

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”)

v2.8a [2023/06/20]

この文書はソースコード説明書です。一般の文書作成者向けの解説については、ユーザマニュアル `bxjscls-manual.pdf` を参照してください。

目次

1	はじめに	4
2	オプション	12
3	和文フォントの変更	42
4	フォントサイズ	52
5	レイアウト	58
5.1	ページレイアウト	59
6	改ページ (日本語 TeX 開発コミュニティ版のみ)	74
7	ページスタイル	75
8	文書のマークアップ	79
8.1	表題	79
8.2	章・節	84
8.3	リスト環境	96
8.4	パラメータの設定	104
8.5	フロート	105
8.6	キャプション	106
9	フォントコマンド	108

10	相互参照	110
10.1	目次の類	110
10.2	参考文献	115
10.3	索引	117
10.4	脚注	118
11	段落の頭へのグルー挿入禁止	121
12	いろいろなロゴ	125
13	amsmath との衝突の回避	126
14	初期設定	126
15	実験的コード	131
付録 A	和文ドライバの仕様 	132
付録 B	和文ドライバ：minimal 	133
B.1	補助マクロ	133
B.2	(u)pTeX 用の設定	136
B.3	pdfTeX 用の処理	140
B.4	X _Y TeX 用の処理	141
B.5	後処理（エンジン共通）	142
付録 C	和文ドライバ：standard 	145
C.1	準備	145
C.2	和文ドライバパラメタ	145
C.3	共通処理 (1)	146
C.4	pTeX 用設定	154
C.5	pdfTeX 用設定：CJK + bxcjkjatype	159
C.6	X _Y TeX 用設定：xeCJK + zxjatype	161
C.7	LuaTeX 用設定：LuaTeX-ja	163
C.8	共通処理 (2)	167
付録 D	和文ドライバ：modern 	168
D.1	フォント設定	168
D.2	fixltx2e 読込	169
D.3	和文カテゴリコード	169
D.4	完了	169
付録 E	和文ドライバ：pandoc 	169
E.1	準備	169

E.2	和文ドライバパラメタ	170
E.3	dupload システム	171
E.4	lang 変数	172
E.5	geometry 変数	176
E.6	CJKmainfont 変数	176
E.7	Option clash 対策	176
E.8	レイアウト上書き禁止	176
E.9	paragraph のマーク	177
E.10	全角空白文字	178
E.11	hyperref 対策	179
E.12	Pandoc 要素に対する和文用の補正	179
E.13	ifPDFTeX スイッチ	180
E.14	完了	181
付録 F	補助パッケージ一覧	181
付録 G	補助パッケージ：bxjscompat	182
G.1	準備	182
G.2	X _g TeX 部分	182
G.3	LuaTeX 部分	184
G.4	完了	185
付録 H	補助パッケージ：bxjscjcat	185
H.1	準備	185
H.2	和文カテゴリコードの設定	186
H.3	ギリシャ・キリル文字の扱い	187
H.4	初期設定	194
H.5	完了	194
付録 I	補助パッケージ：bxjspandoc	194
I.1	準備	194
I.2	パッケージオプション	195
I.3	パッケージ読込の阻止	195
I.4	fixltx2e パッケージ	196
I.5	cmap パッケージ	196
I.6	microtype パッケージ	196
I.7	Unicode 文字変換対策	197
I.8	PandoLa モジュール	198
I.9	完了	198

1 はじめに

この文書は「BXJS ドキュメントクラス」の DocStrip 形式のソースである。インストール時のモジュール指定は以下のようである。

<code><article></code>	<code>bxjsarticle.cls</code>	短いレポート（章なし）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 “jspl”・“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。

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

これは LaTeX3 Project の `classes.dtx` と株式会社アスキーの `jclasses.dtx` に基づいてもともと奥村晴彦により作成されたものです。現在は日本語 TeX 開発コミュニティにより GitHub で管理されています。

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

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

[2016-07-13] 日本語 TeX 開発コミュニティによる管理に移行しました。

[2009-02-22] 田中琢爾氏による upLaTeX 対応パッチを取り込みました。

ここでは次のドキュメントクラス（スタイルファイル）を作ります。

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

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

以下では実際のコードに即して説明します。

`minijs` は, `jsclasses` に似た設定を行うパッケージです。

```

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

```

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

```

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

```

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

```

16 %</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 ($\mathrm{T}_{\mathrm{E}}\mathrm{X}$ の種類) の種別： $j = \mathrm{pT}_{\mathrm{E}}\mathrm{X}$ 系、 $x = \mathrm{X}_{\mathrm{q}}\mathrm{T}_{\mathrm{E}}\mathrm{X}$ 、 $p = \mathrm{pdfT}_{\mathrm{E}}\mathrm{X}$ (含 DVI モード)、 $l = \mathrm{LuaT}_{\mathrm{E}}\mathrm{X}$ 、 $J = \mathrm{NTT}\,j\mathrm{T}_{\mathrm{E}}\mathrm{X}$ 、 $0 = \mathrm{Omega}$ 系、 $n =$ 以上の何れでもない。

```

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}

```

非サポートのエンジンの場合は強制終了させる。

※ $\mathrm{NTT}\,j\mathrm{T}_{\mathrm{E}}\mathrm{X}$ と Omega 系。

```

41 \let\bxjs@tmpa\relax
42 \ifx J\jsEngine \def\bxjs@tmpa{\mathrm{NTT}\,j\mathrm{TeX}}\fi
43 \ifx 0\jsEngine \def\bxjs@tmpa{\mathrm{Omega}}\fi
44 \ifx\bxjs@tmpa\relax \expandafter\@gobble
45 \else
46   \ClassError\bxjs@clsname
47   {The engine in use (\bxjs@tmpa) is not supported}
48   {It's a fatal error. I'll quit right now.}
49   \expandafter\@firstofone
50 \fi{\endinput\@@end}

```

現状での処理系バージョン要件は以下の通りである ($\mathrm{X}_{\mathrm{q}}\mathrm{T}_{\mathrm{E}}\mathrm{X}$ のみの設定で、しかも現実離れしている)。

- $\mathrm{X}_{\mathrm{q}}\mathrm{T}_{\mathrm{E}}\mathrm{X}$ ： 0.997 版 (2007 年) 以上

TODO: 以下で 3.0 版での予定について述べておく。

3.0 版での処理系バージョン要件は以下の通りである。

- $\mathrm{T}_{\mathrm{E}}\mathrm{X}$ ： 3.0 版 [1990/03] 以上
- $\mathrm{pT}_{\mathrm{E}}\mathrm{X}$ ： 2.0 版 [1995/03] 以上
- $\mathrm{upT}_{\mathrm{E}}\mathrm{X}$ ： 0.10 版 [2007/07] 以上
- $\mathrm{pdfT}_{\mathrm{E}}\mathrm{X}$ ： 1.40 版 [2007/01] 以上
- $\mathrm{LuaT}_{\mathrm{E}}\mathrm{X}$ ： 0.60 版 [2010/04] 以上

- Xe_ƒTeX: 0.9994 版 [2009/06] 以上

※標準和文ドライバについてはまた別に要件を定める。

TeX 処理系のバージョンがサポート対象であるかを検査する。

```
51 \@tempswatrue
52 \if x\jsEngine
53   \ifdim\the\XeTeXversion\XeTeXrevision\p@<0.997\p@
54     \@tempswafalse \fi
55 \fi
```

非サポートのバージョン場合は強制終了させる。

```
56 \if@tempswa \expandafter\@gobble
57 \else
58   \ClassError\bxjs@clsname
59   {The engine in use is all too old}
60   {It's a fatal error. I'll quit right now.}
61   \expandafter\@firstofone
62 \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。

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

\ifjsWithupTeX [スイッチ] エンジンが (内部漢字コードが Unicode の) upTeX であるか。

```
73 \newif\ifjsWithupTeX
74 \ifx\ucs\@undefined\else \ifnum\ucs"3000="3000
75   \jsWithupTeXtrue
76 \fi\fi
77 \let\if@jsc@uplatex\ifjsWithupTeX
```

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

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

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

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

\ifjsInPdfMode [スイッチ] pdfTeX / LuaTeX が PDF モードで動作しているか。

```
82 \newif\ifjsInPdfMode
```

```

83 \@nameuse{jsInPdfMode\ifnum0%
84 \ifx\pdfoutput\@undefined\else\the\pdfoutput\fi
85 \ifx\outputmode\@undefined\else\the\outputmode\fi
86 >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。

```

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

```

`\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 年 1 月以降の \LaTeX カーネルにおいて「Unicode を表す \LaTeX 公式のフォントエンコーディング」である “TU” が導入され、これ以降の \LaTeX を \XeTeX または \LuaTeX で動かしている場合は、既定のエンコーディングが TU になる。それ以外の場合は、既定のエンコーディングは OT1 である。

```

89 \newif\ifbxjs@TUenc
90 \def\bxjs@tmpa{TU}\edef\bxjs@tmpb{\f@encoding}
91 \ifx\bxjs@tmpa\bxjs@tmpb
92 \bxjs@TUenctrue
93 \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。

```

94 \newif\ifbxjs@old@hook@system
95 \@ifl@t@r\fmtversion{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。

```
96 \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。

```
97 \RequirePackage{keyval}
```

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

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

```

98 \IfFileExists{iftex.sty}{%
99 \RequirePackage{iftex}
100 }{}
101 \begingroup\expandafter\endgroup
102 \expandafter\ifx\csname ifpdf\endcsname\@undefined\else
103 \expandafter\let\csname ifjsInPdfMode\expandafter\endcsname
104 \csname ifpdf\endcsname
105 \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。)

```
106 \def\jsAtEndOfClass{%
107   \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。

```
108 \IfFileExists{bxjscompat.sty}{%
109   \RequirePackage{bxjscompat}%
110 }{}
```

■BXJS クラス特有の設定

LuaTeX の場合、本クラス用の Lua モジュールを用意する。

```
111 \ifx l\jsEngine
112   \directlua{ bxjs = {} }
113 \fi
```

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

```
114 \ifjsWithTeX \let\bxjs@protected\protected
115 \else \let\bxjs@protected\@empty
116 \fi
```

`\bxjs@robust@def` 無引数の頑強な命令を定義する。

```
117 \ifjsWithTeX
118   \def\bxjs@robust@def{\protected\def}
119 \else
120   \def\bxjs@robust@def{\DeclareRobustCommand*}
121 \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。

```
122 \@ifl@t@r\fmtversion{2021/11/15}{%
123   \def\bxjs@CGHN#1{#1}%
124 }{%else
125   \def\bxjs@CGHN#1{\bxjs@CGHN@a#1//}%
126   \def\bxjs@CGHN@a#1/#2/#3/{#1/#3/#2}}
```

`\bxjs@cond` `\bxjs@cond\ifXXX……\fi{〈真〉}{〈偽〉}`

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

```
127 \@gobbletwo\if\if \def\bxjs@cond#1\fi{%
128   #1\expandafter\@firstoftwo
129   \else\expandafter\@secondoftwo
130   \fi}
```

`\bxjs@cslet \bxjs@cslet{<名前 1>}\制御綴：`

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

`\bxjs@csletcs \bxjs@csletcs{<名前 1>}{<名前 2>}：`

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

`\bxjs@catopt \bxjs@catopt{<文字列 1>}{<文字列 2>}：` 2つの文字列を , で繋いだ文字列。ただし片方が空の場合は , を入れない。完全展開可能。

```
135 \def\bxjs@catopt#1#2{%
136   #1\if\relax#1\relax\else\if\relax#2\relax\else,\fi\fi#2}
```

`\bxjs@ifplus \@ifstar` の + 版。

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

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

```
138 \def\bxjs@trim#1{\expandafter\bxjs@trim@a#1\@nil#1}
139 \def\bxjs@trim@a{\futurelet\bxjs@tmpb\bxjs@trim@b}
140 \def\bxjs@trim@b{\bxjs@cond\ifx\bxjs@tmpb\@sptoken\fi
141   {\bxjs@trim@c\bxjs@trim@a}{\bxjs@trim@d}}
142 \def\bxjs@trim@c#1 {#1}
143 \def\bxjs@trim@d#1\@nil{\bxjs@trim@e#1\@nil: \@nil\@nnil}
144 \def\bxjs@trim@e#1 \@nil#2\@nnil{\bxjs@cond\ifx\@nnil#2\@nnil\fi
145   {\bxjs@trim@f#1\@nnil}{\bxjs@trim@e#1\@nil: \@nil\@nnil}}
146 \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。

```
147 \def\bxjs@set@array@from@clist#1#2{%
148   \@tempcnta\z@
149   \@for\bxjs@tmpa:=\@empty#2\do{%
150     \bxjs@trim\bxjs@tmpa \bxjs@cslet{#1/\the\@tempcnta}\bxjs@tmpa
151     \advance\@tempcnta\@ne}
152   \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。

```
153 \let\c@bxjs@tempcnta\@tempcnta
154 \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）。

```
155 \def\jsSetQHLLength#1#2{%
156   \begingroup
```

```

157 \bxjs@parse@qh{#2}%
158 \ifx\bxjs@tmpb\relax
159 \setlength\@tempdima{#2}%
160 \xdef\bxjs@g@tmpa{\the\@tempdima}%
161 \else \global\let\bxjs@g@tmpa\bxjs@tmpb
162 \fi
163 \endgroup
164 #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。

```

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

```

今の段階では Q/H だけが使用可能。

```

190 \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。

```

191 \newif\ifbxjs@after@preamble

```

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

```

192 \@onlypreamble\bxjs@begin@document@hook

```

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

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

```
195 \@onlypreamble\bxjs@post@option@hook
196 \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。

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

一時的な手続き用の制御綴。

```
199 \@onlypreamble\bxjs@tmpdo
200 \@onlypreamble\bxjs@tmpdo@a
201 \@onlypreamble\bxjs@tmpdo@b
202 \@onlypreamble\bxjs@tmpdo@c
203 \@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。

```
204 \bxjs@robust@def\jsInhibitGlue{%
205   \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。

```
206 \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。

```
207 \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。

```
208 %<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。

```
209 %<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。

```
210 %<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。

```
211 \newif\if@enablejfam \@enablejfamfalse
```

以下で各オプションを宣言します。

■用紙サイズ JIS や ISO の A0 判は面積 1 m^2 、縦横比 $1:\sqrt{2}$ の長方形の辺の長さを mm 単位に切り捨てたものです。これを基準として順に半截しては mm 単位に切り捨てたものが A1, A2, …です。

B 判は JIS と ISO で定義が異なります。JIS では B0 判の面積が 1.5 m^2 ですが、ISO では B1 判の辺の長さが A0 判と A1 判の辺の長さの幾何平均です。したがって ISO の B0 判は $1000\text{ mm} \times 1414\text{ mm}$ です。このため、 \LaTeX 2_ϵ の `b5paper` は $250\text{ mm} \times 176\text{ mm}$ ですが、 $\text{p}\text{\LaTeX 2}_\epsilon$ の `b5paper` は $257\text{ mm} \times 182\text{ mm}$ になっています。ここでは $\text{p}\text{\LaTeX 2}_\epsilon$ にならって JIS に従いました。

デフォルトは `a4paper` です。

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

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

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

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

```
242 \for\bxjs@tmpa:={0,1,2,3}\do{\edef\bxjs@next{%
243   \noexpand\DeclareOption{b\bxjs@tmpa paper}%
244   {\noexpand\bxjs@setpaper@bsize{\bxjs@tmpa}}%
245 }\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。

```
246 \DeclareOption{a4varpaper}{\bxjs@setpaper{{210truemm}{283truemm}}}%
247 \DeclareOption{b5varpaper}{\bxjs@setpaper{{182truemm}{230truemm}}}%
248 \DeclareOption{screenpaper}{\bxjs@setpaper{screen}}%
249 \DeclareOption{custompaper}{}
```

■横置き 用紙の縦と横の長さを入れ換えます。

```
250 \newif\if@landscape
251 \@landscapefalse
252 \DeclareOption{landscape}{\@landscapetrue}
```

■slide オプション slide を新設しました。

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

```
253 \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。

```
254 %<!slide>\@slidefalse
255 %<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。

```
256 \newcommand{\@ptsize}{0}
257 \def\bxjs@param@basefontsize{10pt}
258 \def\jsUnusualPtSize{-20}
```

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

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

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

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

```
260 \jsSetQHLength\@tempdima{#1}%
261 \edef\bxjs@param@basefontsize{\the\@tempdima}%
262 \ifdim\@tempdima=10pt \long\def\@ptsize{0}%
263 \else\ifdim\@tempdima=10.95pt \long\def\@ptsize{1}%
264 \else\ifdim\@tempdima=12pt \long\def\@ptsize{2}%
265 \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。

```
266 \def\bxjs@setjbasefontsize#1{%
267 \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。

```
268 \newif\ifjsc@mag
269 \newif\ifjsc@mag@xreal
270 %\let\jsc@magscale\undefined

271 \DeclareOption{8pt}{\bxjs@setbasefontsize{8pt}}
272 \DeclareOption{9pt}{\bxjs@setbasefontsize{9pt}}
273 \DeclareOption{10pt}{\bxjs@setbasefontsize{10pt}}
274 \DeclareOption{11pt}{\bxjs@setbasefontsize{10.95pt}}
275 \DeclareOption{12pt}{\bxjs@setbasefontsize{12pt}}
276 \DeclareOption{14pt}{\bxjs@setbasefontsize{14.4pt}}
277 \DeclareOption{17pt}{\bxjs@setbasefontsize{17.28pt}}
```

```

278 \DeclareOption{20pt}{\bxjs@setbasefontsize{20pt}}
279 \DeclareOption{21pt}{\bxjs@setbasefontsize{20.74pt}}
280 \DeclareOption{25pt}{\bxjs@setbasefontsize{24.88pt}}
281 \DeclareOption{30pt}{\bxjs@setbasefontsize{29.86pt}}
282 \DeclareOption{36pt}{\bxjs@setbasefontsize{35.83pt}}
283 \DeclareOption{43pt}{\bxjs@setbasefontsize{43pt}}
284 \DeclareOption{12Q}{\bxjs@setjbasefontsize{3mm}}
285 \DeclareOption{14Q}{\bxjs@setjbasefontsize{3.5mm}}
286 \DeclareOption{10ptj}{\bxjs@setjbasefontsize{10pt}}
287 \DeclareOption{10.5ptj}{\bxjs@setjbasefontsize{10.5pt}}
288 \DeclareOption{11ptj}{\bxjs@setjbasefontsize{11pt}}
289 \DeclareOption{12ptj}{\bxjs@setjbasefontsize{12pt}}

```

JS クラス互換の magstyle 設定オプション。

