehavior}}

```

\bxjs@geometry@driver geometry が用いるドライバの名前。

※この値は一度決めた後は変わってほしくないので、\bxjs@postproc@layout において書き戻す処理を入れている。

```

1764 \let\bxjs@geometry@driver\Gm@driver
1765 \bxjs@postproc@layout

```

geometry のドライバ自動判別に対する前処理。

```

1766 \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。

```

1767 \ifpackagelater{geometry}{2010/02/12}{\else
1768 \PackageError\bxjs@clsname

```

```

1769     {Your 'geometry' package is too old (< v5.0)}%
1770     {\@ehc}%
1771     \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。

```

1772     \ifjsWithpTeXng
1773     \ifx\Gm@driver\@empty
1774     \def\Gm@driver{pdftex}%
1775     \fi
1776     \fi}

```

`\setpagelayout` ページレイアウト設定のためのユーザ命令。

```

1777 \def\setpagelayout{%
1778   \bxjs@ifplus{\bxjs@setpagelayout@a\tw@}{%else
1779     \@ifstar{\bxjs@setpagelayout@a\@ne}{\bxjs@setpagelayout@a\z@}}%
1780 \def\bxjs@setpagelayout@a#1#2{%
1781   \ifcase#1% modify
1782     \def\bxjs@next{\ifjsc@mag truedimen,\fi #2}%
1783   \or% reset(*)
1784     \def\bxjs@next{reset,\bxjs@layout@paper,#2}%
1785   \or% semireset(+)
1786     \def\bxjs@next{reset,\bxjs@layout@paper,\bxjs@layout@base,#2}%
1787   \fi
1788   \bxjs@preproc@layout
1789   \edef\bxjs@next{%
1790     \noexpand\geometry{\bxjs@next}%
1791   }\bxjs@next
1792   \bxjs@postproc@layout}

```

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

```

1793 \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。

```

1794 \g@addto@macro\bxjs@begin@document@hook{%
1795   \ifdim\textwidth=.5\maxdimen
1796     \ClassError\bxjs@clsname
1797     {Page layout is not properly set}%
1798     {\@ehd}%
1799   \fi}
1800 \def\jsUseMinimalPageLayout{%
1801   \setlength{\textwidth}{6.5in}%
1802   \setlength{\textheight}{8in}}

```

`\setpagelayout` はとりあえず無効にしておく。

```

1803 \let\bxjs@geometry@driver\relax
1804 \def\setpagelayout{%

```

```

1805 \bxjs@ifplus{\bxjs@pagelayout@a}{%else
1806   \@ifstar{\bxjs@pagelayout@a}{\bxjs@pagelayout@a}}
1807 \def\bxjs@pagelayout@a#1{%
1808   \ClassError\bxjs@clsname
1809   {Command '\string\setpagelayout' is not supported,\MessageBreak
1810     because 'geometry' value is not 'class'}\@eha}
1811 %
1812 \fi\fi

```

■縦方向のスペース

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

```

1813 %<*jsclasses>

```

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

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

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

```

1814 \setlength\topskip{1.38zw}%% from 10\jsc@mpt (2016-08-17)
1815 \if@slide
1816   \setlength\headheight{0\jsc@mpt}
1817 \else
1818   \setlength\headheight{20\jsc@mpt}%% from 2\topskip (2016-08-17); from \topskip (2003-
1819   06-26)
1819 \fi

```

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

```

1820 %<*article|kiyou>
1821 \if@slide
1822   \setlength\footskip{0pt}
1823 \else
1824   \setlength\footskip{0.03367\paperheight}
1825   \ifdim\footskip<\baselineskip
1826     \setlength\footskip{\baselineskip}
1827 \fi

```

```

1828 \fi
1829 %</article|kiyou>
1830 %<jspf>\setlength\footskip{9\jsc@mmm}
1831 %<*book>
1832 \if@report
1833 \setlength\footskip{0.03367\paperheight}
1834 \ifdim\footskip<\baselineskip
1835 \setlength\footskip{\baselineskip}
1836 \fi
1837 \else
1838 \setlength\footskip{0pt}
1839 \fi
1840 %</book>
1841 %<*report>
1842 \setlength\footskip{0.03367\paperheight}
1843 \ifdim\footskip<\baselineskip
1844 \setlength\footskip{\baselineskip}
1845 \fi
1846 %</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)。

```

1847 %<*article>
1848 \if@slide
1849 \setlength\headsep{0\jsc@mppt}
1850 \addtolength\headsep{-\topskip}%% added (2016-10-08)
1851 \addtolength\headsep{10\jsc@mppt}%% added (2016-10-08)
1852 \else
1853 \setlength\headsep{\footskip}
1854 \addtolength\headsep{-\topskip}
1855 \fi
1856 %</article>
1857 %<*book>
1858 \if@report
1859 \setlength\headsep{\footskip}
1860 \addtolength\headsep{-\topskip}
1861 \else
1862 \setlength\headsep{6\jsc@mmm}
1863 \addtolength\headsep{-\topskip}%% added (2016-10-08)
1864 \addtolength\headsep{10\jsc@mppt}%% added (2016-10-08)
1865 \fi
1866 %</book>
1867 %<*report>
1868 \setlength\headsep{\footskip}
1869 \addtolength\headsep{-\topskip}

```

```

1870 %</report>
1871 %<*jspf>
1872 \setlength\headsep{9\jsc@mmm}
1873 \addtolength\headsep{-\topskip}
1874 %</jspf>
1875 %<*kiyou>
1876 \setlength\headheight{0\jsc@empt}
1877 \setlength\headsep{0\jsc@empt}
1878 \addtolength\headsep{-\topskip}%% added (2016-10-08)
1879 \addtolength\headsep{10\jsc@empt}%% added (2016-10-08)
1880 %</kiyou>

```

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

```
1881 \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。

```
1882 \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。

```

1883 %<*article>
1884 \if@slide
1885 \setlength\fullwidth{0.9\paperwidth}
1886 \else
1887 \setlength\fullwidth{0.76\paperwidth}
1888 \fi
1889 \if@twocolumn \@tempdima=2zw \else \@tempdima=1zw \fi
1890 \divide\fullwidth\@tempdima \multiply\fullwidth\@tempdima
1891 \setlength\textwidth{\fullwidth}
1892 %</article>
1893 %<*book>
1894 \if@report
1895 \setlength\fullwidth{0.76\paperwidth}

```

```

1896 \else
1897 \setlength\fullwidth{\paperwidth}
1898 \addtolength\fullwidth{-36\jsc@mmm}
1899 \fi
1900 \if@twocolumn \@tempdima=2zw \else \@tempdima=1zw \fi
1901 \divide\fullwidth\@tempdima \multiply\fullwidth\@tempdima
1902 \setlength\textwidth{\fullwidth}
1903 \if@report \else
1904 \if@twocolumn \else
1905 \ifdim \fullwidth>40zw
1906 \setlength\textwidth{40zw}
1907 \fi
1908 \fi
1909 \fi
1910 %</book>
1911 %<*report>
1912 \setlength\fullwidth{0.76\paperwidth}
1913 \if@twocolumn \@tempdima=2zw \else \@tempdima=1zw \fi
1914 \divide\fullwidth\@tempdima \multiply\fullwidth\@tempdima
1915 \setlength\textwidth{\fullwidth}
1916 %</report>
1917 %<*jspf>
1918 \setlength\fullwidth{50zw}
1919 \addtolength\fullwidth{8\jsc@mmm}
1920 \setlength\textwidth{\fullwidth}
1921 %</jspf>
1922 %<*kiyou>
1923 \setlength\fullwidth{48zw}
1924 \addtolength\fullwidth{\columnsep}
1925 \setlength\textwidth{\fullwidth}
1926 %</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)。

```

1927 %<*article|book|report>
1928 \if@slide
1929 \setlength{\textheight}{0.95\paperheight}
1930 \else
1931 \setlength{\textheight}{0.83\paperheight}
1932 \fi
1933 \addtolength{\textheight}{-10\jsc@empt}%% from -\topskip (2016-10-08); from -
    \headheight (2003-06-26)
1934 \addtolength{\textheight}{-\headsep}
1935 \addtolength{\textheight}{-\footskip}
1936 \addtolength{\textheight}{-\topskip}
1937 \divide\textheight\baselineskip
1938 \multiply\textheight\baselineskip
1939 %</article|book|report>
1940 %<jspf>\setlength{\textheight}{51\baselineskip}
1941 %<kiyou>\setlength{\textheight}{47\baselineskip}
1942 \addtolength{\textheight}{\topskip}
1943 \addtolength{\textheight}{0.1\jsc@empt}
1944 %<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。

```

1945 \def\flushbottom{%
1946 \def\@textbottom{\vskip \z@ \@plus.1\jsc@empt}%
1947 \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。

```

1948 \setlength\marginparsep{\columnsep}
1949 \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 $\text{p}\text{\LaTeX} 2_{\epsilon}$ (`plcore.ltx`) はトンボの内側に `1in` のスペース (`1truein` ではなく) を挿入するので、場合分けしています。

```

1950 \setlength{\oddsidemargin}{\paperwidth}
1951 \addtolength{\oddsidemargin}{-\fullwidth}
1952 \setlength{\oddsidemargin}{.5\oddsidemargin}
1953 \iftombow
1954 \addtolength{\oddsidemargin}{-1in}
1955 \else
1956 \addtolength{\oddsidemargin}{-\inv@mag in}

```



```

1957 \fi
1958 \setlength{\evensidemargin}{\oddsidemargin}
1959 \if@mparswitch
1960 \addtolength{\evensidemargin}{\fullwidth}
1961 \addtolength{\evensidemargin}{-\textwidth}
1962 \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。

```

1963 \setlength\marginparwidth{\paperwidth}
1964 \addtolength\marginparwidth{-\oddsidemargin}
1965 \addtolength\marginparwidth{-\inv@mag in}
1966 \addtolength\marginparwidth{-\textwidth}
1967 \addtolength\marginparwidth{-10\jsc@mmm}
1968 \addtolength\marginparwidth{-\marginparsep}
1969 \@tempdima=1zw
1970 \divide\marginparwidth\@tempdima
1971 \multiply\marginparwidth\@tempdima

```

`\topmargin` 上マージン (紙の上端とヘッダ上端の距離) から 1 インチ引いた値です。

[2003-06-26] `\headheight` を `\topskip` に直しました。以前はこの二つは値が同じであったので、変化はないはずです。

[2016-08-17] `\topskip` を 10pt から 1.38zw に直しましたが、`\topmargin` は従来の値から変わらないように調節しました。…のつもりでしたが、`\textheight` を増やし忘れていたので変わってしまっていました (2016-08-26 修正済み)。

```

1972 \setlength\topmargin{\paperheight}
1973 \addtolength\topmargin{-\textheight}
1974 \if@slide
1975 \addtolength\topmargin{-\headheight}
1976 \else
1977 \addtolength\topmargin{-10\jsc@mpt}%% from -\topskip (2016-10-08); from -
    \headheight (2003-06-26)
1978 \fi
1979 \addtolength\topmargin{-\headsep}
1980 \addtolength\topmargin{-\footskip}
1981 \setlength\topmargin{0.5\topmargin}
1982 %<kiyou>\setlength\topmargin{81truebp}
1983 \iftombow
1984 \addtolength\topmargin{-1in}
1985 \else
1986 \addtolength\topmargin{-\inv@mag in}
1987 \fi
1988 %</jsclasses>

```

■脚注

ここからのコードは以下の点を除いて JS クラスのものを踏襲する。

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

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

ここは元々は

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

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

```
1989 \footnotesep=11\jsc@empt \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。

```
1990 \setlength{\skip\footins}{16\jsc@empt \@plus 5\jsc@empt \@minus 2\jsc@empt}
```

■**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。

```
1991 \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。

```
1992 \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。

```
1993 \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。

```
1994 \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。

1995 `\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。

1996 `\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。

1997 `\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。

1998 `\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。

1999 `\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。

2000 `\renewcommand{\dblfloatpagefraction}{.8}`

`\floatsep` `\floatsep` はページ上部・下部のフロート間の距離です。`\textfloatsep` はページ上部・

`\textfloatsep` 下部のフロートと本文との距離です。`\intextsep` は本文の途中に出力されるフロートと本文との距離です。

`\intextsep`

2001 `\setlength\floatsep {12\jsc@empt \@plus 2\jsc@empt \@minus 2\jsc@empt}`

2002 `\setlength\textfloatsep{20\jsc@empt \@plus 2\jsc@empt \@minus 4\jsc@empt}`

2003 `\setlength\intextsep {12\jsc@empt \@plus 2\jsc@empt \@minus 2\jsc@empt}`

`\dblfloatsep` 二段組のときの段抜きのフロートについての値です。

`\dbltextfloatsep` 2004 `\setlength\dblfloatsep {12\jsc@empt \@plus 2\jsc@empt \@minus 2\jsc@empt}`

2005 `\setlength\dbltextfloatsep{20\jsc@empt \@plus 2\jsc@empt \@minus 4\jsc@empt}`

`\@fptop` フロートだけのページに入るグルーです。`\@fptop` はページ上部, `\@fpbot` はページ下部,

`\@fpsep` `\@fpsep` はフロート間に入ります。

`\@fpbot` 2006 `\setlength\@fptop{0\jsc@empt \@plus 1fil}`

2007 `\setlength\@fpsep{8\jsc@empt \@plus 2fil}`

2008 `\setlength\@fpbot{0\jsc@empt \@plus 1fil}`

`\@dblfpptop` 段抜きフロートについての値です。

`\@dblfpsep` 2009 `\setlength\@dblfpptop{0\jsc@empt \@plus 1fil}`

`\@dblfpbot` 2010 `\setlength\@dblfpsep{8\jsc@empt \@plus 2fil}`

2011 `\setlength\@dblfpbot{0\jsc@empt \@plus 1fil}`

6 改ページ (日本語 TeX 開発コミュニティ版のみ)

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

`\pltx@cleartoleftpage`

`\pltx@cleartooddpage`

`\pltx@cleartoevenpage`

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

となっています。

```

2012 %\def\pltx@cleartorightpage{\clearpage\if@twoside
2013 % \ifodd\c@page
2014 % \iftdir
2015 % \hbox{}\thispagestyle{empty}\newpage
2016 % \if@twocolumn\hbox{}\newpage\fi
2017 % \fi
2018 % \else
2019 % \ifydir
2020 % \hbox{}\thispagestyle{empty}\newpage
2021 % \if@twocolumn\hbox{}\newpage\fi
2022 % \fi
2023 % \fi\fi}
2024 %\def\pltx@cleartoleftpage{\clearpage\if@twoside
2025 % \ifodd\c@page
2026 % \ifydir
2027 % \hbox{}\thispagestyle{empty}\newpage
2028 % \if@twocolumn\hbox{}\newpage\fi
2029 % \fi
2030 % \else
2031 % \iftdir
2032 % \hbox{}\thispagestyle{empty}\newpage
2033 % \if@twocolumn\hbox{}\newpage\fi
2034 % \fi
2035 % \fi\fi}
2036 \def\pltx@cleartooddpage{\clearpage\if@twoside
2037 \ifodd\c@page\else
2038 \hbox{}\thispagestyle{empty}\newpage
2039 \if@twocolumn\hbox{}\newpage\fi
2040 \fi\fi}
2041 \def\pltx@cleartoevenpage{\clearpage\if@twoside
2042 \ifodd\c@page
2043 \hbox{}\thispagestyle{empty}\newpage
2044 \if@twocolumn\hbox{}\newpage\fi
2045 \fi\fi}

```

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

```

2046 \let\pltx@cleartorightpage\pltx@cleartooddpage
2047 \let\pltx@cleartoleftpage\pltx@cleartoevenpage

```

`\vsize` の値がアレな場合は本体開始まで `\clearpage` を無効にする。

```

2048 \ifdim\vsize=\z@
2049 \begingroup

```

```

2050 \toks@\expandafter{\clearpage}
2051 \xdef\clearpage{\noexpand\ifbxjs@after@preamble\the\toks@\noexpand\fi}
2052 \endgroup
2053 \fi

```

`\cleardoublepage` [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、report と book クラスの場合に `\cleardoublepage` を再定義します。

```

2054 %<*book|report>
2055 \if@openleft
2056 \let\cleardoublepage\pltx@cleartoleftpage
2057 \else\if@openright
2058 \let\cleardoublepage\pltx@cleartorightpage
2059 \fi\fi
2060 %</book|report>

```

7 ページスタイル

ページスタイルとして、L^AT_EX 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 L^AT_EX 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, `\@oddhead` フッタ) を出力する命令です。これらは `\fullwidth` 幅の `\hbox` の中で呼び出されます。

`\@evenfoot` `\ps@...` の中で定義しておきます。

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

```

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

```

柱を設定する命令は、右の柱が左の柱の下位にある場合は十分まともに動作します。たとえば左マークを `\chapter`, 右マークを `\section` で変更する場合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、おかしな結果になることがあります。

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

`\ps@empty` `empty` ページスタイルの定義です。L^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。

```

2061 % \def\ps@empty{%
2062 %   \let\@mkboth\@gobbletwo
2063 %   \let\@oddhead\@empty
2064 %   \let\@oddfoot\@empty
2065 %   \let\@evenhead\@empty
2066 %   \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。

```

2067 \def\ps@plainfoot{%
2068   \let\@mkboth\@gobbletwo
2069   \let\@oddhead\@empty
2070   \def\@oddfoot{\normalfont\hfil\thepage\hfil}%
2071   \let\@evenhead\@empty
2072   \let\@evenfoot\@oddfoot}
2073 \def\ps@plainhead{%
2074   \let\@mkboth\@gobbletwo
2075   \let\@oddfoot\@empty
2076   \let\@evenfoot\@empty
2077   \def\@evenhead{%
2078     \if@mparswitch \hss \fi
2079     \hbox to \fullwidth{\textbf{\thepage}\hfil}%
2080     \if@mparswitch\else \hss \fi}%
2081   \def\@oddhead{%
2082     \hbox to \fullwidth{\hfil\textbf{\thepage}}\hss}}
2083 %<book>\let\ps@plain\ps@plainhead
2084 %<!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。

```

2085 %<*article|slide>
2086 \if@twoside
2087   \def\ps@headings{%
2088     \let\@oddfoot\@empty
2089     \let\@evenfoot\@empty
2090     \def\@evenhead{\if@mparswitch \hss \fi
2091       \underline{\hbox to \fullwidth{\textbf{\thepage}\hfil\leftmark}}}%
2092     \if@mparswitch\else \hss \fi}%
2093     \def\@oddhead{%
2094       \underline{%
2095         \hbox to \fullwidth{\rightmark}\hfil\textbf{\thepage}}\hss}%
2096     \let\@mkboth\markboth
2097     \def\sectionmark##1{\markboth{%
2098       \ifnum \c@secnumdepth >\z@ \bxjs@label@sect{section}\hskip1\jsZw\fi
2099       ##1-}}}%
2100     \def\subsectionmark##1{\markright{%

```

```

2101     \ifnum \c@secnumdepth >\@ne \bxjs@label@sect{subsection}\hskip1\jsZw\fi
2102     ##1}}%
2103   }
2104 \else % if not twoside
2105   \def\ps@headings{%
2106     \let\@oddfoot\@empty
2107     \def\@oddhead{%
2108       \underline{%
2109         \hbox to \fullwidth{\rightmark}\hfil\textbf{\thepage}}\hss}%
2110     \let\@mkboth\markboth
2111     \def\sectionmark##1{\markright{%
2112       \ifnum \c@secnumdepth >\z@ \bxjs@label@sect{section}\hskip1\jsZw\fi
2113       ##1}}
2114 \fi
2115 %</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）。

```

2116 %<*book|report>

```

`\bxjs@maybe@autoxspacing` `\autoxspacing` が定義済ならばそれを実行する。

※`\autoxspacing` は未定義の可能性があるので代わりに用いる。

```

2117 \def\bxjs@maybe@autoxspacing{%
2118   \ifx\autoxspacing\@undefined\else \autoxspacing \fi}

```

```

2119 \newif\if@omit@number
2120 \def\ps@headings{%
2121   \let\@oddfoot\@empty
2122   \let\@evenfoot\@empty
2123   \def\@evenhead{%
2124     \if@mparswitch \hss \fi
2125     \underline{\hbox to \fullwidth{\bxjs@maybe@autoxspacing
2126       \textbf{\thepage}\hfil\leftmark}}}%
2127     \if@mparswitch\else \hss \fi}%
2128   \def\@oddhead{\underline{\hbox to \fullwidth{\bxjs@maybe@autoxspacing
2129     {\if@twoside\rightmark\else\leftmark\fi}\hfil\textbf{\thepage}}}\hss}%
2130   \let\@mkboth\markboth
2131   \def\chaptermark##1{\markboth{%
2132     \ifnum \c@secnumdepth >\m@ne
2133       \if@mainmatter
2134         \if@omit@number\else
2135           \@chapapp\thechapter\@chappos\hskip1\jsZw
2136         \fi
2137       \fi
2138     \fi
2139     ##1-{}%

```

```

2140 \def\sectionmark##1{\markright{%
2141   \ifnum \c@secnumdepth >\z@ \bxjs@label@sect{section}\hskip1\jsZw\fi
2142   ##1}}}%
2143 %</book|report>

```

最後は学会誌の場合です。

```

2144 %< *jspf>
2145 \def\ps@headings{%
2146   \def\@oddfoot{\normalfont\hfil\thepage\hfil}
2147   \def\@evenfoot{\normalfont\hfil\thepage\hfil}
2148   \def\@oddhead{\normalfont\hfil \@title \hfil}
2149   \def\@evenhead{\normalfont\hfil プラズマ・核融合学会誌\hfil}}
2150 %</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。

```

2151 \def\ps@myheadings{%
2152   \let\@oddfoot\@empty\let\@evenfoot\@empty
2153   \def\@evenhead{%
2154     \if@mparswitch \hss \fi%
2155     \hbox to \fullwidth{\thepage\hfil\leftmark}%
2156     \if@mparswitch\else \hss \fi}%
2157   \def\@oddhead{%
2158     \hbox to \fullwidth{\rightmark\hfil\thepage}\hss}%
2159   \let\@mkboth\@gobbletwo
2160 %<book|report> \let\chaptermark\@gobble
2161 \let\sectionmark\@gobble
2162 %<!book&!report> \let\subsectionmark\@gobble
2163 }

```

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 2164 % \newcommand*{\title}[1]{\gdef\@title{#1}}
2165 % \newcommand*{\author}[1]{\gdef\@author{#1}}
2166 % \newcommand*{\date}[1]{\gdef\@date{#1}}
2167 % \date{\today}

```

`\subtitle` 副題を設定する。

`\jsSubtitle` ※プレアンブルにおいて `\newcommand*{\subtitle}{...}` が行われることへの対策として、`\subtitle` の定義を `\title` の実行まで遅延させることにする。もしどうしても主題

より前に副題を設定したい場合は、`\jsSubtitle` 命令を直接用いればよい。

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

本体を `\jsSubtitle` として定義する。

```
2168 \newcommand*{\jsSubtitle}[1]{\gdef\bxjs@subtitle{#1}}
2169 %\let\bxjs@subtitle\undefined

\title にフックを入れる。

2170 \renewcommand*{\title}[1]{\bxjs@decl@subtitle\gdef\@title{#1}}
2171 \g@addto@macro\bxjs@begin@document@hook{\bxjs@decl@subtitle}
2172 \def\bxjs@decl@subtitle{%
2173   \global\let\bxjs@decl@subtitle\relax
2174   \ifx\subtitle\undefined
2175     \global\let\subtitle\jsSubtitle
2176   \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。

```
2177 \def\bxjs@annihilate@subtitle{%
2178   \ifx\subtitle\jsSubtitle \global\let\subtitle\relax \fi
2179   \global\let\jsSubtitle\relax}
```

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

```
\eauthor 2180 %<*jspf>
\keywords 2181 \newcommand*{\etitle}[1]{\gdef\@etitle{#1}}
2182 \newcommand*{\eauthor}[1]{\gdef\@eauthor{#1}}
2183 \newcommand*{\keywords}[1]{\gdef\@keywords{#1}}
2184 \newcommand*{\email}[1]{\gdef\authors@mail{#1}}
2185 \newcommand*{\AuthorsEmail}[1]{\gdef\authors@mail{author's e-mail:\ #1}}
2186 %</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。

```
2187 \def\plainifnotempty{%
2188   \ifx \@oddhead \@empty
2189     \ifx \@oddfoot \@empty
2190     \else
2191       \thispagestyle{plainfoot}%
2192     \fi
2193   \else
2194     \thispagestyle{plainhead}%
2195   \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。

```
2196 %<*article|book|report|slide>
2197 \if@titlepage
2198   \newcommand{\maketitle}{%
2199     \begin{titlepage}%
2200       \let\footnotesize\small
2201       \let\footnoterule\relax
2202       \let\footnote\thanks
2203       \null\vfil
2204       \if@slide
2205         {\footnotesize \@date}%
2206         \begin{center}
2207           \mbox{} \\\[1\jsZw]
2208           \large
2209           {\maybeblue\hrule height0\jsc@mpt depth2\jsc@mpt\relax}\par
2210           \jsc@smallskip
2211           \@title
2212           \ifx\bxjs@subtitle\@undefined\else
2213             \par\vskip\z@
2214             {\small \bxjs@subtitle\par}
2215           \fi
2216           \jsc@smallskip
2217           {\maybeblue\hrule height0\jsc@mpt depth2\jsc@mpt\relax}\par
2218           \vfill
2219           {\small \@author}%
2220         \end{center}
2221       \else
2222       \vskip 60\jsc@mpt
2223       \begin{center}%
2224         {\LARGE \@title \par}%
2225         \ifx\bxjs@subtitle\@undefined\else
2226           \vskip5\jsc@mpt
2227           {\normalsize \bxjs@subtitle\par}
2228         \fi
2229         \vskip 3em%
2230         {\large
2231           \lineskip .75em
2232           \begin{tabular}[t]{c}%
2233             \@author
2234           \end{tabular}\par}%
2235         \vskip 1.5em
2236         {\large \@date \par}%
2237       \end{center}%
```

```

2238     \fi
2239     \par
2240     \@thanks\vfil\null
2241 \end{titlepage}%
2242 \setcounter{footnote}{0}%
2243 \global\let\thanks\relax
2244 \global\let\maketitle\relax
2245 \global\let\@thanks\@empty
2246 \global\let\@author\@empty
2247 \global\let\@date\@empty
2248 \global\let\@title\@empty
2249 \global\let\title\relax
2250 \global\let\author\relax
2251 \global\let\date\relax
2252 \global\let\and\relax
2253 \bxjs@annihilate@subtitle
2254 }%
2255 \else
2256 \newcommand{\maketitle}{\par
2257 \begingroup
2258   \renewcommand\thefootnote{\@fnsymbol\c@footnote}%
2259   \def\@makefnmark{\rlap{-\textsuperscript{\normalfont\@thefnmark}}}%
2260   \long\def\@makefntext##1{\advance\leftskip 3\jsZw
2261     \parindent 1\jsZw\noindent
2262     \llap{-\textsuperscript{\normalfont\@thefnmark}\hskip0.3\jsZw}##1}%
2263   \if@twocolumn
2264     \ifnum \col@number=\@ne
2265       \@maketitle
2266     \else
2267       \twocolumn[\@maketitle]%
2268     \fi
2269   \else
2270     \newpage
2271     \global\@topnum\z@ % Prevents figures from going at top of page.
2272     \@maketitle
2273   \fi
2274   \plainifnotempty
2275   \@thanks
2276 \endgroup
2277 \setcounter{footnote}{0}%
2278 \global\let\thanks\relax
2279 \global\let\maketitle\relax
2280 \global\let\@thanks\@empty
2281 \global\let\@author\@empty
2282 \global\let\@date\@empty
2283 \global\let\@title\@empty
2284 \global\let\title\relax
2285 \global\let\author\relax
2286 \global\let\date\relax

```

```

2287 \global\let\and\relax
2288 \bxjs@annihilate@subtitle
2289 }

```

`\@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。

```

2290 \def\@maketitle{%
2291 \newpage\null
2292 \vskip 2em
2293 \begin{center}%
2294 \let\footnote\thanks
2295 {\LARGE \@title \par}%
2296 \ifx\bxjs@subtitle\@undefined\else
2297 \vskip3\jsc@empt
2298 {\normalsize \bxjs@subtitle\par}
2299 \fi
2300 \vskip 1.5em
2301 {\large
2302 \lineskip .5em
2303 \begin{tabular}[t]{c}%
2304 \author
2305 \end{tabular}\par}%
2306 \vskip 1em
2307 {\large \@date}%
2308 \end{center}%
2309 \par\vskip 1.5em
2310 %<article|slide> \ifvoid\@abstractbox\else\centerline{\box\@abstractbox}\vskip1.5em\fi
2311 }
2312 \fi
2313 %</article|book|report|slide>
2314 %<*jspf>
2315 \newcommand{\maketitle}{\par
2316 \begingroup
2317 \renewcommand\thefootnote{\@fnsymbol\c@footnote}%
2318 \def\@makefnmark{\rlap{\@textsuperscript{\normalfont\@thefnmark}}}%
2319 \long\def\@makefntext##1{\advance\leftskip 3\jsZw
2320 \parindent 1\jsZw\noindent
2321 \llap{\@textsuperscript{\normalfont\@thefnmark}\hskip0.3\jsZw}##1}%
2322 \twocolumn[\@maketitle]%
2323 \plainifnotempty
2324 \@thanks
2325 \endgroup
2326 \setcounter{footnote}{0}%
2327 \global\let\thanks\relax
2328 \global\let\maketitle\relax
2329 \global\let\@thanks\@empty
2330 \global\let\@author\@empty
2331 \global\let\@date\@empty
2332 % \global\let\@title\@empty % \@title は柱に使う
2333 \global\let\title\relax

```

```

2334 \global\let\author\relax
2335 \global\let\date\relax
2336 \global\let\and\relax
2337 \ifx\authors@mail\@undefined\else{%
2338   \def\@makefnctext{\advance\leftskip 3\jsZw \parindent -3\jsZw}%
2339   \footnotetext[0]{\itshape\authors@mail}%
2340 } \fi
2341 \global\let\authors@mail\@undefined}
2342 \def\@maketitle{%
2343   \newpage\null
2344   \vskip 6em % used to be 2em
2345   \begin{center}
2346     \let\footnote\thanks
2347     \ifx\@title\@undefined\else{\LARGE\headfont\@title\par} \fi
2348     \lineskip .5em
2349     \ifx\@author\@undefined\else
2350       \vskip 1em
2351       \begin{tabular}[t]{c}%
2352         \@author
2353       \end{tabular}\par
2354     \fi
2355     \ifx\@etitle\@undefined\else
2356       \vskip 1em
2357       {\large \@etitle \par}%
2358     \fi
2359     \ifx\@eauthor\@undefined\else
2360       \vskip 1em
2361       \begin{tabular}[t]{c}%
2362         \@eauthor
2363       \end{tabular}\par
2364     \fi
2365     \vskip 1em
2366     \@date
2367   \end{center}
2368   \vskip 1.5em
2369   \centerline{\box\@abstractbox}
2370   \ifx\@keywords\@undefined\else
2371     \vskip 1.5em
2372     \centerline{\parbox{157\jsc@mmm}{\textsf{Keywords:}}\ \small\@keywords}
2373   \fi
2374   \vskip 1.5em}
2375 %</jspf>

```

8.2 章・節

label-section オプション対応のための処理。

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

```
2376 \def\bxjs@label@sect#1{%
2377   \@ifundefined{label#1}{\@nameuse{the#1}}{\@nameuse{label#1}}
2378 \def\@secntformat#1{\bxjs@label@sect{#1}\quad}
```

`\@secapp` 節番号の接頭辞。

`\@secpos` 節番号の接尾辞。

```
2379 \ifnum\bxjs@label@section=\bxjs@label@section@@compat\else
2380 \def\@secapp{\presectionname}
2381 \def\@secpos{\postsectionname}
2382 \fi
```

`\labelsection` 節番号の出力書式。