```

290 \DeclareOption{usemag}{\let\bxjs@magstyle\bxjs@magstyle@@usemag}
291 \DeclareOption{nomag}{\let\bxjs@magstyle\bxjs@magstyle@@nomag}
292 \DeclareOption{nomag*}{\let\bxjs@magstyle\bxjs@magstyle@@xreal}

```

■**トンボオプション** トンボ (crop marks) を出力します。実際の処理は pL^AT_EX 2_ε 本体で行います (plcore.dtx 参照)。オプション `tombow` で日付付きのトンボ、オプション `tombo` で日付なしのトンボを出力します。これらはアスキー版のままです。カウンタ `\hour`, `\minute` は pL^AT_EX 2_ε 本体で宣言されています。

取りあえず、pT_EX 系の場合に限り、JS クラスのトンボ関連のコードをそのまま活かしておく。正常に動作する保証はない。

```

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

```

309 \if j\jsEngine

```



```

310 \DeclareOption{mentuke}{%
311   \tombowtrue \tombowdatefalse
312   \setlength{\@tombowwidth}{\z@}%
313   \maketombowbox}
314 \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。

```

315 \DeclareOption{oneside}{\@twosidefalse \@mparswitchfalse}
316 \DeclareOption{twoside}{\@twosidetrue \@mparswitchtrue}
317 \DeclareOption{vartwoside}{\@twosidetrue \@mparswitchfalse}

```

■二段組 `twocolumn` で二段組になります。

```

318 \DeclareOption{onecolumn}{\@twocolumnfalse}
319 \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。

```

320 \DeclareOption{titlepage}{\@titlepagetrue}
321 \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。

```

322 %<book|report>\DeclareOption{openright}{\@openrighttrue\@openleftfalse}
323 %<book|report>\DeclareOption{openleft}{\@openlefttrue\@openrightfalse}
324 %<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。

```

325 \def\eqnarray{%
326   \stepcounter{equation}%
327   \def\@currentlabel{\p@equation\theequation}%
328   \def\@currentcounter{equation}%
329   \global\@eqnswtrue
330   \m@th
331   \global\@eqcnt\z@

```

```

332 \tabskip\@centering
333 \let\\\@eqncr
334 $$\everycr{}\halign to\displaywidth\bgroup
335 \hskip\@centering$\displaystyle\tabskip\z@skip{##}$\@eqnse1
336 &\global\@eqcnt\@ne \hfil$\displaystyle{##}$\hfil
337 &\global\@eqcnt\tw@ $\displaystyle{##}$\hfil\tabskip\@centering
338 &\global\@eqcnt\thr@@ \hb@xt@\z@\bgroup\hss##\egroup
339 \tabskip\z@skip
340 \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。森本さんにしがって訂正しました。

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

```

341 \DeclareOption{leqno}{\input{leqno.clo}}
342 \DeclareOption{fleqn}{\input{fleqn.clo}}
343 % fleqn 用の eqnarray 環境の再定義
344 \def\eqnarray{%
345 \stepcounter{equation}%
346 \def\@currentlabel{\p@equation\theequation}%
347 \def\@currentcounter{equation}%
348 \global\@eqnswtrue\m@th
349 \global\@eqcnt\z@
350 \tabskip\mathindent
351 \let\=\@eqncr
352 \setlength\abovedisplayskip{\topsep}%
353 \ifvmode
354 \addtolength\abovedisplayskip{\partopsep}%
355 \fi
356 \addtolength\abovedisplayskip{\parskip}%
357 \setlength\belowdisplayskip{\abovedisplayskip}%
358 \setlength\belowdisplayshortskip{\abovedisplayskip}%
359 \setlength\abovedisplayshortskip{\abovedisplayskip}%
360 $$\everycr{}\halign to\linewidth% $$
361 \bgroup
362 \hskip\@centering$\displaystyle\tabskip\z@skip{##}$\@eqnse1
363 &\global\@eqcnt\@ne \hfil$\displaystyle{##}$\hfil
364 &\global\@eqcnt\tw@
365 $\displaystyle{##}$\hfil \tabskip\@centering
366 &\global\@eqcnt\thr@@ \hb@xt@\z@\bgroup\hss##\egroup
367 \tabskip\z@skip\cr
368 }}

```

■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。

```

369 % \DeclareOption{openbib}{%
370 % \AtEndOfPackage{%
371 % \renewcommand\@openbib@code{%

```

```

372 %      \advance\leftmargin\bibindent
373 %      \itemindent -\bibindent
374 %      \listparindent \itemindent
375 %      \parsep \z@}%
376 %      \renewcommand\newblock{\par}}

```

■数式フォントとして和文フォントを登録しないオプション 数式中では 16 通りのフォントしか使えません。AMSTeX や mathptmx パッケージを使って数式フォントをたくさん使うと “Too many math alphabets ...” というエラーが起こってしまいます。disablejfam オプションを付ければ、明朝・ゴシックを数式用フォントとして登録するのをやめますので、数式用フォントが二つ節約できます。いずれにしても \textmc や \mbox や amsmath パッケージの \text を使えば数式中で和文フォントが使えますので、この新ドキュメントクラスでは標準で和文フォントを数式用に登録しないことにしていたのですが、従来のドキュメントクラスの仕様に合わせることにしました。

\bxjs@enablejfam [暗黙文字トークン] enablejfam オプションの状態：

```

377 %\let\bxjs@enablejfam\@undefined

enablejfam オプションの処理。

378 \def\bxjs@kv@enablejfam@true{\let\bxjs@enablejfam=t}
379 \def\bxjs@kv@enablejfam@false{\let\bxjs@enablejfam=f}
380 \def\bxjs@kv@enablejfam@default{\let\bxjs@enablejfam\@undefined}
381 \define@key{bxjs}{enablejfam}[true]{%
382   \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。

```

383 \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。

```

384 \newif\ifjsDraft
385 \DeclareOption{draft}{\jsDrafttrue \overfullrule=5pt }
386 \DeclareOption{final}{\jsDraftfalse \overfullrule=0pt }

```

■和文フォントメトリックの選択 このクラスファイルでは、和文 TFM として東京書籍印刷の小林肇さんの作られた JIS フォントメトリック (jis, jisg) を標準で使うことにしますが、従来の min10, goth10 などを使いたいときは mingoth というオプションを指定します。また、winjis オプションで winjis メトリック (OTF パッケージと同じ psitau さん作；ソースに書かれた Windows の機種依存文字が dvips, dvipdfmx などでも出力出来るようになる) が使えます。

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

BXJS クラスではここは和文ドライバの管轄。

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

BXJS クラスでは geometry パッケージがこの処理を行う。

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

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

```
387 \newif\ifbxjs@papersize
388 \bxjs@papersizetrue
389 \DeclareOption{nopapersize}{\bxjs@papersizefalse}
390 \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。

```
391 \newif\if@english
392 \@englishfalse
393 \DeclareOption{english}{\@englishttrue}
```

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

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

BXJS では当初から bxjsreport クラスが用意されている。

■jslogo パッケージの読み込み L^AT_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。

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

■複合設定オプション

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

\bxjs@invscale \bxjs@invscale は T_EX 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。

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

複合設定オプションとは、「エンジンやドライバや和文ドライバの設定を含む、複数の設定

を一度に行うオプション」のことである。ある特定の設定を短く書く必要が高いと判断される場合に用意される。

pandoc オプションは、Pandoc で L^AT_EX 用の既定テンプレートを用いて他形式から L^AT_EX (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。

```
422 \DeclareOption{pandoc}{%
423   \bxjs@apply@pandoc@opt}
424 \@onlypreamble\bxjs@apply@pandoc@opt
425 \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。

```
426   \g@addto@macro\bxjs@post@option@hook{%
427     \bxjs@oldfontcommandstrue
428     \setkeys{bxjs}{ja=pandoc}%
429     \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)。

```
430   \ifx\bxjs@driver@opt\@undefined
431     \def\bxjs@driver@opt{dvipdfmx}%
432     \bxjs@dvi@opttrue
433   \fi
434   \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。

```
435 \DeclareOption{pandoc+}{%
436   \g@addto@macro\bxjs@post@option@hook{%
437     \edef\jsJaParam{\bxjs@catopt{_plus}\jsJaParam}}%
438   \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。

```
439 %\let\bxjs@engine@given\@undefined
```

`\bxjs@engine@opt` 明示されたエンジンのオプション名。

```
440 %\let\bxjs@engine@opt\@undefined
```

エンジン明示指定のオプションの処理。

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

```
441 \DeclareOption{autodetect-engine}{%
442   \let\bxjs@engine@given=*}
443 \DeclareOption{latex}{%
```

```

444 \def\bxjs@engine@opt{latex}%
445 \let\bxjs@engine@given=n}
446 \DeclareOption{platex}{%
447 \def\bxjs@engine@opt{platex}%
448 \let\bxjs@engine@given=j}
449 \DeclareOption{uplatex}{%
450 \def\bxjs@engine@opt{uplatex}%
451 \let\bxjs@engine@given=u}
452 \DeclareOption{xelatex}{%
453 \def\bxjs@engine@opt{xelatex}%
454 \let\bxjs@engine@given=x}
455 \DeclareOption{pdflatex}{%
456 \def\bxjs@engine@opt{pdflatex}%
457 \let\bxjs@engine@given=p}
458 \DeclareOption{lualatex}{%
459 \def\bxjs@engine@opt{lualatex}%
460 \let\bxjs@engine@given=l}
461 \DeclareOption{platex-ng}{%
462 \def\bxjs@engine@opt{platex-ng}%
463 \let\bxjs@engine@given=g}
464 \DeclareOption{platex-ng*}{%
465 \def\bxjs@engine@opt{platex-ng*}%
466 \let\bxjs@platexng@nodrv=t%
467 \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。

```

468 %\let\bxjs@driver@given\@undefined
469 \let\bxjs@driver@@dvimode=0
470 \let\bxjs@driver@@dvipdfmx=1
471 \let\bxjs@driver@@pdfmode=2
472 \let\bxjs@driver@@xetex=3
473 \let\bxjs@driver@@dvips=4
474 \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。

```

475 %\let\bxjs@driver@opt\@undefined

```

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

```

476 \DeclareOption{dvips}{%
477 \def\bxjs@driver@opt{dvips}%
478 \let\bxjs@driver@given\bxjs@driver@@dvips}
479 \DeclareOption{dviout}{%
480 \def\bxjs@driver@opt{dviout}%
481 \let\bxjs@driver@given\bxjs@driver@@dvimode}
482 \DeclareOption{xdvi}{%
483 \def\bxjs@driver@opt{xdvi}%
484 \let\bxjs@driver@given\bxjs@driver@@dvimode}
485 \DeclareOption{dvipdfmx}{%

```

```

486 \def\bxjs@driver@opt{dvipdfmx}%
487 \let\bxjs@driver@given\bxjs@driver@@dvipdfmx}
488 \DeclareOption{nodvidriver}{%
489 \def\bxjs@driver@opt{nodvidriver}%
490 \let\bxjs@driver@given\bxjs@driver@@none}
491 \DeclareOption{nodvidriver*}{%
492 \def\bxjs@driver@opt{nodvidriver*}%
493 \let\bxjs@driver@given\bxjs@driver@@none}
494 \DeclareOption{pdftex}{%
495 \def\bxjs@driver@opt{pdftex}%
496 \let\bxjs@driver@given\bxjs@driver@@pdfmode}
497 \DeclareOption{luatex}{%
498 \def\bxjs@driver@opt{luatex}%
499 \let\bxjs@driver@given\bxjs@driver@@pdfmode}
500 \DeclareOption{xetex}{%
501 \def\bxjs@driver@opt{xetex}%
502 \let\bxjs@driver@given\bxjs@driver@@xetex}

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

```

■その他の BXJS 独自オプション 🐼

TODO: 互換用オプションを分離する（2.0 版で？）。

```

\bxjs@depre@opt 非推奨のオプションについて警告を出す。
\bxjs@depre@opt@do 504 \@onlypreamble\bxjs@depre@opt
505 \def\bxjs@depre@opt#1#2{%
506 \ClassWarningNoLine\bxjs@clsname
507 {The old option '#1' is DEPRECATED\MessageBreak
508 and may be abolished in future!\MessageBreak
509 You should instead write:\MessageBreak
510 \space\space #2}}
511 \@onlypreamble\bxjs@depre@opt@do
512 \def\bxjs@depre@opt@do#1#2{%
513 \bxjs@depre@opt{#1}{#2}%
514 \setkeys{bxjs}{#2}}

```

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

※ 2.0 版より、既定値を常に真とする。

```

515 \newif\ifbxjs@bigcode \bxjs@bigcodetrue

nobigcode / bigcode オプションの定義。
516 \DeclareOption{nobigcode}{%

```



```

517 \bxjs@bigcodefalse}
518 \DeclareOption{bigcode}{%
519 \bxjs@bigcodetrue}

```

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

```

520 \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。

```

521 \DeclareOption{nooldfontcommands}{%
522 \bxjs@oldfontcommandsfalse}
523 \DeclareOption{oldfontcommands}{%
524 \bxjs@oldfontcommandstrue}


```

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

```

525 \DeclareOption{winjis}{%
526 \ClassWarningNoLine\bxjs@clsname
527 {This class does not support `winjis' option}%
528 }
529 \DeclareOption{mingoth}{%
530 \ClassWarningNoLine\bxjs@clsname
531 {This class does not support `mingoth' option}%
532 }
533 \DeclareOption{jis}{%
534 \ClassWarningNoLine\bxjs@clsname
535 {This class does not support `jis' option}%
536 }

```

■ `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。

```

537 \DeclareOption*{%
538 \def\bxjs@next{\bxjs@safe@setkeys{bxjs}}%
539 \expandafter\bxjs@next\expandafter{\CurrentOption}}

```

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

※ネスト不可。

```

540 \def\bxjs@safe@setkeys#1#2{%
541 \let\bxjs@save@KV@errx\KV@errx \let\KV@errx\@gobble
542 \setkeys{#1}{#2}%
543 \let\KV@errx\bxjs@save@KV@errx}

```

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

“<オプション名>=<値>” のオプション指定に対して、`\[bxjs@<enum 名>]` を `\[bxjs@<enum 名>@@<値>]` に等置する（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。

```

544 \@onlypreamble\bxjs@declare@enum@option
545 \def\bxjs@declare@enum@option#1#2#3{%
546   \bxjs@csletcs{bxjs@#2}{bxjs@#2@#3}%
547   \define@key{bxjs}{#1}{%
548     \expandafter\ifx\csname bxjs@#2@#1\endcsname\relax
549     \bxjs@error@keyval{#1}{#1}%
550   \else \bxjs@csletcs{bxjs@#2}{bxjs@#2@#1}%
551   \fi}}

```

`\bxjs@declare@bool@option` `\bxjs@declare@bool@option{<オプション名>}{<スイッチ名>}{<初期値>}`

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

```

552 \@onlypreamble\bxjs@declare@bool@option
553 \def\bxjs@declare@bool@option#1#2#3{%
554   \csname newif\expandafter\endcsname\csname ifbxjs@#2\endcsname
555   \@nameuse{bxjs@#2#3}%
556   \define@key{bxjs}{#1}[true]{%
557     \expandafter\ifx\csname bxjs@#2#1\endcsname\relax
558     \bxjs@error@keyval{#1}{#1}%
559   \else \@nameuse{bxjs@#2#1}%
560   \fi}}

```

`\bxjs@set@keyval` `\bxjs@set@keyval{<key>}{<value>}{<error>}`

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

```

561 \def\bxjs@set@keyval#1#2#3{%
562   \bxjs@csletcs{bxjs@next}{bxjs@kv@#1@#2}%
563   \ifx\bxjs@next\relax
564     \bxjs@error@keyval{#1}{#2}%
565     #3%
566   \else \bxjs@next
567   \fi}
568 \@onlypreamble\bxjs@error@keyval
569 \def\bxjs@error@keyval#1#2{%
570   \ClassError\bxjs@clsname
571   {Invalid value '#2' for option #1}\@ehc}

```

`\jsScale` [実数値マクロ] 和文スケール値。

```

572 \def\jsScale{0.924715}

```

`\bxjs@base@opt` 明示された base オプションの値。

```

573 %\let\bxjs@base@opt\undefined

```

base オプションの処理。

```

574 \define@key{bxjs}{base}{%
575   \edef\bxjs@base@opt{#1}%
576   \bxjs@setbasefontsize{#1}}
577 \define@key{bxjs}{fontsize}{\setkeys{bxjs}{base=#1}}

```

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

```
578 %\let\bxjs@jbase@opt\@undefined
```

jbase オプションの処理。

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

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

\bxjs@scale@opt 明示された scale オプションの値。

```
581 %\let\bxjs@scale@opt\@undefined
```

scale オプションの処理。

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

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

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

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

noscale オプションの処理。

TODO: noscale は 3.0 版で廃止の予定。

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

\bxjs@param@mag mag オプションの値。

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

mag オプションの処理。

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

paper オプションの処理。

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

\bxjs@jadriver 和文ドライバの名前。

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

\bxjs@jadriver@opt 明示された和文ドライバの名前。

```
591 %\let\bxjs@jadriver@opt\@undefined
```

ja オプションの処理。

※ jadriver は 0.9 版で用いられた旧称。

TODO: jadriver は 3.0 版で廃止の予定。

※単なる ja という指定は無視される (Pandoc 対策)。

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

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

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

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

\jsJaFont 和文フォント設定の名前。

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

jafont オプションの処理。

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

`\jsJaParam` 和文ドライバパラメタの文字列。

```
598 \let\jsJaParam\@empty
```

`japaram` オプションの処理。

```
599 \define@key{bxjs}{japaram}{%  
600   \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。

```
601 \define@key{bxjs}{pandoc}[]{%  
602   \ExecuteOptions{pandoc}%  
603   \edef\jsJaParam{\bxjs@catopt\jsJaParam{#1}}}  
604 \define@key{bxjs}{pandoc+}[]{%  
605   \ExecuteOptions{pandoc+}%  
606   \edef\jsJaParam{\bxjs@catopt\jsJaParam{#1}}}
```

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

```
607 \let\bxjs@magstyle@@mag=m  
608 \let\bxjs@magstyle@@real=r  
609 \let\bxjs@magstyle@@xreal=x
```

(新しい素敵な名前。)

※ただし制御綴としては、*付の名前は扱い難いので、`\bxjs@magstyle@@xreal` の方を優先させる。

```
610 \let\bxjs@magstyle@@usemag\bxjs@magstyle@@mag  
611 \let\bxjs@magstyle@@nomag\bxjs@magstyle@@real  
612 \bxjs@cslet{bxjs@magstyle@@nomag*}\bxjs@magstyle@@xreal
```

`\bxjs@magstyle@@default` は既定の値を表す。

```
613 \let\bxjs@magstyle@@default\bxjs@magstyle@@usemag  
614 \ifx l\jsEngine \ifnum\luatexversion>86  
615   \let\bxjs@magstyle@@default\bxjs@magstyle@@xreal  
616 \fi\fi  
617 \ifjsWithpTeXng  
618   \let\bxjs@magstyle@@default\bxjs@magstyle@@xreal  
619 \fi  
620 \let\bxjs@magstyle\bxjs@magstyle@@default
```

`magstyle` オプションの処理。

```
621 \define@key{bxjs}{magstyle}{%  
622   \bxjs@csletcs{bxjs@magstyle}{bxjs@magstyle@@#1}%  
623   \ifx\bxjs@magstyle\relax  
624     \bxjs@error@keyval{magstyle}{#1}%  
625     \let\bxjs@magstyle\bxjs@magstyle@@default  
626   \fi}
```

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

```
627 \let\bxjs@geometry@@class=c  
628 \let\bxjs@geometry@@user=u  
629 \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。

```
630 \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。

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

DVI モードのドライバとドライバ種別との対応。

```
632 \let\bxjs@dvidriver@@dvipdfmx=\bxjs@driver@@dvipdfmx
633 \let\bxjs@dvidriver@@dvips=\bxjs@driver@@dvips
634 \let\bxjs@dvidriver@@dviout=\bxjs@driver@@dvimode
635 \let\bxjs@dvidriver@@xdvi=\bxjs@driver@@dvimode
636 \let\bxjs@dvidriver@@nodvidriver=\bxjs@driver@@none
637 \bxjs@cslet{bxjs@dvidriver@@nodvidriver*}\bxjs@driver@@none
```

dvi オプションの処理。

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

`\bxjs@driver@given` を未定義にしていることに注意。

```
643   \def\bxjs@driver@opt{#1}%
644   \let\bxjs@driver@given\undefined
645   \bxjs@dvi@opttrue
646 \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。

```
647 \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。

```
648 \newif\ifbxjs@force@chapterabstract
649 %<book>\bxjs@force@chapterabstracttrue
```

layout オプションの処理。

```
650 \@namedef{bxjs@kv@layout@v1}{%
651 %<book>\bxjs@layout@buggyhmargintrue
652 %<report>\bxjs@force@chapterabstracttrue
653 }
654 \@namedef{bxjs@kv@layout@v2}{%
655 %<book>\bxjs@layout@buggyhmarginfalse
656 %<report>\bxjs@force@chapterabstractfalse
657 }
658 \define@key{bxjs}{layout}{%
659   \bxjs@set@keyval{layout}{#1}{}}
```

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

```

660 %\let\bxjs@textwidth@limit@opt\@undefined
661 \define@key{bxjs}{textwidth-limit}{%
662   \bxjs@depre@opt{textwidth-limit}{textwidth=#1zw}%
663   \edef\bxjs@textwidth@limit@opt{#1}}

```

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

```

664 %\let\bxjs@textwidth@opt\@undefined
665 \define@key{bxjs}{textwidth}{\edef\bxjs@textwidth@opt{#1}}
666 \define@key{bxjs}{line_length}{\setkeys{bxjs}{textwidth=#1}}

```

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

```

667 %\let\bxjs@number@of@lines@opt\@undefined
668 \define@key{bxjs}{number-of-lines}{\edef\bxjs@number@of@lines@opt{#1}}
669 \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。

```

670 %\let\bxjs@paragraph@mark\@undefined
671 \define@key{bxjs}{paragraph-mark}{%
672   \edef\bxjs@paragraph@mark{#1}}

```

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

```

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

```

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

```

674 \bxjs@declare@bool@option{jaspace-cmd}{jaspace@cmd}{true}
675 \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。

```

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

```

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

```

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

```

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

```

678 \chardef\bxjs@everyparhook@none=0
679 \chardef\bxjs@everyparhook@compat=1
680 \chardef\bxjs@everyparhook@modern=2
681 \bxjs@declare@enum@option{everyparhook}{everyparhook}{%
682   \if j\jsEngine compat\else modern\fi}

```

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

```

683 \chardef\bxjs@label@section@none=0
684 \chardef\bxjs@label@section@compat=1
685 \chardef\bxjs@label@section@modern=2
686 \bxjs@declare@enum@option{label-section}{label@section}{compat}

```

`\ifbxjs@usezw` [スイッチ] `use-zw` の指定値。

TODO: `zw/nozw` は 3.0 版で廃止の予定。

```

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

```

```
688 \DeclareOption{nozw}{\bxjs@depre@opt@do{nozw}{use-zw=false}}
689 \DeclareOption{zw}{\bxjs@depre@opt@do{zw}{use-zw=true}}
```

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

TODO: `js/nojs` は 3.0 版で廃止の予定。

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

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

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

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

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

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

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

■オプションの実行

L^AT_EX の実装では、クラスやパッケージのオプションのトークン列の中に { } が含まれると正常に処理ができない。これに対処する為 `\@removeelement` の実装に少し手を加える(仕様は変わらない)。

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

```
702 \let\bxjs@org@removeelement\@removeelement
703 \def\@removeelement#1#2#3{%
704   \def\reserved@a{#2}%
705   \ifx\reserved@a\@empty \let#3\@empty
706   \else \bxjs@org@removeelement{#1}{#2}{#3}%
707   \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。

```

708 %<article>\ExecuteOptions{a4paper,oneside,onecolumn,notitlepage,final}
709 %<book>\ExecuteOptions{a4paper,twoside,onecolumn,titlepage,openright,final}
710 %<report>\ExecuteOptions{a4paper,oneside,onecolumn,titlepage,openany,final}
711 %<slide>\ExecuteOptions{36pt,a4paper,landscape,oneside,onecolumn,titlepage,final}
712 \ProcessOptions\relax
713 \bxjs@post@option@hook

```

後処理

```

714 \if@slide
715   \def\maybeblue{\ifundefined{ver@color.sty}{\color{blue}}{}}
716 \fi
717 \if@landscape
718   \setlength\@tempdima {\paperheight}
719   \setlength\paperheight{\paperwidth}
720   \setlength\paperwidth {\@tempdima}
721 \fi

```

■グローバルオプションの整理

グローバルオプションのトークン列に { } が含まれていると、やはり後のパッケージの読み込み処理で不具合を起こすようである (\ProcessOptions* がエラーになる)。従って、このようなオプションは除外することにする。

```

722 \def\bxjs@tmpdo{%
723   \def\bxjs@tmpa{\@gobble}%
724   \expandafter\bxjs@tmpdo@a\@classoptionslist,\@nil,%
725   \let\@classoptionslist\bxjs@tmpa}
726 \def\bxjs@tmpdo@a#1,{%
727   \ifx\@nil#1\relax\else
728     \bxjs@tmpdo@b#1{}\@nil
729     \if@tempswa \edef\bxjs@tmpa{\bxjs@tmpa,#1}\fi
730     \expandafter\bxjs@tmpdo@a
731   \fi}
732 \def\bxjs@tmpdo@b#1#\bxjs@tmpdo@c}
733 \def\bxjs@tmpdo@c#1\@nil{%
734   \ifx\@nil#1\@nil \@tempwatrue \else \@tempwafalse \fi}
735 \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。

```

736 \@expandtwoargs\@removeelement
737   {papersize}\@classoptionslist\@classoptionslist
738 \@expandtwoargs\@removeelement
739   {10pt}\@classoptionslist\@classoptionslist
740 \@expandtwoargs\@removeelement
741   {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。

```
742 \ifx\bxjs@jadriver@opt\undefined\else
743   \let\bxjs@jadriver\bxjs@jadriver@opt
744 \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。

```
745 \let\bxjs@tmpb\jsEngine
746 \ifx j\bxjs@tmpb\ifjsWithpTeXng
747   \let\bxjs@tmpb=g
748 \fi\fi
749 \ifx j\bxjs@tmpb\ifjsWithupTeX
750   \let\bxjs@tmpb=u
751 \fi\fi
752 \ifx p\bxjs@tmpb\ifjsInPdfMode\else
753   \let\bxjs@tmpb=n
754 \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。)

```
755 \ifx *\bxjs@engine@given
756   \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。

```
757 \ifx j\bxjs@engine@given
758   \g@addto@macro\@classoptionslist{,platex}
759 \else\ifx u\bxjs@engine@given
760   \g@addto@macro\@classoptionslist{,uplatex}
761 \fi\fi
762 \fi
763 \ifx\bxjs@engine@given\undefined\else
764   \ifx\bxjs@engine@given\bxjs@tmpb\else
765     \ClassError\bxjs@clsname
766       {Option '\bxjs@engine@opt' used on wrong engine}\@ehc
```

```
767 \fi
768 \fi
```

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

```
769 \ifjsWithpTeXng
770 \g@addto@macro\@classoptionslist{,uplatex}
771 \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。

```
772 \@tempswatrue
773 \ifx \bxjs@driver@given\@undefined\else
774 \ifjsInPdfMode
775 \ifx\bxjs@driver@given\bxjs@driver@@pdfmode\else
776 \@tempswafalse
777 \fi
778 \else\ifx x\jsEngine
779 \ifx\bxjs@driver@given\bxjs@driver@@xetex\else
780 \@tempswafalse
781 \fi
782 \else
783 \ifx\bxjs@driver@given\bxjs@driver@@pdfmode
784 \@tempswafalse
785 \else\ifx\bxjs@driver@given\bxjs@driver@@xetex
786 \@tempswafalse
787 \fi\fi
788 \ifjsWithpTeXng\ifx\bxjs@driver@given\bxjs@driver@@dvipdfmx\else
789 \@tempswafalse
790 \fi\fi
791 \fi\fi
792 \fi
793 \if@tempswa\else
794 \ClassError\bxjs@clsname
795 {Option '\bxjs@driver@opt' used on wrong engine}\@ehc
796 \fi
```

DVI 出力のエンジンである場合の追加処理。

```
797 \ifjsInPdfMode \@tempswafalse
798 \else\ifx x\jsEngine \@tempswafalse
799 \else\ifjsWithpTeXng \@tempswafalse
800 \else \@tempswatrue
801 \fi\fi\fi
802 \if@tempswa
```

ドライバオプションがない場合は警告を出す。

※ただし ja 非指定の場合はスキップする (0.3 版との互換性のため)。

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

```

806      {A driver option is MISSING!!\MessageBreak
807      You should properly specify one of the valid\MessageBreak
808      driver options according to the DVI driver\MessageBreak
809      that is in use:\MessageBreak
810      \@spaces dvips, dvipdfmx, dviout, xdvi,\MessageBreak
811      \@spaces nodvidriver}
812  \fi
813 \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。)

```

814 \ifbxjs@dvi@opt
815   \edef\bxjs@next{%
816     \let\noexpand\bxjs@driver@given
817     \csname bxjs@dvidriver@@\bxjs@driver@opt\endcsname
818     \noexpand\g@addto@macro\noexpand\@classoptionslist
819     {,\bxjs@driver@opt}%
820   }\bxjs@next
821 \fi
822 \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。

```

823 \ifjsWithpTeXng
824   \ifx\bxjs@driver@given\bxjs@driver@@dvipdfmx
825     \let\bxjs@platexng@nodrv\@undefined
826   \else\ifx t\bxjs@platexng@nodrv\else
827     \g@addto@macro\@classoptionslist{,dvipdfmx}
828   \fi\fi
829 \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。

```

830 \ifx\bxjs@driver@given\bxjs@driver@@none
831   \bxjs@papersizefalse
832 \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。

```

833 \@ifpackageloaded{plautopatch}{%
834   \bxjs@plautopatchtrue
835 }{}

```

標準の和文ドライバの名前の定数。

```

836 \def\bxjs@@minimal{minimal}
837 \def\bxjs@@standard{standard}
838 \def\bxjs@@pandoc{pandoc}
839 \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。

```
840 \ifx\bxjs@jadriver\relax
841   \ifx j\jsEngine
842     \let\bxjs@jadriver\bxjs@@standard
843   \else
844     \ClassWarningNoLine\bxjs@clsname
845     {The option 'ja' is MISSING!!\MessageBreak
846     So 'ja=minimal' is assumed as fallback, but\MessageBreak
847     such implicit setting is now DEPRECATED!\MessageBreak
848     You should write 'ja=minimal' explicitly,\MessageBreak
849     if it is intended}
850   \let\bxjs@jadriver\bxjs@@minimal
851 \fi
852 \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。

```
853 \ifx\bxjs@jadriver\bxjs@@pandoc \ifx\bxjs@plautopatch@given\@undefined
854   \ifjsWitheTeX
855   \bxjs@plautopatchtrue
856 \fi\fi\fi
857 \ifx j\jsEngine \ifbxjs@plautopatch
858   \RequirePackage{plautopatch}[2018/08/22]%v0.3
859 \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。

```
860 \ifx\bxjs@jadriver@opt\@undefined\else
861   \ifx\bxjs@engine@given\@undefined
862     \ClassError\bxjs@clsname
863     {An engine option must be explicitly given}%
864     {When you use a Japanese-driver you must specify a correct\MessageBreak
865     engine option.\MessageBreak\@ehc}
866 \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。)

```
867 \ifx\bxjs@magstyle@@default\bxjs@magstyle@@mag\else
868   \ifx\bxjs@magstyle\bxjs@magstyle@@mag
869     \let\bxjs@magstyle\bxjs@magstyle@@default
870   \ClassError\bxjs@clsname
871   {The engine does not support 'magstyle=usemag'}%
872   {LuaTeX v0.87 or later no longer supports the "mag" feature of TeX.\MessageBreak
873   The default value 'nomag*' is used instead.\MessageBreak \@ehc}
```

```
874 \fi
875 \fi
```

base、jbase、scale の値を用いて和文スケール値を解決する。

※\bxjs@param@basefontsize と \jsScale へのオプション値の反映は既に実施されていることに注意。jbase 非指定の場合はこのままでよい。

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

jbase 指定済で base 未指定の場合は、\jsScale の値を採用して和文基底サイズを決定する。

```
878 \jsSetQHLlength\@tempdima{\bxjs@jbase@opt}%
879 \bxjs@invscale\@tempdima\jsScale
880 \bxjs@setbasefontsize{\@tempdima}%
881 \else
```

jbase と base がともに指定済の場合は、それらの値から和文スケール値を決定する。

```
882 \ifx\bxjs@scale@opt\@undefined\else
883 \ClassWarningNoLine\bxjs@clsname
884 {Redundant 'scale' option is ignored}%
885 \fi
886 \jsSetQHLlength\@tempdima{\bxjs@jbase@opt}%
887 \@tempdimb=\bxjs@param@basefontsize\relax
888 \edef\jsScale{\strip@pt\@tempdimb}%
889 \bxjs@invscale\@tempdima\jsScale
890 \edef\jsScale{\strip@pt\@tempdima}%
891 \fi
892 \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。

```
893 \let\Cjascale\jsScale
```

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

```
894 \if \if p\jsEngine T\else\if n\jsEngine T\else F\fi\fi T
895 \@tempcnta="80 \loop \ifnum\@tempcnta<"100
896 \catcode\@tempcnta\active
897 \advance\@tempcnta\@ne
898 \repeat
899 \fi
```

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。

```
900 \ifbxjs@disguise@js
901 %<book|report>\def\bxjs@js@clsname{jsbook}
```

```

902 %<!book&!report>\def\bxjs@js@clsname{jsarticle}
903 \namedef{ver@\bxjs@js@clsname.cls}{2001/01/01 (bxjs)}
904 \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。

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

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

906 \ifbxjs@oldfontcommands
907 \AtEndOfClass{\allowoldfontcommands}
908 \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。

```

909 \ifx\stockwidth\undefined\newdimen\stockwidth\fi
910 \ifx\stockheight\undefined\newdimen\stockheight\fi

```

```

911 \begingroup\expandafter\expandafter\expandafter\endgroup
912 \expandafter\ifx\csname iftombow\expandafter\endcsname\csname iftrue\endcsname
913   \setlength{\stockwidth}{\paperwidth}
914   \setlength{\stockheight}{\paperheight}
915   \advance \stockwidth 2in
916   \advance \stockheight 2in
917 \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。

```

918 %<slide>\def\n@baseline{13}%
919 %<!slide>\ifdim\bxjs@param@basefontsize<10pt \def\n@baseline{15}%
920 %<!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。

```

921 \ifx\bxjs@magstyle\bxjs@magstyle@@mag
922   \jsc@magtrue
923 \else\ifx\bxjs@magstyle\bxjs@magstyle@@xreal
924   \jsc@mag@xrealtrue
925 \fi\fi

```

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

[2008-12-26] 1000 / \mag に相当する \inv@mag を定義しました。truein を使っていたところを \inv@mag in に直しましたので、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。

```

926 \ifx\bxjs@param@mag\relax
927   \@tempdima=\bxjs@param@basefontsize
928   \advance\@tempdima.001pt \multiply\@tempdima25
929   \divide\@tempdima16384\relax \@tempcnta\@tempdima\relax
930   \edef\bxjs@param@mag{\the\@tempcnta}
931 \else
932 % mag 値が直接指定された場合
933   \bxjs@gset@tempcnta{\bxjs@param@mag}
934   \ifnum\@tempcnta<\z@ \@tempcnta=\z@ \fi
935 % 有効な mag 値の範囲は 1--32768

```