```
2383 \ifnum\bxjs@label@section=\bxjs@label@section@@modern
2384 \def\labelsection{\@secapp\thesection\@secpos}
2385 \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 `\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。

```

2386 \def\@startsection#1#2#3#4#5#6{%
2387 \if@noskipsec \leavevmode \fi
2388 \par
2389 % 見出し上の空きを \@tempskipa にセットする
2390 \@tempskipa #4\relax
2391 % \@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
2392 \if@english \@afterindentfalse \else \@afterindenttrue \fi
2393 % 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
2394 \ifdim \@tempskipa <\z@
2395   \@tempskipa -\@tempskipa \@afterindentfalse
2396 \fi
2397 \if@nobreak
2398 % \everypar{\everyparhook}% これは間違い
2399 \everypar{}%
2400 \else
2401 \addpenalty\@secpenalty
2402 % 次の行は削除
2403 % \addvspace\@tempskipa
2404 % 次の \noindent まで追加
2405 \ifdim \@tempskipa >\z@
2406 \if@slide\else
2407 \null
2408 \vspace*{-\baselineskip}%
2409 \fi
2410 \vskip\@tempskipa
2411 \fi
2412 \fi
2413 \noindent
2414 % 追加終わり
2415 \@ifstar
2416 {\@ssect{#3}{#4}{#5}{#6}}%
2417 {\@dblarg{\@sect{#1}{#2}{#3}{#4}{#5}{#6}}}

```

\@sect と \@xsect は、前のアキがちょうどゼロの場合にもうまくいくように、多少変えてあります。 \everyparhook も挿入しています。

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

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

```

2418 \ifnum\bxjs@everyparhook=\bxjs@everyparhook@@compat
2419 \let\bxjs@if@ceph\@firstofone
2420 \else \let\bxjs@if@ceph\@gobble
2421 \fi

```

```

2422 \def\@sect#1#2#3#4#5#6[#7]#8{%
2423 \ifnum #2>\c@secnumdepth
2424 \let\@svsec\@empty

```

```

2425 \else
2426   \refstepcounter{#1}%
2427   \protected@edef\@svsec{\@secntformat{#1}\relax}%
2428   \fi
2429 % 見出し後の空きを \@tempkipa にセット
2430   \@tempkipa #5\relax
2431 % 条件判断の順序を入れ替えました
2432   \ifdim \@tempkipa<\z@
2433     \def\@svsechd{%
2434       #6{\hskip #3\relax
2435         \@svsec #8}%
2436       \csname #1mark\endcsname{#7}%
2437       \addcontentsline{toc}{#1}{%
2438         \ifnum #2>\c@secnumdepth \else
2439           \protect\numberline{\bxjs@label@sect{#1}}%
2440         \fi
2441         #7}}% 目次にフルネームを載せるなら #8
2442   \else
2443     \begingroup
2444       \interlinepenalty \@M % 下から移動
2445       #6{%
2446         \@hangfrom{\hskip #3\relax\@svsec}%
2447       % \interlinepenalty \@M % 上に移動
2448       #8\@par}%
2449     \endgroup
2450     \csname #1mark\endcsname{#7}%
2451     \addcontentsline{toc}{#1}{%
2452       \ifnum #2>\c@secnumdepth \else
2453         \protect\numberline{\bxjs@label@sect{#1}}%
2454       \fi
2455       #7}% 目次にフルネームを載せるならここは #8
2456   \fi
2457   \@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。

```

2458 \def\@xsect#1{%
2459 % 見出しの後ろの空きを \@tempkipa にセット
2460   \@tempkipa #1\relax
2461 % 条件判断の順序を変えました
2462   \ifdim \@tempkipa<\z@
2463     \@nobreakfalse
2464     \global\@noskipsecttrue
2465     \everypar{%
2466       \if@noskipsec
2467         \global\@noskipsecfalse

```



```

2468     {\setbox\z@\lastbox}%
2469     \clubpenalty\@M
2470     \begingroup \@svsechd \endgroup
2471     \unskip
2472     \@tempskipa #1\relax
2473     \hskip -\@tempskipa
2474     \else
2475     \clubpenalty \clubpenalty
2476     \everypar\expandafter{\bxjs@if@ceph\everyparhook}%

```

TODO: ↑ ナニコレ？

```

2477     \fi\bxjs@if@ceph\everyparhook}%
2478 \else
2479 \par \nobreak
2480 \vskip \@tempskipa
2481 \@afterheading
2482 \fi
2483 \if@slide
2484 {\vskip@if@twocolumn-5\jsc@mpt\else-6\jsc@mpt\fi
2485 \maybeblue\hrule height0\jsc@mpt depth1\jsc@mpt
2486 \vskip@if@twocolumn 4\jsc@mpt\else 7\jsc@mpt\fi\relax}%
2487 \fi
2488 \par % 2000-12-18
2489 \ignorespaces}
2490 \def\@ssect#1#2#3#4#5{%
2491 \@tempskipa #3\relax
2492 \ifdim \@tempskipa<\z@
2493 \def\@svsechd{#4{\hskip #1\relax #5}}%
2494 \else
2495 \begingroup
2496 #4{%
2497 \@hangfrom{\hskip #1}%
2498 \interlinepenalty \@M #5\@par}%
2499 \endgroup
2500 \fi
2501 \@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 2502 \newcommand*\chaptermark[1]{}
\subsubsectionmark 2503 % \newcommand*\sectionmark[1]{}
2504 % \newcommand*\subsectionmark[1]{}
\paragraphmark 2505 % \newcommand*\subsubsectionmark[1]{}
\subparagraphmark 2506 % \newcommand*\paragraphmark[1]{}
2507 % \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。

```
2508 %<!book&!report>\setcounter{secnumdepth}{3}
2509 %<book|report>\setcounter{secnumdepth}{2}
```

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

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

```
\c@subsection 2510 \newcounter{part}
2511 %<book|report>\newcounter{chapter}
\c@subsubsection 2512 %<book|report>\newcounter{section}[chapter]
\c@paragraph 2513 %<!book&!report>\newcounter{section}
\c@subparagraph 2514 \newcounter{subsection}[section]
2515 \newcounter{subsubsection}[subsection]
2516 \newcounter{paragraph}[subsubsection]
2517 \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。

```
2518 \renewcommand{\thepart}{\@Roman\c@part}
2519 %<*!book&!report>
2520 \ifnum\bxjs@label@section=\bxjs@label@section@@compat
2521 \renewcommand{\thesection}{\presectionname\@arabic\c@section\postsectionname}
2522 \renewcommand{\thesubsection}{\@arabic\c@section.\@arabic\c@subsection}
2523 \else
2524 \renewcommand{\thesection}{\@arabic\c@section}
2525 \renewcommand{\thesubsection}{\thesection.\@arabic\c@subsection}
2526 \fi
2527 %</!book&!report>
2528 %<*book|report>
2529 \renewcommand{\thechapter}{\@arabic\c@chapter}
2530 \renewcommand{\thesection}{\thechapter.\@arabic\c@section}
2531 \renewcommand{\thesubsection}{\thesection.\@arabic\c@subsection}
2532 %</book|report>
2533 \renewcommand{\thesubsubsection}{%
2534 \thesubsection.\@arabic\c@subsubsection}
2535 \renewcommand{\theparagraph}{%
2536 \thesubsubsection.\@arabic\c@paragraph}
2537 \renewcommand{\thesubparagraph}{%
2538 \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。

```
2539 %<book|report>\newcommand{\@chapapp}{\prechaptername}
2540 %<book|report>\newcommand{\@chappos}{\postchaptername}
```

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

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

```
2541 %<*book|report>
2542 \newcommand\frontmatter{%
2543   \pltx@cleartooddpage
2544   \@mainmatterfalse
2545   \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。

```
2546 \newcommand\mainmatter{%
2547   \pltx@cleartooddpage
2548   \@mainmattertrue
2549   \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。

```
2550 \newcommand\backmatter{%
2551   \if@openleft
2552     \cleardoublepage
2553   \else\if@openright
2554     \cleardoublepage
2555   \else
2556     \clearpage
2557   \fi\fi
2558   \@mainmatterfalse}
2559 %</book|report>
```

■部

`\part` 新しい部を始めます。

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

```
\secdef{星なし}{星あり}
```

星なし * のない形の定義です。

星あり * のある形の定義です。

`\secdef` は次のようにして使います。

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

まず `book` と `report` のクラス以外です。

```
2560 %<!*book&!report>
2561 \newcommand\part{%
2562   \ifnoskipsec \leavevmode \fi
2563   \par
2564   \addvspace{4ex}%
2565   \if@english \@afterindentfalse \else \@afterindenttrue \fi
2566   \secdef\@part\@spart}
2567 %</!*book&!report>
```

`book` および `report` クラスの場合は、少し複雑です。

```
2568 %<*book|report>
2569 \newcommand\part{%
2570   \if@openleft
2571     \cleardoublepage
2572   \else\if@openright
2573     \cleardoublepage
2574   \else
2575     \clearpage
2576   \fi\fi
2577   \thispagestyle{empty}% 欧文用標準スタイルでは plain
2578   \if@twocolumn
2579     \onecolumn
2580     \@restonecoltrue
2581   \else
2582     \@restonecolfalse
2583   \fi
2584   \null\vfil
2585   \secdef\@part\@spart}
2586 %</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。

```
2587 %<!*book&!report>
2588 \def\@part [#1]#2{%
2589   \ifnum \c@secnumdepth >\m@ne
2590     \refstepcounter{part}%
2591     \addcontentsline{toc}{part}{%
2592       \prepartname\thepart\postpartname\hspace{1\jsZw}#1}%
```

```

2593 \else
2594   \addcontentsline{toc}{part}{#1}%
2595 \fi
2596 \markboth{}{}%
2597 {\parindent\z@
2598   \raggedright
2599   \interlinepenalty \@M
2600   \normalfont
2601   \ifnum \c@secnumdepth >\m@ne
2602     \Large\headfont\prepartname\thepart\postpartname
2603     \par\nobreak
2604   \fi
2605   \huge \headfont #2%
2606   \markboth{}{}\par}%
2607 \nobreak
2608 \vskip 3ex
2609 \@afterheading}
2610 %<!/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。

```

2611 %<*book|report>
2612 \def\@part[#1]#2{%
2613   \ifnum \c@secnumdepth >-2\relax
2614     \refstepcounter{part}%
2615     \addcontentsline{toc}{part}{%
2616       \prepartname\thepart\postpartname\hspace{1\jsZw}#1}%
2617   \else
2618     \addcontentsline{toc}{part}{#1}%
2619   \fi
2620   \markboth{}{}%
2621   {\centering
2622     \interlinepenalty \@M
2623     \normalfont
2624     \ifnum \c@secnumdepth >-2\relax
2625       \huge\headfont \prepartname\thepart\postpartname
2626       \par\vskip20\jsc@mp
2627     \fi
2628     \Huge \headfont #2\par}%
2629   \@endpart}
2630 %</book|report>

```

\@spart 番号を付けない部です。

```

2631 %<*!book&!report>
2632 \def\@spart#1{%
2633   \parindent \z@ \raggedright
2634   \interlinepenalty \@M
2635   \normalfont
2636   \huge \headfont #1\par}%
2637 \nobreak

```

```

2638 \vskip 3ex
2639 \@afterheading}
2640 %</!book&!report>
2641 %<*book|report>
2642 \def\@spart#1{%
2643   \centering
2644   \interlinepenalty \@M
2645   \normalfont
2646   \Huge \headfont #1\par}%
2647 \@endpart}
2648 %</book|report>

```

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

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

```

2649 %<*book|report>
2650 \def\@endpart{\vfil\newpage
2651   \if@twoside
2652     \if@openleft %% added (2017/02/24)
2653       \null\thispagestyle{empty}\newpage
2654     \else\if@openright %% added (2016/12/13)
2655       \null\thispagestyle{empty}\newpage
2656     \fi\fi %% added (2016/12/13, 2017/02/24)
2657   \fi
2658   \if@restonecol
2659     \twocolumn
2660   \fi}
2661 %</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。

```

2662 %<*book|report>
2663 \newcommand{\chapter}{%
2664   \if@openleft\cleardoublepage\else
2665   \if@openright\cleardoublepage\else\clearpage\fi\fi
2666   \plainifnotempty % 元: \thispagestyle{plain}
2667   \global\@topnum\z@
2668   \if@english \@afterindentfalse \else \@afterindenttrue \fi
2669   \secdef
2670     {\@omit@numberfalse\@chapter}%
2671     {\@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。

```

2672 \def\@chapter[#1]#2{%
2673   \ifnum \c@secnumdepth >\m@ne
2674     \if@mainmatter
2675       \refstepcounter{chapter}%
2676       \typeout{\@chapapp\thechapter\@chappos}%
2677       \addcontentsline{toc}{chapter}%
2678         {\protect\numberline
2679 %       %{\if@english\thechapter\else\@chapapp\thechapter\@chappos\fi}%
2680         {\@chapapp\thechapter\@chappos}%
2681         #1}%
2682     \else\addcontentsline{toc}{chapter}{#1}\fi
2683 \else
2684   \addcontentsline{toc}{chapter}{#1}%
2685 \fi
2686 \chaptermark{#1}%
2687 \addtocontents{lof}{\protect\advspace{10\jsc@empt}}%
2688 \addtocontents{lot}{\protect\advspace{10\jsc@empt}}%
2689 \if@twocolumn
2690   \@topnewpage[\@makechapterhead{#2}]%
2691 \else
2692   \@makechapterhead{#2}%
2693   \@afterheading
2694 \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。

```

2695 \def\@makechapterhead#1{%
2696   \vspace*{2\Cvs}% 欧文は 50pt
2697   {\parindent \z@ \raggedright \normalfont
2698     \ifnum \c@secnumdepth >\m@ne
2699       \if@mainmatter
2700         \huge\headfont \@chapapp\thechapter\@chappos
2701         \par\nobreak
2702         \vskip \Cvs % 欧文は 20pt
2703         \fi
2704       \fi
2705       \interlinepenalty\@M
2706       \Huge \headfont #1\par\nobreak
2707       \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。

```

2708 \def\@schapter#1{%
2709   \chaptermark{#1}%
2710   \if@twocolumn
2711     \@topnewpage[\@makeschapterhead{#1}]%
2712   \else
2713     \@makeschapterhead{#1}\@afterheading
2714   \fi}

```

`\@makeschapterhead` 番号なしの章見出しです。

```

2715 \def\@makeschapterhead#1{%
2716   \vspace*{2\Cvs}% 欧文は 50pt
2717   {\parindent \z@ \raggedright
2718     \normalfont
2719     \interlinepenalty\@M
2720     \Huge \headfont #1\par\nobreak
2721     \vskip 3\Cvs}} % 欧文は 40pt
2722 %</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。

```

2723 \if@twocolumn
2724   \newcommand{\section}{%
2725 %<jspf>\ifx\maketitle\relax\else\maketitle\fi
2726   \@startsection{section}{1}{\z@}%
2727 %<kiyou>   {0.6\Cvs}{0.4\Cvs}%
2728 %<kiyou>   {\Cvs}{0.5\Cvs}%
2729 %   {\normalfont\large\headfont\@secapp}}
2730   {\normalfont\large\headfont\raggedright}}
2731 \else
2732   \newcommand{\section}{%
2733     \if@slide\clearpage\fi
2734     \@startsection{section}{1}{\z@}%
2735     {\Cvs \@plus.5\Cdp \@minus.2\Cdp}% 前アキ
2736     {.5\Cvs \@plus.3\Cdp}% 後アキ
2737 %   {\normalfont\Large\headfont\@secapp}}
2738     {\normalfont\Large\headfont\raggedright}}
2739 \fi

```

`\subsection` 同上です。

```

2740 \if@twocolumn
2741   \newcommand{\subsection}{\@startsection{subsection}{2}{\z@}%
2742     {\z@}{\if@slide .4\Cvs \else \z@ \fi}%
2743     {\normalfont\normalsize\headfont}}
2744 \else
2745   \newcommand{\subsection}{\@startsection{subsection}{2}{\z@}%
2746     {\Cvs \@plus.5\Cdp \@minus.2\Cdp}% 前アキ
2747     {.5\Cvs \@plus.3\Cdp}% 後アキ
2748     {\normalfont\large\headfont}}
2749 \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)。

```

2750 \if@twocolumn
2751   \newcommand{\subsubsection}{\@startsection{subsubsection}{3}{\z@}%

```



```

2752     {\z@}{\if@slide .4\Cvs \else \z@ \fi}%
2753     {\normalfont\normalsize\headfont}}
2754 \else
2755   \newcommand{\subsubsection}{\@startsection{subsubsection}{3}{\z@}%
2756     {\Cvs \@plus.5\Cdp \@minus.2\Cdp}%
2757     {\if@slide .5\Cvs \@plus.3\Cdp \else \z@ \fi}%
2758     {\normalfont\normalsize\headfont}}
2759 \fi

```

`\paragraph` 見出しの後ろで改行されません。

`\jsParagraphMark` [2016-11-16] 従来は `\paragraph` の最初に出るマークを「■」に固定していましたが、このマークを変更可能にするため `\jsParagraphMark` というマクロに切り出しました。これで、たとえば

```
\renewcommand{\jsParagraphMark}{★}
```

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

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

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

```
2760 \let\jsJaChar\@empty
```

```

2761 \newcommand\jsParagraphMark{\relax\jsJaChar{■}}
2762 \let\bxjs@org@paragraph@mark\jsParagraphMark
2763 \ifx\bxjs@paragraph@mark\@empty
2764   \let\jsParagraphMark\@empty
2765 \else\ifx\bxjs@paragraph@mark\@undefined\else
2766   \long\edef\jsParagraphMark{\noexpand\jsJaChar{\bxjs@paragraph@mark}}
2767 \fi\fi
2768 \if@twocolumn
2769   \newcommand{\paragraph}{\@startsection{paragraph}{4}{\z@}%
2770     {\z@}{\if@slide .4\Cvs \else -1\jsZw\fi}% 改行せず 1\jsZw のアキ
2771     %<jspf>   {\normalfont\normalsize\headfont}}
2772     %<!jspf>   {\normalfont\normalsize\headfont\jsParagraphMark}}
2773 \else
2774   \newcommand{\paragraph}{\@startsection{paragraph}{4}{\z@}%
2775     {0.5\Cvs \@plus.5\Cdp \@minus.2\Cdp}%
2776     {\if@slide .5\Cvs \@plus.3\Cdp \else -1\jsZw\fi}% 改行せず 1\jsZw のアキ
2777     %<jspf>   {\normalfont\normalsize\headfont}}
2778     %<!jspf>   {\normalfont\normalsize\headfont\jsParagraphMark}}
2779 \fi

```

`\subparagraph` 見出しの後ろで改行されません。

```
2780 \if@twocolumn
```

```

2781 \newcommand{\subparagraph}{\@startsection{subparagraph}{5}{\z@}%
2782   {\z@}{\if@slide .4\Cvs \@plus.3\Cdp \else -1\jsZw\fi}%
2783   {\normalfont\normalsize\headfont}}
2784 \else
2785 \newcommand{\subparagraph}{\@startsection{subparagraph}{5}{\z@}%
2786   {\z@}{\if@slide .5\Cvs \@plus.3\Cdp \else -1\jsZw\fi}%
2787   {\normalfont\normalsize\headfont}}
2788 \fi

```

8.3 リスト環境

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

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

[2002-05-11] 3zw に変更しました。

[2005-03-19] 二段組は 2zw に戻しました。

```

2789 \if@slide
2790   \setlength\leftmargini{1\jsZw}
2791 \else
2792   \if@twocolumn
2793     \setlength\leftmargini{2\jsZw}
2794   \else
2795     \setlength\leftmargini{3\jsZw}
2796   \fi
2797 \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 2798 \if@slide
\leftmarginv 2799   \setlength\leftmarginii {1\jsZw}
\leftmarginvi 2800   \setlength\leftmarginiii{1\jsZw}
2801   \setlength\leftmarginiv {1\jsZw}
2802   \setlength\leftmarginv  {1\jsZw}
2803   \setlength\leftmarginvi {1\jsZw}
2804 \else
2805   \setlength\leftmarginii {2\jsZw}
2806   \setlength\leftmarginiii{2\jsZw}
2807   \setlength\leftmarginiv {2\jsZw}
2808   \setlength\leftmarginv  {1\jsZw}
2809   \setlength\leftmarginvi {1\jsZw}
2810 \fi

```

`\labelsep` `\labelwidth` 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。

```

2811 \setlength \labelsep {0.5\jsZw} % .5em

```

```
2812 \setlength \labelwidth{\leftmarginI}
2813 \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。

```
2814 \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 2815 \@beginparpenalty -\@lowpenalty
\@itempenalty 2816 \@endparpenalty -\@lowpenalty
2817 \@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 $\begin{smallmatrix} +0.2 \\ -0.1 \end{smallmatrix}$ `\baselineskip` を思い切って外しました。

```
2818 \def\@listi{\leftmargin\leftmarginI
2819 \parsep \z@
2820 \topsep 0.5\baselineskip
2821 \itemsep \z@ \relax}
2822 \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）。

```
2823 \@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 2824 \def\@listii{\leftmargin\leftmarginii
\@listiv 2825 \labelwidth\leftmarginii \advance\labelwidth-\labelsep
2826 \topsep \z@
\@listv 2827 \parsep \z@
\@listvi 2828 \itemsep\parsep}
2829 \def\@listiii{\leftmargin\leftmarginiii
2830 \labelwidth\leftmarginiii \advance\labelwidth-\labelsep
2831 \topsep \z@
2832 \parsep \z@
2833 \itemsep\parsep}
2834 \def\@listiv {\leftmargin\leftmarginiv
2835 \labelwidth\leftmarginiv
2836 \advance\labelwidth-\labelsep}
2837 \def\@listv {\leftmargin\leftmarginv
2838 \labelwidth\leftmarginv
2839 \advance\labelwidth-\labelsep}
2840 \def\@listvi {\leftmargin\leftmarginvi
```

```
2841 \labelwidth\leftmarginiv
2842 \advance\labelwidth-\labelsep}
```

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

`\theenumi` 出力する番号の書式を設定します。これらは L^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。

```
2843 \renewcommand{\theenumi}{\@arabic\c@enumi}
2844 \renewcommand{\theenumii}{\@alph\c@enumii}
2845 \renewcommand{\theenumiii}{\@roman\c@enumiii}
2846 \renewcommand{\theenumiv}{\@Alph\c@enumiv}
```

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

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

```
2847 \newcommand*{\jsInJaParen}[1]{%
2848 \mbox{\jsInhibitGlue (#1) \jsInhibitGlue}}
2849 \newcommand{\labelenumi}{\theenumi.}
2850 \newcommand{\labelenumii}{\jsInJaParen{\theenumii}}
2851 \newcommand{\labelenumiii}{\theenumiii.}
2852 \newcommand{\labelenumiv}{\theenumiv.}
```

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

```
\p@enumiv 2853 \renewcommand{\p@enumii}{\theenumi}
2854 \renewcommand{\p@enumiii}{\theenumi\jsInhibitGlue (\theenumii )}
2855 \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 2856 \newcommand\labelitemi{\textbullet}
\labelitemiii 2857 \newcommand\labelitemii{\normalfont\bfseries \textendash}
\labelitemiv 2858 \newcommand\labelitemiii{\textasteriskcentered}
2859 \newcommand\labelitemiv{\textperiodcentered}
```

■description 環境

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

```
2860 \newenvironment{description}{%
2861   \list{}{%
2862     \labelwidth=\leftmargin
2863     \labelsep=1\jsZw
2864     \advance \labelwidth by -\labelsep
2865     \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。

```
2866 \newcommand*\descriptionlabel[1]{\normalfont\headfont #1\hfil}
```

■概要

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

JSPF スタイルでは実際の出力は `\maketitle` で行われます。

bxjsreport クラスの abstract 環境は：

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

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

```
2867 %<*book|report>
2868 \newenvironment{chapterabstract}{%
2869   \begin{list}{}{%
2870     \listparindent=1\jsZw
2871     \itemindent=\listparindent
2872     \rightmargin=0pt
2873     \leftmargin=5\jsZw\item[]}\end{list}\vspace{\baselineskip}}
2874 %</book|report>
```

“普通の” abstract 環境の定義。

```
2875 %<*article|report|slide>
2876 \newbox\@abstractbox
2877 \if@titlepage
2878   \newenvironment{abstract}{%
2879     \titlepage
2880     \null\vfil
```

```

2881 \beginparpenalty\@lowpenalty
2882 \begin{center}%
2883 \headfont \abstractname
2884 \endparpenalty\@M
2885 \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。

```

2886 \par}%
2887 {\par\vfil\null\endtitlepage}
2888 \else
2889 \newenvironment{abstract}{%
2890 \if@twocolumn
2891 \ifx\maketitle\relax
2892 \section*{\abstractname}%
2893 \else
2894 \global\setbox\@abstractbox\hbox\bgroup
2895 \begin{minipage}[b]{\textwidth}
2896 \small\parindent1\jsZw
2897 \begin{center}%
2898 {\headfont \abstractname\vspace{-.5em}\vspace{\z@}}%
2899 \end{center}%
2900 \list{}{%
2901 \listparindent\parindent
2902 \itemindent \listparindent
2903 \rightmargin \leftmargin}%
2904 \item\relax
2905 \fi
2906 \else
2907 \small
2908 \begin{center}%
2909 {\headfont \abstractname\vspace{-.5em}\vspace{\z@}}%
2910 \end{center}%
2911 \list{}{%
2912 \listparindent\parindent
2913 \itemindent \listparindent
2914 \rightmargin \leftmargin}%
2915 \item\relax
2916 \fi}{\if@twocolumn
2917 \ifx\maketitle\relax
2918 \else
2919 \endlist\end{minipage}\egroup
2920 \fi
2921 \else
2922 \endlist
2923 \fi}
2924 \fi
2925 %</article|report|slide>
2926 %<*jspf>
2927 \newbox\@abstractbox

```

```

2928 \newenvironment{abstract}{%
2929 \global\setbox\@abstractbox\hbox\bgroup
2930 \begin{minipage}[b]{157\jsc@mmm}{\sffamily Abstract}\par
2931 \small
2932 \if@english \parindent6\jsc@mmm \else \parindent1\jsZw \fi}%
2933 {\end{minipage}\egroup}
2934 %</jspf>

```

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

```

2935 %<*book|report>
2936 \ifbxjs@force@chapterabstract
2937 \let\abstract\chapterabstract
2938 \let\endabstract\endchapterabstract
2939 \fi
2940 %</book|report>

```

■キーワード

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

```

2941 %<*jspf>
2942 %\newbox\@keywordsbox
2943 %\newenvironment{keywords}{%
2944 % \global\setbox\@keywordsbox\hbox\bgroup
2945 % \begin{minipage}[b]{1570\jsc@mmm}{\sffamily Keywords:}\par
2946 % \small\parindent0\jsZw}%
2947 % {\end{minipage}\egroup}
2948 %</jspf>

```

■verse 環境

`verse (env.)` 詩のための `verse` 環境です。

```

2949 \newenvironment{verse}{%
2950 \let \=\@centercr
2951 \list{}{%
2952 \itemsep \z@
2953 \itemindent -2\jsZw % 元: -1.5em
2954 \listparindent\itemindent
2955 \rightmargin \z@
2956 \advance\leftmargin 2\jsZw}% 元: 1.5em
2957 \item\relax}{\endlist}

```

■quotation 環境

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

```

2958 \newenvironment{quotation}{%
2959   \list{}{%
2960     \listparindent\parindent
2961     \itemindent\listparindent
2962     \rightmargin \z@}%
2963   \item\relax}{\endlist}

```

■quote 環境

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

```

2964 \newenvironment{quote}%
2965   {\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。

```

2966 \def\@begintheorem#1#2{\trivlist\labelsep=1\jsZw
2967   \item[\hskip \labelsep{\headfont #1\ #2}]}
2968 \def\@opargbegintheorem#1#2#3{\trivlist\labelsep=1\jsZw
2969   \item[\hskip \labelsep{\headfont #1\ #2 (#3)}]}

```

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

[2017-02-24] コミュニティ版 pL^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。

```

2970 \newenvironment{titlepage}{%
2971   %<book>   \pltx@cleartooddpage %% 2017-02-24
2972   \if@twocolumn
2973     \@restonecoltrue\onecolumn
2974   \else
2975     \@restonecolfalse\newpage
2976   \fi
2977   \thispagestyle{empty}%
2978   \ifodd\c@page\setcounter{page}\@ne\else\setcounter{page}\z@\fi %% 2017-
2979   }%

```



```

2980 {\if@restonecol\twocolumn \else \newpage \fi
2981 \if@twoside\else
2982 \setcounter{page}\@ne
2983 \fi}

```

■付録

`\appendix` 本文と付録を分離するコマンドです。

```

2984 %<!*book&!report>
2985 \newcommand{\appendix}{\par
2986 \setcounter{section}{0}%
2987 \setcounter{subsection}{0}%
2988 \ifnum\bxjs@label@section=\bxjs@label@section@@compat
2989 \gdef\presectionname{\appendixname}%
2990 \gdef\postsectionname{}%
2991 % \gdef\thesection{\@Alph\c@section}% [2003-03-02]
2992 \gdef\thesection{\presectionname\@Alph\c@section\postsectionname}%
2993 \gdef\thesubsection{\@Alph\c@section.\@arabic\c@subsection}%
2994 \else
2995 \gdef\@secapp{\appendixname}%
2996 \gdef\@secpos{}%
2997 \gdef\thesection{\@Alph\c@section}%
2998 \fi}
2999 %</!*book&!report>
3000 %<*book|report>
3001 \newcommand{\appendix}{\par
3002 \setcounter{chapter}{0}%
3003 \setcounter{section}{0}%
3004 \gdef\@chapapp{\appendixname}%
3005 \gdef\@chappos{}%
3006 \gdef\thechapter{\@Alph\c@chapter}}
3007 %</book|report>

```

8.4 パラメータの設定

■array と tabular 環境

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

```
3008 \setlength\arraycolsep{5\jsc@empt}
```

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

```
3009 \setlength\tabcolsep{6\jsc@empt}
```

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

```
3010 \setlength\arrayrulewidth{.4\p@}
```

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

```
3011 \setlength\doublerulesep{2\p@}
```

■tabbing 環境

`\tabbingsep` \ ' コマンドで入るアキです。

```
3012 \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。

```
3013 \skip\@mpfootins = \skip\footins
```

■framebox 環境

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

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

```
3014 \setlength\fboxsep{3\jsc@mp}
```

```
3015 \setlength\fboxrule{.4\p@}
```

■equation と eqnarray 環境

`\theequation` 数式番号を出力するコマンドです。

```
3016 %<!book&!report>\renewcommand \theequation {\@arabic\c@equation}
```

```
3017 %<*book|report>
```

```
3018 \@addtoreset{equation}{chapter}
```

```
3019 \renewcommand\theequation
```

```
3020 {\ifnum \c@chapter>\z@ \thechapter.\fi \@arabic\c@equation}
```

```
3021 %</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。

```
3022 % \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。

```
3023 % \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。

```
3024 % \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。

```
3025 %<!*book&!report>
3026 \newcounter{figure}
3027 \renewcommand \thefigure {\@arabic\c@figure}
3028 %</!*book&!report>
3029 %<*book|report>
3030 \newcounter{figure}[chapter]
3031 \renewcommand \thefigure
3032     {\ifnum \c@chapter>\z@ \thechapter.\fi \@arabic\c@figure}
3033 %</book|report>
```

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

```
\ext@figure 3034 \def\fps@figure{tbp}
3035 \def\ftype@figure{1}
\fnum@figure 3036 \def\ext@figure{lof}
3037 \def\fnum@figure{\figurename\nobreak\thefigure}
```

`figure (env.)` * 形式は段抜きのフロートです。