```

936 \edef\bxjs@param@mag{\the\@tempcnta}
937 \advance\@tempcnta100000
938 \def\bxjs@tmpa#1#2#3#4#5\@nil{\@tempdima=#2#3#4.#5\p@}
939 \expandafter\bxjs@tmpa\the\@tempcnta\@nil
940 \edef\bxjs@param@basefontsize{\the\@tempdima}
941 \fi
942 \@tempcnta\bxjs@param@mag \advance\@tempcnta100000
943 \def\bxjs@tmpa#1#2#3#4\@nil{\@tempdima=#2#3.#4\p@}
944 \expandafter\bxjs@tmpa\the\@tempcnta\@nil
945 \edef\jsc@magscale{\strip@pt\@tempdima}
946 \let\jsBaseFontSize\bxjs@param@basefontsize

```

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

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

```

947 \newdimen\jsc@empt
948 \newdimen\jsc@mmm
949 \ifjsc@mag
950 \jsc@empt=1\p@
951 \jsc@mmm=1mm
952 \catcode`\?=9 % \p@? read as \p@
953 \else
954 \jsc@empt=\jsc@magscale\p@
955 \jsc@mmm=\jsc@magscale mm
956 \catcode`\?=11 \let\p@?\jsc@empt
957 \fi
958 \chardef\bxjs@qmcc=\catcode`\?\relax
959 \g@addto@macro\bxjs@pre@jadriver@hook{\catcode`\?=12\relax}

```

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。

```

960 \newdimen\jsZw
961 \jsZw=10\jsc@empt \jsZw=\jsScale\jsZw
962 \ifbxjs@usezw
963 \providecommand*\zw{\jsZw}
964 \fi

```

`\zwspace` 全角幅の水平空き。

```

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

```

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


```

966 \ifjsc@mag@xreal
967   \RequirePackage{type1cm}
968   \let\jsc@invscale\bxjs@invscale

```

```

969   \ifbxjs@TUenc
970     \expandafter\let\csname TU/lmr/m/n/10\endcsname\relax
971   \else
972     \expandafter\let\csname OT1/cmr/m/n/10\endcsname\relax
973   \fi
974   \expandafter\let\csname OMX/cmex/m/n/10\endcsname\relax
975   \let\jsc@get@external@font\get@external@font
976   \def\get@external@font{%
977     \jsc@preadjust@extract@font
978     \jsc@get@external@font}
979   \def\jsc@fstrunc#1{%
980     \edef\jsc@tmpa{\strip@pt#1}%
981     \expandafter\jsc@fstrunc@a\jsc@tmpa.****\@nil}
982   \def\jsc@fstrunc@a#1.#2#3#4#5#6\@nil{%
983     \if#5* \else
984       \edef\jsc@tmpa{#1%
985         \ifnum#2#3>\z@ .#2\ifnum#3>\z@ #3\fi\fi}%
986     \fi}
987   \def\jsc@preadjust@extract@font{%
988     \let\jsc@req@size\f@size
989     \dimen@f@size\p@ \jsc@invscale\dimen@\jsc@magscale
990     \advance\dimen@.005pt\relax \jsc@fstrunc\dimen@
991     \let\jsc@ref@size\jsc@tmpa
992     \let\f@size\jsc@ref@size}
993   \def\execute@size@function#1{%
994     \let\jsc@cref@size\f@size
995     \let\f@size\jsc@req@size
996     \csname s@fct@#1\endcsname}
997   \let\jsc@DeclareErrorFont\DeclareErrorFont
998   \def\DeclareErrorFont#1#2#3#4#5{%
999     \@tempdimc#5\p@ \@tempdimc\jsc@magscale\@tempdimc
1000     \edef\jsc@tmpa{{#1}-{#2}-{#3}-{#4}-{#5}\strip@pt\@tempdimc}}
1001   \expandafter\jsc@DeclareErrorFont\jsc@tmpa}
1002   \def\gen@sfcnt{%
1003     \edef\mandatory@arg{\mandatory@arg\jsc@cref@size}%
1004     \empty@sfcnt}
1005   \def\genb@sfcnt{%
1006     \edef\mandatory@arg{%
1007       \mandatory@arg\expandafter\genb@x\jsc@cref@size.. \@@}%
1008     \empty@sfcnt}
1009   \ifbxjs@TUenc\else
1010     \DeclareErrorFont{OT1}{cmr}{m}{n}{10}
1011   \fi
1012 \fi

```

[2016-11-16] latex.ltx (ltspace.dtx) で定義されている `\smallskip` の、単位 `pt` を `\jsc@empt` 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 1013 \def\jsc@smallskip{\vspace\jsc@smallskipamount}
\jsc@bigskip 1014 %\def\jsc@medskip{\vspace\jsc@medskipamount}
1015 %\def\jsc@bigskip{\vspace\jsc@bigskipamount}

\jsc@smallskipamount
\jsc@medskipamount 1016 \newskip\jsc@smallskipamount
\jsc@bigskipamount 1017 \jsc@smallskipamount=3\jsc@empt plus 1\jsc@empt minus 1\jsc@empt
1018 %\newskip\jsc@medskipamount
1019 %\jsc@medskipamount =6\jsc@empt plus 2\jsc@empt minus 2\jsc@empt
1020 %\newskip\jsc@bigskipamount
1021 %\jsc@bigskipamount =12\jsc@empt plus 4\jsc@empt minus 4\jsc@empt
```

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

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

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

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

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

```
1022 % \ifpapersize
1023 %   \setlength{\@tempdima}{\paperwidth}
1024 %   \setlength{\@tempdimb}{\paperheight}
1025 %   \iftombow
1026 %     \advance \@tempdima 2truein
1027 %     \advance \@tempdimb 2truein
1028 %   \fi
1029 %   \AtBeginDvi{\special{pdf: pagesize width \the\@tempdima\space height \the\@tempdimb}}
1030 % \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。

```
1031 %</class>
1032 %<*jsclasses>
```

1033 %<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, $\mathrm{T}_{\mathrm{E}}\mathrm{X}$ 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, $\mathrm{pT}_{\mathrm{E}}\mathrm{X}$ (アスキーが日本語化した $\mathrm{T}_{\mathrm{E}}\mathrm{X}$) の公称 10 ポイントの和文フォント (min10 など) は, 実寸 (標準の字送り量) が 9.62216pt です。これは 3.3818mm, 写研の写植機の単位では 13.527 級, PostScript の単位では 9.5862 ポイントになります。jis フォントなどもこの値を踏襲しています。

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

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

1034 %</class>

1035 %<minijs>

1036 %% min/goth -> jis/jisg (for pLaTeX only)

1037 \ifnum\jis"2121="3000 \else

1038 \@for\@tempa:=5,6,7,8,9,10,10.95,12,14.4,17.28,20.74,24.88\do{%

1039 \expandafter\let\csname JY1/mc/m/n/\@tempa\endcsname\relax

1040 \expandafter\let\csname JY1/gt/m/n/\@tempa\endcsname\relax

1041 \expandafter\let\csname JT1/mc/m/n/\@tempa\endcsname\relax

1042 \expandafter\let\csname JT1/gt/m/n/\@tempa\endcsname\relax

1043 }

1044 \def\Cjascale{0.924690}

1045 \DeclareFontShape{JY1}{mc}{m}{n}{<-> s * [0.961] jis}{}

1046 \DeclareFontShape{JY1}{gt}{m}{n}{<-> s * [0.961] jisg}{}

1047 \DeclareFontShape{JT1}{mc}{m}{n}{<-> s * [0.961] tmin10}{}

1048 \DeclareFontShape{JT1}{gt}{m}{n}{<-> s * [0.961] tgoth10}{}

1049 \fi

1050 %</minijs>

1051 %<class>

1052 %<!*jspf>

1053 \def\Cjascale{0.924690}

1054 \ifmingoth

1055 \DeclareFontShape{\jsc@JYn}{mc}{m}{n}{<-> s * [0.961] \jsc@pfx@ min10}{}

1056 \DeclareFontShape{\jsc@JYn}{gt}{m}{n}{<-> s * [0.961] \jsc@pfx@ goth10}{}

1057 \DeclareFontShape{\jsc@JTn}{mc}{m}{n}{<-> s * [0.961] \jsc@pfx@ tmin10}{}

1058 \DeclareFontShape{\jsc@JTn}{gt}{m}{n}{<-> s * [0.961] \jsc@pfx@ tgoth10}{}

```

1059 \else
1060   \ifjisfont
1061     \DeclareFontShape{\jsc@JYn}{mc}{m}{n}{<-> s * [0.961] \jsc@pfx@ jis}{\}
1062     \DeclareFontShape{\jsc@JYn}{gt}{m}{n}{<-> s * [0.961] \jsc@pfx@ jisg}{\}
1063     \DeclareFontShape{\jsc@JTn}{mc}{m}{n}{<-> s * [0.961] \jsc@pfx@ tmin10}{\}
1064     \DeclareFontShape{\jsc@JTn}{gt}{m}{n}{<-> s * [0.961] \jsc@pfx@ tgoth10}{\}
1065   \else
1066     \if@jsc@uplatex
1067       \DeclareFontShape{JY2}{mc}{m}{n}{<-> s * [0.924690] upjisr-h}{\}
1068       \DeclareFontShape{JY2}{gt}{m}{n}{<-> s * [0.924690] upjisg-h}{\}
1069       \DeclareFontShape{JT2}{mc}{m}{n}{<-> s * [0.924690] upjisr-v}{\}
1070       \DeclareFontShape{JT2}{gt}{m}{n}{<-> s * [0.924690] upjisg-v}{\}
1071     \else
1072       \DeclareFontShape{\jsc@JYn}{mc}{m}{n}{<-> s * [0.961] \jsc@pfx@ jis}{\}
1073       \DeclareFontShape{\jsc@JYn}{gt}{m}{n}{<-> s * [0.961] \jsc@pfx@ jisg}{\}
1074       \DeclareFontShape{\jsc@JTn}{mc}{m}{n}{<-> s * [0.961] \jsc@pfx@ tmin10}{\}
1075       \DeclareFontShape{\jsc@JTn}{gt}{m}{n}{<-> s * [0.961] \jsc@pfx@ tgoth10}{\}
1076     \fi
1077   \fi
1078 \fi
1079 %<*/!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。

```

1080 %<*jspf>
1081 \def\Cjascale{0.903375}
1082 \ifmingoth
1083   \DeclareFontShape{\jsc@JYn}{mc}{m}{n}{<-> s * [0.93885] \jsc@pfx@ min10}{\}
1084   \DeclareFontShape{\jsc@JYn}{gt}{m}{n}{<-> s * [0.93885] \jsc@pfx@ goth10}{\}
1085   \DeclareFontShape{\jsc@JTn}{mc}{m}{n}{<-> s * [0.93885] \jsc@pfx@ tmin10}{\}
1086   \DeclareFontShape{\jsc@JTn}{gt}{m}{n}{<-> s * [0.93885] \jsc@pfx@ tgoth10}{\}
1087 \else
1088   \ifjisfont
1089     \DeclareFontShape{\jsc@JYn}{mc}{m}{n}{<-> s * [0.93885] \jsc@pfx@ jis}{\}
1090     \DeclareFontShape{\jsc@JYn}{gt}{m}{n}{<-> s * [0.93885] \jsc@pfx@ jisg}{\}
1091     \DeclareFontShape{\jsc@JTn}{mc}{m}{n}{<-> s * [0.93885] \jsc@pfx@ tmin10}{\}
1092     \DeclareFontShape{\jsc@JTn}{gt}{m}{n}{<-> s * [0.93885] \jsc@pfx@ tgoth10}{\}
1093   \else
1094     \if@jsc@uplatex
1095       \DeclareFontShape{JY2}{mc}{m}{n}{<-> s * [0.903375] upjisr-h}{\}
1096       \DeclareFontShape{JY2}{gt}{m}{n}{<-> s * [0.903375] upjisg-h}{\}
1097       \DeclareFontShape{JT2}{mc}{m}{n}{<-> s * [0.903375] upjisr-v}{\}
1098       \DeclareFontShape{JT2}{gt}{m}{n}{<-> s * [0.903375] upjisg-v}{\}
1099     \else
1100       \DeclareFontShape{\jsc@JYn}{mc}{m}{n}{<-> s * [0.93885] \jsc@pfx@ jis}{\}
1101       \DeclareFontShape{\jsc@JYn}{gt}{m}{n}{<-> s * [0.93885] \jsc@pfx@ jisg}{\}
1102       \DeclareFontShape{\jsc@JTn}{mc}{m}{n}{<-> s * [0.93885] \jsc@pfx@ tmin10}{\}
1103       \DeclareFontShape{\jsc@JTn}{gt}{m}{n}{<-> s * [0.93885] \jsc@pfx@ tgoth10}{\}

```

```

1104 \fi
1105 \fi
1106 \fi
1107 %</jpsf>

```

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, $\mathrm{T}_{\mathrm{E}}\mathrm{X}$ が数学で多用されることを考えると, イタリック体に明朝体を当てたほうがいいように思えてきましたので, イタリック体・斜体に対応する和文を明朝体に変えることにしました。

[2004-11-03] `\rmfamily` も和文対応にしました。

```

1108 % \DeclareFontShape{\jsc@JYn}{mc}{bx}{n}{<->ssub*gt/m/n}{ } % in \jsc@JYnmc
1109 % \DeclareFontShape{\jsc@JYn}{gt}{bx}{n}{<->ssub*gt/m/n}{ } % in \jsc@JYngt
1110 \DeclareFontShape{\jsc@JYn}{mc}{m}{it}{<->ssub*mc/m/n}{ }
1111 \DeclareFontShape{\jsc@JYn}{mc}{m}{sl}{<->ssub*mc/m/n}{ }
1112 \DeclareFontShape{\jsc@JYn}{mc}{m}{sc}{<->ssub*mc/m/n}{ }
1113 \DeclareFontShape{\jsc@JYn}{gt}{m}{it}{<->ssub*gt/m/n}{ }
1114 \DeclareFontShape{\jsc@JYn}{gt}{m}{sl}{<->ssub*gt/m/n}{ }
1115 \DeclareFontShape{\jsc@JYn}{mc}{bx}{it}{<->ssub*gt/m/n}{ }
1116 \DeclareFontShape{\jsc@JYn}{mc}{bx}{sl}{<->ssub*gt/m/n}{ }
1117 % \DeclareFontShape{\jsc@JTn}{mc}{bx}{n}{<->ssub*gt/m/n}{ } % in \jsc@JTnmc
1118 % \DeclareFontShape{\jsc@JTn}{gt}{bx}{n}{<->ssub*gt/m/n}{ } % in \jsc@JTngt
1119 \DeclareFontShape{\jsc@JTn}{mc}{m}{it}{<->ssub*mc/m/n}{ }
1120 \DeclareFontShape{\jsc@JTn}{mc}{m}{sl}{<->ssub*mc/m/n}{ }
1121 \DeclareFontShape{\jsc@JTn}{mc}{m}{sc}{<->ssub*mc/m/n}{ }
1122 \DeclareFontShape{\jsc@JTn}{gt}{m}{it}{<->ssub*gt/m/n}{ }
1123 \DeclareFontShape{\jsc@JTn}{gt}{m}{sl}{<->ssub*gt/m/n}{ }
1124 \DeclareFontShape{\jsc@JTn}{mc}{bx}{it}{<->ssub*gt/m/n}{ }
1125 \DeclareFontShape{\jsc@JTn}{mc}{bx}{sl}{<->ssub*gt/m/n}{ }

```

[2020-02-02] $\mathrm{IAT}_{\mathrm{E}}\mathrm{X} 2_{\epsilon}$ 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 $\mathrm{IAT}_{\mathrm{E}}\mathrm{X} 2_{\epsilon}$ 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] $\mathrm{IAT}_{\mathrm{E}}\mathrm{X} 2_{\epsilon}$ 2020-10-01 では `\AddToHook` を利用します。

```

1126 %</class>
1127 %<*class|minijs>
1128 %% ad-hoc "relation font"
1129 \@ifl@t@r\fmtversion{2020/10/01}
1130 { \jsc@needspace@tchfalse } { \jsc@needspace@tchtrue }
1131 \ifjsc@needspace@tch % --- for 2020-02-02 or older BEGIN
1132 \ifx\@rmfamilyhook\@undefined % old

```

```

1133 \DeclareRobustCommand\rmfamily
1134     {\not@math@alphabet\rmfamily\mathrm
1135       \romanfamily\rmdefault\kanjifamily\mcdefault\selectfont}
1136 \DeclareRobustCommand\sffamily
1137     {\not@math@alphabet\sffamily\mathsf
1138       \romanfamily\sfdefault\kanjifamily\gtdefault\selectfont}
1139 \DeclareRobustCommand\ttfamily
1140     {\not@math@alphabet\ttfamily\mathtt
1141       \romanfamily\ttdefault\kanjifamily\gtdefault\selectfont}
1142 \AtBeginDocument{%
1143   \ifx\mweights@init\@undefined\else % mweights.sty is loaded
1144     % my definitions above should have been overwritten, recover it!
1145     % \selectfont is executed twice but I don't care about speed...
1146     \expandafter\g@addto@macro\csname rmfamily \endcsname
1147       {\kanjifamily\mcdefault\selectfont}%
1148     \expandafter\g@addto@macro\csname sffamily \endcsname
1149       {\kanjifamily\gtdefault\selectfont}%
1150     \expandafter\g@addto@macro\csname ttfamily \endcsname
1151       {\kanjifamily\gtdefault\selectfont}%
1152   \fi}
1153 \else % 2020-02-02
1154 \g@addto@macro\@rmfamilyhook
1155   {\prepare@family@series@update@kanji{mc}\mcdefault}
1156 \g@addto@macro\@sffamilyhook
1157   {\prepare@family@series@update@kanji{gt}\gtdefault}
1158 \g@addto@macro\@ttfamilyhook
1159   {\prepare@family@series@update@kanji{gt}\gtdefault}
1160 \fi
1161 \else % --- for 2020-02-02 or older END & for 2020-10-01 BEGIN
1162 \AddToHook{rmfamily}%
1163   {\prepare@family@series@update@kanji{mc}\mcdefault}
1164 \AddToHook{sffamily}%
1165   {\prepare@family@series@update@kanji{gt}\gtdefault}
1166 \AddToHook{ttfamily}%
1167   {\prepare@family@series@update@kanji{gt}\gtdefault}
1168 \fi % --- for 2020-10-01 END
1169 %</class|minijs>
1170 %<*class>

```

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

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

[2017-09-03] Yue ZHANG さん作の fixjfm パッケージが\documentclass より前に

`\RequirePackage{fixjfm}` として読み込まれていた場合には、その定義を優先するため、このクラスファイルでは再定義しません。

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

```
1171 \ifx\DeclareFixJFMCJKTextFontCommand\undefined
1172 \DeclareRobustCommand\textmc[1]{%
1173   \relax\ifmmode \expandafter\nfss@text \fi{\mcfamily #1}}
1174 \DeclareRobustCommand\textgt[1]{%
1175   \relax\ifmmode \expandafter\nfss@text \fi{\gtfamily #1}}
1176 \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。

```
1177 \def\reDeclareMathAlphabet#1#2#3{%
1178   \edef\@tempa{\expandafter@gobble\string#2}%
1179   \edef\@tempb{\expandafter@gobble\string#3}%
1180   \edef\@tempc{\string @\expandafter@gobbletwo\string#2}%
1181   \ifx\@tempc\@tempa%
1182     \edef\@tempa{\expandafter@gobbletwo\string#2}%
1183     \edef\@tempb{\expandafter@gobbletwo\string#3}%
1184   \fi
1185   \begingroup
1186     \let\protect\noexpand
1187     \def\@tempaa{\relax}%
1188     \expandafter\ifx\csname RDMAorg@\@tempa\endcsname\relax
1189       \edef\@tempaa{\expandafter\def\expandafter\noexpand%
1190         \csname RDMAorg@\@tempa\endcsname{%
1191           \expandafter\noexpand\csname\@tempa\endcsname}}%
1192     \fi
1193     \def\@tempbb{\relax}%
1194     \expandafter\ifx\csname RDMAorg@\@tempb\endcsname\relax
1195       \edef\@tempbb{\expandafter\def\expandafter\noexpand%
1196         \csname RDMAorg@\@tempb\endcsname{%
1197           \expandafter\noexpand\csname\@tempb\endcsname}}%
1198     \fi
1199     \edef\@tempc{\@tempaa\@tempbb}%
1200   \expandafter\endgroup\@tempc%
1201   \edef#1{\noexpand\protect\expandafter\noexpand\csname%
1202     \expandafter@gobble\string#1\space\space\endcsname}%
1203   \expandafter\edef\csname\expandafter@gobble\string#1\space\space\endcsname%
```

```

1204     {\noexpand\DualLang@mathalph@bet%
1205       {\expandafter\noexpand\csname RDMAorg@\@tempa\endcsname}%
1206       {\expandafter\noexpand\csname RDMAorg@\@tempb\endcsname}%
1207     }%
1208 }
1209 \@onlypreamble\reDeclareMathAlphabet
1210 \def\DualLang@mathalph@bet#1#2{%
1211   \relax\ifmmode
1212     \ifx\math@bgroup\bgroup%      2e normal style      (\mathrm{...})
1213     \bgroup\let\DualLang@Mfontsw\DLMfontsw@standard
1214   \else
1215     \ifx\math@bgroup\relax%      2e two letter style (\rm->\mathrm)
1216     \let\DualLang@Mfontsw\DLMfontsw@oldstyle
1217   \else
1218     \ifx\math@bgroup\@empty% 2.09 oldfont style ({\mathrm ...})
1219     \let\DualLang@Mfontsw\DLMfontsw@oldfont
1220   \else%                          panic! assume 2e normal style
1221     \bgroup\let\DualLang@Mfontsw\DLMfontsw@standard
1222   \fi
1223   \fi
1224   \fi
1225   \else
1226     \let\DualLang@Mfontsw\@firstoftwo
1227   \fi
1228   \DualLang@Mfontsw{#1}{#2}%
1229 }
1230 \def\DLMfontsw@standard#1#2#3{#1{#2{#3}}\egroup}
1231 \def\DLMfontsw@oldstyle#1#2{#1\relax\@fontswitch\relax{#2}}
1232 \def\DLMfontsw@oldfont#1#2{#1\relax#2\relax}
1233 \if@enablejfam
1234   \DeclareSymbolFont{mincho}{\jsc@JYn}{mc}{m}{n}
1235   \DeclareSymbolFontAlphabet{\mathmc}{mincho}
1236   \SetSymbolFont{mincho}{bold}{\jsc@JYn}{gt}{m}{n}
1237   \jfam\symmincho
1238   \DeclareMathAlphabet{\mathgt}{\jsc@JYn}{gt}{m}{n}
1239   \AtBeginDocument{%
1240     \reDeclareMathAlphabet{\mathrm}{\@mathrm}{\@mathmc}
1241     \reDeclareMathAlphabet{\mathbf}{\@mathbf}{\@mathgt}}
1242 \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。


```
1243 % \DeclareTextCommand{\textsterling}{0T1}{\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。

```
1244 \prebreakpenalty\jis"2147=10000      % 5000  '
1245 \postbreakpenalty\jis"2148=10000     % 5000  "
1246 \prebreakpenalty\jis"2149=10000     % 5000  "
```

「 \TeX !」「 π 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。

```
1247 \inhibitxspcode`!=1
1248 \inhibitxspcode`\pi=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

```
1249 % \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。

```
1250 \xspcode`+=3
1251 \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。

```
1252 \xspcode`^^80=3
1253 \xspcode`^^81=3
1254 \xspcode`^^82=3
1255 \xspcode`^^83=3
1256 \xspcode`^^84=3
1257 \xspcode`^^85=3
1258 \xspcode`^^86=3
1259 \xspcode`^^87=3
1260 \xspcode`^^88=3
1261 \xspcode`^^89=3
1262 \xspcode`^^8a=3
1263 \xspcode`^^8b=3
1264 \xspcode`^^8c=3
1265 \xspcode`^^8d=3
1266 \xspcode`^^8e=3
1267 \xspcode`^^8f=3
1268 \xspcode`^^90=3
1269 \xspcode`^^91=3
1270 \xspcode`^^92=3
1271 \xspcode`^^93=3
1272 \xspcode`^^94=3
```

1273 \xspcode`^^95=3
1274 \xspcode`^^96=3
1275 \xspcode`^^97=3
1276 \xspcode`^^98=3
1277 \xspcode`^^99=3
1278 \xspcode`^^9a=3
1279 \xspcode`^^9b=3
1280 \xspcode`^^9c=3
1281 \xspcode`^^9d=3
1282 \xspcode`^^9e=3
1283 \xspcode`^^9f=3
1284 \xspcode`^^a0=3
1285 \xspcode`^^a1=3
1286 \xspcode`^^a2=3
1287 \xspcode`^^a3=3
1288 \xspcode`^^a4=3
1289 \xspcode`^^a5=3
1290 \xspcode`^^a6=3
1291 \xspcode`^^a7=3
1292 \xspcode`^^a8=3
1293 \xspcode`^^a9=3
1294 \xspcode`^^aa=3
1295 \xspcode`^^ab=3
1296 \xspcode`^^ac=3
1297 \xspcode`^^ad=3
1298 \xspcode`^^ae=3
1299 \xspcode`^^af=3
1300 \xspcode`^^b0=3
1301 \xspcode`^^b1=3
1302 \xspcode`^^b2=3
1303 \xspcode`^^b3=3
1304 \xspcode`^^b4=3
1305 \xspcode`^^b5=3
1306 \xspcode`^^b6=3
1307 \xspcode`^^b7=3
1308 \xspcode`^^b8=3
1309 \xspcode`^^b9=3
1310 \xspcode`^^ba=3
1311 \xspcode`^^bb=3
1312 \xspcode`^^bc=3
1313 \xspcode`^^bd=3
1314 \xspcode`^^be=3
1315 \xspcode`^^bf=3
1316 \xspcode`^^c0=3
1317 \xspcode`^^c1=3
1318 \xspcode`^^c2=3
1319 \xspcode`^^c3=3
1320 \xspcode`^^c4=3
1321 \xspcode`^^c5=3

1322 \xspcode`^^c6=3
1323 \xspcode`^^c7=3
1324 \xspcode`^^c8=3
1325 \xspcode`^^c9=3
1326 \xspcode`^^ca=3
1327 \xspcode`^^cb=3
1328 \xspcode`^^cc=3
1329 \xspcode`^^cd=3
1330 \xspcode`^^ce=3
1331 \xspcode`^^cf=3
1332 \xspcode`^^d0=3
1333 \xspcode`^^d1=3
1334 \xspcode`^^d2=3
1335 \xspcode`^^d3=3
1336 \xspcode`^^d4=3
1337 \xspcode`^^d5=3
1338 \xspcode`^^d6=3
1339 \xspcode`^^d7=3
1340 \xspcode`^^d8=3
1341 \xspcode`^^d9=3
1342 \xspcode`^^da=3
1343 \xspcode`^^db=3
1344 \xspcode`^^dc=3
1345 \xspcode`^^dd=3
1346 \xspcode`^^de=3
1347 \xspcode`^^df=3
1348 \xspcode`^^e0=3
1349 \xspcode`^^e1=3
1350 \xspcode`^^e2=3
1351 \xspcode`^^e3=3
1352 \xspcode`^^e4=3
1353 \xspcode`^^e5=3
1354 \xspcode`^^e6=3
1355 \xspcode`^^e7=3
1356 \xspcode`^^e8=3
1357 \xspcode`^^e9=3
1358 \xspcode`^^ea=3
1359 \xspcode`^^eb=3
1360 \xspcode`^^ec=3
1361 \xspcode`^^ed=3
1362 \xspcode`^^ee=3
1363 \xspcode`^^ef=3
1364 \xspcode`^^f0=3
1365 \xspcode`^^f1=3
1366 \xspcode`^^f2=3
1367 \xspcode`^^f3=3
1368 \xspcode`^^f4=3
1369 \xspcode`^^f5=3
1370 \xspcode`^^f6=3

```

1371 \xspcode^^f7=3
1372 \xspcode^^f8=3
1373 \xspcode^^f9=3
1374 \xspcode^^fa=3
1375 \xspcode^^fb=3
1376 \xspcode^^fc=3
1377 \xspcode^^fd=3
1378 \xspcode^^fe=3
1379 \xspcode^^ff=3

1380 %</class>
1381 %</jsclasses>
1382 %<*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 `sfcode` 値を使う。
- 「防御的な \@」での不具合を防ぐため、大文字直後の \@ は標準と同等の動作にする。

```

1383 \chardef\bxjs@periodchar=`\
1384 \bxjs@protected\def\bxjs@SE{%
1385   \ifnum\spacefactor<\@m \spacefactor\@m
1386   \else \spacefactor\sfcode\bxjs@periodchar
1387   \fi}
1388 \ifbxjs@fix@at@cmd
1389   \def\@{\bxjs@SE{}}
1390 \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>\@xiipt</code>	12	<code>\@xivpt</code>	14.4

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

`\kanjiskip` は pL^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。

`\set@fontsize` `\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。

```

1391 \def\bxjs@tmpa{\def\set@fontsize##1##2##3{
1392 \expandafter\bxjs@tmpa\expandafter{%
1393   \set@fontsize{##1}{##2}{##3}%
1394 % 末尾にコードを追加
1395   \expandafter\def\expandafter\size@update\expandafter{%
1396     \size@update
1397     \jsFontSizeChanged}%
1398 }
```

`\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。

```

1399 \newcommand*\jsFontSizeChanged{%
1400   \jsZw=\f@size\p@
1401   \jsZw=\jsScale \jsZw
1402   \ifdim\parindent>\z@
1403     \if@english \parindent=1em
1404     \else       \parindent=1\jsZw
1405   \fi
1406   \fi\relax
1407   \jsResetDimen}
```

`\jsResetDimen` ユーザ定義用のフック。

```
1408 \newcommand*\jsResetDimen{}
```

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

```
1409 \ifjsc@mag
1410   \let\jsc@setfontsize\@setfontsize
1411 \else
1412   \def\jsc@setfontsize#1#2#3{%
1413     \@setfontsize#1{#2\jsc@mp}{#3\jsc@mp}}
1414 % microtype 対策
1415   \ifjsWitheTeX\if j\jsEngine\else
1416     \def\jsc@setfontsize#1#2#3{%
1417       \edef\bxjs@sfs@next{%
1418         \unexpanded{\@setfontsize#1}%
1419         {\the\dimexpr#2\jsc@mp\relax}{\the\dimexpr#3\jsc@mp\relax}%
1420       }\bxjs@sfs@next}
1421   \fi\fi
1422 \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。

```
1423 \emergencystretch 3\jsZw
```

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

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

`\widebaselines` `\abovedisplayskip` 等が初期化されてしまうという shintok さんのご指摘に対して、しばしば愛好家さんが次の修正を教えてくださいました。

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

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

別行立て数式に入るときに `\narrowbaselines` が呼ばれるが、このコードでは「数式中で `\normalsize` などのサイズ命令 (`\@currsize` の実体) が呼ばれた」ことになり警告が出る。JS クラスでは、`\@setfontsize` 中の `\@nomath` 実行を消して「そもそもサイズ命令で警告が出ない」ようにしている。警告が常に出ないのも望ましくないので、BXJS クラスの実装では、`\narrowbaselines` の時だけ警告が出ないようにする。

```
1424 \newif\ifnarrowbaselines
```

```
1425 \if@english
```

```

1426 \narrowbaselinestrue
1427 \fi
1428 \def\narrowbaselines{%
1429 \narrowbaselinestrue
1430 \skip0=\abovedisplayskip
1431 \skip2=\abovedisplayshortskip
1432 \skip4=\belowdisplayskip
1433 \skip6=\belowdisplayshortskip
1434 % 一時的に警告を無効化する
1435 \let\bxjs@save@nomath\@nomath
1436 \let\@nomath\@gobble
1437 \@currsize\selectfont
1438 \let\@nomath\bxjs@save@nomath
1439 \abovedisplayskip=\skip0
1440 \abovedisplayshortskip=\skip2
1441 \belowdisplayskip=\skip4
1442 \belowdisplayshortskip=\skip6\relax}
1443 \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。

```

1444 \def\bxjs@if@narrowbaselines{%
1445 \ifnarrowbaselines\expandafter\@firstoftwo
1446 \else \expandafter\@secondoftwo
1447 \fi
1448 }

```

`\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。

```

1449 \renewcommand{\normalsize}{%
1450 \bxjs@if@narrowbaselines{%
1451 \jsc@setfontsize\normalsize\@xpt\@xiipt
1452 }{%else

```

```

1453 \jsc@setfontsize\normalsize\@xpt{\n@baseline}%
1454 }%

```

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。

```

1455 \abovedisplayskip 11\p@? \@plus3\p@? \@minus4\p@?
1456 \abovedisplayshortskip \z@ \@plus3\p@?
1457 \belowdisplayskip 9\p@? \@plus3\p@? \@minus4\p@?
1458 \belowdisplayshortskip \belowdisplayskip

```

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

```

1459 \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。

```

1460 %</class>
1461 %<*class|minijs>
1462 %% initialize
1463 \normalsize
1464 %</class|minijs>
1465 %<*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
\Cdp ラメータに実際の値を設定します。たとえば \Cwd は \normalfont の全角幅 (1zw) です。
\Cwd [2017-08-31] 基準とする文字を「全角空白」(EUC コード 0xA1A1) から「漢」(JIS コー
\Cvs ド 0x3441) へ変更しました。

\Chs

\Cwd 等の変数は pT_EX 系以外では未定義なのでここで定義する。

```

1466 \ifx\Cht\undefined \newdimen\Cht \fi
1467 \ifx\Cdp\undefined \newdimen\Cdp \fi
1468 \ifx\Cwd\undefined \newdimen\Cwd \fi
1469 \ifx\Cvs\undefined \newdimen\Cvs \fi
1470 \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。

```

1471 \setlength\Cht{0.88\jsZw}
1472 \setlength\Cdp{0.12\jsZw}
1473 \setlength\Cwd{1\jsZw}
1474 \setlength\Cvs{\baselineskip}
1475 \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。

```

1476 \newcommand{\small}{%
1477   \bxjs@if@narrowbaselines{%
1478     %<!kiyou>   \jsc@setfontsize\small\@ixpt{11}%
1479     %<kiyou>    \jsc@setfontsize\small{8.8888}{11}%
1480   }{%else
1481     %<!kiyou>   \jsc@setfontsize\small\@ixpt{13}%
1482     %<kiyou>    \jsc@setfontsize\small{8.8888}{13.2418}%
1483   }%
1484   \abovedisplayskip 9\p@? \@plus3\p@? \@minus4\p@?
1485   \abovedisplayshortskip \z@ \@plus3\p@?
1486   \belowdisplayskip \abovedisplayskip
1487   \belowdisplayshortskip \belowdisplayskip
1488   \def\@listif{\leftmargin\leftmargini
1489               \topsep \z@
1490               \parsep \z@
1491               \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。

```

1492 \newcommand{\footnotesize}{%
1493   \bxjs@if@narrowbaselines{%
1494     %<!kiyou>   \jsc@setfontsize\footnotesize\@viipt{9.5}%
1495     %<kiyou>    \jsc@setfontsize\footnotesize{8.8888}{11}%
1496   }{%else
1497     %<!kiyou>   \jsc@setfontsize\footnotesize\@viipt{11}%
1498     %<kiyou>    \jsc@setfontsize\footnotesize{8.8888}{13.2418}%
1499   }%
1500   \abovedisplayskip 6\p@? \@plus2\p@? \@minus3\p@?
1501   \abovedisplayshortskip \z@ \@plus2\p@?
1502   \belowdisplayskip \abovedisplayskip
1503   \belowdisplayshortskip \belowdisplayskip
1504   \def\@listif{\leftmargin\leftmargini
1505               \topsep \z@
1506               \parsep \z@
1507               \itemsep \parsep}}

```

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

`\LARGE` [2004-11-03] `\HUGE` を追加。