```
figure* (env.) 3038 \newenvironment{figure}%
3039     {\@float{figure}}%
3040     {\end@float}
3041 \newenvironment{figure*}%
3042     {\@dblfloat{figure}}%
3043     {\end@dblfloat}
```

■table 環境

`\c@table` 表番号カウンタと表番号を出力するコマンドです。アスキー版では `\thechapter.` が `\thetable \thechapter{}` になっていますが、ここではオリジナルのままにしています。

```
3044 %<!*book&!report>
3045 \newcounter{table}
3046 \renewcommand\thetable{\@arabic\c@table}
3047 %</!*book&!report>
3048 %<*book|report>
3049 \newcounter{table}[chapter]
```

```

3050 \renewcommand \thetable
3051     {\ifnum \c@chapter>\z@ \thechapter.\fi \@arabic\c@table}
3052 %</book|report>

```

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

```

\ext@table 3053 \def\fps@table{tbp}
3054 \def\ftype@table{2}
\fnun@table 3055 \def\ext@table{lot}
3056 \def\fnun@table{\tablename\nobreak\thetable}

```

`table (env.) *` は段抜きのフロートです。

```

table* (env.) 3057 \newenvironment{table}%
3058             {\@float{table}}%
3059             {\end@float}
3060 \newenvironment{table*}%
3061             {\@dblfloat{table}}%
3062             {\end@dblfloat}

```

8.6 キャプション

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

`\abovecaptionskip` それぞれキャプションの前後に挿入されるスペースです。 `\belowcaptionskip` が 0 になっ
`\belowcaptionskip` ていたので、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。

```

3063 \newlength\abovecaptionskip
3064 \newlength\belowcaptionskip
3065 \setlength\abovecaptionskip{5\jsc@mp} % 元: 10\p@
3066 \setlength\belowcaptionskip{5\jsc@mp} % 元: 0\p@

```

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

[2003-11-05] ロジックを少し変えてみました。

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

```

3067 %<*\jspf>
3068 % \long\def\@makecaption#1#2{\small
3069 %   \advance\leftskip10\jsc@mm
3070 %   \advance\rightskip10\jsc@mm
3071 %   \vskip\abovecaptionskip
3072 %   \sbox\@tempboxa{#1\hskip1\jsZw\relax #2}%
3073 %   \ifdim \wd\@tempboxa >\hsize
3074 %     #1\hskip1\jsZw\relax #2\par
3075 %   \else

```

```

3076 % \global \@minipagefalse
3077 % \hb@xt@\hsize{\hfil\box\@tempboxa\hfil}%
3078 % \fi
3079 % \vskip\belowcaptionskip}}
3080 \long\def\@makecaption#1#2{{\small
3081 \advance\leftskip .0628\linewidth
3082 \advance\rightskip .0628\linewidth
3083 \vskip\abovcaptionskip
3084 \sbox\@tempboxa{#1\zspace#2}%
3085 \ifdim \wd\@tempboxa <\hsize \centering \fi
3086 #1\zspace#2\par
3087 \vskip\belowcaptionskip}}
3088 %<!/jspf>
3089 %<*jspf>
3090 \long\def\@makecaption#1#2{%
3091 \vskip\abovcaptionskip
3092 \sbox\@tempboxa{\small\sffamily #1\quad #2}%
3093 \ifdim \wd\@tempboxa >\hsize
3094 {\small\sffamily
3095 \list{#1}{%
3096 \renewcommand{\makelabel}[1]{##1\hfil}
3097 \itemsep \z@
3098 \itemindent \z@
3099 \labelsep \z@
3100 \labelwidth 11\jsc@mmm
3101 \listparindent\z@
3102 \leftmargin 11\jsc@mmm}\item\relax #2\endlist}
3103 \else
3104 \global \@minipagefalse
3105 \hb@xt@\hsize{\hfil\box\@tempboxa\hfil}%
3106 \fi
3107 \vskip\belowcaptionskip}
3108 %</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 `\scr@DeclareOldFontCommand` に倣い、これらの命令を使うときには警告を発することにしました。

[2016-07-16] 警告を最初の一回だけ発することにしました。また、例外的に警告を出さないようにするスイッチも付けます。

```
\if@jsc@warnoldfontcmd
```

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

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

```
3109 \newif\if@jsc@warnoldfontcmd
3110 \@jsc@warnoldfontcmdtrue
3111 \newif\if@jsc@warnoldfontcmdexception
3112 \@jsc@warnoldfontcmdexceptionfalse
```

`\jsc@DeclareOldFontCommand`

```
3113 \newcommand*{\jsc@DeclareOldFontCommand}[3]{%
3114   \g@addto@macro\bxjs@oldfontcmd@list{\do#1}%
3115   \DeclareOldFontCommand{#1}{%
3116     \bxjs@oldfontcmd{#1}#2%
3117   }{%
3118     \bxjs@oldfontcmd{#1}#3%
3119   }%
3120 }
3121 \DeclareRobustCommand*{\jsc@warnoldfontcmd}[1]{%
3122   \ClassInfo\bxjs@clsname
3123   {Old font command '\string#1' is used!!\MessageBreak
3124     The first occurrence is}%
3125 }
```

`\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。

```
3126 \newcommand*{\allowoldfontcommands}{%
3127   \@jsc@warnoldfontcmdexceptiontrue}
3128 \newcommand*{\disallowoldfontcommands}{%
3129   \@jsc@warnoldfontcmdexceptionfalse}
3130 \let\bxjs@oldfontcmd@list\@empty
3131 \def\bxjs@oldfontcmd#1{%
3132   \expandafter\bxjs@oldfontcmd@a\csname bxjs@ofc/\string#1\endcsname#1}
3133 \def\bxjs@oldfontcmd@a#1#2{%
3134   \if@jsc@warnoldfontcmdexception\else
3135     \global\@jsc@warnoldfontcmdfalse
3136     \ifx#1\relax
3137       \global\let#1=t%
3138       \jsc@warnoldfontcmd{#2}%
3139     \fi
3140   \fi}
3141 \def\bxjs@warnoldfontcmd@final{%
3142 % \par
3143   \global\let\bxjs@warnoldfontcmd@final\@empty
3144   \let\@tempa\@empty
3145   \def\do##1{%
```

```

3146 \ifundefined{bxjs@ofc/\string##1}{\%else
3147 \edef\@tempa{\@tempa \space\string##1}}}%
3148 \bxjs@oldfontcmd@list
3149 \ifx\@tempa\@empty\else
3150 \ClassWarningNoLine{bxjs@clsname
3151 {Some old font commands were used in text:\MessageBreak
3152 \space\@tempa\MessageBreak
3153 You should note, that since 1994 LaTeX2e provides a\MessageBreak
3154 new font selection scheme called NFSS2 with several\MessageBreak
3155 new, combinable font commands. The
3156 class provides\MessageBreak
3157 the old font commands only for compatibility}}
3158 \fi}

```

単純に `\AtEndDocument` のフックの中で `\bxjs@warnoldfontcmd@final` を実行した場合、最終ページのヘッダ・フッタの中にある二文字フォント命令はそれより後に実行されるため捕捉できない。これに対処するため、`\end{document}` 中に実行される `\clearpage` の処理の直後に `\bxjs...final` が呼ばれるようにする。

※新しい L^AT_EX ではフックシステムの機能を利用する。

```

3159 \ifbxjs@old@hook@system
3160 \AtEndDocument{%
3161 \g@addto@macro\clearpage{\bxjs@warnoldfontcmd@final}}
3162 \else
3163 \AddToHook{enddocument/afterlastpage}{\bxjs@warnoldfontcmd@final}
3164 \fi

```

`\mc` フォントファミリーを変更します。

```

\gt 3165 \jsc@DeclareOldFontCommand{\mc}{\normalfont\mcfamily}{\mathmc}
\rm 3166 \jsc@DeclareOldFontCommand{\gt}{\normalfont\gtfamily}{\mathgt}
3167 \jsc@DeclareOldFontCommand{\rm}{\normalfont\rmfamily}{\mathrm}
\sf 3168 \jsc@DeclareOldFontCommand{\sf}{\normalfont\sffamily}{\mathsf}
\tt 3169 \jsc@DeclareOldFontCommand{\tt}{\normalfont\ttfamily}{\mathtt}

```

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

```

3170 \jsc@DeclareOldFontCommand{\bf}{\normalfont\bfseries}{\mathbf}

```

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

```

3171 \jsc@DeclareOldFontCommand{\it}{\normalfont\itshape}{\mathit}
3172 \jsc@DeclareOldFontCommand{\sl}{\normalfont\slshape}{\@nomath\sl}
3173 \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 3174 \DeclareRobustCommand*\cal{\@fontswitch\relax\mathcal}
3175 \DeclareRobustCommand*\mit{\@fontswitch\relax\mathnormal}

```

10 相互参照

10.1 目次の類

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

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

たとえば `\section` に見出し番号が付く場合、上の「タイトル」は

```
\numberline{番号}{見出し}
```

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

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

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

この「番号」は `\thefigure` コマンドで生成された図番号です。

`table` 環境も同様です。

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

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

という書式です。

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

インデント 左側の字下げ量です。

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

`\@pnumwidth` ページ番号の入る箱の幅です。

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

`\@dotsep` 点の間隔です (単位 `mu`)。

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

```
3176 \newcommand\@pnumwidth{1.55em}
```

```
3177 \newcommand\@tocrmarg{2.55em}
```

```
3178 \newcommand\@dotsep{4.5}
```

```
3179 %<!book&!report>\setcounter{tocdepth}{2}
```

```
3180 %<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)

```
3181 \newdimen\jsc@tocl@width
3182 \newcommand{\tableofcontents}{%
3183 %<*book|report>
3184 \settowidth\jsc@tocl@width{\headfont\prechaptername\postchaptername}%
3185 \settowidth\@tempdima{\headfont\appendixname}%
3186 \ifdim\jsc@tocl@width<\@tempdima \setlength\jsc@tocl@width{\@tempdima}\fi
3187 \ifdim\jsc@tocl@width<2\jsZw \divide\jsc@tocl@width by 2 \advance\jsc@tocl@width 1\jsZw\fi
3188 \if@twocolumn
3189 \@restonecoltrue\onecolumn
3190 \else
3191 \@restonecolfalse
3192 \fi
3193 \chapter*{\contentsname}%
3194 \@mkboth{\contentsname}{}%
3195 %</book|report>
3196 %<!*book&!report>
3197 \settowidth\jsc@tocl@width{\headfont\presectionname\postsectionname}%
3198 \settowidth\@tempdima{\headfont\appendixname}%
3199 \ifdim\jsc@tocl@width<\@tempdima\relax\setlength\jsc@tocl@width{\@tempdima}\fi
3200 \ifdim\jsc@tocl@width<2\jsZw \divide\jsc@tocl@width by 2 \advance\jsc@tocl@width 1\jsZw\fi
3201 \section*{\contentsname}%
3202 \@mkboth{\contentsname}{\contentsname}%
3203 %</!book&!report>
3204 \@starttoc{toc}%
3205 %<book|report> \if@restonecol\twocolumn\fi
3206 }
```

`\l@part` 部の目次です。

```
3207 \newcommand*{\l@part}[2]{%
3208 \ifnum \c@tocdepth >-2\relax
3209 %<!book&!report> \addpenalty\@secpenalty
3210 %<book|report> \addpenalty{-\@highpenalty}%
3211 \addvspace{2.25em \@plus\jsc@empt}%
3212 \begingroup
3213 \parindent \z@
3214 % \@pnumwidth should be \@tocrmarg
3215 % \rightskip \@pnumwidth
3216 \rightskip \@tocrmarg
3217 \parfillskip -\rightskip
3218 {\leavevmode
3219 \large \headfont
3220 \setlength\@lnumwidth{4\jsZw}%
3221 #1\hfil \hb@xt@\@pnumwidth{\hss #2}}\par
3222 \nobreak
3223 %<book|report> \global\@nobreaktrue
3224 %<book|report> \everypar{\global\@nobreakfalse\everypar{}}%
```

```
3225 \endgroup
3226 \fi}
```

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

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

```
3227 %<*book|report>
3228 \newcommand*{\l@chapter}[2]{%
3229 \ifnum \c@tocdepth >\m@ne
3230 \addpenalty{-\@highpenalty}%
3231 \addvspace{1.0em \@plus\jsc@mp}
3232 % \vskip 1.0em \@plus\p@ % book.cls では↑がこうなっている
3233 \begingroup
3234 \parindent\z@
3235 % \rightskip\@pnumwidth
3236 \rightskip\@tocrmarg
3237 \parfillskip-\rightskip
3238 \leavevmode\headfont
3239 % % \if@english\setlength\@lnumwidth{5.5em}\else\setlength\@lnumwidth{4.683\jsZw}\fi
3240 \setlength\@lnumwidth{\jsc@tocl@width}\advance\@lnumwidth 2.683\jsZw
3241 \advance\leftskip\@lnumwidth \hskip-\leftskip
3242 #1\nobreak\hfil\nobreak\hbox to\@pnumwidth{\hss#2}\par
3243 \penalty\@highpenalty
3244 \endgroup
3245 \fi}
3246 %</book|report>
```

`\l@section` 節の目次です。

```
3247 %<!*book&!report>
3248 \newcommand*{\l@section}[2]{%
3249 \ifnum \c@tocdepth >\z@
3250 \addpenalty{\@secpenalty}%
3251 \addvspace{1.0em \@plus\jsc@mp}%
3252 \begingroup
3253 \parindent\z@
3254 % \rightskip\@pnumwidth
3255 \rightskip\@tocrmarg
3256 \parfillskip-\rightskip
3257 \leavevmode\headfont
3258 % % \setlength\@lnumwidth{4\jsZw}% 元 1.5em [2003-03-02]
3259 \setlength\@lnumwidth{\jsc@tocl@width}\advance\@lnumwidth 2\jsZw
3260 \advance\leftskip\@lnumwidth \hskip-\leftskip
3261 #1\nobreak\hfil\nobreak\hbox to\@pnumwidth{\hss#2}\par
3262 \endgroup
3263 \fi}
3264 %</!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。

```
3265 %<book|report> % \newcommand*{\l@section}{\@dottedtocline{1}{1\jsZw}{3.683\jsZw}}
```

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

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

\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 3266 %<!book&!report>
3267 % \newcommand*\l@subsection {\@dottedtocline{2}{1.5em}{2.3em}}
3268 % \newcommand*\l@subsubsection{\@dottedtocline{3}{3.8em}{3.2em}}
3269 % \newcommand*\l@paragraph {\@dottedtocline{4}{7.0em}{4.1em}}
3270 % \newcommand*\l@subparagraph {\@dottedtocline{5}{10em}{5em}}
3271 %
3272 % \newcommand*\l@subsection {\@dottedtocline{2}{1zw}{3zw}}
3273 % \newcommand*\l@subsubsection{\@dottedtocline{3}{2\jsZw}{3\jsZw}}
3274 % \newcommand*\l@paragraph {\@dottedtocline{4}{3\jsZw}{3\jsZw}}
3275 % \newcommand*\l@subparagraph {\@dottedtocline{5}{4\jsZw}{3\jsZw}}
3276 %
3277 \newcommand*\l@subsection}{%
3278     \@tempdima\jsc@tocl@width \advance\@tempdima -1\jsZw
3279     \@dottedtocline{2}{\@tempdima}{3\jsZw}}
3280 \newcommand*\l@subsubsection}{%
3281     \@tempdima\jsc@tocl@width \advance\@tempdima 0\jsZw
3282     \@dottedtocline{3}{\@tempdima}{4\jsZw}}
3283 \newcommand*\l@paragraph}{%
3284     \@tempdima\jsc@tocl@width \advance\@tempdima 1\jsZw
3285     \@dottedtocline{4}{\@tempdima}{5\jsZw}}
3286 \newcommand*\l@subparagraph}{%
3287     \@tempdima\jsc@tocl@width \advance\@tempdima 2\jsZw
3288     \@dottedtocline{5}{\@tempdima}{6\jsZw}}
3289 %</!book&!report>
3290 %<*book|report>
3291 % \newcommand*\l@subsection {\@dottedtocline{2}{3.8em}{3.2em}}
3292 % \newcommand*\l@subsubsection{\@dottedtocline{3}{7.0em}{4.1em}}
3293 % \newcommand*\l@paragraph {\@dottedtocline{4}{10em}{5em}}
3294 % \newcommand*\l@subparagraph {\@dottedtocline{5}{12em}{6em}}
3295 \newcommand*\l@section}{%
3296     \@tempdima\jsc@tocl@width \advance\@tempdima -1\jsZw
3297     \@dottedtocline{1}{\@tempdima}{3.683\jsZw}}
3298 \newcommand*\l@subsection}{%
3299     \@tempdima\jsc@tocl@width \advance\@tempdima 2.683\jsZw
3300     \@dottedtocline{2}{\@tempdima}{3.5\jsZw}}
3301 \newcommand*\l@subsubsection}{%
3302     \@tempdima\jsc@tocl@width \advance\@tempdima 6.183\jsZw
3303     \@dottedtocline{3}{\@tempdima}{4.5\jsZw}}
3304 \newcommand*\l@paragraph}{%
3305     \@tempdima\jsc@tocl@width \advance\@tempdima 10.683\jsZw
3306     \@dottedtocline{4}{\@tempdima}{5.5\jsZw}}
3307 \newcommand*\l@subparagraph}{%
3308     \@tempdima\jsc@tocl@width \advance\@tempdima 16.183\jsZw
```

```
3309         \@dottedtocline{5}{\@tempdima}{6.5\jsZw}}
3310 %</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
`\@lnumwidth` すが、アスキー版では `\@tempdima` の代わりに `\@lnumwidth` という変数で幅を決めるよう
に再定義しています。後続文字が全角か半角かでスペースが変わらないように `\hspace` を
入れておきました。

```
3311 \newdimen\@lnumwidth
3312 \def\numberline#1{\hb@xt@\@lnumwidth{#1\hfil}\hspace{0pt}}
```

`\@dottedtocline` L^AT_EX 本体 (l^tsect.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。

```
3313 \def\jsTocLine{\leaders\hbox{%
3314   $\m@th \mkern \@dotsep mu\hbox{.}\mkern \@dotsep mu$\}\hfill}
3315 \def\@dottedtocline#1#2#3#4#5{\ifnum #1>\c@tocdepth \else
3316   \vskip \z@ \@plus.2\jsc@mp
3317   {\leftskip #2\relax \rightskip \@tocrmarg \parfillskip -\rightskip
3318    \parindent #2\relax\@afterindenttrue
3319    \interlinepenalty\@M
3320    \leavevmode
3321    \@lnumwidth #3\relax
3322    \advance\leftskip \@lnumwidth \null\nobreak\hskip -\leftskip
3323    {#4}\nobreak
3324    \jsTocLine \nobreak\hb@xt@\@pnumwidth{%
3325      \hfil\normalfont \normalcolor #5}\par}\fi}
```

■ 図目次と表目次

`\listoffigures` 図目次を出力します。

```
3326 \newcommand{\listoffigures}{%
3327 %<*book|report>
3328   \if@twocolumn\@restonecoltrue\onecolumn
3329   \else\@restonecolfalse\fi
3330   \chapter*{\listfigurename}%
3331   \@mkboth{\listfigurename}{}%
3332 %</book|report>
3333 %<!*book&!report>
3334   \section*{\listfigurename}%
3335   \@mkboth{\listfigurename}{\listfigurename}%
3336 %</!*book&!report>
3337   \@starttoc{lof}}%
```

```
3338 %<book|report> \if@restonecol\twocolumn\fi
3339 }
```

`\l@figure` 図目次の項目を出力します。

```
3340 \newcommand*\l@figure{\@dottedtocline{1}{1\jsZw}{3.683\jsZw}}
```

`\listoftables` 表目次を出力します。

```
3341 \newcommand*\listoftables{%
3342 %<*book|report>
3343 \if@twocolumn\@restonecoltrue\onecolumn
3344 \else\@restonecolfalse\fi
3345 \chapter*\listtablename}%
3346 \@mkboth{\listtablename}{}%
3347 %</book|report>
3348 %<!*book&!report>
3349 \section*\listtablename}%
3350 \@mkboth{\listtablename}{\listtablename}%
3351 %</!*book&!report>
3352 \@starttoc{lot}%
3353 %<book|report> \if@restonecol\twocolumn\fi
3354 }
```

`\l@table` 表目次は図目次と同じです。

```
3355 \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。

```
3356 \newdimen\bibindent
3357 \setlength\bibindent{2\jsZw}
```

`thebibliography (env.)` 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。

```
3358 \newenvironment{thebibliography}[1]{%
3359 \@jsc@warnoldfontcmdexceptiontrue
3360 \global\let\presectionname\relax
3361 \global\let\postsectionname\relax
3362 %<article|slide> \section*\refname\@mkboth{\refname}{\refname}%
3363 %<*kiyou>
3364 \vspace{1.5\baselineskip}
3365 \subsubsection*\refname\@mkboth{\refname}{\refname}%
3366 \vspace{0.5\baselineskip}
3367 %</kiyou>
3368 %<book|report> \chapter*\bibname\@mkboth{\bibname}{}%
3369 %<book|report> \addcontentsline{toc}{chapter}{\bibname}%
```

```

3370 \list{\@biblabel{\@arabic\c@enumiv}}%
3371     {\settowidth\labelwidth{\@biblabel{#1}}%
3372     \leftmargin\labelwidth
3373     \advance\leftmargin\labelsep
3374     \@openbib@code
3375     \usecounter{enumiv}%
3376     \let\p@enumiv\@empty
3377     \renewcommand\theenumiv{\@arabic\c@enumiv}}%
3378 %<kiyou> \small
3379 \sloppy
3380 \clubpenalty4000
3381 \@clubpenalty\clubpenalty
3382 \widowpenalty4000%
3383 \sfcode`. \m}
3384 {\def\@noitemerr
3385   {\@latex@warning{Empty `thebibliography' environment}}%
3386 \endlist}

```

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

```
3387 \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。

```
3388 \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。

```
3389 % \def\@biblabel#1{\jsInhibitGlue [#1] \jsInhibitGlue}
```

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

```

3390 % \def\@citex[#1]#2{\leavevmode
3391 %   \let\@citea\@empty
3392 %   \@cite{\@for\@citeb:=#2\do
3393 %     {\@citea\def\@citea{ \inhibitglue\penalty\@m }%
3394 %     \edef\@citeb{\expandafter\@firstofone\@citeb\@empty}}%
3395 %     \if@filesw\immediate\write\@auxout{\string\citation{\@citeb}}\fi
3396 %     \ifundefined{b@\@citeb}{\mbox{\normalfont\bfseries ?}}%
3397 %     \G@refundefinedtrue
3398 %     \@latex@warning
3399 %       {Citation `@\citeb' on page \thepage \space undefined}}%
3400 %     {\@citeofmt{\csname b@\@citeb\endcsname}}}{#1}}
3401 % \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。

```
3402 % \DeclareRobustCommand\cite{\unskip
3403 %   \@ifnextchar [{\@tempswatrue\@citex}{\@tempswafalse\@citex []}]
3404 % \def\@cite#1#2{${\hbox{\scriptsize{#1}\if@tempswa
3405 %   , \jsInhibitGlue\ #2\fi}) }}$}
```

10.3 索引

`theindex (env.)` 2~3 段組の索引を作成します。最後が偶数ページのとときにマージンがずれる現象を直しました (Thanks: 藤村さん)。

```
3406 \newenvironment{theindex}{% 索引を 3 段組で出力する環境
3407   \if@twocolumn
3408     \onecolumn\@restonecolfalse
3409   \else
3410     \clearpage\@restonecoltrue
3411   \fi
3412   \columnseprule.4pt \columnsep 2\jsZw
3413   \ifx\multicols\undefined
3414 %<book|report>   \twocolumn[\@makeschapterhead{\indexname}]%
3415 %<book|report>   \addcontentsline{toc}{chapter}{\indexname}]%
3416 %<!book&!report> \def\presectionname{}\def\postsectionname{}%
3417 %<!book&!report> \twocolumn[\section*{\indexname}]%
3418   \else
3419     \ifdim\textwidth<\fullwidth
3420       \setlength{\evensidemargin}{\oddsidemargin}
3421       \setlength{\textwidth}{\fullwidth}
3422       \setlength{\linewidth}{\fullwidth}
3423 %<book|report>   \begin{multicols}{3}[\chapter*{\indexname}]%
3424 %<book|report>   \addcontentsline{toc}{chapter}{\indexname}]%
3425 %<!book&!report> \def\presectionname{}\def\postsectionname{}%
3426 %<!book&!report> \begin{multicols}{3}[\section*{\indexname}]%
3427   \else
3428 %<book|report>   \begin{multicols}{2}[\chapter*{\indexname}]%
3429 %<book|report>   \addcontentsline{toc}{chapter}{\indexname}]%
3430 %<!book&!report> \def\presectionname{}\def\postsectionname{}%
3431 %<!book&!report> \begin{multicols}{2}[\section*{\indexname}]%
3432   \fi
3433   \fi
3434 %<book|report>   \@mkboth{\indexname}{}%
3435 %<!book&!report> \mkboth{\indexname}{\indexname}%
3436   \plainifnotempty % \thispagestyle{plain}
3437   \parindent\z@
3438   \parskip\z@ \@plus .3\jsc@mpt\relax
3439   \let\item\@idxitem
3440   \raggedright
3441   \footnotesize\narrowbaselines
3442 }{
```

```

3443 \ifx\multicols\undefined
3444 \if@restonecol\onecolumn\fi
3445 \else
3446 \end{multicols}
3447 \fi
3448 \clearpage
3449 }

```

`\@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 3450 \newcommand{\@idxitem}{\par\hangindent 4\jsZw} % 元 40pt
\subsubitem 3451 \newcommand{\subitem}{\@idxitem \hspace*{2\jsZw}} % 元 20pt
3452 \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。

```
3453 \newcommand{\indexspace}{\par \vskip 10\jsc@empt \@plus5\jsc@empt \@minus3\jsc@empt\relax}
```

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

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

```

3454 \newcommand\seename{\if@english see\else →\fi}
3455 \newcommand\alsoname{\if@english see also\else →\fi}

```

10.4 脚注

`\footnote` 和文の句読点・閉じかっこ類の直後で用いた際に余分なアキが入るのを防ぐため、`\footnotemark` `\inhibitglue` を入れることにします。pL^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。

```

3456 \ifx\pltx@foot@penalty\undefined
3457 \let\footnotes@ve=\footnote
3458 \def\footnote{\jsInhibitGlue\footnotes@ve}
3459 \let\footnotemarks@ve=\footnotemark
3460 \def\footnotemark{\jsInhibitGlue\footnotemarks@ve}
3461 \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 `\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 pT_EX 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 \LaTeX の変更に追随しました (Thanks: 角藤さん)。p \LaTeX 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 \TeX 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 `newpertext` 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)。

```
3462 \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。

```
3463 % \def\thefootnote{\ifnum\c@footnote>\z@ 注\kern0.1\jsw\@arabic\c@footnote\fi}
```

`\footnoterule` 本文と脚注の間の罫線です。

```
3464 \renewcommand{\footnoterule}{%
3465   \kern-2.6\jsc@mpt \kern-.4\p@
3466   \hrule width .4\columnwidth
3467   \kern 2.6\jsc@mpt}
```

`\c@footnote` 脚注番号は章ごとにリセットされます。

```
3468 %<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 \E X and TUG NEWS*, Vol. 2, No. 4 (1993), p. 9)

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

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

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

[2022-09-13] \LaTeX 2 ϵ 2021-11-15 (lfloat.dtx 2021/10/14 v1.2g) で `\@currentcounter` が追加されましたので、追随します。なお、 \LaTeX 2 ϵ 2021-06-01 (lfloat.dtx 2021/02/10 v1.2e) で `parhook` 対応として `\par` が追加されていますが、実は同時に `\color@endgroup` も `\endgraf` するように変更されていますので、不要だと思います。というわけで追加しません。

```
3469 \long\def\@footnotetext{%
3470   \insert\footins\bgroup
3471   \normalfont\footnotesize
```

```

3472 \interlinepenalty\interfootnotelinepenalty
3473 \splittopskip\footnotesep
3474 \splitmaxdepth \dp\strutbox \floatingpenalty \@MM
3475 \hsize\columnwidth \@parboxrestore
3476 \def\@currentcounter{footnote}%
3477 \protected@edef\@currentlabel{%
3478   \csname p@footnote\endcsname\@thefnmark
3479 }%
3480 \color@begingroup
3481   \@makefnmark{%
3482     \rule{z@\footnotesep\ignorespaces}%
3483     \futurelet\jsc@next\jsc@fo@t}
3484 \def\jsc@fo@t{\ifcat\bgroup\noexpand\jsc@next \let\jsc@next\jsc@fo@t
3485               \else \let\jsc@next\jsc@f@t\fi \jsc@next}
3486 \def\jsc@f@t{\bgroup\aftergroup\jsc@@foot\let\jsc@next}
3487 \def\jsc@f@t#1{#1\jsc@@foot}
3488 \def\jsc@@foot{\@finalstrut\strutbox\color@endgroup\egroup
3489 \ifx\pltx@foot@penalty\@undefined\else
3490   \ifhmode\null\fi
3491   \ifnum\pltx@foot@penalty=z@\else
3492     \penalty\pltx@foot@penalty
3493     \pltx@foot@penaltyz@
3494   \fi
3495 \fi}

```

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

```

3496 \newcommand\@makefnmark[#1]{%
3497   \advance\leftskip 3\jsZw
3498   \parindent 1\jsZw
3499   \noindent
3500   \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。

```

3501 % \def\@xfootnotenext[#1]{%
3502 %   \begingroup
3503 %     \ifnum#1>z@
3504 %       \csname c@\mpfn\endcsname #1\relax
3505 %       \unrestored@protected@xdef\@thefnmark{\thempfn}%
3506 %     \else
3507 %       \unrestored@protected@xdef\@thefnmark{}%
3508 %     \fi
3509 %   \endgroup

```

```
3510 % \@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。

```
3511 \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。

```
3512 \def\everyparhook{\jsInhibitGlueAtParTop}
3513 \ifnum\bxjs@everyparhook=\bxjs@everyparhook@@compat
3514 \g@addto@macro\bxjs@begin@document@hook{\everypar{\everyparhook}}
3515 \fi
```

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

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

[2017-02-13] `\jsc@tempa` は実はテンポラリーではなく「この処理専用のユニーク制御綴」である必要があります。間違っって別の箇所です使う危険性が高いので、専用の命令 `\jsc@ig@temp` に置き換えました (Issue #54)。

次の `\@inhibitglue` は JS クラスでの `\jsInhibitGlueAtParTop` の実装である。エンジンが (u)platex の場合はこれを採用する。

```
3516 \ifx j\jsEngine
3517 \def\@inhibitglue{%
3518 \futurelet\@let@token\@inhibitglue}
3519 \begingroup
3520 \let\GDEF=\gdef
3521 \let\CATCODE=\catcode
3522 \let\ENDGROUP=\endgroup
3523 \CATCODE`k=12
3524 \CATCODE`a=12
3525 \CATCODE`n=12
3526 \CATCODE`j=12
```

```

3527 \CATCODE`i=12
3528 \CATCODE`c=12
3529 \CATCODE`h=12
3530 \CATCODE`r=12
3531 \CATCODE`t=12
3532 \CATCODE`e=12
3533 \GDEF\KANJI@CHARACTER{kanji character }
3534 \ENDGROUP
3535 \def\@inhibitglue{%
3536 \expandafter\expandafter\expandafter\jsc@inhibitglue\expandafter\meaning\expandafter\@let@
3537 \expandafter\def\expandafter\jsc@inhibitglue\expandafter#\expandafter1\KANJI@CHARACTER#2#3\j
3538 \def\jsc@ig@temp{#1}%
3539 \ifx\jsc@ig@temp\empty
3540 \ifnum\the\inhibitxspcode`#2=2\relax
3541 \inhibitglue
3542 \fi
3543 \fi}
3544 \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。

```

3545 \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。

```

3546 \def\@doendpe{%
3547 \@endpetrue
3548 \def\par{%
3549 \restorepar\clubpenalty\@clubpenalty\everypar{\everyparhook}\par\@endpefalse}%
3550 \everypar{\setbox\z@\lastbox}\everypar{\everyparhook}\@endpefalse\everyparhook}}

```

[2017-08-31] `minipage` 環境にも対策します。

```

3551 \def\@setminipage{%
3552 \@minipagetrue
3553 \everypar{\@minipagefalse\everypar{\everyparhook}}%
3554 }

```

`\item` 命令の直後です。

```

3555 \def\@item[#1]{%
3556 \if@noperitem
3557 \@donoperitem
3558 \else
3559 \if@inlabel
3560 \indent \par

```

```

3561 \fi
3562 \ifhmode
3563 \unskip\unskip \par
3564 \fi
3565 \if@newlist
3566 \if@nobreak
3567 \@nbitem
3568 \else
3569 \addpenalty\@beginparpenalty
3570 \addvspace\@topsep
3571 \addvspace{-\parskip}%
3572 \fi
3573 \else
3574 \addpenalty\@itempenalty
3575 \addvspace\itemsep
3576 \fi
3577 \global\@inlabeltrue
3578 \fi
3579 \everypar{%
3580 \@minipagefalse
3581 \global\@newlistfalse
3582 \if@inlabel
3583 \global\@inlabelfalse
3584 {\setbox\z@\lastbox
3585 \ifvoid\z@
3586 \kern-\itemindent
3587 \fi}%
3588 \box\@labels
3589 \penalty\z@
3590 \fi
3591 \if@nobreak
3592 \@nobreakfalse
3593 \clubpenalty \@M
3594 \else
3595 \clubpenalty \@clubpenalty
3596 \everypar{\everyparhook}%
3597 \fi\everyparhook}%
3598 \if@noitemarg
3599 \@noitemargfalse
3600 \if@nmbrrlist
3601 \refstepcounter\@listctr
3602 \fi
3603 \fi
3604 \sbox\@tempboxa{\makelabel{#1}}%
3605 \global\setbox\@labels\hbox{%
3606 \unhbox\@labels
3607 \hskip \itemindent
3608 \hskip -\labelwidth
3609 \hskip -\labelsep

```

```

3610 \ifdim \wd\@tempboxa >\labelwidth
3611 \box\@tempboxa
3612 \else
3613 \hbox to\labelwidth {\unhbox\@tempboxa}%
3614 \fi
3615 \hskip \labelsep}%
3616 \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。

```

3617 \def\@afterheading{%
3618 \@nbreaktrue
3619 \everypar{%
3620 \if@nbreak
3621 \@nbreakfalse
3622 \clubpenalty \@M
3623 \if@afterindent \else
3624 {\setbox\z@\lastbox}%
3625 \fi
3626 \else
3627 \clubpenalty \@clubpenalty
3628 \everypar{\everyparhook}%
3629 \fi\everyparhook}}

```

「`\everyparhook` 用のパッチ処理」はここまで。

```
3630 \fi
```

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

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

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

```

3631 \ifnum\bxjs@everyparhook=\bxjs@everyparhook@@none\else
3632 \AtEndOfClass{%
3633 \def\@gnewline #1{%
3634 \ifvmode
3635 \@nolnerr
3636 \else
3637 \unskip \reserved@e {\reserved@f#1}\nbreak \hfil \break \null
3638 \jsInhibitGlue \ignorespaces
3639 \fi}

```

```
3640 }
3641 \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。

※`\小`、`\上小` の制御綴は定義しない。

```
3642 \if@jslogo
3643   \IfFileExists{jslogo.sty}{%
3644     \RequirePackage{jslogo}%
3645   }{%
3646     \ClassWarningNoLine\bxjs@clsname
3647     {The package 'jslogo' is not installed.\MessageBreak
3648     It is included in the recent release of\MessageBreak
3649     the 'jsclasses' bundle}
3650   }
3651 \fi
```

13 amsmath との衝突の回避

`\ltx@ifnextchar` `amsmath` パッケージでは行列中で `\@ifnextchar` を再定義していますが、これが L^AT_EX の `\ProvidesFile` `\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: しっぽ愛好家さん)。

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

```
3652 %\let\ltx@ifnextchar\@ifnextchar
3653 %\def\ProvidesFile#1{%
3654 %   \begingroup
3655 %     \catcode`\ 10 %
3656 %     \ifnum \endlinechar<256 %
3657 %       \ifnum \endlinechar>\m@ne
3658 %         \catcode\endlinechar 10 %
3659 %     \fi
```

```

3660 % \fi
3661 % \@makeother\/%
3662 % \@makeother\&%
3663 % \ltx@ifnextchar[{\@providesfile{#1}}{\@providesfile{#1}[]}]

```

14 初期設定

■いろいろな語

```

\prepartname
\postpartname 3664 \newcommand{\prepartname}{\if@english Part~\else 第\fi}
\prechaptername 3665 \newcommand{\postpartname}{\if@english\else 部\fi}
3666 %<book|report>\newcommand{\prechaptername}{\if@english Chapter~\else 第\fi}
\postchaptername 3667 %<book|report>\newcommand{\postchaptername}{\if@english\else 章\fi}
\presectionname 3668 \newcommand{\presectionname}{}% 第
\postsectionname 3669 \newcommand{\postsectionname}{}% 節

\contentsname
\listfigurename 3670 \newcommand{\contentsname}{\if@english Contents\else 目次\fi}
\listtablename 3671 \newcommand{\listfigurename}{\if@english List of Figures\else 図目次\fi}
3672 \newcommand{\listtablename}{\if@english List of Tables\else 表目次\fi}

\refname
\bibName 3673 \newcommand{\refname}{\if@english References\else 参考文献\fi}
\indexname 3674 \newcommand{\bibname}{\if@english Bibliography\else 参考文献\fi}
3675 \newcommand{\indexname}{\if@english Index\else 索引\fi}

\figurename
\tablename 3676 %<!jspf>\newcommand{\figurename}{\if@english Fig.~\else 図\fi}
3677 %<jspf>\newcommand{\figurename}{Fig.~}
3678 %<!jspf>\newcommand{\tablename}{\if@english Table~\else 表\fi}
3679 %<jspf>\newcommand{\tablename}{Table~}

\appendixname
\abstractname 3680 % \newcommand{\appendixname}{\if@english Appendix~\else 付録\fi}
3681 \newcommand{\appendixname}{\if@english \else 付録\fi}
3682 %<!book>\newcommand{\abstractname}{\if@english Abstract\else 概要\fi}

```

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

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

※ 2.0 版より、完全に `bxwareki` に任せる。

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

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

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

T_EX エンジンの種類により処理を分ける。

```
3683 \@onlypreamble\bxjs@decl@Seireki@cmds
3684 \@tempswafalse
3685 \if p@jsEngine \@tempwattrue \fi
3686 \if n@jsEngine \@tempwattrue \fi
3687 \bxjs@cond\if@tempswa\fi{%
```

8 ビット欧文 T_EX の場合。

\ifjsSeireki [スイッチ] 西暦 スイッチ (\if 西暦) の代わりに用いる。

```
3688 \newif\ifjsSeireki \jsSeirekitrue
```

\bxjs@decl@Seireki@cmds 本クラス用の \西暦/\和暦 の命令を定義するためのマクロ。

※\def\西暦 は実際には \^^e8 の定義文であることに注意。

```
3689 \def\bxjs@decl@Seireki@cmds{%
3690   \def\西暦{\jsSeirekitrue}%
3691   \def\和暦{\jsSeirekifalse\bxjs@wareki@used}}
```

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

```
\Wareki 3692 \def\Seireki{\jsSeirekitrue}
3693 \def\Wareki{\jsSeirekifalse\bxjs@wareki@used}

3694 \def\bxjs@if@use@seireki{\bxjs@cond\ifjsSeireki\fi}
3695 \def\bxjs@iai{\noexpand-}
3696 }{%
```

8 ビット欧文 T_EX ではない場合。ここでは JS クラスと合わせるため 西暦 スイッチを使う。

```
3697 \newif\if 西暦 \西暦 true
3698 \def\bxjs@decl@Seireki@cmds{%
3699   \def\西暦{\西暦 true}%
3700   \def\和暦{\西暦 false\bxjs@wareki@used}}
3701 \def\Seireki{\西暦 true}
3702 \def\Wareki{\西暦 false\bxjs@wareki@used}
3703 \def\bxjs@if@use@seireki{\bxjs@cond\if 西暦\fi}
3704 \let\bxjs@iai\@empty
3705 }
3706 \bxjs@decl@Seireki@cmds
```

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

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

```
3707 \newif\ifbxjs@bxwareki@avail
3708 \IfFileExists{bxwareki.sty}{%
3709   \if \if n\jsEngine \ifjsWitheTeX T\else F\fi\else T\fi T%
3710   \RequirePackage{bxwareki}[2018/04/08]%v0.2
3711   \bxjs@bxwareki@availtrue
3712 \fi}{}
```

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

```
3713 \ifbxjs@bxwareki@avail \let\bxjs@wareki@used\empty
3714 \else
3715   \bxjs@robust@def\bxjs@wareki@used{%
3716     \global\let\bxjs@wareki@used\empty
3717     \ClassWarning\bxjs@clsname
3718     {Wareki mode is not supported, since\MessageBreak
3719     'bxwareki' is unavailable, reported}}
3720   \g@addto@macro\bxjs@begin@document@hook{%
3721     \let\bxjs@wareki@used\empty}
3722 \fi
```

`\jayeare` 和暦における年の表記の「年」以前の部分（元号+年数）。

※ `\heisei` の代替となる機能（だから常に和暦を扱う）。

`\heisei` 年数を表す整数レジスタで、元号が「平成」である場合にのみ定義される。

※ JS クラスと互換の機能。

```
3723 \ifbxjs@bxwareki@avail
3724   \let\jayeare\warekiyear
3725   \def\bxjs@tmpa{H}\ifx\bxjs@tmpa\warekigengoinitial
3726     \newcount\heisei \heisei=\value{warekiyear}
3727 \fi
```

ただし `bxwareki` が使えない場合は西暦表示にフォールバックする。

```
3728 \else
3729   \edef\jayeare{\the\year \bxjs@iaai}
3730 \fi
```

`\today` 英語、西暦、和暦で場合分けをする。

※ `diff` の都合のためまた `jsclasses` のコードを挿入する。

```
3731 %<*jsclasses>
3732 \newif\if 西暦 \西暦 true
3733 \def\西暦{\西暦 true}
3734 \def\和暦{\西暦 false}
3735 \newcount\heisei \heisei\year \advance\heisei-1988\relax
3736 \def\pltx@today@year@#1{%
3737   \ifnum\numexpr\year-#1=1 元\else
3738     \ifnum1=\iftdir\ifmdir0\else1\fi\else0\fi
```

```

3739     \kansuji\numexpr\year-#1\relax
3740   \else
3741     \number\numexpr\year-#1\relax\nobreak
3742   \fi
3743 \fi 年
3744 }
3745 \def\pltx@today@year{%
3746   \ifnum\numexpr\year*10000+\month*100+\day<19890108
3747     昭和\pltx@today@year@{1925}%
3748   \else\ifnum\numexpr\year*10000+\month*100+\day<20190501
3749     平成\pltx@today@year@{1988}%
3750   \else
3751     令和\pltx@today@year@{2018}%
3752   \fi\fi}
3753 %</jsclasses>
3754 \begingroup
3755 \let\bxjs@next\relax
3756 \ifbxjs@bxwareki@avail \ifx\warekigengo\@empty\else
3757   \def\bxjs@next{\warekitoday}
3758   \bxjs@test@engine\unexpanded{%
3759     \def\bxjs@next{\unexpanded\expandafter{\warekitoday}}}}
3760 \fi\fi
3761 \def!#1#2#3{\noexpand#1\noexpand#2\noexpand#3}
3762 \ifx\bxjs@iaai\@empty \let!\@empty \fi
3763 \xdef\bxjs@today{%
3764   \if@english
3765     \ifcase\month\or
3766       January\or February\or March\or April\or May\or June\or
3767       July\or August\or September\or October\or November\or December\fi
3768     \space\number\day, \number\year
3769   \else
3770     \ifx\bxjs@next\relax \expandafter\@firstoftwo
3771     \else \noexpand\bxjs@if@use@seireki
3772     \fi {%
3773       \number\year\bxjs@iaai!年%
3774       \bxjs@iaai\number\month\bxjs@iaai!月%
3775       \bxjs@iaai\number\day\bxjs@iaai!日%
3776     }{\bxjs@next}%
3777   \fi}
3778 \endgroup
3779 \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。

```

3780 \g@addto@macro\bxjs@begin@document@hook{%
3781   \ifx\bb1@jpn@maybekansuji\@undefined\else
3782     \bxjs@decl@Seireki@cmds

```

```

3783 \g@addto@macro\datejapanese{%
3784 \let\today\bxjs@today}%
3785 \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）

```

3786 \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。

```

3787 %<slide>\pagestyle{empty}%
3788 %<article|report>\pagestyle{plain}%
3789 %<book>\pagestyle{headings}%
3790 \pagenumbering{arabic}
3791 \if@twocolumn
3792 \twocolumn
3793 \sloppy
3794 \flushbottom
3795 \else
3796 \onecolumn
3797 \raggedbottom
3798 \fi
3799 %<*/slide>
3800 \renewcommand\familydefault{\sfdefault}
3801 \raggedright
3802 %</slide>

```

■BXJS 独自の追加処理  フックを実行する。

```

3803 \bxjs@pre@jadriver@hook

```

和文ドライバのファイルを読み込む。

```

3804 \input{bxjsja-\bxjs@jadriver.def}

```

15 実験的コード

この節は JS クラスの話で、BXJS クラスには当てはまらない。

[2016-11-29] コミュニティ版 $\text{p}\text{L}\text{A}\text{T}\text{E}\text{X}$ で新設されたテスト用パッケージ（`expp12e` パッケージ）が文書クラスより先に読み込まれていた場合は、`jsclasses` もテスト版として動作します。この処置は `jsarticle`, `jsbook`, `jsreport` にのみ行い、`jspf` と `kiyou` は除外しておきます。`expp12e` パッケージが読みこまれていない場合は通常版として動作しますので、ここで終了します。

```

3805 %</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` `XqTeX`
 - `j` `pTeX` または `upTeX`
 - `n` 以上の何れでもない
- `\ifjsWithupTeX` [スイッチ] 使用されているエンジンが `upTeX` であるか。
- `\ifjsWitheTeX` [スイッチ] 使用されているエンジンが `ε-TeX` 拡張であるか。
- `\ifjsInPdfMode` [スイッチ] 使用されているエンジンが (`pdfTeX`・`LuaTeX` の) PDF モードであるか。
- `\jsUnusualPtSize` [整数定数を表す文字列のマクロ] 基底フォントサイズが `10pt`、`11pt`、`12pt` のいずれでもない場合の `\@ptsize` の値。 (`\@ptsize` 自体があまり有用でないと思われる。)
- `\jsScale` [実数を表す文字列のマクロ] 和文フォントサイズの要求サイズに対するスケール。クラスオプション `scale` で指定される。(既定値は `0.924715`。)
- `\jsJaFont` [マクロ] 和文フォント設定を表す文字列。クラスオプション `jafont` で指定された値。
- `\jsJaParam` [マクロ] 和文モジュールに渡すパラメタを表す文字列。この値が何を表すかは決まっておらず、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」はこの設定と辻褃が合うように和文フォントサイズを調整する必要がある。ほとんどの場合、和文フォントを NFSS で規定する際に `\jsScale` の値をスケール値として与えれば上手くいく。

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

以下で標準で用意されている和文ドライバの実装を示す。

```
3806 %<*drv>
```

付録 B 和文ドライバ：minimal

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

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

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

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

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

B.1 準備

```
3807 %<*minimal>
```

```
3808 %% このファイルは日本語文字を含みます
```

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

ただし「公式にはサポート外」のエンジンが使われている場合は強制終了させる。

※ NTT jI_εX と Omega 系。

```
3809 \let\bxjs@tmpa\relax
```

```
3810 \ifx J\jsEngine \def\bxjs@tmpa{NTT-jTeX}\fi
```

```
3811 \ifx O\jsEngine \def\bxjs@tmpa{Omega}\fi
```

```
3812 \ifx\bxjs@tmpa\relax \expandafter\@gobble
```

```
3813 \else
```

```
3814 \ClassError\bxjs@clsname
```

```
3815 {The engine in use (\bxjs@tmpa) is not supported}
```

```

3816 {It's a fatal error. I'll quit right now.}
3817 \expandafter\@firstofone
3818 \fi{\endinput\@@end}

```

■補助マクロ

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

```

3819 \def\DeclareJaTextFontCommand#1#2{%
3820   \DeclareRobustCommand#1[1]{%
3821     \relax
3822     \ifmmode \expandafter\nfss@text \fi
3823     {#2##1}}%
3824 }

```

`\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。

```

3825 \def\DeclareJaMathFontCommand#1#2{%
3826   \DeclareRobustCommand#1[1]{%
3827     \relax
3828     \ifmmode\else \non@alpherr{#1\space}\fi
3829     \nfss@text{\fontfamily\familydefault
3830               \fontseries{m}\fontshape{n}\selectfont\relax
3831               #2##1}%
3832   }%
3833 }

```

`\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。

```

3834 \long\def\bxjs@@CSsfdefault{\sfdefault}%
3835 \@onlypreamble\bxjs@if@sf@default
3836 \def\bxjs@if@sf@default#1{%
3837   \ifx\familydefault\bxjs@@CSsfdefault#1\fi
3838   \g@addto@macro\bxjs@begin@document@hook{%
3839     \ifx\familydefault\bxjs@@CSsfdefault#1\fi}%
3840 }

```

`\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。

```

3841 \@tempdima\p@ \bxjs@invscale\@tempdima\jsScale
3842 \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。


```

3843 \def\jsLetHeadChar#1#2{%
3844   \begingroup
3845     \escapechar=`\ %
3846     \let\bxjs@tmpa={% brace-match-hack
3847       \bxjs@let@hchar@exp#2}%
3848   \endgroup
3849   \let#1\bxjs@g@tmpa}
3850 \def\bxjs@let@hchar@exp{%
3851   \futurelet\@let@token\bxjs@let@hchar@exp@a}
3852 \def\bxjs@let@hchar@exp@a{%
3853   \bxjs@cond@ifcat\noexpand\@let@token\bgroup\fi{% 波括弧
3854     \bxjs@let@hchar@out\let\relax
3855   }\bxjs@cond@ifcat\noexpand\@let@token\@sptoken\fi{% 空白
3856     \bxjs@let@hchar@out\let\space%
3857   }\bxjs@cond@if\noexpand\@let@token\@backslashchar\fi{% バックスラッシュ
3858     \bxjs@let@hchar@out\let\@backslashchar
3859   }\bxjs@let@hchar@exp@b}}}}
3860 \def\bxjs@let@hchar@exp@b#1{%
3861   \expandafter\bxjs@let@hchar@exp@c\string#1?\@nil#1}
3862 \def\bxjs@let@hchar@exp@c#1#2\@nil{%
3863   %\message{<#1#2>}%
3864   \bxjs@cond@if#1\@backslashchar\fi{% 制御綴
3865     \bxjs@cond\expandafter\ifx\noexpand\@let@token\@let@token\fi{%
3866       \bxjs@let@hchar@out\let\relax
3867     }{%else
3868       \expandafter\bxjs@let@hchar@exp
3869     }%
3870   }{%else
3871     \bxjs@let@hchar@chr#1%
3872   }}
3873 \def\bxjs@let@hchar@chr#1{%
3874   \bxjs@let@hchar@out\def{{#1}}}
3875 \def\bxjs@let@hchar@out#1#2{%
3876   \global#1\bxjs@g@tmpa#2\relax
3877   \toks@\bgroup}% skip to right brace

```

UTF-8 のバイト列を扱うコード。

```

3878 \chardef\bxjs@let@hchar@csta=128
3879 \chardef\bxjs@let@hchar@cstb=192
3880 \chardef\bxjs@let@hchar@cstc=224
3881 \chardef\bxjs@let@hchar@cstd=240
3882 \chardef\bxjs@let@hchar@cste=248
3883 \let\bxjs@let@hchar@chr@ue@a\bxjs@let@hchar@chr
3884 \def\bxjs@let@hchar@chr@ue#1{%
3885   \@tempcnta=`#1\relax
3886   %\message{\the\@tempcnta}%
3887   \bxjs@cond\ifnum\@tempcnta<\bxjs@let@hchar@csta\fi{%
3888     \bxjs@let@hchar@chr@ue@a#1%
3889   }\bxjs@cond\ifnum\@tempcnta<\bxjs@let@hchar@cstb\fi{%

```

```

3890 \bxjs@let@hchar@out\let\relax
3891 }{\bxjs@cond@ifnum\@tempcnta<\bxjs@let@hchar@cstc\fi{%
3892 \bxjs@let@hchar@chr@ue@b
3893 }{\bxjs@cond@ifnum\@tempcnta<\bxjs@let@hchar@cstd\fi{%
3894 \bxjs@let@hchar@chr@ue@c
3895 }{\bxjs@cond@ifnum\@tempcnta<\bxjs@let@hchar@cste\fi{%
3896 \bxjs@let@hchar@chr@ue@d
3897 }{%else
3898 \bxjs@let@hchar@out\let\relax
3899 }}}}
3900 \def\bxjs@let@hchar@chr@ue@a#1{%
3901 \bxjs@let@hchar@out\def{{#1}}}
3902 \def\bxjs@let@hchar@chr@ue@b#1#2{%
3903 \bxjs@let@hchar@out\def{{#1#2}}}
3904 \def\bxjs@let@hchar@chr@ue@c#1#2#3{%
3905 \bxjs@let@hchar@out\def{{#1#2#3}}}
3906 \def\bxjs@let@hchar@chr@ue@d#1#2#3#4{%
3907 \bxjs@let@hchar@out\def{{#1#2#3#4}}}

```

B.2 (u)pTeX 用の設定

```
3908 \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。

```

3909 \def\bxjs@let@hchar@chr@pp#1#2{%
3910 \expandafter\bxjs@let@hchar@chr@pp@a\meaning#2\relax#1#2}
3911 \def\bxjs@let@hchar@chr@pp@a#1#2\relax#3#4{%
3912 %\message{(\meaning#3:\meaning#4)}%
3913 \bxjs@cond@if#1k\fi{%
3914 \bxjs@let@hchar@out\def{{#4}}%
3915 }{%else
3916 \bxjs@let@hchar@chr@ue#3#4%
3917 }}
3918 \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。

```

3919 \edef\jsc@JYn{\ifjsWithupTeX JY2\else JY1\fi}
3920 \edef\jsc@JTn{\ifjsWithupTeX JT2\else JT1\fi}
3921 \edef\jsc@pfx0{\ifjsWithupTeX u\fi}

```

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

めるため一旦マクロにしておく。`\bxjs@sizereference` は全角幅を測定する時に参照するフォント。

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

```
3922 \@onlypreamble\bxjs@declarefontshape
3923 \ifjsWithupTeX
3924 \def\bxjs@declarefontshape{%
3925 \DeclareFontShape{JY2}{mc}{m}{n}{<->s*[\bxjs@scale]upjpnrm-h}{}%
3926 \DeclareFontShape{JY2}{gt}{m}{n}{<->s*[\bxjs@scale]upjpngt-h}{}%
3927 \DeclareFontShape{JT2}{mc}{m}{n}{<->s*[\bxjs@scale]upjpnrm-v}{}%
3928 \DeclareFontShape{JT2}{gt}{m}{n}{<->s*[\bxjs@scale]upjpngt-v}{}%
3929 }
3930 \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。

```
3931 \else
3932 \def\bxjs@declarefontshape{%
3933 \DeclareFontShape{JY1}{mc}{m}{n}{<->s*[\bxjs@scale]jis}{}%
3934 \DeclareFontShape{JY1}{gt}{m}{n}{<->s*[\bxjs@scale]jisg}{}%
3935 \DeclareFontShape{JT1}{mc}{m}{n}{<->s*[\bxjs@scale]tmin10}{}%
3936 \DeclareFontShape{JT1}{gt}{m}{n}{<->s*[\bxjs@scale]tgoth10}{}%
3937 }
3938 \def\bxjs@sizereference{jis}
3939 \fi
```

既で使用されている標準和文フォント定義がもしあれば取り消す。

```
3940 \def\bxjs@next#1/#2/#3/#4/#5\relax{%
3941 \def\bxjs@tmpb{#5}}
3942 \ifjsWithpTeXng \def\bxjs@tmpb{10}%
3943 \else
3944 \expandafter\expandafter\expandafter\bxjs@next
3945 \expandafter\string\the\jfont\relax
3946 \fi
3947 \@for\bxjs@tmpa:={\jsc@JYn/mc/m/n,\jsc@JYn/gt/m/n,%
3948 \jsc@JTn/mc/m/n,\jsc@JTn/gt/m/n}\do
3949 {\expandafter\let\csname\bxjs@tmpa/10\endcsname=\@undefined
3950 \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。

```
3951 \begingroup
3952 % 参照用フォント (\bxjs@sizereference) の全角空白の幅を取得
3953 \font\bxjs@tmpa=\bxjs@sizereference\space at 10pt
```

```

3954 \setbox\z@\hbox{\bxjs@tmpa\char\jis"2121\relax}
3955 % 幅が丁度 10pt なら補正は不要
3956 \ifdim\wd\z@=10pt
3957   \global\let\bxjs@scale\jsScale
3958 \else
3959 % (10*s)/(10*f) として計算、\bxjs@invscale は BXJS で定義
3960   \edef\bxjs@tmpa{\strip@pt\wd\z@}
3961   \@tempdima=10pt \@tempdima=\jsScale\@tempdima
3962   \bxjs@invscale\@tempdima\bxjs@tmpa
3963   \xdef\bxjs@scale{\strip@pt\@tempdima}
3964 \fi
3965 \endgroup
3966 %\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。

```
3967 \bxjs@declarefontshape
```

フォント代替の明示的定義。

```

3968 \DeclareFontShape{\jsc@JYn}{mc}{m}{it}{<->ssub*mc/m/n}{}
3969 \DeclareFontShape{\jsc@JYn}{mc}{m}{sl}{<->ssub*mc/m/n}{}
3970 \DeclareFontShape{\jsc@JYn}{mc}{m}{sc}{<->ssub*mc/m/n}{}
3971 \DeclareFontShape{\jsc@JYn}{gt}{m}{it}{<->ssub*gt/m/n}{}
3972 \DeclareFontShape{\jsc@JYn}{gt}{m}{sl}{<->ssub*gt/m/n}{}
3973 \DeclareFontShape{\jsc@JYn}{mc}{bx}{it}{<->ssub*gt/m/n}{}
3974 \DeclareFontShape{\jsc@JYn}{mc}{bx}{sl}{<->ssub*gt/m/n}{}
3975 \DeclareFontShape{\jsc@JYn}{gt}{bx}{it}{<->ssub*gt/m/n}{}
3976 \DeclareFontShape{\jsc@JYn}{gt}{bx}{sl}{<->ssub*gt/m/n}{}
3977 \DeclareFontShape{\jsc@JYn}{mc}{b}{n}{<->ssub*mc/bx/n}{}
3978 \DeclareFontShape{\jsc@JYn}{mc}{b}{it}{<->ssub*mc/bx/n}{}
3979 \DeclareFontShape{\jsc@JYn}{mc}{b}{sl}{<->ssub*mc/bx/n}{}
3980 \DeclareFontShape{\jsc@JYn}{gt}{b}{n}{<->ssub*gt/bx/n}{}
3981 \DeclareFontShape{\jsc@JYn}{gt}{b}{it}{<->ssub*gt/bx/n}{}
3982 \DeclareFontShape{\jsc@JYn}{gt}{b}{sl}{<->ssub*gt/bx/n}{}
3983 \DeclareFontShape{\jsc@JTn}{mc}{m}{it}{<->ssub*mc/m/n}{}
3984 \DeclareFontShape{\jsc@JTn}{mc}{m}{sl}{<->ssub*mc/m/n}{}
3985 \DeclareFontShape{\jsc@JTn}{mc}{m}{sc}{<->ssub*mc/m/n}{}
3986 \DeclareFontShape{\jsc@JTn}{gt}{m}{it}{<->ssub*gt/m/n}{}
3987 \DeclareFontShape{\jsc@JTn}{gt}{m}{sl}{<->ssub*gt/m/n}{}
3988 \DeclareFontShape{\jsc@JTn}{mc}{bx}{it}{<->ssub*gt/m/n}{}
3989 \DeclareFontShape{\jsc@JTn}{mc}{bx}{sl}{<->ssub*gt/m/n}{}
3990 \DeclareFontShape{\jsc@JTn}{gt}{bx}{it}{<->ssub*gt/m/n}{}
3991 \DeclareFontShape{\jsc@JTn}{gt}{bx}{sl}{<->ssub*gt/m/n}{}
3992 \DeclareFontShape{\jsc@JTn}{mc}{b}{n}{<->ssub*mc/bx/n}{}
3993 \DeclareFontShape{\jsc@JTn}{mc}{b}{it}{<->ssub*mc/bx/n}{}
3994 \DeclareFontShape{\jsc@JTn}{mc}{b}{sl}{<->ssub*mc/bx/n}{}
3995 \DeclareFontShape{\jsc@JTn}{gt}{b}{n}{<->ssub*gt/bx/n}{}
3996 \DeclareFontShape{\jsc@JTn}{gt}{b}{it}{<->ssub*gt/bx/n}{}
3997 \DeclareFontShape{\jsc@JTn}{gt}{b}{sl}{<->ssub*gt/bx/n}{}

```

欧文総称フォント命令で和文フォントが連動するように修正する。その他の和文フォント関係の定義を行う。

※ 2020/02/02 の NFSS の改修に対する jsclasses の対策を取り入れた。

```
3998 \@ifl@t@r\fmtversion{2020/10/01}
3999   {\jsc@needspace@tchfalse}{\jsc@needspace@tchtrue}
4000 \ifjsc@needspace@tch      % --- for 2020-02-02 or older BEGIN
4001 \ifx\@rmfamilyhook\@undefined % old
4002 \DeclareRobustCommand\rmfamily
4003   {\not@math@alphabet\rmfamily\mathrm
4004    \romanfamily\rmdefault\kanjifamily\mcdefault\selectfont}
4005 \DeclareRobustCommand\sffamily
4006   {\not@math@alphabet\sffamily\mathsf
4007    \romanfamily\sfdefault\kanjifamily\gtdefault\selectfont}
4008 \DeclareRobustCommand\ttfamily
4009   {\not@math@alphabet\ttfamily\mathtt
4010    \romanfamily\ttdefault\kanjifamily\gtdefault\selectfont}
4011 \g@addto@macro\bxjs@begin@document@hook{%
4012   \ifx@mweights@init\@undefined\else % mweights.sty is loaded
4013     % my definitions above should have been overwritten, recover it!
4014     % \selectfont is executed twice but I don't care about speed...
4015     \expandafter\g@addto@macro\csname rmfamily \endcsname
4016       {\kanjifamily\mcdefault\selectfont}%
4017     \expandafter\g@addto@macro\csname sffamily \endcsname
4018       {\kanjifamily\gtdefault\selectfont}%
4019     \expandafter\g@addto@macro\csname ttfamily \endcsname
4020       {\kanjifamily\gtdefault\selectfont}%
4021   \fi}
4022 \else % 2020-02-02
4023 \g@addto@macro\@rmfamilyhook
4024   {\prepare@family@series@update@kanji{mc}\mcdefault}
4025 \g@addto@macro\@sffamilyhook
4026   {\prepare@family@series@update@kanji{gt}\gtdefault}
4027 \g@addto@macro\@ttfamilyhook
4028   {\prepare@family@series@update@kanji{gt}\gtdefault}
4029 \fi
4030 \else % --- for 2020-02-02 or older END & for 2020-10-01 BEGIN
4031 \AddToHook{rmfamily}%
4032   {\prepare@family@series@update@kanji{mc}\mcdefault}
4033 \AddToHook{sffamily}%
4034   {\prepare@family@series@update@kanji{gt}\gtdefault}
4035 \AddToHook{ttfamily}%
4036   {\prepare@family@series@update@kanji{gt}\gtdefault}
4037 \fi % --- for 2020-10-01 END
4038 \ifx\DeclareFixJFMCJKTextFontCommand\@undefined
4039 \DeclareJaTextFontCommand{\textmc}{\mcfamily}
4040 \DeclareJaTextFontCommand{\textgt}{\gtfamily}
4041 \fi
4042 \bxjs@if@sf@default{%
```

```
4043 \renewcommand\kanjifamilydefault{\gtdefault}}
```