```

\huge 1508 \newcommand{\scriptsize}{\jsc@setfontsize\scriptsize\@viipt\@viipt}
1509 \newcommand{\tiny}{\jsc@setfontsize\tiny\@vpt\@vpt}
\LARGE 1510 \if@twocolumn
\HUGE 1511 %<!kiyou> \newcommand{\large}{\jsc@setfontsize\large\@xipt{\n@baseline}}

```

```

1512 %<kiyou> \newcommand{\large}{\jsc@setfontsize\large{11.111}{\n@baseline}}
1513 \else
1514 %<!kiyou> \newcommand{\large}{\jsc@setfontsize\large\xiip{17}}
1515 %<kiyou> \newcommand{\large}{\jsc@setfontsize\large{11.111}{17}}
1516 \fi
1517 %<!kiyou>\newcommand{\Large}{\jsc@setfontsize\Large\xivpt{21}}
1518 %<kiyou>\newcommand{\Large}{\jsc@setfontsize\Large{12.222}{21}}
1519 \newcommand{\LARGE}{\jsc@setfontsize\LARGE\xvipt{25}}
1520 \newcommand{\huge}{\jsc@setfontsize\huge\xxpt{28}}
1521 \newcommand{\Huge}{\jsc@setfontsize\Huge\xxvpt{33}}
1522 \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。

```

1523 \everydisplay=\expandafter{\the\everydisplay \narrowbaselines}

```

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

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

```

1524 % \newcommand{\headfont}{\bfseries}
1525 \newcommand{\headfont}{\sffamily}
1526 % \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。

```

1527 %<!kiyou>\setlength\columnsep{2\Cwd}
1528 %<kiyou>\setlength\columnsep{28truebp}
1529 \setlength\columnseprule{\z@}

```

■段落

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

```

\lineskiplimit

```

```

\normallineskiplimit

```

```

1530 \setlength\lineskip{1\jsc@empt}
1531 \setlength\normallineskip{1\jsc@empt}
1532 \setlength\lineskiplimit{1\jsc@empt}
1533 \setlength\normallineskiplimit{1\jsc@empt}

```

`\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。

```
1534 \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。

```

1535 \setlength\parskip{\z@}
1536 \if@slide
1537   \setlength\parindent{0\p@}
1538 \else
1539   \setlength\parindent{1\Cwd}
1540 \fi

```

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

```

\@highpenalty 1541 \@lowpenalty   51
                  1542 \@medpenalty 151
                  1543 \@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。

```
1544 % \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。

```
1545 % \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。

```
1546 \@onlypreamble\bxjs@bd@pre@geometry@hook
```

```
1547 \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。

```
1548 \ifjsc@mag
```

```
1549 \mag=\bxjs@param@mag
```

```
1550 \fi
```

```
1551 \setlength{\topskip}{10\p@?}
```

\jsSetQHLength のための和文単位の定義。

```
1552 \def\bxjs@unit@trueQ{0.25trueem}\let\bxjs@unit@trueH\bxjs@unit@trueQ
```

```
1553 \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。

```
1554 \@tempwafalse
```

```
1555 \def\bxjs@tmpdo{\@ifnextchar\bgroup\bxjs@tmpdo@a\remove@to@nnil}
```

```
1556 \def\bxjs@tmpdo@a#1{\edef\bxjs@tmpa{#1}%
```

```
1557 \@ifnextchar\bgroup\bxjs@tmpdo@b\remove@to@nnil}
```

```
1558 \def\bxjs@tmpdo@b#1{\edef\bxjs@tmpa{\bxjs@tmpa,#1}%
```

```
1559 \@ifnextchar\@nnil\bxjs@tmpdo@c\remove@to@nnil}
```

```
1560 \def\bxjs@tmpdo@c\@nnil{\@tempwattrue
```

```
1561 \edef\bxjs@param@paper{papersize={\bxjs@tmpa}}}
```

```
1562 \expandafter\bxjs@tmpdo\bxjs@param@paper\@nnil
```

W,H の形式について。

```
1563 \if@tempwa\else
```

```
1564 \def\bxjs@tmpa{\@nil,\@nil}
```

```
1565 \def\bxjs@tmpdo#1,#2,#3\@nnil{%
```

```
1566 \def\bxjs@tmpb{#3}\ifx\bxjs@tmpa\bxjs@tmpb
```

```
1567 \@tempwattrue\edef\bxjs@param@paper{papersize={#1,#2}}\fi}
```

```
1568 \expandafter\bxjs@tmpdo\bxjs@param@paper,\@nil,\@nil\@nnil
```

```
1569 \fi
```

W*H の形式について。

```
1570 \if@tempwa\else
```

```
1571 \def\bxjs@tmpa{\@nil*\@nil}
```

```
1572 \def\bxjs@tmpdo#1*#2*#3\@nnil{%
```

```
1573 \def\bxjs@tmpb{#3}\ifx\bxjs@tmpa\bxjs@tmpb
```

```
1574 \@tempwattrue\edef\bxjs@param@paper{papersize={#1,#2}}\fi}
```

```
1575 \expandafter\bxjs@tmpdo\bxjs@param@paper*\@nil*\@nil\@nnil
```

```
1576 \fi
```

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

```
1577 \edef\bxjs@layout@paper{%
```

```
1578 \ifjsc@mag truedimen,\fi
```

```
1579 \if@landscape landscape,\fi
```

```
1580 \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。

```

1581 %<*article|report>
1582 \def\bxjs@layout@base{%
1583   headheight=\topskip,footskip=0.03367\paperheight,%
1584   headsep=\footskip-\topskip,includeheadfoot,%
1585 }
1586 \edef\bxjs@layout{\bxjs@layout@base
1587   hscale=0.76,hmarginratio=1:1,%
1588   vscale=0.83,vmarginratio=1:1,%
1589 }
1590 %</article|report>
1591 %<*book>
1592 \def\bxjs@layout@base{%
1593   headheight=\topskip,headsep=6\jsc@mmm,nofoot,includeheadfoot,%
1594 }
1595 \ifbxjs@layout@buggyhmargin      %---
1596 % アレ
1597 \edef\bxjs@layout{\bxjs@layout@base
1598   hmargin=36\jsc@mmm,hmarginratio=1:1,%
1599   vscale=0.83,vmarginratio=1:1,%
1600 }
1601 \else                            %---
1602 % 非アレ
1603 \edef\bxjs@layout{\bxjs@layout@base
1604   hmargin=18\jsc@mmm,%
1605   vscale=0.83,vmarginratio=1:1,%
1606 }
1607 \fi                              %---
1608 %</book>
1609 %<*slide>
1610 \def\bxjs@layout@base{%
1611   noheadfoot,%
1612 }
1613 \edef\bxjs@layout{\bxjs@layout@base
1614   hscale=0.9,hmarginratio=1:1,%
1615   vscale=0.944,vmarginratio=1:1,%
1616 }
1617 %</slide>

textwidth オプションの設定を反映する。

1618 %<!*book>
1619 \ifx\bxjs@textwidth@opt\undefined\else
1620   \jsSetQHLLength\@tempdima{\bxjs@textwidth@opt}
1621   \edef\bxjs@layout{\bxjs@layout width=\the\@tempdima,}
1622 \fi
1623 %</!book>
1624 \ifx\bxjs@number@of@lines@opt\undefined\else
1625   \bxjs@gset@tempcnta{\bxjs@number@of@lines@opt}
1626   \edef\bxjs@layout{\bxjs@layout lines=\the\@tempcnta,}
1627 \fi

```

`\fullwidth` [寸法レジスタ] ヘッダ・フッタ領域の横幅。

```
1628 \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。

```
1629 %<*book>
1630 \newcommand\jsTextWidthLimit{40}
1631 \@tempdima=\jsTextWidthLimit\Cwd
1632 \ifx\bxjs@textwidth@limit@opt\undefined\else
1633   \bxjs@gset@tempcnta{\bxjs@textwidth@limit@opt}
1634   \@tempdima=\@tempcnta\Cwd
1635 \fi
1636 \ifx\bxjs@textwidth@opt\undefined\else
1637   \jsSetQHLLength\@tempdima{\bxjs@textwidth@opt}
1638 \fi
1639 \edef\bxjs@textwidth@limit{\the\@tempdima}
1640 \ifdim\@tempdima=\jsTextWidthLimit\Cwd\else
1641   \bxjs@invscale\@tempdima{\strip@pt\Cwd}
1642   \long\edef\jsTextWidthLimit{\strip@pt\@tempdima}
1643 \fi
1644 %</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。

```
1645 \def\bxjs@preproc@layout{%
1646   \edef\bxjs@save@ht@strutbox{\the\ht\strutbox}\ht\strutbox=10\jsc@empt}
```

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

```
1647 \def\bxjs@postproc@layout{%
```

`geometry` のドライバを再設定する。

```
1648   \ifx\bxjs@geometry@driver\relax\else
1649     \let\Gm@driver\bxjs@geometry@driver
1650   \fi
```

`\ht\strutbox` の値を元に戻す。

```
1651   \ht\strutbox=\bxjs@save@ht@strutbox\relax
```

`\textwidth` の値を補正する。

```
1652   \ifbxjs@whole@zw@lines
1653     \@tempdimb=\textwidth
1654     \if@twocolumn \@tempdima=2\Cwd \else \@tempdima=1\Cwd \fi
1655     \advance\textwidth.005pt\relax
1656     \divide\textwidth\@tempdima \multiply\textwidth\@tempdima
1657     \advance\@tempdimb-\textwidth
1658     \advance\oddsidemargin 0.5\@tempdimb
```

```

1659 \advance\evensidemargin 0.5\@tempdimb
1660 \fi
1661 \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。

```

1662 %<*book>
1663 \@tempdima=\bxjs@textwidth@limit\relax
1664 \ifbxjs@whole@zw@lines
1665 \advance\@tempdima.005pt\relax
1666 \divide\@tempdima\Cwd \multiply\@tempdima\Cwd
1667 \fi
1668 \ifdim\textwidth>\@tempdima
1669 \textwidth=\@tempdima
1670 \addtolength\evensidemargin{\fullwidth-\textwidth}
1671 \fi
1672 %</book>

```

\textheight 関連の調整。

```

1673 \@tempdimb=\textheight
1674 \advance\textheight-\topskip
1675 \advance\textheight.005pt\relax
1676 \divide\textheight\baselineskip \multiply\textheight\baselineskip
1677 \advance\textheight\topskip
1678 \advance\@tempdimb-\textheight
1679 \advance\topmargin0.5\@tempdimb

```

\headheight 関連の調整。

```

1680 \@tempdima=\topskip
1681 \advance\headheight\@tempdima
1682 \advance\topmargin-\@tempdima

```

marginpar 関連の調整。

```

1683 \setlength\marginparsep{\columnsep}
1684 \setlength\marginparpush{\baselineskip}
1685 \setlength\marginparwidth{\paperwidth-\oddsidemargin-1truein%
1686   -\textwidth-10\jsc@mmm-\marginparsep}
1687 \ifbxjs@whole@zw@lines
1688 \divide\marginparwidth\Cwd \multiply\marginparwidth\Cwd
1689 \fi

```

連動する変数。

```

1690 \maxdepth=.5\topskip
1691 \stockwidth=\paperwidth
1692 \stockheight=\paperheight
1693 }

```

\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。

```

1694 \edef\jsGeometryOptions{%

```

```
1695 \bxjs@layout@paper,\bxjs@layout}
```

■geometry パッケージを読み込む

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

※ L^AT_EX のフックシステムがある場合はムニャムニャ。

```
1696 \def\bxjs@geometry@name{geometry}
1697 \ifbxjs@old@hook@system
1698 \let\bxjs@apply@bd@pre@geometry@hook\AtBeginDocument
1699 \else
1700 \def\bxjs@apply@bd@pre@geometry@hook{%
1701 \AddToHook{begindocument}{\bxjs@geometry@name}}
1702 \fi
```

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

```
1703 \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。

```
1704 \ifbxjs@papersize
1705 \ifx\bxjs@driver@given\bxjs@driver@@dvipdfmx
1706 \PassOptionsToPackage{dvipdfm}{geometry}
1707 \else\ifx\bxjs@driver@given\bxjs@driver@@dvimode
1708 \PassOptionsToPackage{dvipdfm}{geometry}
1709 \fi\fi
1710 \let\bxPapersizeSpecialDone=t
1711 \else
1712 \PassOptionsToPackage{driver=none}{geometry}
1713 \fi
```

ここで geometry を読み込む。

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

```
1714 \bxjs@apply@bd@pre@geometry@hook{%
1715 \bxjs@bd@pre@geometry@hook
1716 \@nameuse{ImposeOldLuaTeXBehavior}}
1717 \bxjs@preproc@layout
1718 \edef\bxjs@next{%
1719 \noexpand\RequirePackage[\bxjs@layout@paper,\bxjs@layout]{geometry}%
1720 }\bxjs@next
1721 \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。


```

1722 \let\bxjs@geometry@driver\Gm@driver
1723 \bxjs@postproc@layout

```

geometry のドライバ自動判別に対する前処理。

```

1724 \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。

```

1725   \@ifpackagelater{geometry}{2010/02/12}{\}%else
1726   \PackageError\bxjs@clsname
1727     {Your 'geometry' package is too old (< v5.0)}%
1728     {\@ehc}%
1729   \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。

```

1730   \ifjsWithpTeXng
1731   \ifx\Gm@driver\@empty
1732     \def\Gm@driver{pdftex}%
1733   \fi
1734   \fi}

```

`\setpagelayout` ページレイアウト設定のためのユーザ命令。

```

1735 \def\setpagelayout{%
1736   \bxjs@ifplus{\bxjs@setpagelayout@a\tw@}{\}%else
1737   \@ifstar{\bxjs@setpagelayout@a\@ne}{\bxjs@setpagelayout@a\z@}}
1738 \def\bxjs@setpagelayout@a#1#2{%
1739   \ifcase#1% modify
1740     \def\bxjs@next{\ifjsc@mag truedimen,\fi #2}%
1741   \or% reset(*)
1742     \def\bxjs@next{reset,\bxjs@layout@paper,#2}%
1743   \or% semireset(+)
1744     \def\bxjs@next{reset,\bxjs@layout@paper,\bxjs@layout@base,#2}%
1745   \fi
1746   \bxjs@preproc@layout
1747   \edef\bxjs@next{%
1748     \noexpand\geometry{\bxjs@next}%
1749   }\bxjs@next
1750   \bxjs@postproc@layout}

```

geometry パッケージを読み込まない

geometry=user の場合の処理。

```

1751 \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。

```

1752 \g@addto@macro\bxjs@begin@document@hook{%
1753   \ifdim\textwidth=.5\maxdimen
1754     \ClassError\bxjs@clsname
1755       {Page layout is not properly set}%
1756       {\@ehd}%
1757   \fi}
1758 \def\jsUseMinimalPageLayout{%
1759   \setlength{\textwidth}{6.5in}%
1760   \setlength{\textheight}{8in}}

   \setpagelayout はとりあえず無効にしておく。
1761 \let\bxjs@geometry@driver\relax
1762 \def\setpagelayout{%
1763   \bxjs@ifplus{\bxjs@pagelayout@a}{\%else
1764     \@ifstar{\bxjs@pagelayout@a}{\bxjs@pagelayout@a}}
1765 \def\bxjs@pagelayout@a#1{%
1766   \ClassError\bxjs@clsname
1767     {Command '\string\setpagelayout' is not supported,\MessageBreak
1768     because 'geometry' value is not 'class'}\@eha}
1769 %
1770 \fi\fi

```

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。

```

1771 %<*jsclasses>

```

■縦方向のスペース

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

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

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

```

1772 \setlength\topskip{1.38zw}%% from 10\jsc@mpt (2016-08-17)
1773 \if@slide
1774   \setlength\headheight{0\jsc@mpt}
1775 \else
1776   \setlength\headheight{20\jsc@mpt}%% from 2\topskip (2016-08-17); from \topskip (2003-
      06-26)
1777 \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) になっていましたが、ここでは A4 判のときちょうど 1cm となるように、`\paperheight` の 0.03367 倍 (最小 `\baselineskip`) としました。書籍については、フッタは使わないことにして、ゼロにしました。

```
1778 %<*article|kiyou>
1779 \if@slide
1780   \setlength\footskip{0pt}
1781 \else
1782   \setlength\footskip{0.03367\paperheight}
1783   \ifdim\footskip<\baselineskip
1784     \setlength\footskip{\baselineskip}
1785   \fi
1786 \fi
1787 %</article|kiyou>
1788 %<jspf>\setlength\footskip{9\jsc@mmm}
1789 %<*book>
1790 \if@report
1791   \setlength\footskip{0.03367\paperheight}
1792   \ifdim\footskip<\baselineskip
1793     \setlength\footskip{\baselineskip}
1794   \fi
1795 \else
1796   \setlength\footskip{0pt}
1797 \fi
1798 %</book>
1799 %<*report>
1800 \setlength\footskip{0.03367\paperheight}
1801 \ifdim\footskip<\baselineskip
1802   \setlength\footskip{\baselineskip}
1803 \fi
1804 %</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)。

```
1805 %<*article>
1806 \if@slide
1807   \setlength\headsep{0\jsc@mpt}
1808   \addtolength\headsep{-\topskip}%% added (2016-10-08)
1809   \addtolength\headsep{10\jsc@mpt}%% added (2016-10-08)
1810 \else
1811   \setlength\headsep{\footskip}
1812   \addtolength\headsep{-\topskip}
1813 \fi
1814 %</article>
```

```

1815 %<*book>
1816 \if@report
1817   \setlength\headsep{\footskip}
1818   \addtolength\headsep{-\topskip}
1819 \else
1820   \setlength\headsep{6\jsc@mmm}
1821   \addtolength\headsep{-\topskip}%% added (2016-10-08)
1822   \addtolength\headsep{10\jsc@mpt}%% added (2016-10-08)
1823 \fi
1824 %</book>
1825 %<*report>
1826 \setlength\headsep{\footskip}
1827 \addtolength\headsep{-\topskip}
1828 %</report>
1829 %<*jspf>
1830 \setlength\headsep{9\jsc@mmm}
1831 \addtolength\headsep{-\topskip}
1832 %</jspf>
1833 %<*kiyou>
1834 \setlength\headheight{0\jsc@mpt}
1835 \setlength\headsep{0\jsc@mpt}
1836 \addtolength\headsep{-\topskip}%% added (2016-10-08)
1837 \addtolength\headsep{10\jsc@mpt}%% added (2016-10-08)
1838 %</kiyou>

```

`\maxdepth` `\maxdepth` は本文最下行の最大の深さで、plain TeX や L^AT_EX 2.09 では 4pt に固定でした。L^AT_EX2e 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。

```
1839 \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。

```
1840 \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。

```

1841 %<*article>
1842 \if@slide
1843   \setlength\fullwidth{0.9\paperwidth}
1844 \else
1845   \setlength\fullwidth{0.76\paperwidth}
1846 \fi
1847 \if@twocolumn \@tempdima=2zw \else \@tempdima=1zw \fi
1848 \divide\fullwidth\@tempdima \multiply\fullwidth\@tempdima
1849 \setlength\textwidth{\fullwidth}
1850 %</article>
1851 %<*book>
1852 \if@report
1853   \setlength\fullwidth{0.76\paperwidth}
1854 \else
1855   \setlength\fullwidth{\paperwidth}
1856   \addtolength\fullwidth{-36\jsc@mmm}
1857 \fi
1858 \if@twocolumn \@tempdima=2zw \else \@tempdima=1zw \fi
1859 \divide\fullwidth\@tempdima \multiply\fullwidth\@tempdima
1860 \setlength\textwidth{\fullwidth}
1861 \if@report \else
1862   \if@twocolumn \else
1863     \ifdim \fullwidth>40zw
1864       \setlength\textwidth{40zw}
1865     \fi
1866   \fi
1867 \fi
1868 %</book>
1869 %<*report>
1870 \setlength\fullwidth{0.76\paperwidth}
1871 \if@twocolumn \@tempdima=2zw \else \@tempdima=1zw \fi
1872 \divide\fullwidth\@tempdima \multiply\fullwidth\@tempdima
1873 \setlength\textwidth{\fullwidth}
1874 %</report>
1875 %<*jspf>
1876 \setlength\fullwidth{50zw}
1877 \addtolength\fullwidth{8\jsc@mmm}
1878 \setlength\textwidth{\fullwidth}
1879 %</jspf>
1880 %<*kiyou>
1881 \setlength\fullwidth{48zw}
1882 \addtolength\fullwidth{\columnsep}
1883 \setlength\textwidth{\fullwidth}
1884 %</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)。

```
1885 %<*article|book|report>
1886 \if@slide
1887   \setlength{\textheight}{0.95\paperheight}
1888 \else
1889   \setlength{\textheight}{0.83\paperheight}
1890 \fi
1891 \addtolength{\textheight}{-10\jsc@empt}%% from -\topskip (2016-10-08); from -
    \headheight (2003-06-26)
1892 \addtolength{\textheight}{-\headsep}
1893 \addtolength{\textheight}{-\footskip}
1894 \addtolength{\textheight}{-\topskip}
1895 \divide\textheight\baselineskip
1896 \multiply\textheight\baselineskip
1897 %</article|book|report>
1898 %<jspf>\setlength{\textheight}{51\baselineskip}
1899 %<kiyou>\setlength{\textheight}{47\baselineskip}
1900 \addtolength{\textheight}{\topskip}
1901 \addtolength{\textheight}{0.1\jsc@empt}
1902 %<jspf>\setlength{\mathindent}{10\jsc@mmm}
```

`\flushbottom` [2016-07-18] `\textheight` に念のため 0.1 ポイント余裕を持たせているのと同様に、`\flushbottom` にも余裕を持たせます。元の \LaTeX 2_ϵ 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。

```
1903 \def\flushbottom{%
1904   \def\@textbottom{\vskip \z@ \@plus.1\jsc@empt}%
1905   \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。

```
1906 \setlength\marginparsep{\columnsep}
1907 \setlength\marginparpush{\baselineskip}
```

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

```

1908 \setlength{\oddsidemargin}{\paperwidth}
1909 \addtolength{\oddsidemargin}{-\fullwidth}
1910 \setlength{\oddsidemargin}{.5\oddsidemargin}
1911 \iftombow
1912   \addtolength{\oddsidemargin}{-1in}
1913 \else
1914   \addtolength{\oddsidemargin}{-\inv@mag in}
1915 \fi
1916 \setlength{\evensidemargin}{\oddsidemargin}
1917 \if@mparswitch
1918   \addtolength{\evensidemargin}{\fullwidth}
1919   \addtolength{\evensidemargin}{-\textwidth}
1920 \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。

```

1921 \setlength\marginparwidth{\paperwidth}
1922 \addtolength\marginparwidth{-\oddsidemargin}
1923 \addtolength\marginparwidth{-\inv@mag in}
1924 \addtolength\marginparwidth{-\textwidth}
1925 \addtolength\marginparwidth{-10\jsc@mmm}
1926 \addtolength\marginparwidth{-\marginparsep}
1927 \@tempdima=1zw
1928 \divide\marginparwidth\@tempdima
1929 \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)。

```

1930 \setlength\topmargin{\paperheight}
1931 \addtolength\topmargin{-\textheight}
1932 \if@slide
1933   \addtolength\topmargin{-\headheight}
1934 \else
1935   \addtolength\topmargin{-10\jsc@empt}%% from -\topskip (2016-10-08); from -
     \headheight (2003-06-26)
1936 \fi
1937 \addtolength\topmargin{-\headsep}

```

```

1938 \addtolength\topmargin{-\footskip}
1939 \setlength\topmargin{0.5\topmargin}
1940 %<kiyou>\setlength\topmargin{81truebp}
1941 \iftombow
1942 \addtolength\topmargin{-1in}
1943 \else
1944 \addtolength\topmargin{-\inv@mag in}
1945 \fi
1946 %</jsclasses>

```

■JS クラスと共通処理の開始

ここからのコードは以下の点を除いて JS クラスのものを踏襲する。

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

■脚注

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

ここは元々は

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

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

```
1947 \footnotesep=11\p@? \footnotesep=0.7\footnotesep
```

`\footins` `\skip\footins` は本文の最終行と最初の脚注との間の距離です。標準の 10 ポイントクラスでは 9 plus 4 minus 2 ポイントになっていますが、和文の行送りを考えてもうちょっと大きくします。

```
1948 \setlength{\skip\footins}{16\p@? \@plus 5\p@? \@minus 2\p@?}
```

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

`\c@topnumber` `topnumber` カウンタは本文ページ上部のフロートの最大数です。

[2003-08-23] ちょっと増やしました。

1949 `\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。

1950 `\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。

1951 `\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。

1952 `\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。

1953 `\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。

1954 `\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。

1955 `\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。

1956 `\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。

1957 `\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。

1958 `\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。

1959 `\setlength\floatsep {12\p@? \@plus 2\p@? \@minus 2\p@?}`

1960 `\setlength\textfloatsep{20\p@? \@plus 2\p@? \@minus 4\p@?}`

1961 `\setlength\intextsep {12\p@? \@plus 2\p@? \@minus 2\p@?}`

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

`\dbltextfloatsep` 1962 `\setlength\dblfloatsep {12\p@? \@plus 2\p@? \@minus 2\p@?}`

1963 `\setlength\dbltextfloatsep{20\p@? \@plus 2\p@? \@minus 4\p@?}`

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

```
\@fpbot 1964 \setlength\@fptop{0\p@? \@plus 1fil}
          1965 \setlength\@fpsep{8\p@? \@plus 2fil}
          1966 \setlength\@fpbot{0\p@? \@plus 1fil}
```

`\@dblftop` 段抜きフロートについての値です。

```
\@dblfpsep 1967 \setlength\@dblftop{0\p@? \@plus 1fil}
\@dblfpbot 1968 \setlength\@dblfpsep{8\p@? \@plus 2fil}
          1969 \setlength\@dblfpbot{0\p@? \@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 \LaTeX の標準クラス 2017/02/15 に合わせて, 同じ命令を追
`\pltx@cleartoleftpage` 加しました。

```
\pltx@cleartooddpage 1. \pltx@cleartorightpage: 右ページになるまでページを繰る命令
\pltx@cleartoevenpage 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。

```
1970 %\def\pltx@cleartorightpage{\clearpage\if@twoside
1971 % \ifodd\c@page
1972 % \iftdir
1973 % \hbox{}\thispagestyle{empty}\newpage
1974 % \if@twocolumn\hbox{}\newpage\fi
1975 % \fi
1976 % \else
1977 % \ifydir
1978 % \hbox{}\thispagestyle{empty}\newpage
1979 % \if@twocolumn\hbox{}\newpage\fi
1980 % \fi
1981 % \fi\fi}
1982 %\def\pltx@cleartoleftpage{\clearpage\if@twoside
1983 % \ifodd\c@page
1984 % \ifydir
1985 % \hbox{}\thispagestyle{empty}\newpage
1986 % \if@twocolumn\hbox{}\newpage\fi
1987 % \fi
1988 % \else
1989 % \iftdir
1990 % \hbox{}\thispagestyle{empty}\newpage
1991 % \if@twocolumn\hbox{}\newpage\fi
1992 % \fi
1993 % \fi\fi}
```

```

1994 \def\pltx@cleartooddpage{\clearpage\if@twoside
1995   \ifodd\c@page\else
1996     \hbox{}\thispagestyle{empty}\newpage
1997     \if@twocolumn\hbox{}\newpage\fi
1998   \fi\fi}
1999 \def\pltx@cleartoevenpage{\clearpage\if@twoside
2000   \ifodd\c@page
2001     \hbox{}\thispagestyle{empty}\newpage
2002     \if@twocolumn\hbox{}\newpage\fi
2003   \fi\fi}

```

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

```

2004 \let\pltx@cleartorightpage\pltx@cleartooddpage
2005 \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。
2006 \ifdim\vsize=\z@
2007 \begingroup
2008 \toks@{\expandafter{\clearpage}}
2009 \xdef\clearpage{\noexpand\ifbxjs@after@preamble\the\toks@\noexpand\fi}
2010 \endgroup
2011 \fi

```

`\cleardoublepage` [2017-02-24] コミュニティ版 p \LaTeX の標準クラス 2017/02/15 に合わせて、`report` と `book` クラスの場合に `\cleardoublepage` を再定義します。

```

2012 %<*book|report>
2013 \if@openleft
2014   \let\cleardoublepage\pltx@cleartoleftpage
2015 \else\if@openright
2016   \let\cleardoublepage\pltx@cleartorightpage
2017 \fi\fi
2018 %</book|report>

```

7 ページスタイル

ページスタイルとして、 \LaTeX 2_ε (欧文版) の標準クラスでは `empty`, `plain`, `headings`, `myheadings` があります。このうち `empty`, `plain` スタイルは \LaTeX 2_ε 本体で定義されています。

アスキーのクラスファイルでは `headnombre`, `footnombre`, `bothstyle`, `jpl@in` が追加されていますが、ここでは欧文標準のものだけにしました。

ページスタイルは `\ps@...` の形のマクロで定義されています。

`\@evenhead` `\@oddhead`, `\@oddfoot`, `\@evenhead`, `\@evenfoot` は偶数・奇数ページの柱 (ヘッダ, `\@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。

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

柱を設定する命令は、右の柱が左の柱の下位にある場合は十分まともに動作します。たとえば左マークを `\chapter`、右マークを `\section` で変更する場合はこれにあたります。しかし、同一ページに複数の `\markboth` があると、おかしい結果になることがあります。

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

`\ps@empty empty` ページスタイルの定義です。L^AT_EX 本体で定義されているものをコメントアウトした形で載せておきます。

```

2019 % \def\ps@empty{%
2020 %   \let\mkboth\gobbletwo
2021 %   \let\oddhead\empty
2022 %   \let\oddfoot\empty
2023 %   \let\evenhead\empty
2024 %   \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。

```

2025 \def\ps@plainfoot{%
2026   \let\mkboth\gobbletwo
2027   \let\oddhead\empty
2028   \def\oddfoot{\normalfont\hfil\thepage\hfil}%
2029   \let\evenhead\empty
2030   \let\evenfoot\oddfoot}
2031 \def\ps@plainhead{%
2032   \let\mkboth\gobbletwo
2033   \let\oddfoot\empty
2034   \let\evenfoot\empty
2035   \def\evenhead{%
2036     \ifmparswitch \hss \fi
2037     \hbox to \fullwidth{\textbf{\thepage}\hfil}%
2038     \ifmparswitch\else \hss \fi}%
2039   \def\oddhead{%
2040     \hbox to \fullwidth{\hfil\textbf{\thepage}}\hss}}
2041 %<book>\let\ps@plain\ps@plainhead
2042 %<!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。

```
2043 %<*article|slide>
2044 \if@twoside
2045   \def\ps@headings{%
2046     \let\@oddfoot\@empty
2047     \let\@evenfoot\@empty
2048     \def\@evenhead{\if@mparswitch \hss \fi
2049       \underline{\hbox to \fullwidth{\textbf{\thepage}\hfil\leftmark}}}%
2050     \if@mparswitch\else \hss \fi}%
2051   \def\@oddhead{%
2052     \underline{%
2053       \hbox to \fullwidth{\rightmark}\hfil\textbf{\thepage}}}\hss}%
2054   \let\@mkboth\markboth
2055   \def\sectionmark##1{\markboth{%
2056     \ifnum \c@secnumdepth >\z@ \bxjs@label@sect{section}\hskip1\jsZw\fi
2057     ##1}}}%
2058   \def\subsectionmark##1{\markright{%
2059     \ifnum \c@secnumdepth >\@ne \bxjs@label@sect{subsection}\hskip1\jsZw\fi
2060     ##1}}}%
2061 }
2062 \else % if not twoside
2063   \def\ps@headings{%
2064     \let\@oddfoot\@empty
2065     \def\@oddhead{%
2066       \underline{%
2067         \hbox to \fullwidth{\rightmark}\hfil\textbf{\thepage}}}\hss}%
2068     \let\@mkboth\markboth
2069     \def\sectionmark##1{\markright{%
2070       \ifnum \c@secnumdepth >\z@ \bxjs@label@sect{section}\hskip1\jsZw\fi
2071       ##1}}}%
2072 \fi
2073 %</article|slide>
```

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

```
2074 %<*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。

```
2075 \def\bxjs@maybe@autoxspacing{%
2076   \ifx\autoxspacing\undefined\else \autoxspacing \fi}
```

```
2077 \newif\if@omit@number
2078 \def\ps@headings{%
2079   \let\@oddfoot\@empty
2080   \let\@evenfoot\@empty
```

```

2081 \def\@evenhead{%
2082   \if@mparswitch \hss \fi
2083   \underline{\hbox to \fullwidth{\bxjs@maybe@autoxspacing
2084     \textbf{\thepage}\hfil\leftmark}}}%
2085   \if@mparswitch\else \hss \fi}%
2086 \def\@oddhead{\underline{\hbox to \fullwidth{\bxjs@maybe@autoxspacing
2087   {\if@twoside\rightmark\else\leftmark\fi}\hfil\textbf{\thepage}}}\hss}%
2088 \let\@mkboth\markboth
2089 \def\chaptermark##1{\markboth{%
2090   \ifnum \c@secnumdepth >\m@ne
2091     \if@mainmatter
2092       \if@omit@number\else
2093         \@chapapp\thechapter\@chappos\hskip1\jsZw
2094       \fi
2095     \fi
2096   \fi
2097   ##1}{}}}%
2098 \def\sectionmark##1{\markright{%
2099   \ifnum \c@secnumdepth >\z@ \bxjs@label@sect{section}\hskip1\jsZw\fi
2100   ##1}}}%
2101 %</book|report>

```

最後は学会誌の場合です。

```

2102 %<*jspf>
2103 \def\ps@headings{%
2104   \def\@oddfoot{\normalfont\hfil\thepage\hfil}
2105   \def\@evenfoot{\normalfont\hfil\thepage\hfil}
2106   \def\@oddhead{\normalfont\hfil \@title \hfil}
2107   \def\@evenhead{\normalfont\hfil プラズマ・核融合学会誌\hfil}
2108 %</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。

```

2109 \def\ps@myheadings{%
2110   \let\@oddfoot\@empty\let\@evenfoot\@empty
2111   \def\@evenhead{%
2112     \if@mparswitch \hss \fi%
2113     \hbox to \fullwidth{\thepage\hfil\leftmark}%
2114     \if@mparswitch\else \hss \fi}%
2115   \def\@oddhead{%
2116     \hbox to \fullwidth{\rightmark\hfil\thepage}\hss}%
2117   \let\@mkboth\@gobbletwo
2118 %<book|report> \let\chaptermark\@gobble
2119   \let\sectionmark\@gobble
2120 %<!\book&!\report> \let\subsectionmark\@gobble
2121 }

```

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 2122 % \newcommand*{\title}[1]{\gdef\@title{#1}}
2123 % \newcommand*{\author}[1]{\gdef\@author{#1}}
2124 % \newcommand*{\date}[1]{\gdef\@date{#1}}
2125 % \date{\today}
```

`\subtitle` 副題を設定する。

`\jsSubtitle` ※プレアンプルにおいて `\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: `\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。

```
2126 \newcommand*{\jsSubtitle}[1]{\gdef\bxjs@subtitle{#1}}
2127 %\let\bxjs@subtitle\@undefined

\title にフックを入れる。
2128 \renewcommand*{\title}[1]{\bxjs@decl@subtitle\gdef\@title{#1}}
2129 \g@addto@macro\bxjs@begin@document@hook{\bxjs@decl@subtitle}
2130 \def\bxjs@decl@subtitle{%
2131   \global\let\bxjs@decl@subtitle\relax
2132   \ifx\subtitle\@undefined
2133     \global\let\subtitle\jsSubtitle
2134   \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。

```
2135 \def\bxjs@annihilate@subtitle{%
2136   \ifx\subtitle\jsSubtitle \global\let\subtitle\relax \fi
2137   \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 2138 %<*jspf>
2139 \newcommand*{\etitle}[1]{\gdef\@etitle{#1}}
\keywords 2140 \newcommand*{\eauthor}[1]{\gdef\@eauthor{#1}}
2141 \newcommand*{\keywords}[1]{\gdef\@keywords{#1}}
2142 \newcommand*{\email}[1]{\gdef\authors@mail{#1}}
2143 \newcommand*{\AuthorsEmail}[1]{\gdef\authors@mail{author's e-mail:\ #1}}
2144 %</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。

```

2145 \def\plainifnotempty{%
2146   \ifx \@oddhead \@empty
2147     \ifx \@oddfoot \@empty
2148       \else
2149         \thispagestyle{plainfoot}%
2150       \fi
2151     \else
2152       \thispagestyle{plainhead}%
2153     \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。

```

2154 %<article|book|report|slide>
2155 \if@titlepage
2156   \newcommand{\maketitle}{%
2157     \begin{titlepage}%
2158       \let\footnotesize\small
2159       \let\footnoterule\relax
2160       \let\footnote\thanks
2161       \null\vfil
2162       \if@slide
2163         {\footnotesize \@date}%
2164         \begin{center}
2165           \mbox{} \ll[1\jsZw]
2166           \large
2167           {\maybeblue\hrule height0\p@? depth2\p@?\relax}\par
2168           \jsc@smallskip
2169           \@title
2170           \ifx\bxjs@subtitle\@undefined\else
2171             \par\vskip\z@
2172             {\small \bxjs@subtitle\par}
2173           \fi
2174           \jsc@smallskip
2175           {\maybeblue\hrule height0\p@? depth2\p@?\relax}\par
2176           \vfill
2177           {\small \@author}%
2178         \end{center}
2179       \else
2180       \vskip 60\p@?