念のため。

```
4044 \selectfont
```

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

```
4045 \def\bxjs@parse@qh#1{\let\bxjs@tmpb\relax}
```

```
4046 \let\bxjs@parse@qh@a\@undefined
```

```
4047 \let\bxjs@parse@qh@b\@undefined
```

■パラメタの設定

```
4048 \prebreakpenalty\jis"2147=10000
```

```
4049 \postbreakpenalty\jis"2148=10000
```

```
4050 \prebreakpenalty\jis"2149=10000
```

```
4051 \inhibitxspcode`!=1
```

```
4052 \inhibitxspcode`¯=2
```

```
4053 \xspcode`+=3
```

```
4054 \xspcode`%=3
```

"80~"FF の範囲の `\spcode` を 3 に変更。

```
4055 \@tempcnta="80 \@whilenum\@tempcnta<"100 \do{%
```

```
4056 \xspcode\@tempcnta=3\advance\@tempcnta\@ne}
```

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

```
4057 \let\jsInhibitGlueAtParTop\@inhibitglue
```

`\jsResetDimen` は空のままよい。

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

```
4058 \begingroup
```

```
4059 \catcode`\!=0
```

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

```
4060 \gdef\bxjs@ptex@dir{%
```

```
4061 !iftdir t%
```

```
4062 !else!ifydir y%
```

```
4063 !else ?%
```

```
4064 !fi!fi}
```

新版の pTeX で脚注番号の周囲の空気が過大になる現象への対処。

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

```
4065 % 古い \@makefnmark の定義
```

```
4066 \long\def\bxjs@tmpa{\hbox{%
```

```
4067 !ifydir \@textsuperscript{\normalfont\@thefnmark}}%
```

```
4068 !else\hbox{\yoko\@textsuperscript{\normalfont\@thefnmark}}!fi}}
```

```

4069 \ifx\@makefnmark\bxjs@tmpa
4070 \long\gdef\@makefnmark{%
4071 !ifydir \hbox{ }\hbox{\@textsuperscript{\normalfont\@thefnmark}}\hbox{ }%
4072 !else\hbox{\yoko\@textsuperscript{\normalfont\@thefnmark}}!fi}
4073 \fi

```

エスケープ文字の変更はここまで。

```

4074 \endgroup

```

■minijs パッケージのブロック やっておく。

```

4075 \@namedef{ver@minijs.sty}{ }

```

B.3 pdfTeX 用の処理

```

4076 \else\if \if p\jsEngine T\else\if n\jsEngine T\else F\fi\fi T

```

\jsLetHeadChar を UTF-8 バイト列に対応させる。

```

4077 \let\bxjs@let@hchar@chr\bxjs@let@hchar@chr@ue

```

ムニャムニャ。

```

4078 \@onlypreamble\bxjs@cjk@loaded
4079 \def\bxjs@cjk@loaded{%
4080 \def\@footnotemark{%
4081 \leavevmode
4082 \ifhmode
4083 \edef\@x@sf{\the\spacefactor}%
4084 \ifdim\lastkern>\z@\ifdim\lastkern<5sp\relax
4085 \unkern\unkern
4086 \ifdim\lastskip>\z@ \unskip \fi
4087 \fi\fi
4088 \nobreak
4089 \fi
4090 \@makefnmark
4091 \ifhmode \spacefactor\@x@sf \fi
4092 \relax}%
4093 \let\bxjs@cjk@loaded\relax
4094 }
4095 \g@addto@macro\bxjs@begin@document@hook{%
4096 \@ifpackageloaded{CJK}{%
4097 \bxjs@cjk@loaded
4098 }{ }%
4099 }

```

B.4 X₃TeX 用の処理

```

4100 \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。

```

4101 \def\bxjs@let@hchar@chr#1{%
4102 \@tempcnta`#1\relax \divide\@tempcnta"800\relax

```

```

4103 \bxjs@cond@ifnum\@tempcnta=27 \fi{%
4104   \bxjs@let@hchar@chr@xe
4105 }{\bxjs@let@hchar@out\def{#{1}}}}
4106 \def\bxjs@let@hchar@chr@xe#1{%
4107   \lccode`0=`#1\relax
4108   \lowercase{\bxjs@let@hchar@out\def{0}}}}

```

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

```

4109 \@onlypreamble\bxjs@do@precisetext
4110 \ifx\XeTeXgenerateactualtext\@undefined\else
4111   \def\bxjs@do@precisetext{%
4112     \XeTeXgenerateactualtext=\@ne}
4113 \fi

```

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

TODO:3.0 バージョン要件を見直して暫定措置を解除する。

```

4114 \@onlypreamble\bxjs@do@simplejasetup
4115 \def\bxjs@do@simplejasetup{%
4116   \@namedef{bxjs@zeroglue/0.0pt}{T}%
4117   \ifnum\XeTeXinterchartokenstate>\z@
4118   \else\expandafter\ifx\csname bxjs@zeroglue/\the\XeTeXlinebreakskip\endcsname\relax\else
4119     \jsSimpleJaSetup
4120     \ClassInfo\bxjs@clsname
4121     {'\string\jsSimpleJaSetup' is applied\@gobble}%
4122   \fi\fi}

```

`\jsSimpleJaSetup` 日本語出力用の超簡易的な設定。

```

4123 \newcommand*{\jsSimpleJaSetup}{%
4124   \XeTeXlinebreaklocale "ja"\relax
4125   \XeTeXlinebreakskip=0pt plus 1pt minus 0.1pt\relax
4126   \XeTeXlinebreakpenalty=0\relax}

```

B.5 後処理 (エンジン共通)

```
4127 \fi\fi\fi
```

`simplejasetup` オプションの処理。

```

4128 \ifx\bxjs@do@simplejasetup\@undefined\else
4129   \g@addto@macro\bxjs@begin@document@hook{%
4130     \ifbxjs@simplejasetup
4131       \bxjs@do@simplejasetup
4132     \fi}
4133 \fi

```

`precisetext` オプションの処理。

```

4134 \ifbxjs@precisetext
4135   \ifx\bxjs@do@precisetext\@undefined
4136     \ClassWarning\bxjs@clsname
4137     {The current engine does not support the\MessageBreak

```



```

4138     'precise-text' option\@gobble}
4139   \else
4140     \bxjs@do@precisetext
4141   \fi
4142 \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。

```

4143 \g@addto@macro\bxjs@begin@document@hook{%
4144   \ifx\jsInhibitGlueAtParTop\@empty
4145     \def\bxjs@tmpa{\jsInhibitGlueAtParTop}%
4146     \ifx\everyparhook\bxjs@tmpa
4147       \let\everyparhook\@empty
4148     \fi
4149 \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。

```
4150 \ifnum\bxjs@everyparhook=\bxjs@everyparhook@@modern
```

まず `\everypar` を“乗っ取る” 処理を行う。

```

4151 \let\bxjs@everypar\everypar
4152 \newtoks\everypar
4153 \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。

```

4154 \bxjs@everypar{\the\expandafter\everypar\everyparhook}%
4155 \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。

```
4156 \ifbxjs@fancyhdr
```

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

```

4157 \@onlypreamble\bxjs@adjust@fancyhdr
4158 \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。

```

4159 \def\bxjs@tmpa{\fancyplain-{\sl\rightmark}\strut}%
4160 \def\bxjs@tmpb{\fancyplain-{\rightmark}\strut}%
4161 \ifx\f@ncyelh\bxjs@tmpa \global\let\f@ncyelh\bxjs@tmpb \fi

```

```

4162 \ifx\@ncyerh\bxjs@tmpa \global\let\@ncyerh\bxjs@tmpb \fi
4163 \ifx\@ncyolh\bxjs@tmpa \global\let\@ncyolh\bxjs@tmpb \fi
4164 \ifx\@ncyorh\bxjs@tmpa \global\let\@ncyorh\bxjs@tmpb \fi
4165 \def\bxjs@tmpa{\fancyplain{}{\s1\leftmark}\strut}%
4166 \def\bxjs@tmpb{\fancyplain{}{\leftmark}\strut}%
4167 \ifx\@ncyelh\bxjs@tmpa \global\let\@ncyelh\bxjs@tmpb \fi
4168 \ifx\@ncyerh\bxjs@tmpa \global\let\@ncyerh\bxjs@tmpb \fi
4169 \ifx\@ncyolh\bxjs@tmpa \global\let\@ncyolh\bxjs@tmpb \fi
4170 \ifx\@ncyorh\bxjs@tmpa \global\let\@ncyorh\bxjs@tmpb \fi
4171 \def\bxjs@tmpa{\rm\thepage\strut}%
4172 \def\bxjs@tmpb{\thepage\strut}%
4173 \ifx\@ncyecf\bxjs@tmpa \global\let\@ncyecf\bxjs@tmpb \fi
4174 \ifx\@ncyocf\bxjs@tmpa \global\let\@ncyocf\bxjs@tmpb \fi

```

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

```

4175 \ifx\fullwidth\undefined\else \ifdim\textwidth<\fullwidth
4176   \setlength{\@tempdima}{\fullwidth-\textwidth}%
4177   \edef\bxjs@tmpa{\noexpand\fancyhoffset [EL,OR]{\the\@tempdima}%
4178   }\bxjs@tmpa
4179 \fi\fi
4180 \PackageInfo\bxjs{clsname
4181   {Patch to fancyhdr is applied\@gobble}}

```

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

```

4182 \def\bxjs@pagestyle@hook{%
4183   \@ifpackageloaded{fancyhdr}{%
4184     \bxjs@adjust@fancyhdr
4185     \global\let\bxjs@adjust@fancyhdr\relax
4186   }{}}

```

\pagestyle にフックを入れ込む。

```

4187 \let\bxjs@org@pagestyle\pagestyle
4188 \def\pagestyle{%
4189   \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。

```

4190 \g@addto@macro\bxjs@begin@document@hook{%
4191   \bxjs@pagestyle@hook
4192   \global\let\bxjs@pagestyle@hook\relax}

```

終わり。

```
4193 \fi
```

■和文空白命令

```
4194 \ifbxjs@jaspace@cmd
```

\jaenspace 半角幅の水平空き。

```
4195 \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。

```
4196 \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。

```
4197 \def\_ {\zwspace}
```

`\jaspace` `jlreq` クラスと互換の命令。

```
4198 \DeclareRobustCommand*{\jaspace}[1]{%
4199   \expandafter\ifx\csname bxjs@jaspace@@#1\endcsname\relax
4200   \ClassError\bxjs@clsname
4201   {Unknown jaspac: #1}{\@eha}%
4202   \else
4203   \csname bxjs@jaspace@@#1\endcsname
4204   \fi}
4205 \def\bxjs@jaspace@@zenkaku{\hskip 1\jsZw\relax}
4206 \def\bxjs@jaspace@@nibu{\hskip .5\jsZw\relax}
4207 \def\bxjs@jaspace@@shibu{\hskip .25\jsZw\relax}
```

終わり。

```
4208 \fi
```

以上で終わり。

```
4209 %</minimal>
```

付録 C 和文ドライバ：standard 🍡

標準のドライバ。

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

C.1 準備

```
4210 %<*standard>
```

```
4211 %% このファイルは日本語文字を含みます
```

まず `minimal` ドライバを読み込む。

```
4212 \input{bxjsja-minimal.def}
```

simplejasetup は standard では無効になる。

```
4213 \bxjs@simplejasetupfalse
```

■環境検査

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

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

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

加えて、以下の要件を定める。

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

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

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

```
4214 \ifjswitheTeX
```

```
4215 \IfFileExists{etoolbox.sty}{%
```

```
4216 \RequirePackage{etoolbox}[2007/10/08]% v1.3
```

```
4217 }{}
```

```
4218 \fi
```

C.2 和文ドライバパラメタ

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

\ifbxjs@jp@jismmiv 2004JIS 字形を優先させるか。

```
4219 \newif\ifbxjs@jp@jismmiv
```

jis2004 オプションの処理。

```
4220 \bxjs@cslet{bxjs@kv@jis2004@true}\bxjs@jp@jismmivtrue
```

```
4221 \bxjs@cslet{bxjs@kv@jis2004@false}\bxjs@jp@jismmivfalse
```

```
4222 \define@key{bxjsStd}{jis2004}[true]{%
```

```
4223 \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。

```
4224 \newif\ifbxjs@jp@units
```

units オプションの処理。

```

4225 \let\bxjs@kv@units@true\bxjs@jp@unitstrue
4226 \let\bxjs@kv@units@false\bxjs@jp@unitsfalse
4227 \define@key{bxjsStd}{units}[true]{%
4228   \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。

```
4229 \let\bxjs@jp@font\@empty
```

`font` オプションの処理。

※ 2.9 版より、複数回指定した場合には累積させる。

```

4230 \define@key{bxjsStd}{font}{%
4231   \edef\bxjs@jp@font{\bxjs@catopt\bxjs@jp@font{#1}}}

```

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

```
4232 \newif\ifbxjs@jp@strong@cmd \bxjs@jp@strong@cmdtrue
```

`strong-cmd` オプションの処理。

```

4233 \let\bxjs@kv@strongcmd@true\bxjs@jp@strong@cmdtrue
4234 \let\bxjs@kv@strongcmd@false\bxjs@jp@strong@cmdfalse
4235 \define@key{bxjs}{strong-cmd}[true]{\bxjs@set@keyval{strongcmd}{#1}}

```

実際の `japaram` の値を適用する。

```

4236 \def\bxjs@next#1{\bxjs@safe@setkeys{bxjsStd}{#1}}
4237 \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。

```

4238 \@onlypreamble\bxjs@apply@mmiv
4239 \def\bxjs@apply@mmiv{%
4240   \g@addto@macro\@classoptionslist{,jis2004}
4241   \% \ifpackagewith 判定への対策
4242   \PassOptionsToPackage{jis2004}{otf}
4243   \global\let\bxjs@apply@mmiv\relax}
4244 \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。

```

4245 \if j\jsEngine
4246   \bxjs@jp@unitsfalse
4247 \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。

```

4248 \ifbxjs@jp@units
4249   \IfFileExists{bxcalc.sty}{%
4250     \RequirePackage{bxcalc}[2018/01/28]%v1.0a

```

```

4251 \ifx\usepTeXunits\@undefined
4252 \PackageWarningNoLine\bxjs@clsname
4253 {Cannot support pTeX units (zw etc.), since\MessageBreak
4254 the package 'bxcalc' is too old}%
4255 \bxjs@jp@unitsfalse
4256 \else \usepTeXunits
4257 \fi
4258 }{%else
4259 \PackageWarningNoLine\bxjs@clsname
4260 {Cannot support pTeX units (zw etc.), since\MessageBreak
4261 the package 'bxcalc' is unavailable}%
4262 \bxjs@jp@unitsfalse
4263 }
4264 \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。

```

4265 \ifbxjs@jp@units
4266 \def\bxjs@parse@qh#1{\let\bxjs@tmpb\relax}
4267 \let\bxjs@parse@qh@a\@undefined
4268 \let\bxjs@parse@qh@b\@undefined
4269 \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。

```

4270 \ifbxjs@jp@units
4271 \def\bxjs@let@lenexpr#1#2{%
4272 \edef#1{#2}%
4273 \expandafter\CUXParseExpr\expandafter#1\expandafter{#1}}
4274 \else
4275 \def\bxjs@let@lenexpr{\edef}
4276 \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 strongenv (env.) にする。

※既に利用可能である場合は何もしない。

```

4277 \ifbxjs@jp@strong@cmd\jsAtEndOfClass{%
4278 \ifx\strong\@undefined\ifx\strongenv\@undefined
4279 \DeclareRobustCommand{\strongenv}{\bxjs@strong@font}%
4280 \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。

```

4281 \newcommand*{\strongfontdeclare}{\bxjs@strongfontdeclare}%
4282 \newcount\bxjs@strong@level

```

```

4283 \bxjs@protected\def\bxjs@strongfontdeclare#1{%
4284 \bxjs@set@array@from@clist{bxjs@strong}{#1}%
4285 \bxjs@strong@level\z@}%
4286 \bxjs@strongfontdeclare{\bfseries}%
4287 \def\bxjs@strong@font{%
4288 \bxjs@csletcs{bxjs@tmpa}{bxjs@strong/\the\bxjs@strong@level}%
4289 \ifx\bxjs@tmpa\relax
4290 \advance\bxjs@strong@level\m@ne \bxjs@strong@font
4291 \else \advance\bxjs@strong@level\@ne \bxjs@tmpa
4292 \fi}%
4293 \fi\fi
4294 }\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。

```
4295 \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 4296 \@tempdima=0.25mm
4297 \protected\edef\jQ{\dimexpr\the\@tempdima\relax}
4298 \let\jH\jQ

```

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

```

\trueH 4299 \ifjsc@mag
4300 \@tempdimb=\jsBaseFontSize\relax
4301 \edef\bxjs@tmpa{\strip@pt\@tempdimb}%
4302 \@tempdima=2.5mm
4303 \bxjs@invscale\@tempdima\bxjs@tmpa
4304 \protected\edef\trueQ{\dimexpr\the\@tempdima\relax}
4305 \@tempdima=10pt
4306 \bxjs@invscale\@tempdima\bxjs@tmpa
4307 \protected\edef\bxjs@truept{\dimexpr\the\@tempdima\relax}
4308 \else \let\trueQ\jQ \let\bxjs@truept\p@
4309 \fi
4310 \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。

```

4311 \@tempdima\trueQ \bxjs@invscale\@tempdima\jsScale
4312 \protected\edef\ascQ{\dimexpr\the\@tempdima\relax}
4313 \@tempdima\bxjs@truept \bxjs@invscale\@tempdima\jsScale
4314 \protected\edef\ascpt{\dimexpr\the\@tempdima\relax}
4315 \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。

```
4316 \def\jafontsize#1#2{%
4317   \begingroup
4318     \bxjs@jafontsize@a{#1}%
4319     \@tempdimb\jsInverseScale\@tempdima
4320     \bxjs@jafontsize@a{#2}%
4321     \xdef\bxjs@g@tmpa{%
4322       \noexpand\fontsize{\the\@tempdimb}{\the\@tempdima}}%
4323   \endgroup\bxjs@g@tmpa}
4324 \def\bxjs@jafontsize@a#1{%
4325   \bxjs@parse@qh{#1}%
4326   \ifx\bxjs@tmpb\relax \def\bxjs@tmpb{#1}\fi
4327   \@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。

```
4328 \def\bxjs@kanjiskip{0pt}
```

`\setkanjiskip` 和文間空白の量を設定する。

```
4329 \newcommand*\setkanjiskip[1]{%
4330   \bxjs@let@lenexpr\bxjs@kanjiskip{#1}%
4331   \bxjs@reset@kanjiskip}
```

`\getkanjiskip` 和文間空白の量を表すテキストに展開する。

```
4332 \newcommand*\getkanjiskip{%
4333   \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。

```
4334 \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 4335 \bxjs@robust@def\bxjs@enable@kanjiskip{%
4336   \bxjs@kanjiskip@enabledtrue
4337   \bxjs@reset@kanjiskip}
4338 \bxjs@robust@def\bxjs@disable@kanjiskip{%
4339   \bxjs@kanjiskip@enabledfalse
4340   \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。

```
4341 \bxjs@robust@def\bxjs@reset@kanjiskip{%
4342   \ifbxjs@kanjiskip@enabled
4343     \setlength{\@tempskipa}{\bxjs@kanjiskip}%
4344   \else \@tempskipa\z@
4345   \fi
4346   \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 4347 \def\bxjs@xkanjiskip{0pt}
\getxkanjiskip 4348 \newcommand*\setxkanjiskip[1]{%
\ifbxjs@xkanjiskip@enabled 4349 \bxjs@let@lenexpr\bxjs@xkanjiskip{#1}%
4350 \bxjs@reset@xkanjiskip}
\bxjs@enable@xkanjiskip 4351 \newcommand*\getxkanjiskip{%
4352 \bxjs@xkanjiskip}
\bxjs@disable@xkanjiskip 4353 \newif\ifbxjs@xkanjiskip@enabled \bxjs@xkanjiskip@enabledtrue
\bxjs@reset@xkanjiskip 4354 \bxjs@robust@def\bxjs@enable@xkanjiskip{%
4355 \bxjs@xkanjiskip@enabledtrue
4356 \bxjs@reset@xkanjiskip}
4357 \bxjs@robust@def\bxjs@disable@xkanjiskip{%
4358 \bxjs@xkanjiskip@enabledfalse
4359 \bxjs@reset@xkanjiskip}
4360 \bxjs@robust@def\bxjs@reset@xkanjiskip{%
4361 \ifbxjs@xkanjiskip@enabled
4362 \setlength{\@tempskipa}{\bxjs@xkanjiskip}%
4363 \else \@tempskipa\z@
4364 \fi
4365 \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。

```
4366 \g@addto@macro\jsResetDimen{%
4367 \bxjs@reset@kanjiskip
4368 \bxjs@reset@xkanjiskip}
4369 \let\bxjs@apply@kanjiskip\relax
4370 \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、`TeX Live` の `kanji-config-updmap` コマンドで使う“ファミリ”と同じにすることを想定する。特別な値として、`auto` は `kanji-config-updmap` で現在指定されているファミリを表す。

`\bxjs@adjust@jafont` `\jsJaFont` に入っている和文フォント設定の値を“調整”して、その結果を `\bxjs@tmpa` に返す。`#1` が `f` の場合は“非埋込 (noEmbed)”の設定が禁止される。この禁止の場合も含め、何か異常がある場合は `\bxjs@tmpa` は空になる。

```
4371 \@onlypreamble\bxjs@adjust@jafont
4372 \def\bxjs@adjust@jafont#1{%
4373 \ifx\jsJaFont\bxjs@auto
4374 \bxjs@get@kanjiEmbed
4375 \ifx\bxjs@jaEmbed\relax
4376 \let\bxjs@tmpa\@empty
4377 \else
4378 \let\bxjs@tmpa\bxjs@jaEmbed
4379 \ifx\bxjs@jaVariant\bxjs@hziv
4380 \bxjs@apply@mmiv
```

```

4381     \fi
4382     \fi
4383 \else
4384     \let\bxjs@tmpa\jsJaFont
4385     \fi
4386 \if f#1\ifx\bxjs@tmpa\bxjs@noEmbed
4387     \ClassWarningNoLine\bxjs@clsname
4388     {Option 'jafont=noEmbed' is ignored, because it is\MessageBreak
4389     not available on the current situation}%
4390     \let\bxjs@tmpa@empty
4391     \fi\fi
4392 }
4393 \def\bxjs@@auto{auto}
4394 \def\bxjs@@noEmbed{noEmbed}
4395 \def\bxjs@@hziv{-04}

```

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

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

```

4396 \let\bxjs@jaEmbed\relax
4397 \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。

```

4398 \@onlypreamble\bxjs@get@kanjiEmbed
4399 \def\bxjs@get@kanjiEmbed{%
4400     \begingroup\setbox\z@=\hbox{%
4401         \global\let\bxjs@tmpdo@empty
4402         \def\bxjs@next##1##2##3{%
4403             \def##1###1###3 #####2\@nil###3\@nnil{%
4404                 \ifx$###1$\gdef##2{###2}\fi}%
4405             \g@addto@macro\bxjs@tmpdo{%
4406                 \expandafter##1\bxjs@tmpa\@nil##3 \@nil\@nnil}}%
4407             \bxjs@next\bxjs@tmpdo@a\bxjs@g@tmpa{kanjiEmbed}%
4408             \bxjs@next\bxjs@tmpdo@b\bxjs@g@tmpa{jaEmbed}%
4409             \bxjs@next\bxjs@tmpdo@c\bxjs@g@tmpb{kanjiVariant}%
4410             \bxjs@next\bxjs@tmpdo@d\bxjs@g@tmpb{jaVariant}%
4411         }%
4412         \global\let\bxjs@g@tmpa\relax
4413         \global\let\bxjs@g@tmpb\relax
4414         \endlinechar\m@ne
4415         \let\do\@makeother\dospecials
4416         \catcode32=10 \catcode12=10 %form-feed
4417         \let\bxjs@tmpa@empty
4418         \openin\@inputcheck="|kpsewhich updmap.cfg"\relax
4419         \ifeof\@inputcheck\else
4420             \read\@inputcheck to\bxjs@tmpa
4421             \closein\@inputcheck
4422         \fi
4423     \ifx\bxjs@tmpa@empty\else

```

```

4424 \openin\@inputcheck="\bxjs@tmpa"\relax
4425 \@tempwattrue
4426 \loop\if@tempswa
4427 \read\@inputcheck to\bxjs@tmpa
4428 \bxjs@tmpdo
4429 \ifeof\@inputcheck \@tempwafalse \fi
4430 \repeat
4431 \fi
4432 }\endgroup
4433 \let\bxjs@jaEmbed\bxjs@g@tmpa
4434 \let\bxjs@jaVariant\bxjs@g@tmpb
4435 }

```

`\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。

```

4436 \@onlypreamble\bxjs@resolve@jafont@paren
4437 \def\bxjs@resolve@jafont@paren#1{%
4438 \def\bxjs@tmpb{\let#1}%
4439 \expandafter\bxjs@resolve@jafont@paren@a#1\@nil()\@nil\@nnil#1}
4440 \@onlypreamble\bxjs@resolve@jafont@paren@a
4441 \def\bxjs@resolve@jafont@paren@a#1(#2)#3\@nil#4\@nnil#5{%
4442 \ifx\relax#4\relax \bxjs@tmpb#5%
4443 \else
4444 \edef\bxjs@tmpa{#1\bxjs@jafont@paren{#2}#3}%
4445 \bxjs@tmpb\bxjs@tmpa
4446 \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。

```

4447 \newcommand*\jachar[1]{%
4448 \begingroup

```

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

```

4449 \jsLetHeadChar\bxjs@tmpa{#1}%
4450 \ifx\bxjs@tmpa\relax
4451 \ClassWarningNoLine\bxjs@clsname
4452 {Illegal argument given to \string\jachar}%
4453 \else
4454 \expandafter\bxjs@jachar\expandafter{\bxjs@tmpa}%
4455 \fi
4456 \endgroup}

```

`\jsJaChar` を `\jachar` と等価にする。

```
4457 \let\jsJaChar\jachar
```

下請けの `\bxjs@jachar` の実装はエンジンにより異なる。

```
4458 \let\bxjs@jachar\@firstofone
```

■hyperref 対策 出力ページサイズに飽する処理は geometry パッケージが行うので、hyperref 側の処理は無効にしておく。

```
4459 \PassOptionsToPackage{setpagesize=false}{hyperref}
```

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

```
4460 \@onlypreamble\bxjs@fix@hyperref@unicode
4461 \def\bxjs@fix@hyperref@unicode#1{%
4462   \PassOptionsToPackage{bxjs/hook=#1}{hyperref}%
4463   \@namedef{KV@Hyp@bxjs/hook}##1{%
4464     \KV@Hyp@unicode{##1}%
4465     \def\KV@Hyp@unicode####1{%
4466       \expandafter\ifx\csname if##1\expandafter\endcsname
4467       \csname if####1\endcsname\else
4468       \ClassWarningNoLine\bxjs@clsname
4469       {Blcoked hyperref option 'unicode=####1'}%
4470       \fi
4471     }%
4472   }%
4473 }
```

\jsCheckHyperrefUnicode 「hyperref の unicode オプションの値を検証する」ための本体開始時のフック。

※ pxjahyper-uni.def はこのフックを \relax に上書きすることで検証を無効化している。

```
4474 \@onlypreamble\jsCheckHyperrefUnicode
4475 \let\jsCheckHyperrefUnicode\empty
4476 \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。

```
4477 \@onlypreamble\bxjs@check@hyperref@unicode
4478 \def\bxjs@check@hyperref@unicode#1{%
4479   \g@addto@macro\jsCheckHyperrefUnicode{%
4480     \@tempwafalse
4481     \begingroup
4482       \expandafter\ifx\csname ifHy@unicode\endcsname\relax
4483       \aftergroup\@tempwatrue \fi
4484       \expandafter\ifx\csname ifHy@unicode\expandafter\endcsname
4485       \csname if#1\endcsname
4486       \aftergroup\@tempwatrue \fi
4487     \endgroup
4488     \if@tempwa\else
4489       \ClassError\bxjs@clsname
4490       {The value of hyperref 'unicode' key is not suitable\MessageBreak
4491       for the present engine (must be #1)}%
4492       {\@ehc}%
4493     \fi}}}
```

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

```

4494 \@onlypreamble\bxjs@urgent@special
4495 \@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。

```

4496 \ifbxjs@old@hook@system
4497 \def\bxjs@urgent@special#1{%
4498   \AtBeginDvi{\special{#1}}%
4499   \g@addto@macro\bxjs@begin@document@hook{%
4500     \ifpackageloaded{atbegshi}{%
4501       \begingroup
4502         \toks\z@{\special{#1}}%
4503         \toks\tw@\expandafter{\AtBegShi@HookFirst}%
4504         \xdef\AtBegShi@HookFirst{\the\toks@the\toks\tw@}%
4505       \endgroup
4506     }{}%
4507   }%
4508 }

```

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。

```

4509 \else
4510   \def\bxjs@urgent@special#1{%
4511     \bxjs@urgent@special@a
4512     \AddToHook{shipout/firstpage}[pxjahyper/enc]{\special{#1}}
4513   \def\bxjs@urgent@special@a{%
4514     \DeclareHookRule{shipout/firstpage}{pxjahyper/enc}{<}{hyperref}%
4515     \global\let\bxjs@urgent@special@a\relax}
4516 \fi

```

C.4 pTeX 用設定

```
4517 \if j\jsEngine
```

■ 共通命令の実装

```

4518 \def\bxjs@apply@kanjiskip{%
4519   \kanjiskip\@tempskipa}
4520 \def\bxjs@apply@xkanjiskip{%
4521   \xkanjiskip\@tempskipa}

```

\jaJaChar のサブマクロ。

```

4522 \def\bxjs@jachar#1{%
4523   \bxjs@jachar@a#1...\@nil}
4524 \def\bxjs@jachar@a#1#2#3#4#5\@nil{%

```

引数が単一トークンなら和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。

```
4525 \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。

```

4526 \else\ifx.#3%
4527   \@tempcnta`#1 \multiply\@tempcnta64
4528   \advance\@tempcnta`#2 \advance\@tempcnta-"3080
4529   \bxjs@jachar@b
4530 \else\ifx.#4%
4531   \@tempcnta`#1 \multiply\@tempcnta64
4532   \advance\@tempcnta`#2 \multiply\@tempcnta64
4533   \advance\@tempcnta`#3 \advance\@tempcnta-"E2080
4534   \bxjs@jachar@b
4535 \else
4536   \@tempcnta`#1 \multiply\@tempcnta64
4537   \advance\@tempcnta`#2 \multiply\@tempcnta64
4538   \advance\@tempcnta`#3 \multiply\@tempcnta64
4539   \advance\@tempcnta`#4 \advance\@tempcnta-"3C82080
4540   \bxjs@jachar@b
4541 \fi\fi\fi}

```

符号値が \@tempcnta の和文文字を出力する処理。

```

4542 \ifjsWithupTeX
4543   \def\bxjs@jachar@b{\kchar\@tempcnta}
4544 \else
4545   \def\bxjs@jachar@b{%
4546     \ifx\bxUInt\@undefined\else
4547       \bxUInt{\@tempcnta}%
4548     \fi}
4549 \fi

```

和欧文間空白の命令 \jathinspace の実装。

```

4550 \ifbxjs@jaspace@cmd
4551   \def\jathinspace{\hskip\xkanjiskip}
4552 \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。

```

4553 \ifbxjs@jp@jismmiv
4554   \PassOptionsToPackage{prefer2004jis}{pxchfon}
4555   \PassOptionsToPackage{prefer2004jis}{pxbabel}
4556 \fi

```

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

```

4557 \let\bxjs@jafont@paren\@firstofone
4558 \let\bxjs@tmpa\jsJaFont
4559 \ifx\bxjs@tmpa\bxjs@@auto
4560   \let\bxjs@tmpa\@empty
4561 \else\ifx\bxjs@tmpa\bxjs@@noEmbed

```

```

4562 \def\bxjs@tmpa{noembed}
4563 \fi\fi
4564 \bxjs@resolve@jafont@paren\bxjs@tmpa
4565 \edef\bxjs@tmpa{\bxjs@catopt\bxjs@tmpa\bxjs@jp@font}
4566 \ifx\bxjs@tmpa@empty\else
4567 \edef\bxjs@next{%
4568 \noexpand\RequirePackage[\bxjs@tmpa]{pxchfon}[2010/05/12]% v0.5
4569 }\bxjs@next
4570 \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……。)

```

4571 \begingroup
4572 \global\let\bxjs@g@tmpa\relax
4573 \catcode`\|=0 \catcode`\|=12
4574 |def|bxjs@tmpdo#1|@nil{%
4575 |bxjs@tmpdo@a#1|@nil\RequirePackage|@nnil}%
4576 |def|bxjs@tmpdo@a#1\RequirePackage#2|@nnil{%
4577 |ifx$#1$|bxjs@tmpdo@b#2|@nil keyval|@nnil |fi}%
4578 |catcode`\|=0 \catcode`\|=12
4579 \def\bxjs@tmpdo@b#1keyval#2\@nnil{%
4580 \ifx$#2$\else
4581 \xdef\bxjs@g@tmpa{%
4582 \noexpand\PassOptionsToPackage{scale=\jsScale}{otf}}%
4583 \fi}
4584 \@firstofone{%
4585 \catcode10=12 \endlinechar\m@ne
4586 \let\do\@makeother \dospecials \catcode32=10
4587 \openin\@inputcheck=otf.sty\relax
4588 \@tempswatrue
4589 \loop\if@tempswa
4590 \ifeof\@inputcheck \@tempswafalse \fi
4591 \if@tempswa
4592 \read\@inputcheck to\bxjs@next
4593 \expandafter\bxjs@tmpdo\bxjs@next\@nil
4594 \fi
4595 \repeat
4596 \closein\@inputcheck
4597 \endgroup}
4598 \bxjs@g@tmpa

```

■**hyperref 対策** hyperref の unicode オプションに対する調整を行う。

※ pxjahyper パッケージの「unicode 対応」サポートの履歴：

- 0.7 版 [2021-02-13] : upL^AT_EX 上に限り unicode 対応。
- 0.9c 版 [2021-06-06] : pxjahyper-uni.def ファイルを追加。
- 1.0 版 [2022-04-01] : pL^AT_EX 上の unicode 対応を試験的サポート。
- 1.3 版 [2023-03-01] : pL^AT_EX 上の unicode 対応を正式サポート。

```
4599 \ifbxjs@hyperref@enc
```

unicode オプションが偽であることを検証する。ただし、pxjahyper パッケージまたは pxjahyper-uni.def が読み込まれて（前提条件を満たして）「unicode 対応」が行われた場合は検証は無効化される。

```
4600 \bxjs@check@hyperref@unicode{false}
```

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

```
4601 \bxjs@if@package@at@least{plautopatch}{2020/05/25}{% v0.9g
```

```
4602 \let\bxjs@plautopatch@new=t}{}
```

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

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

```
4603 \let\bxjs@avail@hy@unicode=f
```

```
4604 \if \ifx t\bxjs@plautopatch@new T%
```

```
4605 \else\ifbxjs@old@hook@system F\else T\fi\fi T%
```

```
4606 \IfFileExists{pxjahyper-uni.def}{\let\bxjs@avail@hy@unicode=t}{}
```

```
4607 \fi
```

```
4608 \if t\bxjs@avail@hy@unicode
```

```
4609 \ifjswithupteX
```

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

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

```
4610 \ifbxjs@old@hook@system\else
```

```
4611 \AddToHook{\bxjs@CGHN{package/hyperref/after}}{%
```

```
4612 \input{pxjahyper-uni.def}}
```

```
4613 \fi
```

```
4614 \else
```

必要条件が満たされていて、かつ pL^AT_EX である場合の処理。pxjahyper が「pL^AT_EX 上の unicode 対応をもつほど新しい版（1.3 版以降）」であるかを判定する方法はない。しかし、新しい L^AT_EX システムで unicode を無効にするのは避けたいので、L^AT_EX カーネルが 2023/06/01 版以降である場合に pxjahyper も十分に新しいと推定することにする。すなわち「pxjahyper が読み込まれるはず」かつ「L^AT_EX がカーネルが新しい」かを判定する。

```
4615 \let\bxjs@avail@hy@unicode=f
```

```
4616 \ifx t\bxjs@plautopatch@new
```



```

4617         \bxjs@if@format@at@least{2023/06/01}{\let\bxjs@avail@hy@unicode=t}{}
4618         \fi
4619     \fi
4620 \fi

```

この時点で「unicode を有効にできるか」の判定結果がフラグ `\bxjs@avail@hy@unicode` に入っている。unicode を有効にできない場合は unicode の既定値を偽に設定する。

```

4621 \if f\bxjs@avail@hy@unicode
4622     \PassOptionsToPackage{unicode=false}{hyperref}
4623 \fi
4624 \fi

```

tounicode special 命令を出力する。

```

4625 \if \ifx\bxjs@driver@given\bxjs@driver@@dvipdfmx T%
4626     \else\ifjsWithpTeXng T\else F\fi\fi T%
4627 \IfFileExists{pxjahyper-enc.sty}{\@tempwatrue}{\@tempwafalse}
4628 \if@tempswa
4629     \RequirePackage{pxjahyper-enc}[2020/10/05]%v0.6
4630     \ifbxjs@bigcode\else \suppressbigcode \fi
4631 \else
4632     \ifnum\jis"2121="A1A1 %euc
4633         \bxjs@urgent@special{pdf:tounicode EUC-UCS2}
4634     \else\ifnum\jis"2121="8140 %sjis
4635         \bxjs@urgent@special{pdf:tounicode 90ms-RKSJ-UCS2}
4636     \else\ifnum\jis"2121="3000 %uptex
4637         \ifbxjs@bigcode
4638             \bxjs@urgent@special{pdf:tounicode UTF8-UTF16}
4639             \PassOptionsToPackage{bigcode}{pxjahyper}
4640         \else
4641             \bxjs@urgent@special{pdf:tounicode UTF8-UCS2}
4642             \PassOptionsToPackage{nobigcode}{pxjahyper}
4643         \fi
4644     \fi\fi\fi
4645 \let\bxToUnicodeSpecialDone=t
4646 \fi
4647 \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。

```

4648 \ifx f\bxjs@enablejfam\else
4649     \@enablejfamtrue
4650 \fi

```

実際に和文用の数式ファミリの設定を行う。

```

4651 \if@enablejfam
4652     \DeclareSymbolFont{mincho}{\jsc@JYn}{mc}{m}{n}
4653     \DeclareSymbolFontAlphabet{\mathmc}{mincho}
4654     \SetSymbolFont{mincho}{bold}{\jsc@JYn}{gt}{m}{n}
4655     \jfam\symmincho

```

```

4656 \DeclareMathAlphabet{\mathgt}{\jsc@JYn}{gt}{m}{n}
4657 \g@addto@macro\bxjs@begin@document@hook{%
4658   \ifx\reDeclareMathAlphabet\undefined\else
4659     \reDeclareMathAlphabet{\mathrm}{\@mathrm}{\@mathmc}%
4660     \reDeclareMathAlphabet{\mathbf}{\@mathbf}{\@mathgt}%
4661     \reDeclareMathAlphabet{\mathsf}{\@mathsf}{\@mathgt}%
4662   \fi}
4663 \fi

```

C.5 pdfTeX 用設定：CJK + bxcjkatype

```

4664 \else\if \if p\jsEngine T\else\if n\jsEngine T\else F\fi\fi T

```

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

※ Pandoc モードでは `autotilde` を指定しない。

```

4665 \bxjs@adjust@jafont{f}
4666 \let\bxjs@jafont@paren\@firstofone
4667 \bxjs@resolve@jafont@paren\bxjs@tmpa
4668 \edef\bxjs@tmpa{\bxjs@catopt\bxjs@tmpa\bxjs@jp@font}
4669 \edef\bxjs@tmpa{\bxjs@catopt\bxjs@tmpa{whole}}
4670 \ifx\bxjs@jadriver\bxjs@pandoc\else
4671   \edef\bxjs@tmpa{\bxjs@catopt\bxjs@tmpa{autotilde}}
4672 \fi
4673 \edef\bxjs@next{%
4674   \noexpand\RequirePackage[\bxjs@tmpa]{bxcjkatype}[2013/10/15]% v0.2c
4675 }\bxjs@next
4676 \bxjs@cjk@loaded

```

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

```

4677 \ifbxjs@hyperref@enc
4678   \PassOptionsToPackage{unicode}{hyperref}
4679 \fi

```

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

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

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

```

4680 \ifx\bxcjkatypeHyperrefPatchDone\undefined
4681 \begingroup
4682   \CJK@input{UTF8.bdg}
4683 \endgroup
4684 \g@addto@macro\pdfstringdefPreHook{%
4685   \@nameuse{CJK@UTF8@Binding}%

```

```

4686 }
4687 \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。

4688 \ifx\bxckjatypeHyperrefPatchDone\@undefined
4689 \g@addto@macro\pdfstringdefPreHook{%
4690   \ifx~\bxjs@CJktilde
4691     \let\bxjs@org@LetUnexpandableSpace\HyPsd@LetUnexpandableSpace
4692     \let\HyPsd@LetUnexpandableSpace\bxjs@LetUnexpandableSpace
4693     \let~\@empty
4694   \fi
4695 }
4696 \def\bxjs@CJktilde{\CJkceglue\ignorespaces}
4697 \def\bxjs@tildecmd{~}
4698 \def\bxjs@LetUnexpandableSpace#1{%
4699   \def\bxjs@tmpa{#1}\ifx\bxjs@tmpa\bxjs@tildecmd\else
4700     \bxjs@org@LetUnexpandableSpace#1%
4701   \fi}
4702 \fi

```

■ 共通命令の実装

```

4703 \newskip\jsKanjiSkip
4704 \newskip\jsXKanjiSkip
4705 \ifx\CJkceglue\@undefined
4706   \def\CJktilde{\CJk@global\def~{\CJkceglue\ignorespaces}}
4707 \fi
4708 \let\autospacing\bxjs@enable@kanjiskip
4709 \let\noautospacing\bxjs@disable@kanjiskip
4710 \protected\def\bxjs@CJkglue{\hskip\jsKanjiSkip}
4711 \def\bxjs@apply@kanjiskip{%
4712   \jsKanjiSkip\@tempkipa
4713   \let\CJkglue\bxjs@CJkglue}
4714 \let\autoxspacing\bxjs@enable@xkanjiskip
4715 \let\noautoxspacing\bxjs@disable@xkanjiskip
4716 \protected\def\bxjs@CJkceglue{\hskip\jsXKanjiSkip}
4717 \def\bxjs@apply@xkanjiskip{%
4718   \jsXKanjiSkip\@tempkipa
4719   \let\CJkceglue\bxjs@CJkceglue}

```

\jachar のサブマクロの実装。

```

4720 \def\bxjs@jachar#1{%
4721   \CJkforced{#1}}

```

和欧文間空白の命令 \jathinspace の実装。

```

4722 \ifbxjs@jaspace@cmd
4723   \protected\def\jathinspace{\CJkceglue}
4724 \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。

```
4725 \ifx t\bxjs@enablejfam
4726 \ClassWarningNoLine\bxjs@clsname
4727 {You cannot use 'enablejfam=true', since the\MessageBreak
4728 CJK package does not support Japanese math}
4729 \fi
```

C.6 X_ƎT_EX 用設定：xeCJK + zxjatype

```
4730 \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。

```
4731 \RequirePackage{zxjatype}
4732 \PassOptionsToPackage{no-math}{fontspec}%!
4733 \PassOptionsToPackage{xetex}{graphicx}%!
4734 \PassOptionsToPackage{xetex}{graphics}%!
4735 \ifx\zxJaFamilyName\@undefined
4736 \ClassError\bxjs@clsname
4737 {xeCJK or zxjatype is too old}\@ehc
4738 \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。

```
4739 \bxjs@adjust@jafont{f}
4740 \let\bxjs@jafont@paren@gobble
4741 \bxjs@resolve@jafont@paren\bxjs@tmpa
4742 \edef\bxjs@tmpa{\bxjs@catopt\bxjs@tmpa\bxjs@jp@font}
4743 \ifx\bxjs@tmpa\@empty
4744 \setCJKmainfont[BoldFont=HaranoAjiGothic-Medium.otf]{HaranoAjiMincho-
Regular.otf}
4745 \setCJKsansfont[BoldFont=HaranoAjiGothic-Medium.otf]{HaranoAjiGothic-
Medium.otf}
4746 \else
4747 \edef\bxjs@next{%
4748 \noexpand\RequirePackage[\bxjs@tmpa]{zxjafont}[2013/01/28]% v0.2a
4749 }\bxjs@next
4750 \fi
```

■hyperref 対策 unicode オプションの指定に関する話。

X_ƎT_EX 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_ƎT_EX の場合にもパッケージ側で文字コード変換を行う方が望ましいと考えている。実際、unicode を無効にしていると警告が出て強制的に有効化される。一方で、過去 (r35125 まで) の xdvipdfmx では、文字列を UTF-16 に変換した状態で与えるのは不正

と見なして警告が発生する。

これを踏まえて、ここでは、「Xe_{La}TeX のバージョンが 0.9992 以上の場合に `unicode` を既定で有効にする」ことにする。

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

TODO:3.0 バージョン要件を見直して暫定措置を解除する。

```
4751 \ifx\stricmp@undefined\else % 未定義なら条件を満たさない
4752 \ifnum\stricmp{\the\XeTeXversion\XeTeXrevision}{0.9992}>\m@ne
4753 \ifbxjs@hyperref@enc
4754 \PassOptionsToPackage{unicode}{hyperref}
4755 \fi
4756 \fi
4757 \fi
```

■段落頭でのグルー挿入禁止 どうやら、`zsjatype` 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。

```
4758 \AtEndOfClass{%
4759 \def\@gnewline #1{%
4760 \ifvmode \@nolnerr
4761 \else
4762 \unskip \reserved@e {\reserved@f#1}\nobreak \hfil \break \null
4763 \nobreak \hskip-1sp\hskip1sp\relax
4764 \ignorespaces
4765 \fi}
4766 }
```

■共通命令の実装

```
4767 \newskip\jsKanjiSkip
4768 \newskip\jsXKanjiSkip
4769 \ifx\CJKecglue@undefined
4770 \def\CJKtilde{\CJK@global\def~{\CJKecglue\ignorespaces}}
4771 \fi
4772 \let\autospacing\bxjs@enable@kanjiskip
4773 \let\noautospacing\bxjs@disable@kanjiskip
4774 \protected\def\bxjs@CJKglue{\hskip\jsKanjiSkip}
4775 \def\bxjs@apply@kanjiskip{%
4776 \jsKanjiSkip\@tempkipa
4777 \xeCJKsetup{CJKglue={\bxjs@CJKglue}}}
4778 \let\autoxspacing\bxjs@enable@xkanjiskip
4779 \let\noautoxspacing\bxjs@disable@xkanjiskip
4780 \protected\def\bxjs@CJKecglue{\hskip\jsXKanjiSkip}
```

```

4781 \def\bxjs@apply@xkanjiskip{%
4782 \jsXKanjiSkip\@tempskipa
4783 \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。

```

4784 \ifx\mcfamily\@undefined
4785 \protected\def\mcfamily{\CJKfamily{\CJKrmddefault}}
4786 \protected\def\gtfamily{\CJKfamily{\JKsfdefault}}
4787 \fi

```

`\jachar` のサブマクロの実装。

```

4788 \def\bxjs@jachar#1{%
4789 \xeCJKDeclareCharClass{CJK}{`#1}\relax
4790 #1}

```

`\jathinspace` の実装。

```

4791 \ifbxjs@jaspace@cmd
4792 \protected\def\jathinspace{\CJKecglue}
4793 \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。

```

4794 \ifx t\bxjs@enablejfam
4795 \@enablejfamtrue
4796 \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。

```

4797 \if@enablejfam
4798 \xeCJKsetup{CJKmath=true}
4799 \fi

```

C.7 LuaTeX 用設定：LuaTeX-ja

```

4800 \else\if l\jsEngine

```

■LuaTeX-ja パッケージの読み込み `luatexja` とともに `luatexja-fontspec` パッケージを読み込む。

`luatexja` は自前の `\zw` (これは実際の現在和文フォントに基づく値を返す) を定義するので、`\zw` の定義を消しておく。なお、レイアウト定義の「全角幅」は「規定」に基づく `\jsZw` であることに注意が必要。

※ 1.0b 版から「graphics パッケージに `pdftex` オプションを渡す」処理を行っていたが、1.4 版で廃止された。

```

4801 \let\zw\@undefined
4802 \RequirePackage{luatexja}
4803 \edef\bxjs@next{%
4804 \noexpand\RequirePackage[scale=\jsScale]{luatexja-fontspec}[2015/08/26]%

```

```
4805 }\bxjs@next
```

\set@fontsize へのパッチ適用を再度行う。

```
4806 \bxjs@patch@set@fontsize
```

フォント代替の明示的定義。

```
4807 \DeclareFontShape{JY3}{mc}{m}{it}{<->ssub*mc/m/n}{  
4808 \DeclareFontShape{JY3}{mc}{m}{sl}{<->ssub*mc/m/n}{  
4809 \DeclareFontShape{JY3}{mc}{m}{sc}{<->ssub*mc/m/n}{  
4810 \DeclareFontShape{JY3}{gt}{m}{it}{<->ssub*gt/m/n}{  
4811 \DeclareFontShape{JY3}{gt}{m}{sl}{<->ssub*gt/m/n}{  
4812 \DeclareFontShape{JY3}{mc}{bx}{it}{<->ssub*gt/m/n}{  
4813 \DeclareFontShape{JY3}{mc}{bx}{sl}{<->ssub*gt/m/n}{  
4814 \DeclareFontShape{JY3}{gt}{bx}{it}{<->ssub*gt/m/n}{  
4815 \DeclareFontShape{JY3}{gt}{bx}{sl}{<->ssub*gt/m/n}{  
4816 \DeclareFontShape{JY3}{mc}{b}{n}{<->ssub*mc/bx/n}{  
4817 \DeclareFontShape{JY3}{mc}{b}{it}{<->ssub*mc/bx/n}{  
4818 \DeclareFontShape{JY3}{mc}{b}{sl}{<->ssub*mc/bx/n}{  
4819 \DeclareFontShape{JY3}{gt}{b}{n}{<->ssub*gt/bx/n}{  
4820 \DeclareFontShape{JY3}{gt}{b}{it}{<->ssub*gt/bx/n}{  
4821 \DeclareFontShape{JY3}{gt}{b}{sl}{<->ssub*gt/bx/n}{  
4822 \DeclareFontShape{JT3}{mc}{m}{it}{<->ssub*mc/m/n}{  
4823 \DeclareFontShape{JT3}{mc}{m}{sl}{<->ssub*mc/m/n}{  
4824 \DeclareFontShape{JT3}{mc}{m}{sc}{<->ssub*mc/m/n}{  
4825 \DeclareFontShape{JT3}{gt}{m}{it}{<->ssub*gt/m/n}{  
4826 \DeclareFontShape{JT3}{gt}{m}{sl}{<->ssub*gt/m/n}{  
4827 \DeclareFontShape{JT3}{mc}{bx}{it}{<->ssub*gt/m/n}{  
4828 \DeclareFontShape{JT3}{mc}{bx}{sl}{<->ssub*gt/m/n}{  
4829 \DeclareFontShape{JT3}{gt}{bx}{it}{<->ssub*gt/m/n}{  
4830 \DeclareFontShape{JT3}{gt}{bx}{sl}{<->ssub*gt/m/n}{  
4831 \DeclareFontShape{JT3}{mc}{b}{n}{<->ssub*mc/bx/n}{  
4832 \DeclareFontShape{JT3}{mc}{b}{it}{<->ssub*mc/bx/n}{  
4833 \DeclareFontShape{JT3}{mc}{b}{sl}{<->ssub*mc/bx/n}{  
4834 \DeclareFontShape{JT3}{gt}{b}{n}{<->ssub*gt/bx/n}{  
4835 \DeclareFontShape{JT3}{gt}{b}{it}{<->ssub*gt/bx/n}{  
4836 \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。

```
4837 \bxjs@adjust@jafont{t}  
4838 \ifx\bxjs@tmpa\bxjs@@noEmbed  
4839 \def\bxjs@tmpa{noembed}  
4840 \fi  
4841 \let\bxjs@jafont@paren\@gobble  
4842 \bxjs@resolve@jafont@paren\bxjs@tmpa  
4843 \edef\bxjs@tmpa{\bxjs@catopt\bxjs@tmpa\bxjs@jp@font}  
4844 \ifx\bxjs@tmpa\@empty
```

```

4845 \defaultjfontfeatures{ Kerning=Off }
4846 \setmainfont [BoldFont=HaranoAjiGothic-Medium.otf,JFM=ujis]{HaranoAjiMincho-
Regular.otf}
4847 \setsansfont [BoldFont=HaranoAjiGothic-Medium.otf,JFM=ujis]{HaranoAjiGothic-
Medium.otf}
4848 \else
4849 \edef\bxjs@next{%
4850 \noexpand\RequirePackage[\bxjs@tmpa]{luatexja-preset}%
4851 }\bxjs@next
4852 \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。

```

4853 \ifpackagelater{luatexja}{2016/03/31}{}{%else
4854 \DeclareRobustCommand\rmfamily
4855 {\not@math@alphabet\rmfamily\mathrm
4856 \romanfamily\rmdefault\kanjifamily\mcdefault\selectfont}
4857 \DeclareRobustCommand\sffamily
4858 {\not@math@alphabet\sffamily\mathsf
4859 \romanfamily\sfdefault\kanjifamily\gtdefault\selectfont}
4860 \DeclareRobustCommand\ttfamily
4861 {\not@math@alphabet\ttfamily\mathtt
4862 \romanfamily\ttdefault\kanjifamily\gtdefault\selectfont}
4863 }
4864 \long\def\jttdefault{\gtdefault}
4865 \unless\ifx\@ltj@match@familytrue\@undefined
4866 \@ltj@match@familytrue
4867 \fi
4868 \g@addto@macro\bxjs@begin@document@hook{%
4869 \reDeclareMathAlphabet{\mathrm}{\mathrm}{\mathmc}%
4870 \reDeclareMathAlphabet{\mathbf}{\mathbf}{\mathgt}%
4871 \reDeclareMathAlphabet{\mathsf}{\mathsf}{\mathgt}}%
4872 \bxjs@if@sf@default{%
4873 \renewcommand\kanjifamilydefault{\gtdefault}}

```

■和文パラメタの設定

```

4874 % 次の3つは既定値の通り
4875 %\ltjsetparameter{prebreakpenalty={` ,10000}}
4876 %\ltjsetparameter{postbreakpenalty={` ",10000}}
4877 %\ltjsetparameter{prebreakpenalty={` ",10000}}
4878 \ltjsetparameter{jaxspmode={` ! ,1}}
4879 \ltjsetparameter{jaxspmode={` ¯ ,2}}
4880 \ltjsetparameter{alxspmode={` + ,3}}
4881 \ltjsetparameter{alxspmode={` % ,3}}

```

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

※\jsInhibitGlueAtParTop は使わない。

\ltjfakeparbegin 現在の LuaTeX-ja で定義されているマクロで、段落中で段落冒頭用の処理を発動する。未定

義である場合に備えて同等のものを用意する。

```
4882 \ifx\ltjfakeparbegin\@undefined
4883 \protected\def\ltjfakeparbegin{%
4884   \ifhmode
4885     \relax\directlua{%
4886       luatexja.jfmglue.create_beginpar_node()}}
4887 \fi}
4888 \fi
```

ltjs* クラスの定義と同等になるようにパッチを当てる。

```
4889 \unless\ifnum\bxjs@everyparhook=\bxjs@everyparhook@none
4890 \begingroup
4891 \let%\@percentchar \def\@#1{[[\detokenize{#1}]]}
4892 \@gobble\if\def\bxjs@tmpa{\@{\everypar{}}\fi}
4893 \ifnum\bxjs@everyparhook=\bxjs@everyparhook@compat
4894   \@gobble\if\def\bxjs@tmpa{\@{\everypar{\everyparhook}}\fi}\fi
4895 \directlua{
4896   local function patchcmd(cs, code, from, to)
4897     tex.sprint(code:gsub(from:gsub("%W", "%\\%\\%0"), "%0"..to)
4898       :gsub("macro:", \@gdef..cs, 1):gsub("->", "{", 1).."}")
4899   end
4900   patchcmd(\@\@xsect, [[\meaning\@xsect]],
4901     \@{\hskip-\@tempskipa}, \@{\ltjfakeparbegin})
4902   patchcmd(\@\@item, [[\meaning\@item]],
4903     \bxjs@tmpa, \@{\ltjfakeparbegin})}
4904 \endgroup
4905 \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。

```
4906 \ifbxjs@hyperref@enc
4907 \PassOptionsToPackage{unicode}{hyperref}
4908 \bxjs@check@hyperref@unicode{true}
4909 \fi
```

■共通命令の実装

```
4910 \protected\def\autospacing{%
4911   \ltjsetparameter{autospacing=true}}
4912 \protected\def\noautospacing{%
4913   \ltjsetparameter{autospacing=false}}
4914 \protected\def\autoxspacing{%
4915   \ltjsetparameter{autoxspacing=true}}
4916 \protected\def\noautoxspacing{%
4917   \ltjsetparameter{autoxspacing=false}}
4918 \def\bxjs@apply@kanjiskip{%
4919   \ltjsetparameter{kanjiskip={\@tempskipa}}}
4920 \def\bxjs@apply@xkanjiskip{%
4921   \ltjsetparameter{xkanjiskip={\@tempskipa}}}
```

`\jachar` のサブマクロの実装。

```
4922 \def\bxjs@jachar#1{%
4923   \ltjjachar`#1\relax}
```

`\jathinspace` の実装。

```
4924 \ifbxjs@jaspac@cmd
4925   \protected\def\jathinspace{%
4926     \hskip\ltjgetparameter{xkanjiskip}\relax}
4927 \fi
```

■和文数式ファミリー LuaTeX-ja 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。

```
4928 \ifx f\bxjs@enablejfam
4929   \ClassWarningNoLine\bxjs@clsname
4930     {You cannot use 'enablejfam=false', since the\MessageBreak
4931       LuaTeX-ja always provides Japanese math families}
4932 \fi
```

C.8 共通処理 (2)

```
4933 \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 4934 \ifx\DeclareFixJFMCJKTextFontCommand\@undefined
4935   \DeclareJaTextFontCommand{\textmc}{\mcfamily}
4936   \DeclareJaTextFontCommand{\textgt}{\gtfamily}
4937 \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。

```
4938 \ifx\mathmc\@undefined
4939   \DeclareJaMathFontCommand{\mathmc}{\mcfamily}
4940   \DeclareJaMathFontCommand{\mathgt}{\gtfamily}
4941 \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)。

```
4942 \ifbxjs@jaspac@cmd
4943   \bxjs@protected\def\bxjs@choice@jathinspace{%
4944     \relax\ifmode \mskip\medmuskip
4945     \else \jathinspace\ignorespaces
4946     \fi}
4947 \jsAtEndOfClass{%
4948   \ifjsWitheTeX \let\>\bxjs@choice@jathinspace
```

```

4949 \else \def\>{\protect\bxjs@choice@jathinspace}%
4950 \fi}
4951 \fi

```

■和文・和欧文間空白の初期値

```

4952 \setkanjiskip{0pt plus.1\jsZw minus.01\jsZw}
4953 \ifx\jsDocClass\jsSlide \setxkanjiskip{0.1em}
4954 \else \setxkanjiskip{0.25em plus 0.15em minus 0.06em}
4955 \fi

```

以上で終わり。

```
4956 %</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。

```

4957 %<*modern>
4958 \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。

```

4959 \ifnum0\if x\jsEngine1\fi\if 1\jsEngine1\fi=\z@
4960 \def\encodingdefault{T1}%
4961 \input{t1enc.def}%
4962 \fontencoding\encodingdefault\selectfont
4963 \fi

```

基本フォントを Latin Modern フォントファミリに変更する。

※以下は `\usepackage[noamth]{lmodern}` と同じ。ユーザは後で `lmodern` を好きなオプションを付けて読み込むことができる。

```

4964 \ifnum0\if x\jsEngine1\fi\if 1\jsEngine1\fi=\z@
4965 \renewcommand{\rmdefault}{lmr}
4966 \renewcommand{\sfdefault}{lmss}
4967 \renewcommand{\ttdefault}{lmtt}
4968 \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。

```

4969 \DeclareFontShape{OMX}{cmex}{m}{n}{%
4970 <-7.5>cmex7<7.5-8.5>cmex8%
4971 <8.5-9.5>cmex9<9.5->cmex10}{}%
4972 \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。

```
4973 \def\cmex@opt{10}
```

D.2 fixltx2e 読込

※ fixltx2e 廃止前の L^AT_EX カーネルの場合。

```
4974 \ifx\@IncludeInRelease\undefined
```

```
4975 \RequirePackage{fixltx2e}
```

```
4976 \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。

```
4977 \RequirePackage{bxjscssjkat}
```

D.4 完了

おしまい。

```
4978 %</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

```
4979 %<*pandoc>
```

xeCJK で space が有効になるのを阻止する。

※ bxjsja-standard.def の中で xeCJK が読み込まれるためこの位置に置いている。

```
4980 \if x\jsEngine
```

```
4981 \PassOptionsToPackage{nospace}{xeCJK}
```

```
4982 \fi
```

standard ドライバの設定を引き継ぐ。

```
4983 \input{bxjsja-standard.def}
```

■環境検査

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

pandoc 和文ドライバの処理系バージョン要件は standard と同じとする。加えて、以下の要件を定める。

- p_TE_X 系も含めて全てのエンジン種別で e-_TE_X 拡張を要求する。

- 特に etoolbox の 2.0 版以上を要求する。
※もちろん他にも追加の依存パッケージがある。

■パッケージ読込 bxjspandoc パッケージを読み込む。

```
4984 \RequirePackage{bxjspandoc}
```

ϵ -TeX ではない場合に警告を出す。

```
4985 \ifjswitheTeXelse
4986 \ClassWarningNoLine{bxjs@clsname
4987 {!!!!!!! WARNING !!!!!!!\MessageBreak
4988 This engine does not support e-TeX extension!\MessageBreak
4989 Some feature might not work properly}
4990 \fi
```

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

```
4991 \newif\ifbxjs@bxghost@available
4992 \ifjswitheTeX
4993 \RequirePackage{pdftexcmds}[2009/09/22]% v0.5
4994 \IfFileExists{bxghost.sty}{%
4995 \bxjs@bxghost@availabletrue
4996 \@namedef{bxjs@bgbv/79E70A0991967E27981832C84DB5DF99}{1}%v0.2.0
4997 \ifx\pdf@filemdfivesum\undefined\else
4998 \expandafter\ifx\cname bxjs@bgbv/\pdf@filemdfivesum{bxghost.sty}%
4999 \endcsname\relax\else \bxjs@bxghost@availablefalse \fi
5000 \fi
5001 }{}
5002 \fi
```

その他の依存パッケージを読み込む。

```
5003 \RequirePackage{iftex}[2013/04/04]% v0.2
5004 \ifjswitheTeX
5005 \RequirePackage{etoolbox}[2010/08/21]% v2.0
5006 \RequirePackage{filehook}[2011/10/12]% v0.5d
5007 \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。

```
5008 \newif\ifbxjs@jp@fix@strong \bxjs@jp@fix@strongtrue
5009
5010 fix-strong オプションの処理。
5011
5012 \let\bxjs@kv@fixstrong@true\bxjs@jp@fix@strongtrue
5013 \let\bxjs@kv@fixstrong@false\bxjs@jp@fix@strongfalse
5014 \define@key{bxjsPan}{fix-strong}[true]{%
5015 \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。

```
5013 \newif\ifbxjs@jp@fix@code \bxjs@jp@fix@codetrue
```

`fix-code` オプションの処理。

```
5014 \let\bxjs@kv@fixcode@true\bxjs@jp@fix@codetrue
5015 \let\bxjs@kv@fixcode@false\bxjs@jp@fix@codefalse
5016 \define@key{bxjsPan}{fix-code}[true]{%
5017 \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。

```
5018 \chardef\bxjs@jp@strong=0
```

`strong` オプションの処理。

```
5019 \def\bxjs@kv@strong@bold{\chardef\bxjs@jp@strong=0 }
5020 \def\bxjs@kv@strong@sans{\chardef\bxjs@jp@strong=1 }
5021 \def\bxjs@kv@strong@boldsans{\chardef\bxjs@jp@strong=2 }
5022 \define@key{bxjsPan}{strong}{%
5023 \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 5024 \newif\ifbxjs@jp@or@indent \bxjs@jp@or@indenttrue
```

```
\ifbxjs@jp@or@block@heading 5025 \newif\ifbxjs@jp@or@secnumdepth \bxjs@jp@or@secnumdepthtrue
```

```
5026 \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。

```
5027 \define@key{bxjsPan}{_plus}[]{%
5028 \bxjs@jp@or@indentfalse
5029 \bxjs@jp@or@secnumdepthfalse
5030 \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。

```
5031 \let\bxjs@kv@orindent@true\bxjs@jp@or@indenttrue
5032 \let\bxjs@kv@orindent@false\bxjs@jp@or@indentfalse
5033 \define@key{bxjsPan}{or-indent}[true]{%
5034 \bxjs@set@keyval{orindent}{#1}{}}
5035 \let\bxjs@kv@orsecnumdepth@true\bxjs@jp@or@secnumdepthtrue
5036 \let\bxjs@kv@orsecnumdepth@false\bxjs@jp@or@secnumdepthfalse
5037 \define@key{bxjsPan}{or-secnumdepth}[true]{%
5038 \bxjs@set@keyval{orsecnumdepth}{#1}{}}
5039 \let\bxjs@kv@orblockheading@true\bxjs@jp@or@block@headingtrue
5040 \let\bxjs@kv@orblockheading@false\bxjs@jp@or@block@headingfalse
5041 \define@key{bxjsPan}{or-block-heading}[true]{%
5042 \bxjs@set@keyval{orblockheading}{#1}{}}
```

実際の `japaram` の値を適用する。

```
5043 \def\bxjs@next#1{\bxjs@safe@setkeys{bxjsPan}{#1}}
5044 \expandafter\bxjs@next\expandafter{\jsJaParam}
```

E.3 dupload システム

TODO: 新しいカーネルで利用可能な機構での代替を検討する。カーネルへのパッチは排除したいので。

パッケージが重複して読み込まれたときに “option clash” の検査をスキップする。この時に何らかのコードを実行させることができる。

`\bxjs@set@dupload@proc \bxjs@set@dupload@proc{<ファイル名>}{<定義本体>}`： 指定の名前の特定のファイルの読み込みが `\@filewithoptions` で指示されて、しかもそのファイルが読み込み済である場合に、オプション重複検査をスキップして、代わりに `<定義本体>` のコードを実行する。このコード中で `#1` は渡されたオプション列のテキストに置換される。

```
5045 \@onlypreamble\bxjs@set@dupload@proc
5046 \def\bxjs@set@dupload@proc#1{%
5047   \expandafter\bxjs@set@dupload@proc@a\csname bxjs@dlp/#1\endcsname}
5048 \@onlypreamble\bxjs@set@dupload@proc@a
5049 \def\bxjs@set@dupload@proc@a#1{%
5050   \@onlypreamble#1\def#1##1}
5051 \def\bxjs@unset@dupload@proc#1{%
5052   \bxjs@cslet{bxjs@dlp/#1}\@undefined}
```

`\@if@options \@if@options` の再定義。

```
5053 \@onlypreamble\bxjs@org@if@options
5054 \let\bxjs@org@if@options\@if@options
5055 \@onlypreamble\bxjs@org@reset@options
5056 \let\bxjs@org@reset@options\relax
5057 \def\@if@options#1#2#3{%
5058   \let\bxjs@next\@secondoftwo
5059   \def\bxjs@tmpa{#1}\def\bxjs@tmpb{\@currentx}%
5060   \ifx\bxjs@tmpa\bxjs@tmpb
5061     \expandafter\ifx\csname bxjs@dlp/#2.#1\endcsname\relax\else
5062     \let\bxjs@next\@firstoftwo \fi
5063   \fi
5064   \bxjs@next\bxjs@do@dupload@proc\bxjs@org@if@options{#1}{#2}{#3}}
5065 \g@addto@macro\bxjs@begin@document@hook{%
5066   \let\@if@options\bxjs@org@if@options}
5067 \@onlypreamble\bxjs@do@dupload@proc
5068 \def\bxjs@do@dupload@proc#1#2#3{%
5069   \ifx\bxjs@org@reset@options\relax
5070     \let\bxjs@org@reset@options\@reset@options
5071   \fi
5072   \bxjs@csletcs{bxjs@next}{bxjs@dlp/#2.#1}%
5073   \def\@reset@options{%
5074     \let\@reset@options\bxjs@org@reset@options
5075     \@reset@options
5076     \bxjs@next{#3}}%
5077   \@firstoftwo}
```

E.4 lang 変数

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

※そもそも BXJS クラスは日本語用の文書クラスであるため、もし言語設定が行われているのであれば「メイン言語は日本語である」であるはずなので、「サブ言語が日本語である」ことは考慮しない。

■**Polyglossia について** 現在 CTAN に登録されている日本語用の gloss ファイルは超絶アレでかつ有害な設定を行うため、この読込を避ける必要がある。そのため、メイン言語が `japanese` である場合（古い Pandoc ではこの場合に引数が空の `\setmainlanguage{}` が実行されるがこのパターンも同様に扱う）には、Polyglossia の処理を無効化してしまうことにする。つまり、Polyglossia が提供する命令について、何もしないダミーの定義を与える。※ Polyglossia は古い Pandoc のテンプレートにおいて、エンジンが X_YL^AT_EX か Lua^AT_EX の場合に利用されていた。

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

```
5078 \let\bxjs@polyglossia@options\relax
```

エンジンが X_YL^AT_EX か Lua^AT_EX の場合が対象になる。

※この場合 `etoolbox` が使用可能になっている。

```
5079 \ifnum0\if x\jsEngine1\fi\if 1\jsEngine1\fi>0
```

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

```
5080 \pandocSkipLoadPackage{polyglossia}
```

```
5081 \bxjs@set@dupload@proc{polyglossia.sty}{%
```

```
5082 \bxjs@unset@dupload@proc{polyglossia.sty}{%
```

```
5083 \ClassWarning\bxjs@clsname
```

```
5084 {Package polyglossia is requested}%
```

```
5085 \def\bxjs@polyglossia@options{#1}%
```

`polyglossia` の読込が指示された場合、直後に `\setmainlanguage` が実行されることを想定して、フック用の `\setmainlanguage` を定義する。

※最初に `\setmainlanguage` 以外が実行された場合はエラーになる。

```
5086 \newcommand*\setmainlanguage[2][]{%
```

もし、`\setmainlanguage` の引数が空または `japanese` だった場合はメインが日本語である (`lang=ja` 指定) と見なす。

```
5087 \ifboolexpr{test{\ifblank{##2}}or test{\ifstrequal{##2}{japanese}}}{%
```

```
5088 \ClassWarning\bxjs@clsname
```

```
5089 {Main language is 'japanese', thus fallback\MessageBreak
```

```
5090 definitions will be employed}%
```



```

5091     \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。
5092   }{%else
5093     \ClassWarning\bxjs@clsname
5094     {Main language is '##2',\MessageBreak
5095     thus polyglossia will be loaded}%
5096     \csundef{ver@polyglossia.sty}%
5097     \edef\bxjs@next{%
5098     \noexpand\RequirePackage[\bxjs@polyglossia@options]{polyglossia}[]%
5099     }\bxjs@next
5100     \setmainlanguage[##1]{##2}%
5101   }}

```

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

```

5102 \g@addto@macro\bxjs@endpreamble@hook{%
5103   \ifx\bxjs@polyglossia@options\relax
5104     \csundef{ver@polyglossia.sty}%
5105   \fi}

```

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

```

5106 \@onlypreamble\bxjs@pandoc@polyglossia@ja
5107 \def\bxjs@pandoc@polyglossia@ja{%
5108   \renewcommand*\setmainlanguage[2] []{%
5109   \newcommand*\setotherlanguage[2] []{%
5110     \ifblank{##2}{-}{%else
5111     \cslet{##2}\@empty \cslet{end##2}\@empty
5112     \cslet{text##2}\@firstofone}}%
5113   \newcommand*\setotherlanguages[2] []{%
5114     \@for\bxjs@tmpa:={##2}\do{%
5115     \setotherlanguage{\bxjs@tmpa}}}%

```

Polyglossia の読込済マークは外れるようにしておく。

```

5116   \let\bxjs@polyglossia@options\relax}%
5117 \fi

```

■Babel について 現在の Pandoc では、テンプレートで用いられる多言語パッケージとしてエンジンの種別によらずに Babel が使われる。

※ X_qTeX では 2.15 版で、LuaTeX は 2.6 版で Polyglossia から Babel に変更されている。

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

```

5118 \let\bxjs@babel@options\relax

```

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

```

5119 \pandocSkipLoadPackage{babel}
5120 \bxjs@set@dupload@proc{babel.sty}{%
5121 \bxjs@unset@dupload@proc{babel.sty}%
5122 \ClassWarning\bxjs@clsname
5123 {Package babel is requested}%

```

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

```

5124 \@tempwafalse \let\bxjs@babel@options\@empty
5125 \def\bxjs@tmpb{main=}
5126 \@for\bxjs@tmpa:=#1\do{%
5127 \ifx\bxjs@tmpa\bxjs@tmpb \def\bxjs@tmpa{main=japanese}\fi
5128 \edef\bxjs@babel@options{\bxjs@babel@options,\bxjs@tmpa}}%
5129 \bxjs@cslet{ver@babel.sty}\@undefined
5130 \edef\bxjs@next{%
5131 \noexpand\RequirePackage[\bxjs@babel@options]{babel}\relax
5132 }\bxjs@next
5133 \RequirePackage{bxorigcapt}\relax}

```

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

```

5134 \g@addto@macro\bxjs@endpreamble@hook{%
5135 \ifx\bxjs@babel@options\relax
5136 \bxjs@cslet{ver@babel.sty}\@undefined
5137 \fi}

```

3.0 版より前の japanese.1df はサポート対象エンジンが限られていた。ここでは、エンジンの種類を問わず、「japanese.1df が古い場合は読込を回避してダミー定義で代替する」という対策を入れる。実は japanese.1df で行う定義は bxorigcapt の機能等により実質的に全て無効化されている。最新の環境においては「japanese 指定の Babel + bxorigcapt パッケージ」の状態にしておきたい。

```
5138 \ifjswitheTeX
```

filehook の機能を用いて japanese.1df の読込にフックを仕込む。

```

5139 \AtBeginOfFile{japanese.1df}{\bxjs@begin@japanese@ldf@hook}
5140 \def\bxjs@begin@japanese@ldf@hook{%
5141 \let\bxjs@begin@japanese@ldf@hook\relax
5142 \let\bxjs@save@ProvidesLanguage\ProvidesLanguage
5143 \let\bxjs@save@LdfInit\LdfInit
5144 \def\ProvidesLanguage##1[##2]{\bxjs@do@japanese@ldf{##2}}%
5145 \def\LdfInit##1##2{\bxjs@do@japanese@ldf{0000/00/00}}

```

バージョンを判定する部分。

※\LdfInit にも細工を入れている理由は、初期の japanese.1df には \ProvidesLanguage が記述されていないため。

```

5146 \def\bxjs@do@japanese@ldf#1{\bxjs@do@japanese@ldf@a#1\@nil}
5147 \def\bxjs@do@japanese@ldf@a#1/#2/#3#4#5\@nil{%
5148 \let\LdfInit\bxjs@save@LdfInit
5149 \ClassInfo\bxjs@clsname
5150 {Release date of japanese.1df is:\MessageBreak
5151 \@spaces #1/#2/#3#4\@gobble}%

```

```

5152 \ifnum#1#2#3#4<20201206 % v3.0
5153 \let\bxjs@japanese@ldf@skipped=t\csuse{endinput}%
5154 \fi}
5155 \AtEndOfFile{japanese. ldf}{\bxjs@end@japanese@ldf@hook}
5156 \def\bxjs@end@japanese@ldf@hook{%
5157 \let\bxjs@end@japanese@ldf@hook\relax
5158 \let\ProvidesLanguage\bxjs@save@ProvidesLanguage
5159 \let\LdfInit\bxjs@save@LdfInit
5160 \ifx t\bxjs@japanese@ldf@skipped
5161 \ClassWarningNoLine\bxjs@clsname
5162 {Loading japanese. ldf is skipped}%
ダミーの言語定義。
5163 \ifundef\l@japanese{\chardef\l@japanese\z@}{}%
5164 \let\datejapanese\@empty\let\captionjapanese\@empty
5165 \let\extrajapanese\@empty\let\noextrajapanese\@empty
5166 \main@language{japanese}%
5167 \fi}
5168 \g@addto@macro\bxjs@begin@document@hook{%
5169 \let\bxjs@begin@japanese@ldf@hook\relax
5170 \let\bxjs@end@japanese@ldf@hook\relax}
5171 \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。

```

5172 \bxjs@set@dupload@proc{geometry. sty}{%
5173 \setpagelayout*{#1}}

```

E.6 CJKmainfont 変数

LuaTeX (+ LuaTeX-ja) の場合に CJKmainfont 変数が指定された場合は `\setmainjfont` の指定にまわす。

```

5174 \if l\jsEngine
5175 \pandocSkipLoadPackage{xeCJK}
5176 \providecommand*\setCJKmainfont{\setmainjfont}
5177 \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。

```

5178 \if x\jsEngine

```

```

5179 \expandafter\g@addto@macro\csname opt@xeCJK.sty\endcsname{%
5180     ,space}
5181 \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。

```

5182 \ifjswitheTeX
5183 \@onlypreamble\bxjs@info@or@ban
5184 \def\bxjs@info@or@ban#1{%
5185     \PackageInfo\bxjs@clsname
5186     {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。

```

5187 \unless\ifbxjs@jp@or@indent
5188     \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。

```

5189 \IfFileExists{parskip.sty}{%
5190     \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。

```

5191 }{%else
5192     \eappto\bxjs@endpreamble@hook{%
5193         \parindent=\the\parindent\relax
5194         \parskip=\the\parskip\relax}}
5195 \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。

```

5196 \ifbxjs@jp@or@secnumdepth\else
5197     \bxjs@info@or@ban{secnumdepth}
5198     \eappto\bxjs@endpreamble@hook{%
5199         \c@secnumdepth=\the\c@secnumdepth\relax}
5200 \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。

```

5201 \ifbxjs@jp@or@block@heading\else
5202     \let\bxjs@frozen@paragraph\paragraph
5203     \let\bxjs@frozen@subparagraph\subparagraph
5204     \bxjs@info@or@ban{block-heading}
5205     \appto\bxjs@endpreamble@hook{%
5206         \let\oldparagraph\undefined

```

```

5207 \let\paragraph\bxjs@frozen@paragraph
5208 \let\subparagraph\bxjs@frozen@subparagraph}
5209 \fi

    以上。
5210 \fi

```

E.9 paragraph のマーク

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

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

```

5211 \g@addto@macro\bxjs@begin@document@hook{%
5212 \@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。

```

5213 \ifx\oldparagraph\@undefined\else
5214 \@tempswatruetrue
5215 \fi

```

`\paragraph` が番号付きの場合は、マーク変更が必要である。

```

5216 \ifnum\c@secnumdepth>3
5217 \@tempswatruetrue
5218 \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。

```

5219 \if@tempswa\ifx\jsParagraphMark\bxjs@org@paragraph@mark
5220 \let\jsParagraphMark\@empty
5221 \fi\fi}

```

E.10 全角空白文字

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

※ (u)pLaTeX では対応できないので対象外。

`\pandocZWSpace` 全角空白文字の入力で実行されるコード。

```

5222 \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。

```
5223 \ifnum0\if x\jsEngine1\fi\if 1\jsEngine1\fi>\z@
5224 \catcode"3000=\active
5225 \begingroup \catcode`\!=7
5226 \protected\gdef!!!3000{\pandocZWSpace}
5227 \endgroup
5228 \else\ifx\DeclareUnicodeCharacter\@undefined\else
5229 \DeclareUnicodeCharacter{3000}{\bxjs@zsp@char}
5230 \bxjs@protected\def\bxjs@zsp@char{\pandocZWSpace}
5231 \fi\fi
```

E.11 hyperref 対策

`hyperref` の `unicode` オプションの固定を行う。

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

```
5232 \if j\jsEngine
5233 \bxjs@fix@hyperref@unicode{false}
5234 \else
5235 \bxjs@fix@hyperref@unicode{true}
5236 \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 $\text{L}^{\text{T}}\text{E}_\text{X}$ 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 (うわぁ)。

```
5237 \ifbxjs@jp@fix@strong\ifbxjs@jp@strong@cmd
5238 \let\orgtextbf\textbf
5239 \DeclareRobustCommand\pandocTextbf[1]{%
5240 \begingroup
5241 \let\textbf\orgtextbf
5242 \strong{#1}%
5243 \endgroup}%
5244 \g@addto@macro\bxjs@begin@document@hook{%
5245 \let\textbf\pandocTextbf}
5246 \fi\fi
```

`\strong` の書体を設定する。

```
5247 \jsAtEndOfClass{%
5248 \ifx\strongfontdeclare\@undefined\else
5249 \ifcase\bxjs@jp@strong
5250 \or \strongfontdeclare{\sffamily}%
5251 \or \strongfontdeclare{\sffamily\bfseries}%
5252 \fi
5253 \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。

```
5254 \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。

```
5255 \ifbxjs@bxghost@available
5256   \RequirePackage[verb]{bxghost}[2020/01/31]% v0.3.0
5257   \let\bxjs@eghostguarded\eghostguarded
5258   \else
5259   \chardef\bxjs@eghost@c=23
5260   \ifx j\jsEngine \xspcode\bxjs@eghost@c=3
5261   \else\ifx l\jsEngine \ltjsetparameter{alxspmode={\bxjs@eghost@c,3}}
5262   \else\ifx x\jsEngine %no-op
5263   \else \let\bxjs@eghost@c\@undefined
5264   \fi\fi\fi
5265   \ifx\bxjs@eghost@c\@undefined\else
5266   \font\bxjs@eghost@f=ec-lmr10 at 1.23456pt
5267   \def\bxjs@pan@eghost{\bgroup\bxjs@eghost@f\bxjs@eghost@c\egroup}
5268   \def\bxjs@eghostguarded#1{%
5269     \bxjs@pan@eghost\null#1\null\bxjs@pan@eghost}
5270   \fi
5271   \fi
5272   \ifx\bxjs@eghostguarded\@undefined\else
5273   \let\orgtexttt\texttt
5274   \DeclareRobustCommand\pandocTexttt[1]{%
5275     \ifmmode \nfss@text{\ttfamily #1}%
5276     \else
5277     \ifvmode \leavevmode \fi
5278     \bxjs@eghostguarded{\begingroup\ttfamily#1\endgroup}%
5279     \fi}
5280   \g@addto@macro\bxjs@begin@document@hook{%
5281     \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。

```
5282   \ifbxjs@bxghost@available\else
5283   \expandafter\def\expandafter\verb\expandafter{%
5284     \expandafter\bxjs@pan@eghost\verb}
5285   \g@addto@macro\verb\egroup{\bxjs@pan@eghost}
5286   \fi
5287   \fi
5288 \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。

```
5289 \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  
のみ \CS を実行する。
```

```
5290 \def\bxjs@check@frontier{%  
5291   \expandafter\bxjs@check@frontier@a\@currnamestack\noindent...\@nil}  
5292 \def\bxjs@check@frontier@a#1#2#3#4#5\@nil#6{%  
5293   \ifx\noindent#4#6\fi}
```

```
\bxjs@unforge@ifPDFTeX \ifPDFTeX を偽（正しい値）にする。
```

```
5294 \@onlypreamble\bxjs@unforge@ifPDFTeX  
5295 \def\bxjs@unforge@ifPDFTeX{\global\bxjs@csletcs{ifPDFTeX}{iffalse}}
```

```
\bxjs@forge@ifPDFTeX \ifPDFTeX を真（偽装した値）にする。
```

```
5296 \@onlypreamble\bxjs@forge@ifPDFTeX  
5297 \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。
```

```
5298 \def\bxjs@unload@forge@ifPDFTeX{%  
5299   \bxjs@unforge@ifPDFTeX  
5300   \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。

```
5301 \jsAtEndOfClass{\bxjs@forge@ifPDFTeX}  
5302 \ifjsWitheTeX  
5303   \AtBeginOfEveryFile{\bxjs@check@frontier\bxjs@unforge@ifPDFTeX}%  
5304   \AtEndOfEveryFile{\bxjs@check@frontier\bxjs@forge@ifPDFTeX}%  
5305   \g@addto@macro\bxjs@endpreamble@hook{\bxjs@unload@forge@ifPDFTeX}  
5306 \else  
5307   \g@addto@macro\bxjs@begin@document@hook{\bxjs@unload@forge@ifPDFTeX}  
5308 \fi  
5309 \fi
```


E.14 完了

おしまい。

```
5310 %</pandoc>
```

和文ドライバ実装はここまで。

```
5311 %</drv>
```

付録 F 補助パッケージ一覧

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

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

```
5312 %<*anc>
```

付録 G 補助パッケージ：bxjscompat

古いやつをどうにかするためのムニャムニャ。

※すなわち BXJS クラスにおいては「新しいシステムにおいては `bxjscompat` がなくても正常に動作する」状態を保つべき。

G.1 準備

```
5313 %<*compat>
```

```
5314 \def\bxac@pkgname{bxjscompat}
```

`\bxjx@engine` エンジンの種別。

```
5315 \let\bxac@engine=n
```

```
5316 \def\bxac@do#1#2{%
```

```
5317   \edef\bxac@tmpa{\string#1}%
```

```
5318   \edef\bxac@tmpb{\meaning#1}%
```

```
5319   \ifx\bxac@tmpa\bxac@tmpb #2\fi}
```

```
5320 \bxac@do\kanjiskip{\let\bxac@engine=j}
```

```
5321 \bxac@do\XeTeXversion{\let\bxac@engine=x}
```

```
5322 \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。

```
5323 \ifx\jsAtEndOfClass\undefined
```

```
5324   \let\bxac@delayed@if@bxjs\firstofone
```

```
5325 \else \let\bxac@delayed@if@bxjs\jsAtEndOfClass
```

```
5326 \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` で元に戻ることができる。

※エンジンが LuaTeX 以外の場合は何もしない。

```
5327 \newif\ifbxac@in@old@behavior
5328 \let\ImposeOldLuaTeXBehavior\relax
5329 \let\RevokeOldLuaTeXBehavior\relax
```

G.2 8bit 欧文 TeX

```
5330 \ifx n\bxac@engine
```

和文を含むマクロ定義を通用させるため、高位バイトをアクティブ化しておく。

```
5331 \@tempcnta="80 \loop \ifnum\@tempcnta<"100
5332 \catcode\@tempcnta\active
5333 \advance\@tempcnta\@ne
5334 \repeat
```

以上。

```
5335 \fi
```

G.3 XeTeX

```
5336 \ifx x\bxac@engine
```

■文字クラスの設定 XeTeX の文字クラス (`\XeTeXcharclass`) の Unicode 規定に基づく設定は、初期の版ではフォーマットに組み込まれていたが、2016/02/01 以降の L^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、XeTeX に「文字間トークン挿入」の機能が導入されたのは 0.997 版 (2007 年頃) からのようだ。

ただし xeCJK が読込済ならば (そちらが適切に設定しているはずなので) 何もしない。

```
5337 \ifx\XeTeXcharclass\@undefined\else
5338 \bxac@delayed@if@bxjs{%
5339 \@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。

```
5340 \ifx\Xe@alloc@intercharclass\@undefined\else
5341 \ifnum\Xe@alloc@intercharclass=\z@
5342 \PackageInfo\bxac@pkgname
5343 {Setting up interchar class for CJK...\@gobble}%
5344 \InputIfFileExists{load-unicode-xetex-classes.tex}{%
5345 \Xe@alloc@intercharclass=3
5346 }{%else
5347 \PackageWarning\bxac@pkgname
5348 {Cannot find file 'load-unicode-xetex-classes.tex'%
5349 \@gobble}%
5350 }%
5351 \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。

```
5352 \ifnum\XeTeXcharclass"3041=\z@
5353 \PackageInfo\bxac@pkgname
5354 {Adjusting interchar class for CJK...\@gobble}%
5355 \@for\bxac@tmpb:={%
5356 3041,3043,3045,3047,3049,3063,3083,3085,3087,308E,%
5357 3095,3096,30A1,30A3,30A5,30A7,30A9,30C3,30E3,30E5,%
5358 30E7,30EE,30F5,30F6,30FC,31F0,31F1,31F2,31F3,31F4,%
5359 31F5,31F6,31F7,31F8,31F9,31FA,31FB,31FC,31FD,31FE,%
5360 31FF%
5361 }\do{\XeTeXcharclass"\bxac@tmpb=\@ne}%
5362 \fi
5363 }%
5364 }
5365 \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。

```
5366 \chardef\bxac@tmpb=11
5367 \def\bxac@do#1#2{%
5368 \@tempcnta=#1\relax
5369 \unless\ifnum\catcode\@tempcnta=\bxac@tmpb
5370 \chardef\bxac@tmpa=#2\relax
5371 \@whilenum{\@tempcnta<\bxac@tmpa}\do{%
5372 \catcode\@tempcnta\bxac@tmpb \advance\@tempcnta\@ne}%
5373 \fi}
5374 \bxac@do{"4E00}{"9FCD}
5375 \fi
```

以上。

G.4 LuaTeX

```
5376 \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。

```
5377 %\unless\ifnum\luatexversion<80 \ifnum\luatexversion<85
5378 % \chardef\pdftexversion=200
5379 % \def\pdftexrevision{0}
5380 % \let\pdftexbanner\luatexbanner
5381 %\fi\fi
```

```
\ImposeOldLuaTeXBehavior 0.85 版以降であるかを検査する。
```

```
\RevokeOldLuaTeXBehavior 5382 \begingroup\expandafter\expandafter\expandafter\endgroup
5383 \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。

```
5384 \def\bxac@ob@list{%
5385 \do{\let}\pdfoutput{\outputmode}%
```

```

5386 \do{\let}\pdfpagewidth{\pagewidth}%
5387 \do{\let}\pdfpageheight{\pageheight}%
5388 \do{\protected\edef}\pdfhorigin{\pdfvariable horigin}}%
5389 \do{\protected\edef}\pdfvorigin{\pdfvariable vorigin}}
5390 \def\bxac@ob@do#1#2{\begingroup
5391 \expandafter\bxac@ob@do@a\csname bxac@\string#2\endcsname{#1}#2}
5392 \def\bxac@ob@do@a#1#2#3#4{\endgroup
5393 \ifbxac@in@old@behavior \let#1#3\relax #2#3#4\relax
5394 \else \let#3#1\relax \let#1\@undefined
5395 \fi}
5396 \protected\def\ImposeOldLuaTeXBehavior{%
5397 \unless\ifbxac@in@old@behavior
5398 \bxac@in@old@behaviortrue
5399 \let\do\bxac@ob@do \bxac@ob@list
5400 \fi}
5401 \protected\def\RevokeOldLuaTeXBehavior{%
5402 \ifbxac@in@old@behavior
5403 \bxac@in@old@behaviorfalse
5404 \let\do\bxac@ob@do \bxac@ob@list
5405 \fi}
5406 \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。

```

5407 \directlua{
5408   local function range(cs, ce, cc, ff)
5409     if ff or not tex.getcatcode(cs) == cc then
5410       local setcc = tex.setcatcode
5411       for c = cs, ce do setcc(c, cc) end
5412     end
5413   end
5414   range(0x3400, 0x4DB5, 11, false)
5415   \ifnum\luatexversion>64
5416     range(0x4DB5, 0x4DBF, 11, true)
5417     range(0x4E00, 0x9FCC, 11, false)
5418     range(0x9FCD, 0x9FFF, 11, true)
5419     range(0xAC00, 0xD7A3, 11, false)
5420     range(0x20000, 0x2A6D6, 11, false)
5421     range(0x2A6D7, 0x2A6FF, 11, true)
5422     range(0x2A700, 0x2B734, 11, false)
5423     range(0x2B735, 0x2B73F, 11, true)
5424     range(0x2B740, 0x2B81D, 11, false)
5425     range(0x2B81E, 0x2B81F, 11, true)
5426     range(0x2B820, 0x2CEA1, 11, false)
5427     range(0x2CEA2, 0x2FFFD, 11, true)
5428   \fi
5429 }

```

以上。

```

5430 \fi

```

G.5 完了

おしまい。

```
5431 %</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

```
5432 %<*cjkcat>
5433 \def\bxjx@pkgname{bxjscjkat}
5434 \newcount\bxjx@cnta
5435 \@onlypreamble\bxjx@tmpdo
5436 \@onlypreamble\bxjx@tmpdo@a
5437 \@onlypreamble\bxjx@tmpdo@b
```

\bxjx@engine エンジンの種別。

```
5438 \let\bxjx@engine=n
5439 \def\bxjx@tmpdo#1#2{%
5440   \edef\bxjx@tmpa{\string#1}%
5441   \edef\bxjx@tmpb{\meaning#1}%
5442   \ifx\bxjx@tmpa\bxjx@tmpb #2\fi}
5443 \bxjx@tmpdo\kanjiskip{\let\bxjx@engine=j}
5444 \bxjx@tmpdo\enablecjktoken{%
5445   \ifx\ucs\@undefined\else \ifnum\ucs"3000="3000
5446     \let\bxjx@engine=u\fi\fi}
5447 \bxjx@tmpdo\XeTeXversion{\let\bxjx@engine=x}
5448 \bxjx@tmpdo\pdftexversion{\let\bxjx@engine=p}
5449 \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。

```
5450 \def\bxjx@tmpdo#1#2{%
5451   \if#1\bxjx@engine
5452     \@ifpackageloaded{#2}{-}{%else
5453       \PackageError\bxjx@pkgname
5454         {Package '#2' must be loaded}%
5455         {Package loading is aborted.\MessageBreak\@ehc}%
5456     \endinput}
5457   \fi}
5458 \bxjx@tmpdo{p}{bxcjkatype}
5459 \bxjx@tmpdo{x}{xeCJK}
5460 \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。

```

5461 \ifx\TextOrMath\@undefined
5462 \RequirePackage{fixltx2e}
5463 \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。

```

5464 \if u\bxjx@engine
5465 \@for\bxjx@tmpa:={%
5466 0080,0100,0180,0250,02B0,0300,0500,0530,0590,0600,%
5467 0700,0750,0780,07C0,0800,0840,0860,08A0,0900,0980,%
5468 0A00,0A80,0B00,0B80,0C00,0C80,0D00,0D80,0E00,0E80,%
5469 0F00,1000,10A0,1200,1380,13A0,1400,1680,16A0,1700,%
5470 1720,1740,1760,1780,1800,18B0,1900,1950,1980,19E0,%
5471 1A00,1A20,1AB0,1B00,1B80,1BC0,1C00,1C50,1C80,1CC0,%
5472 1CD0,1D00,1D80,1DC0,1E00,1F00,2440,27C0,27F0,2800,%
5473 2A00,2C00,2C60,2C80,2D00,2D30,2D80,2DE0,2E00,4DC0,%
5474 A4D0,A500,A640,A6A0,A700,A720,A800,A830,A840,A880,%
5475 A8E0,A900,A930,A980,A9E0,AA00,AA60,AA80,AAE0,AB00,%
5476 AB30,AB70,ABCO,DB80,DC00,E000,FB00,FB50,FE00,%
5477 FE70,FFFO,%
5478 10000,10080,10100,10140,10190,101D0,10280,102A0,%
5479 102E0,10300,10330,10350,10380,103A0,10400,10450,%
5480 10480,104B0,10500,10530,10600,10800,10840,10860,%
5481 10880,108E0,10900,10920,10980,109A0,10A00,10A60,%
5482 10A80,10AC0,10B00,10B40,10B60,10B80,10C00,10C80,%
5483 10E60,11000,11080,110D0,11100,11150,11180,111E0,%
5484 11200,11280,112B0,11300,11400,11480,11580,11600,%
5485 11660,11680,11700,118A0,11A00,11A50,11AC0,11C00,%
5486 11C70,11D00,12000,12400,12480,13000,14400,16800,%
5487 16A40,16AD0,16B00,16F00,1BC00,1BCA0,1D000,1D100,%
5488 1D200,1D300,1D360,1D400,1D800,1E000,1E800,1E900,%
5489 1EE00,1F000,1F030,1F0A0,1F300,1F600,1F650,1F680,%
5490 1F700,1F780,1F800,1F900,E0000,E0100,F0000,100000,%
5491 00C0%
5492 }\do{%
5493 \@tempcnta="\bxjx@tmpa\relax
5494 \@tempcntb\@tempcnta \advance\@tempcntb\m@ne
5495 \chardef\bxjx@tmpb\kcatcode\@tempcntb
5496 \kcatcode\@tempcnta=15 \kcatcode\@tempcntb\bxjx@tmpb}
5497 \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。

```

5498 \@onlypreamble\bxjx@grkcyr@list
5499 \def\bxjx@grkcyr@list{%
5500 \do{0391}{LGR}{\textAlpha}{A}%           % GR. C. L. ALPHA
5501 \do{0392}{LGR}{\textBeta}{B}%           % GR. C. L. BETA
5502 \do{0393}{LGR}{\textGamma}{\Gamma}%     % GR. C. L. GAMMA
5503 \do{0394}{LGR}{\textDelta}{\Delta}%     % GR. C. L. DELTA
5504 \do{0395}{LGR}{\textEpsilon}{E}%       % GR. C. L. EPSILON
5505 \do{0396}{LGR}{\textZeta}{Z}%           % GR. C. L. ZETA
5506 \do{0397}{LGR}{\textEta}{H}%           % GR. C. L. ETA
5507 \do{0398}{LGR}{\textTheta}{\Theta}%     % GR. C. L. THETA
5508 \do{0399}{LGR}{\textIota}{I}%           % GR. C. L. IOTA
5509 \do{039A}{LGR}{\textKappa}{K}%          % GR. C. L. KAPPA
5510 \do{039B}{LGR}{\textLambda}{\Lambda}%    % GR. C. L. LAMDA
5511 \do{039C}{LGR}{\textMu}{M}%             % GR. C. L. MU
5512 \do{039D}{LGR}{\textNu}{N}%            % GR. C. L. NU
5513 \do{039E}{LGR}{\textXi}{\Xi}%           % GR. C. L. XI
5514 \do{039F}{LGR}{\textOmicron}{O}%       % GR. C. L. OMICRON
5515 \do{03A0}{LGR}{\textPi}{\Pi}%           % GR. C. L. PI
5516 \do{03A1}{LGR}{\textRho}{P}%           % GR. C. L. RHO
5517 \do{03A3}{LGR}{\textSigma}{\Sigma}%     % GR. C. L. SIGMA
5518 \do{03A4}{LGR}{\textTau}{T}%           % GR. C. L. TAU
5519 \do{03A5}{LGR}{\textUpsilon}{\Upsilon}% % GR. C. L. UPSILON
5520 \do{03A6}{LGR}{\textPhi}{\Phi}%         % GR. C. L. PHI
5521 \do{03A7}{LGR}{\textChi}{X}%           % GR. C. L. CHI
5522 \do{03A8}{LGR}{\textPsi}{\Psi}%         % GR. C. L. PSI
5523 \do{03A9}{LGR}{\textOmega}{\Omega}%     % GR. C. L. OMEGA
5524 \do{03B1}{LGR}{\textalpha}{\alpha}%     % GR. S. L. ALPHA
5525 \do{03B2}{LGR}{\textbeta}{\beta}%       % GR. S. L. BETA
5526 \do{03B3}{LGR}{\textgamma}{\gamma}%     % GR. S. L. GAMMA
5527 \do{03B4}{LGR}{\textdelta}{\delta}%     % GR. S. L. DELTA
5528 \do{03B5}{LGR}{\textepsilon}{\epsilon}% % GR. S. L. EPSILON
5529 \do{03B6}{LGR}{\textzeta}{\zeta}%       % GR. S. L. ZETA
5530 \do{03B7}{LGR}{\texteta}{\eta}%         % GR. S. L. ETA
5531 \do{03B8}{LGR}{\texttheta}{\theta}%     % GR. S. L. THETA

```

5532 \do{03B9}{LGR}{\textiota}{\iota}%	% GR. S. L. IOTA
5533 \do{03BA}{LGR}{\textkappa}{\kappa}%	% GR. S. L. KAPPA
5534 \do{03BB}{LGR}{\textlambda}{\lambda}%	% GR. S. L. LAMDA
5535 \do{03BC}{LGR}{\textmu}{\mu}%	% GR. S. L. MU
5536 \do{03BD}{LGR}{\textnu}{\nu}%	% GR. S. L. NU
5537 \do{03BE}{LGR}{\textxi}{\xi}%	% GR. S. L. XI
5538 \do{03BF}{LGR}{\textomicron}{\omicron}%	% GR. S. L. OMICRON
5539 \do{03C0}{LGR}{\textpi}{\pi}%	% GR. S. L. PI
5540 \do{03C1}{LGR}{\textrho}{\rho}%	% GR. S. L. RHO
5541 \do{03C2}{LGR}{\textvarsigma}{\varsigma}%	% GR. S. L. FINAL SIGMA
5542 \do{03C3}{LGR}{\textsigma}{\sigma}%	% GR. S. L. SIGMA
5543 \do{03C4}{LGR}{\texttau}{\tau}%	% GR. S. L. TAU
5544 \do{03C5}{LGR}{\textupsilon}{\upsilon}%	% GR. S. L. UPSILON
5545 \do{03C6}{LGR}{\textphi}{\phi}%	% GR. S. L. PHI
5546 \do{03C7}{LGR}{\textchi}{\chi}%	% GR. S. L. CHI
5547 \do{03C8}{LGR}{\textpsi}{\psi}%	% GR. S. L. PSI
5548 \do{03C9}{LGR}{\textomega}{\omega}%	% GR. S. L. OMEGA
5549 \do{0401}{T2A}{\CYRYO}{}	% CY. C. L. IO
5550 \do{0410}{T2A}{\CYRA}{}	% CY. C. L. A
5551 \do{0411}{T2A}{\CYRB}{}	% CY. C. L. BE
5552 \do{0412}{T2A}{\CYRV}{}	% CY. C. L. VE
5553 \do{0413}{T2A}{\CYRG}{}	% CY. C. L. GHE
5554 \do{0414}{T2A}{\CYRD}{}	% CY. C. L. DE
5555 \do{0415}{T2A}{\CYRE}{}	% CY. C. L. IE
5556 \do{0416}{T2A}{\CYRZH}{}	% CY. C. L. ZHE
5557 \do{0417}{T2A}{\CYRZ}{}	% CY. C. L. ZE
5558 \do{0418}{T2A}{\CYRI}{}	% CY. C. L. I
5559 \do{0419}{T2A}{\CYRISHRT}{}	% CY. C. L. SHORT I
5560 \do{041A}{T2A}{\CYRK}{}	% CY. C. L. KA
5561 \do{041B}{T2A}{\CYRL}{}	% CY. C. L. EL
5562 \do{041C}{T2A}{\CYRM}{}	% CY. C. L. EM
5563 \do{041D}{T2A}{\CYRN}{}	% CY. C. L. EN
5564 \do{041E}{T2A}{\CYRO}{}	% CY. C. L. O
5565 \do{041F}{T2A}{\CYRP}{}	% CY. C. L. PE
5566 \do{0420}{T2A}{\CYRR}{}	% CY. C. L. ER
5567 \do{0421}{T2A}{\CYRS}{}	% CY. C. L. ES
5568 \do{0422}{T2A}{\CYRT}{}	% CY. C. L. TE
5569 \do{0423}{T2A}{\CYRU}{}	% CY. C. L. U
5570 \do{0424}{T2A}{\CYRF}{}	% CY. C. L. EF
5571 \do{0425}{T2A}{\CYRH}{}	% CY. C. L. HA
5572 \do{0426}{T2A}{\CYRC}{}	% CY. C. L. TSE
5573 \do{0427}{T2A}{\CYRCH}{}	% CY. C. L. CHE
5574 \do{0428}{T2A}{\CYRSH}{}	% CY. C. L. SHA
5575 \do{0429}{T2A}{\CYRSHCH}{}	% CY. C. L. SHCHA
5576 \do{042A}{T2A}{\CYRHRDSN}{}	% CY. C. L. HARD SIGN
5577 \do{042B}{T2A}{\CYRERY}{}	% CY. C. L. YERU
5578 \do{042C}{T2A}{\CYRSFTSN}{}	% CY. C. L. SOFT SIGN
5579 \do{042D}{T2A}{\CYREREV}{}	% CY. C. L. E
5580 \do{042E}{T2A}{\CYRYU}{}	% CY. C. L. YU


```

5581 \do{042F}{T2A}{\CYRYA}{}% % CY. C. L. YA
5582 \do{0430}{T2A}{\cyra}{}% % CY. S. L. A
5583 \do{0431}{T2A}{\cyrb}{}% % CY. S. L. BE
5584 \do{0432}{T2A}{\cyrv}{}% % CY. S. L. VE
5585 \do{0433}{T2A}{\cyrg}{}% % CY. S. L. GHE
5586 \do{0434}{T2A}{\cyrd}{}% % CY. S. L. DE
5587 \do{0435}{T2A}{\cyre}{}% % CY. S. L. IE
5588 \do{0436}{T2A}{\cyrrh}{}% % CY. S. L. ZHE
5589 \do{0437}{T2A}{\cyrz}{}% % CY. S. L. ZE
5590 \do{0438}{T2A}{\cyri}{}% % CY. S. L. I
5591 \do{0439}{T2A}{\cyrishrt}{}% % CY. S. L. SHORT I
5592 \do{043A}{T2A}{\cyrk}{}% % CY. S. L. KA
5593 \do{043B}{T2A}{\cyr1}{}% % CY. S. L. EL
5594 \do{043C}{T2A}{\cyrm}{}% % CY. S. L. EM
5595 \do{043D}{T2A}{\cyrn}{}% % CY. S. L. EN
5596 \do{043E}{T2A}{\cyro}{}% % CY. S. L. O
5597 \do{043F}{T2A}{\cyrp}{}% % CY. S. L. PE
5598 \do{0440}{T2A}{\cyrr}{}% % CY. S. L. ER
5599 \do{0441}{T2A}{\cyrs}{}% % CY. S. L. ES
5600 \do{0442}{T2A}{\cyrt}{}% % CY. S. L. TE
5601 \do{0443}{T2A}{\cyru}{}% % CY. S. L. U
5602 \do{0444}{T2A}{\cyrf}{}% % CY. S. L. EF
5603 \do{0445}{T2A}{\cyrh}{}% % CY. S. L. HA
5604 \do{0446}{T2A}{\cyrc}{}% % CY. S. L. TSE
5605 \do{0447}{T2A}{\cyrch}{}% % CY. S. L. CHE
5606 \do{0448}{T2A}{\cyrrh}{}% % CY. S. L. SHA
5607 \do{0449}{T2A}{\cyrrhch}{}% % CY. S. L. SHCHA
5608 \do{044A}{T2A}{\cyrrhdsn}{}% % CY. S. L. HARD SIGN
5609 \do{044B}{T2A}{\cyrrery}{}% % CY. S. L. YERU
5610 \do{044C}{T2A}{\cyrrsftsn}{}% % CY. S. L. SOFT SIGN
5611 \do{044D}{T2A}{\cyrrerev}{}% % CY. S. L. E
5612 \do{044E}{T2A}{\cyryu}{}% % CY. S. L. YU
5613 \do{044F}{T2A}{\cyrya}{}% % CY. S. L. YA
5614 \do{0451}{T2A}{\cyryo}{}% % CY. S. L. IO
5615 \do{00A7}{TS1}{\textsection}{\mathsection}% SECTION SYMBOL
5616 \do{00A8}{TS1}{\textasciidieresis}{}% % DIAERESIS
5617 \do{00B0}{TS1}{\textdegree}{\mathdegree}% % DEGREE SIGN
5618 \do{00B1}{TS1}{\textpm}{\pm}% % PLUS-MINUS SIGN
5619 \do{00B4}{TS1}{\textasciicircum}{}% % ACUTE ACCENT
5620 \do{00B6}{TS1}{\textparagraph}{\mathparagraph}% PILCROW SIGN
5621 \do{00D7}{TS1}{\texttimes}{\times}% % MULTIPLICATION SIGN
5622 \do{00F7}{TS1}{\textdiv}{\div}% % DIVISION SIGN
5623 }

```

`\mathdegree` 面倒なので補っておく。

```
5624 \providecommand*{\mathdegree}{\textcircled{0}}
```

`\ifbxjx@gcc@cjkk` [スイッチ] 「特定 CJK 曖昧文字」を和文扱いにするか。

```
5625 \newif\ifbxjx@gcc@cjkk
```

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

```
5626 \newcommand*\greekasCJK{%
5627   \bxjx@gcc@cjctrue}
```

`\nogreekasCJK` [公開命令] 「特定 CJK 曖昧文字」を欧文扱いにする。

```
5628 \newcommand*\nogreekasCJK{%
5629   \bxjx@gcc@cj>false}
```

`\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。

```
5630 \def\bxjx@tmpdo#1\relax{%
5631   \def\bxjx@fake@grk##1##2{%
5632     \expandafter\bxjx@fake@grk@a\meaning##2#1\@nil{##1}{##2}}%
5633   \def\bxjx@fake@grk@a###1##2\@nil##3##4{%
5634     \ifx\##1\%
5635       \bxjx@cmta##4\divide\bxjx@cmta\@cclvi
5636       \multiply\bxjx@cmta\@cclvi \advance\bxjx@cmta`##3\relax
5637       \mathchar\bxjx@cmta
5638     \else ##3\fi}
5639 }\expandafter\bxjx@tmpdo\string\mathchar\relax
```

■pdfTeX・upTeXの場合

```
5640 \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。

```
5641 \@ifpackageloaded{inputenc}{\fi}{%else
5642   \RequirePackage[utf8]{inputenc}}
5643 \def\bxjx@tmpa{utf8}
5644 \ifx\bxjx@tmpa\inputencdoingname
5645   \PackageWarningNoLine\bxjx@pkgname
5646     {Input encoding changed to utf8}%
5647   \inputencoding{utf8}%
5648 \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。

```

5649 \if u\bxjx@engine
5650 \kcatcode"0370=15
5651 \kcatcode"0400=15
5652 \kcatcode"0500=15
5653 \fi

```

各文字について `\DeclareUnicodeCharacter` を実行する。

```

5654 \def\bxjx@tmpdo#1{%
5655   \@tempcnta=#1\relax
5656   \expandafter\bxjx@tmpdo@a\csname bxjx@KC/\the\@tempcnta\endcsname{#1}}
5657 \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。

```

5658   \ifx\#5\%
5659     \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。

```

5660   \else\ifcat A\noexpand#5%
5661     \edef\bxjx@tmpa{\noexpand\bxjx@fake@grk{#5}%
5662       {\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。

```

5663   \else \def\bxjx@tmpa{#5}%
5664   \fi\fi
5665   \def\bxjx@tmpb{\bxjx@tmpdo@b{#1}{#2}{#3}{#4}}%
5666   \expandafter\bxjx@tmpb\expandafter{\bxjx@tmpa}}

```

以降はエンジン種別で分岐する。up \TeX の場合。

```

5667 \if u\bxjx@engine
5668 \def\bxjx@tmpdo@b#1#2#3#4#5{%

```

引数 = `\[bxjx@KC/⟨符号値⟩]{⟨符号値⟩}{⟨fontenc⟩}{⟨LICR⟩}{⟨数式中の動作⟩}`

当該の Unicode 文字の動作は「テキストでは ⟨LICR⟩、数式では ⟨数式中の動作⟩」となる。LICR は現在エンコーディングで有効な定義がある場合はそれが実行されるはずである。（つまり、現在が LGR である場合はギリシャ文字は常に欧文扱いになる。）それ以外の場合は LICR を `\bxjx@ja@or@not` に帰着させる。この際に、和文用の定義として当該の `kchardef` を使用し、その制御綴として `\[bxjx@KC/...]` を流用している。

```

5669   \kchardef#1=\@tempcnta
5670   \DeclareTextCommandDefault{#4}{\bxjx@ja@or@not{#1}{#3}{#4}}%
5671   \DeclareUnicodeCharacter{#2}{\TextOrMath{#4}{#5}}

```

pdf \TeX の場合も処理はほとんど同じ。ただし、和文用の定義として `\UTF{⟨符号値⟩}` を使う（`\UTF` は `bxckjatype` の命令）。`\[bxjx@KC/...]` は使わないが定義済にする必要がある。

```

5672 \else\if p\bxjx@engine
5673 \def\bxjx@tmpdo@b#1#2#3#4#5{%
5674   \mathchardef#1=\@tempcnta
5675   \DeclareTextCommandDefault{#4}{\bxjx@ja@or@not{\UTF{#2}}{#3}{#4}}%

```

```
5676 \DeclareUnicodeCharacter{#2}{\TextOrMath{#4}{#5}}
5677 \fi\fi
```

以上の処理を「特定 CJK 曖昧文字」の各々に適用する。

```
5678 \let\do\bxjx@tmpdo \bxjx@grkcyr@list
```

`\bxjx@DeclareUnicodeCharacter` `\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。

```
5679 \@onlypreamble\bxjx@org@DeclareUnicodeCharacter
5680 \let\bxjx@org@DeclareUnicodeCharacter\DeclareUnicodeCharacter
5681 \@onlypreamble\bxjx@DeclareUnicodeCharacter
5682 \def\bxjx@DeclareUnicodeCharacter#1#2{%
5683   \count@=#1\relax
5684   \expandafter\ifx\csname bxjx@KC/\the\count@\endcsname\relax
5685     \bxjx@org@DeclareUnicodeCharacter{#1}{#2}%
5686   \else
5687     \wlog{ \space\space skipped defining Unicode char U+#1}%
5688   \fi}
```

`\bxjx@ja@or@not` `\bxjx@ja@or@not{<和文用定義>}{<対象 fontenc>}{<LICR>}` : `\[no]greekasCJK` の状態に応じて和文または欧文で文字を出力する。

```
5689 \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。

```
5690 \ifbxjx@gcc@CJK #1%
```

`\nogreekasCJK` の場合は、対象のエンコーディングに変更して LICR を実行するが、そのエンコーディングが未定義の場合は (フォールバックとして) 和文用定義を使う。

```
5691 \else\expandafter\ifx\csname T@#2\endcsname\relax #1%
5692 \else \UseTextSymbol{#2}{#3}%
5693 \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。

```
5694 \begingroup
5695 \toks@\expandafter{\DeclareFontEncoding@{#1}{#2}{#3}}
5696 \xdef\next{\def\noexpand\DeclareFontEncoding@##1##2##3{%
5697   \noexpand\bxjx@swap@DUC@cmd
5698   \the\toks@
5699   \noexpand\bxjx@swap@DUC@cmd}}
5700 \endgroup\next
5701 \def\bxjx@swap@DUC@cmd{%
5702   \let\bxjx@tmpa\DeclareUnicodeCharacter
5703   \let\DeclareUnicodeCharacter\bxjx@DeclareUnicodeCharacter
5704   \let\bxjx@DeclareUnicodeCharacter\bxjx@tmpa
5705   \let\bxjx@tmpa\relax}
```

以上。

■Xe_{La}TeX・LuaTeX の場合

```
5706 \else\ifnum0\if x\bxjx@engine1\fi\if l\bxjx@engine1\fi>0
```

各文字について、数式中の動作を定義する。

```
5707 \def\bxjx@tmpdo#1{%
5708   \bxjx@ccta="#1\relax
5709   \beginngroup
5710     \lccode`~=\bxjx@ccta
5711   \lowercase{\endgroup
5712     \bxjx@tmpdo@a{-}}{#1}}
5713 \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、up_{La}TeX 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。

```
5714   \ifx\\#5\\\let\bxjx@tmpa\relax
5715   \else\ifcat A\noexpand#5%
5716     \edef\bxjx@tmpa{\noexpand\bxjx@fake@grk{#5}%
5717       {\ifnum\uccode`#5=`#5\noexpand\Pi\else\noexpand\pi\fi}}%
5718   \else \def\bxjx@tmpa{#5}%
5719   \fi\fi
5720   \ifx\bxjx@tmpa\relax\else
5721     \mathcode\bxjx@ccta"8000 \let#1\bxjx@tmpa
5722   \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。

```
5723 \mathchardef\bxjx@tmpa="119
5724 \ifx\bxjx@tmpa\pi \let\do\bxjx@tmpdo \bxjx@grkcyr@list \fi
```

次に、テキストにおいて「特定 CJK 曖昧文字」の扱いが `\[no]greekasCJK` で切り替わるようにする。

LuaTeX の場合は、 LuaTeX-ja の `jacharrange` の設定を変更する。

※ “範囲 2” がギリシャ・キリル文字、“範囲 8” が Latin-1 の記号。

```
5725 \if l\bxjx@engine
5726   \protected\def\greekasCJK{%
5727     \bxjx@gcc@cjktrue
5728     \ltjsetparameter{jacharrange={+2, +8}}
5729   \protected\def\nogreekasCJK{%
5730     \bxjx@gcc@cjkfalse
5731     \ltjsetparameter{jacharrange={-2, -8}}
5732 \fi
```

X_{La}TeX の場合、xeCJK は X_{La}TeX の文字クラス定義を参照しているので、対象文字の文字クラスを変更する。

```
5733 \if x\bxjx@engine
5734   \let\bxjx@gcc@cjk@list\@empty
5735   \def\do#1#2#3#4{%
5736     \edef\bxjx@gcc@cjk@list{\bxjx@gcc@cjk@list
```

```

5737     \noexpand\XeTeXcharclass"#1\bxjx@cnta}}
5738 \bxjx@grkcyr@list
5739 \protected\def\greekasCJK{%
5740     \bxjx@gcc@cjctrue
5741     \bxjx@cnta=@ne \bxjx@gcc@cjkl@list}
5742 \protected\def\nogreekasCJK{%
5743     \bxjx@gcc@cjcfalse
5744     \bxjx@cnta=@z@ \bxjx@gcc@cjkl@list}
5745 \fi

```

以上。

```
5746 \fi\fi
```

H.4 初期設定

「特定 CJK 曖昧文字」を欧文扱いにする。

```
5747 \nogreekasCJK
```

H.5 完了

おしまい。

```
5748 %</cjkcat>
```

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

```

5749 %<*ancpandoc>
5750 %% このファイルは日本語文字を含みます。
5751 \def\bxjsp@pkgname{bxjspandoc}

```

\bxjsp@engine エンジンの種別。

```

5752 \let\bxjsp@engine=n
5753 \@onlypreamble\bxjsp@do
5754 \def\bxjsp@do#1#2{%
5755     \edef\bxjsp@tmpa{\string#1}%
5756     \edef\bxjsp@tmpb{\meaning#1}%
5757     \ifx\bxjsp@tmpa\bxjsp@tmpb #2\fi}
5758 \bxjsp@do\kanjiskip{\let\bxjsp@engine=j}
5759 \bxjsp@do\XeTeXversion{\let\bxjsp@engine=x}

```

```
5760 \bxjsp@do\pdftexversion{\let\bxjsp@engine=p}
5761 \bxjsp@do\luatexversion{\let\bxjsp@engine=l}
```

`\bxjsp@begin@document@hook` 文書本体開始時フック。

```
5762 \@onlypreamble\bxjsp@begin@document@hook
5763 \let\bxjsp@begin@document@hook\@empty
5764 \AtBeginDocument{\bxjsp@begin@document@hook}
```

`\ifbxjsp@babel@used` [スイッチ] Babel が読み込まれたか。

```
5765 \newif\ifbxjsp@babel@used
5766 \g@addto@macro\bxjsp@begin@document@hook{%
5767   \ifpackage@loaded{babel}{\bxjsp@babel@usedtrue}{}}
```

1.2 パッケージオプション

`english` オプションが指定されている場合、`\ldots` の調整を抑制する。
 ※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制の対象になる。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。

```
5768 \newif\ifbxjsp@english
5769 \DeclareOption{english}{\bxjsp@englishtrue}
```

オプション定義はおしまい。

```
5770 \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。

```
5771 \@onlypreamble\pandocSkipLoadFile
5772 \newcommand*\pandocSkipLoadFile[1]{%
5773   \expandafter\bxjsp@skip@load@file@a\csname ver@#1\endcsname{#1}}
5774 \def\bxjsp@skip@load@file@a#1#2{%
5775   \ifx#1\relax
5776     \def#1{2001/01/01}%
5777     \PackageInfo\bxjsp@pkgname
5778     {File '#2' marked as loaded\@gobble}%
5779   \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。

```
5780 \@onlypreamble\pandocSkipLoadPackage
5781 \newcommand*\pandocSkipLoadPackage[1]{%
5782   \pandocSkipLoadFile{#1.sty}}
```

1.4 fixltx2e パッケージ

テンプレートでは `fixltx2e` パッケージを読み込むが、最近（2015 年版以降）の \LaTeX ではこれで警告が出る。これを抑止する。

\LaTeX カーネルが新しい場合は `fixltx2e` を読込済にする。

```
5783 \ifx\@IncludeInRelease\undefined\else
5784   \pandocSkipLoadPackage{fixltx2e}
5785 \fi
```

1.5 cmap パッケージ

エンジンが $(u)\text{\pLaTeX}$ のときに `cmap` パッケージが読み込まれるのを阻止する。（実際は警告が出るだけで無害であるが。）

```
5786 \if j\bxjsp@engine
5787   \pandocSkipLoadPackage{cmap}
5788 \fi
```

1.6 microtype パッケージ

警告が多すぎなので消す。

```
5789 \if j\bxjsp@engine \else
5790   \PassOptionsToPackage{verbose=silent}{microtype}
5791 \fi
```

エンジンが $(u)\text{\pLaTeX}$ のときに `microtype` パッケージが読み込まれるのを阻止し、さらにテンプレートで使われている命令を通すためにダミーの定義を行う。

※昔は `standard` ドライバでこの処理を行っていたが、元来は Pandoc 用の処理なので、1.5 版で `pandoc` に移動。

```
5792 \if j\bxjsp@engine
5793   \pandocSkipLoadPackage{microtype}
5794   \newcommand*\UseMicrotypeSet[2][]{\}
5795 \fi
```

1.7 Unicode 文字変換対策

Pandoc で \LaTeX 形式に書き出す場合は、元データ中の一部の Unicode 文字を「 \LaTeX の表記」に置き換える。その中には日本語文書で問題になるものが含まれる。

```
…→\ldots{ } ‘→` ’→! “→` ”→! !
```

日本語 \LaTeX では「 \LaTeX の表記」は欧文扱い、Unicode 文字は和文扱いとして使い分ける習慣があるので、このような置換が行われるのは好ましくない。

これらの置換のうち、後の 4 つは Pandoc の `--no-tex-ligatures` オプションを指定すれば抑止できるが、「…」の置換を抑止する機能はないようである。そこで、「`\ldots`」を『…』

に戻す」という処置を行う。

`\pandocLdots` Pandoc 用の `\ldots` の実装。非数式である場合は代わりに `…` を実行する。

※以前は「Pandoc が必ず `\ldots{}` の形で書き出す」ことを利用して後続に `{}` があるかで「元が `…` であるか」を判断していた。ところが、Pandoc 2.7 版で `{}` を必ずしも付けなくなったため、1.9f 版で非数式の `\ldots` を全て `…` に戻す動作に変更した。

```
5796 \DeclareRobustCommand{\pandocLdots}{%
5797   \let\bxjsp@do\bxjsp@ja@ellipsis
5798   \ifmmode \let\bxjsp@do\bxjsp@org@ldots
5799   \else\ifbxjsp@babel@used
5800     \expandafter\ifx\csname bxjsp@ld/\language\endcsname\relax
5801     \let\bxjsp@do\bxjsp@org@ldots \fi
5802   \fi\fi \bxjsp@do}
5803 \@namedef{bxjsp@ld/japanese}{1}
5804 \def\bxjsp@ja@ellipsis{…}
5805 \let\bxjsp@org@ldots\ldots
```

`\ldots` の実装を `\pandocLdots` に置き換える。

```
5806 \g@addto@macro\bxjsp@begin@document@hook{%
5807   \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。

```
5808 \long\def\bxjsp@tmpa{\ldots}%
5809 \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。

```
5810   \ifnum0\ifbxjsp@english\ifbxjsp@babel@used\else1\fi\fi=0
5811   \let\ldots\pandocLdots
5812   \fi
5813 \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_YTeX・LuaTeX は英字と見なす（または将来その可能性がある）」文字で、特に日本語文書に現れるものについて、非英字扱いにしておく。

※ Pandoc は「Unicode 7.0 で GC が Letter」な文字を英字と判定している。

```
5814 \chardef\bxjsp@cc@other=12
5815 \@onlypreamble\bxjsp@makeother@range
5816 \def\bxjsp@makeother@range#1#2{%
5817   \@tempcnta"#1\relax \@tempcntb"#2\relax
5818   \loop\ifnum\@tempcnta<\@tempcntb
5819     \catcode\@tempcnta\bxjsp@cc@other
5820     \advance\@tempcnta\@ne
5821   \repeat}
5822 \ifnum0\if x\bxjsp@engine1\fi\if 1\bxjsp@engine1\fi>0
5823   \catcode"1F23B=\bxjsp@cc@other
5824   \bxjsp@makeother@range{9FCD}{A000}
```

```
5825 \bxjsp@makeother@range{1B002}{1B170}  
5826 \bxjsp@makeother@range{2B820}{2EBF0}  
5827 \fi
```

1.8 PandoLa モジュール

インストール済であれば読み込む。

```
5828 \IfFileExists{bxpandola.sty}{%  
5829 \RequirePackage{bxpandola}\relax  
5830 \PackageInfo{bxjsp@pkgname  
5831 {PandoLa module is loaded\@gobble}  
5832 }{}
```

1.9 完了

おしまい。

```
5833 %</ancpandoc>
```

補助パッケージ実装はここまで。

```
5834 %</anc>
```