```



```

2181 \begin{center}%
2182 {\LARGE \@title \par}%
2183 \ifx\bxjs@subtitle\@undefined\else
2184 \vskip5\p@?
2185 {\normalsize \bxjs@subtitle\par}
2186 \fi
2187 \vskip 3em%
2188 {\large
2189 \lineskip .75em
2190 \begin{tabular}[t]{c}%
2191 \author
2192 \end{tabular}\par}%
2193 \vskip 1.5em
2194 {\large \@date \par}%
2195 \end{center}%
2196 \fi
2197 \par
2198 \@thanks\vfil\null
2199 \end{titlepage}%
2200 \setcounter{footnote}{0}%
2201 \global\let\thanks\relax
2202 \global\let\maketitle\relax
2203 \global\let\@thanks\@empty
2204 \global\let\@author\@empty
2205 \global\let\@date\@empty
2206 \global\let\@title\@empty
2207 \global\let\title\relax
2208 \global\let\author\relax
2209 \global\let\date\relax
2210 \global\let\and\relax
2211 \bxjs@annihilate@subtitle
2212 }%
2213 \else
2214 \newcommand{\maketitle}{\par
2215 \begingroup
2216 \renewcommand\thefootnote{\@fnsymbol\c@footnote}%
2217 \def\@makefnmark{\rlap{\@textsuperscript{\normalfont\@thefnmark}}}%
2218 \long\def\@makefntext##1{\advance\leftskip 3\jsZw
2219 \parindent 1\jsZw\noindent
2220 \llap{\@textsuperscript{\normalfont\@thefnmark}\hskip0.3\jsZw}##1}%
2221 \if@twocolumn
2222 \ifnum \col@number=\@ne
2223 \maketitle
2224 \else
2225 \twocolumn[\maketitle]%
2226 \fi
2227 \else
2228 \newpage
2229 \global\@topnum\z@ % Prevents figures from going at top of page.

```

```

2230     \@maketitle
2231     \fi
2232     \plainifnotempty
2233     \@thanks
2234 \endgroup
2235 \setcounter{footnote}{0}%
2236 \global\let\thanks\relax
2237 \global\let\maketitle\relax
2238 \global\let\@thanks\@empty
2239 \global\let\@author\@empty
2240 \global\let\@date\@empty
2241 \global\let\@title\@empty
2242 \global\let\title\relax
2243 \global\let\author\relax
2244 \global\let\date\relax
2245 \global\let\and\relax
2246 \bxjs@annihilate@subtitle
2247 }

```

`\@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。

```

2248 \def\@maketitle{%
2249   \newpage\null
2250   \vskip 2em
2251   \begin{center}%
2252     \let\footnote\thanks
2253     {\LARGE \@title \par}%
2254     \ifx\bxjs@subtitle\@undefined\else
2255       \vskip3\p@?
2256       {\normalsize \bxjs@subtitle\par}
2257     \fi
2258     \vskip 1.5em
2259     {\large
2260       \lineskip .5em
2261       \begin{tabular}[t]{c}%
2262         \@author
2263       \end{tabular}\par}%
2264     \vskip 1em
2265     {\large \@date}%
2266   \end{center}%
2267   \par\vskip 1.5em
2268 %<article|slide>   \ifvoid\@abstractbox\else\centerline{\box\@abstractbox}\vskip1.5em\fi
2269 }
2270 \fi
2271 %</article|book|report|slide>
2272 %<*jspf>
2273 \newcommand{\maketitle}{\par
2274   \begin{group
2275     \renewcommand\thefootnote{\@fnsymbol\c@footnote}%
2276     \def\@makefnmark{\rlap{\@textsuperscript{\normalfont\@thefnmark}}}%

```

```

2277 \long\def\@makefntext##1{\advance\leftskip 3\jsZw
2278 \parindent 1\jsZw\noindent
2279 \llap{\@textsuperscript{\normalfont\@thefnmark}\hskip0.3\jsZw}##1}%
2280 \twocolumn[\@maketitle]%
2281 \plainifnotempty
2282 \@thanks
2283 \endgroup
2284 \setcounter{footnote}{0}%
2285 \global\let\thanks\relax
2286 \global\let\maketitle\relax
2287 \global\let\@thanks\@empty
2288 \global\let\@author\@empty
2289 \global\let\@date\@empty
2290 % \global\let\@title\@empty % \@title は柱に使う
2291 \global\let\title\relax
2292 \global\let\author\relax
2293 \global\let\date\relax
2294 \global\let\and\relax
2295 \ifx\authors@mail\@undefined\else{%
2296 \def\@makefntext{\advance\leftskip 3\jsZw \parindent -3\jsZw}%
2297 \footnotetext[0]{\itshape\authors@mail}%
2298 }\fi
2299 \global\let\authors@mail\@undefined}
2300 \def\@maketitle{%
2301 \newpage\null
2302 \vskip 6em % used to be 2em
2303 \begin{center}
2304 \let\footnote\thanks
2305 \ifx\@title\@undefined\else{\LARGE\headfont\@title\par}\fi
2306 \lineskip .5em
2307 \ifx\@author\@undefined\else
2308 \vskip 1em
2309 \begin{tabular}[t]{c}%
2310 \@author
2311 \end{tabular}\par
2312 \fi
2313 \ifx\@etitle\@undefined\else
2314 \vskip 1em
2315 {\large \@etitle \par}%
2316 \fi
2317 \ifx\@eauthor\@undefined\else
2318 \vskip 1em
2319 \begin{tabular}[t]{c}%
2320 \@eauthor
2321 \end{tabular}\par
2322 \fi
2323 \vskip 1em
2324 \@date
2325 \end{center}

```

```

2326 \vskip 1.5em
2327 \centerline{\box\@abstractbox}
2328 \ifx\@keywords\@undefined\else
2329 \vskip 1.5em
2330 \centerline{\parbox{157\jscmmm}{\textsf{Keywords:}}\ \small\@keywords}}
2331 \fi
2332 \vskip 1.5em}
2333 %</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。

```

2334 \def\bxjs@label@sect#1{%
2335 \expandafter\ifx\csname label#1\endcsname\relax
2336 \csname the#1\endcsname
2337 \else \csname label#1\endcsname
2338 \fi}
2339 \def\@secntformat#1{\bxjs@label@sect{#1}\quad}

```

`\@secapp` 節番号の接頭辞。

`\@secpos` 節番号の接尾辞。

```

2340 \ifnum\bxjs@label@section=\bxjs@label@section@@compat\else
2341 \def\@secapp{\presectionname}
2342 \def\@secpos{\postsectionname}
2343 \fi

```

`\labelsection` 節番号の出力書式。

```

2344 \ifnum\bxjs@label@section=\bxjs@label@section@@modern
2345 \def\labelsection{\@secapp\thesection\@secpos}
2346 \fi

```

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

```

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

```

それぞれの引数の意味は次の通りです。

名 ユーザレベルコマンドの名前です (例: section)。

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

字下げ 見出しの字下げ量です。

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

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

スタイル 見出しの文字スタイルの設定です。

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

別見出し 目次や柱に出力する見出しです。

見出し 見出しです。

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

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

```
2347 \def\@startsection#1#2#3#4#5#6{%
2348   \if@noskipsec \leavevmode \fi
2349   \par
2350   % 見出し上の空きを \@tempskipa にセットする
2351   \@tempskipa #4\relax
2352   % \@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
2353   \if@english \@afterindentfalse \else \@afterindenttrue \fi
2354   % 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
2355   \ifdim \@tempskipa <\z@
2356     \@tempskipa -\@tempskipa \@afterindentfalse
2357   \fi
2358   \if@nobreak
2359     \everypar{\everyparhook}% これは間違い
2360     \everypar{}%
2361   \else
2362     \addpenalty\@secpenalty
2363     % 次の行は削除
2364     % \addvspace\@tempskipa
2365     % 次の \noindent まで追加
2366     \ifdim \@tempskipa >\z@
2367       \if@slide\else
2368         \null
2369         \vspace*{-\baselineskip}%
2370       \fi
2371       \vskip\@tempskipa
2372     \fi
2373   \fi
2374   \noindent
2375   % 追加終わり
2376   \@ifstar
```

```

2377     {\@ssect{#3}{#4}{#5}{#6}}%
2378     {\@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。

```

2379 \ifnum\bxjs@everyparhook=\bxjs@everyparhook@@compat
2380   \let\bxjs@if@ceph\@firstofone
2381 \else \let\bxjs@if@ceph\@gobble
2382 \fi

```

```

2383 \def\@sect#1#2#3#4#5#6[#7]#8{%
2384   \ifnum #2>\c@secnumdepth
2385     \let\@svsec\@empty
2386   \else
2387     \refstepcounter{#1}%
2388     \protected@edef\@svsec{\@secntformat{#1}\relax}%
2389   \fi
2390 % 見出し後の空きを \@tempskipa にセット
2391   \@tempskipa #5\relax
2392 % 条件判断の順序を入れ替えました
2393   \ifdim \@tempskipa<\z@
2394     \def\@svsechd{%
2395       #6{\hskip #3\relax
2396         \@svsec #8}%
2397       \csname #1mark\endcsname{#7}%
2398       \addcontentsline{toc}{#1}{%
2399         \ifnum #2>\c@secnumdepth \else
2400           \protect\numberline{\bxjs@label@sect{#1}}%
2401         \fi
2402         #7}}% 目次にフルネームを載せるなら #8
2403   \else
2404     \begingroup
2405       \interlinepenalty \@M % 下から移動
2406       #6{%
2407         \@hangfrom{\hskip #3\relax\@svsec}%
2408       %   \interlinepenalty \@M % 上に移動
2409         #8\@par}%
2410     \endgroup
2411     \csname #1mark\endcsname{#7}%
2412     \addcontentsline{toc}{#1}{%
2413       \ifnum #2>\c@secnumdepth \else
2414         \protect\numberline{\bxjs@label@sect{#1}}%
2415       \fi

```

```

2416      #7}% 目次にフルネームを載せるならここは #8
2417    \fi
2418    \@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。

```

2419 \def\@xsect#1{%
2420 % 見出しの後ろの空きを \@tempskipa にセット
2421   \@tempskipa #1\relax
2422 % 条件判断の順序を変えました
2423   \ifdim \@tempskipa<\z@
2424     \nobreakfalse
2425     \global\@noskipsecttrue
2426     \everypar{%
2427       \if@noskipsec
2428         \global\@noskipsecfalse
2429         {\setbox\z@\lastbox}%
2430         \clubpenalty\@M
2431         \begingroup \@svsechd \endgroup
2432         \unskip
2433         \@tempskipa #1\relax
2434         \hskip -\@tempskipa
2435       \else
2436         \clubpenalty \@clubpenalty
2437         \everypar\expandafter{\bxjs@if@ceph\everyparhook}%

```

TODO: ↑ ナニコレ？

```

2438     \fi\bxjs@if@ceph\everyparhook}%
2439   \else
2440     \par \nobreak
2441     \vskip \@tempskipa
2442     \@afterheading
2443   \fi
2444   \if@slide
2445     {\vskip\if@twocolumn-5\jsc@mpt\else-6\jsc@mpt\fi
2446     \maybeblue\hrule height0\jsc@mpt depth1\jsc@mpt
2447     \vskip\if@twocolumn 4\jsc@mpt\else 7\jsc@mpt\fi\relax}%
2448   \fi
2449   \par % 2000-12-18
2450   \ignorespaces}
2451 \def\@ssect#1#2#3#4#5{%
2452   \@tempskipa #3\relax
2453   \ifdim \@tempskipa<\z@
2454     \def\@svsechd{#4{\hskip #1\relax #5}}%
2455   \else
2456     \begingroup
2457     #4{%

```

```

2458      \@hangfrom{\hskip #1}%
2459      \interlinepenalty \@M #5\@@par}%
2460  \endgroup
2461  \fi
2462  \@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 2463 \newcommand*\chaptermark[1]{%
\subsubsectionmark 2464 % \newcommand*\sectionmark[1]{%
2465 % \newcommand*\subsectionmark[1]{%
\paragraphmark 2466 % \newcommand*\subsubsectionmark[1]{%
\subparagraphmark 2467 % \newcommand*\paragraphmark[1]{%
2468 % \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。

```

2469 %<!book&!report>\setcounter{secnumdepth}{3}
2470 %<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 2471 \newcounter{part}
2472 %<book|report>\newcounter{chapter}
\c@subsubsection 2473 %<book|report>\newcounter{section}[chapter]
\c@paragraph 2474 %<!book&!report>\newcounter{section}
\c@subparagraph 2475 \newcounter{subsection}[section]
2476 \newcounter{subsubsection}[subsection]
2477 \newcounter{paragraph}[subsubsection]
2478 \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。

```

2479 \renewcommand{\thepart}{\@Roman\c@part}
2480 %<*&!book&!report>

```



```

2481 \ifnum\bxjs@label@section=\bxjs@label@section@@compat
2482 \renewcommand{\thesection}{\presectionname\@arabic\c@section\postsectionname}
2483 \renewcommand{\thesubsection}{\@arabic\c@section.\@arabic\c@subsection}
2484 \else
2485 \renewcommand{\thesection}{\@arabic\c@section}
2486 \renewcommand{\thesubsection}{\thesection.\@arabic\c@subsection}
2487 \fi
2488 %</!book&!report>
2489 %<*book|report>
2490 \renewcommand{\thechapter}{\@arabic\c@chapter}
2491 \renewcommand{\thesection}{\thechapter.\@arabic\c@section}
2492 \renewcommand{\thesubsection}{\thesection.\@arabic\c@subsection}
2493 %</book|report>
2494 \renewcommand{\thesubsubsection}{%
2495   \thesubsection.\@arabic\c@subsubsection}
2496 \renewcommand{\theparagraph}{%
2497   \thesubsubsection.\@arabic\c@paragraph}
2498 \renewcommand{\thesubparagraph}{%
2499   \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。

```

2500 %<book|report>\newcommand{\@chapapp}{\prechaptername}
2501 %<book|report>\newcommand{\@chappos}{\postchaptername}

```

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

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

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

```

2502 %<*book|report>
2503 \newcommand\frontmatter{%
2504   \pltx@cleartooddpage
2505   \@mainmatterfalse
2506   \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。

```

2507 \newcommand\mainmatter{%
2508   \pltx@cleartooddpage
2509   \@mainmattertrue
2510   \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。

```
2511 \newcommand\backmatter{%
2512   \if@openleft
2513     \cleardoublepage
2514   \else\if@openright
2515     \cleardoublepage
2516   \else
2517     \clearpage
2518   \fi\fi
2519   \@mainmatterfalse}
2520 %</book|report>
```

■部

`\part` 新しい部を始めます。

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

```
\secdef{星なし}{星あり}
```

星なし * のない形の定義です。

星あり * のある形の定義です。

`\secdef` は次のようにして使います。

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

まず `book` と `report` のクラス以外です。

```
2521 %<!*book&!report>
2522 \newcommand\part{%
2523   \if@noskipsec \leavevmode \fi
2524   \par
2525   \addvspace{4ex}%
2526   \if@english \@afterindentfalse \else \@afterindenttrue \fi
2527   \secdef\@part\@spart}
2528 %</!book&!report>
```

`book` および `report` クラスの場合は、少し複雑です。

```
2529 %<*book|report>
2530 \newcommand\part{%
2531   \if@openleft
2532     \cleardoublepage
2533   \else\if@openright
2534     \cleardoublepage
2535   \else
2536     \clearpage
2537   \fi\fi
2538   \thispagestyle{empty}% 欧文用標準スタイルでは plain
```

```

2539 \if@twocolumn
2540 \onecolumn
2541 \@restonecoltrue
2542 \else
2543 \@restonecolfalse
2544 \fi
2545 \null\vfil
2546 \secdef\@part\@spart}
2547 %</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。

```

2548 %<!*book&!report>
2549 \def\@part[#1]#2{%
2550 \ifnum \c@secnumdepth >\m@ne
2551 \refstepcounter{part}%
2552 \addcontentsline{toc}{part}{%
2553 \prepartname\thepart\postpartname\hspace{1\jsZw}#1}%
2554 \else
2555 \addcontentsline{toc}{part}{#1}%
2556 \fi
2557 \markboth{}{}%
2558 {\parindent\z@
2559 \raggedright
2560 \interlinepenalty \@M
2561 \normalfont
2562 \ifnum \c@secnumdepth >\m@ne
2563 \Large\headfont\prepartname\thepart\postpartname
2564 \par\nobreak
2565 \fi
2566 \huge \headfont #2%
2567 \markboth{}{}\par}%
2568 \nobreak
2569 \vskip 3ex
2570 \@afterheading}
2571 %</!*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。

```

2572 %<*book|report>
2573 \def\@part[#1]#2{%
2574 \ifnum \c@secnumdepth >-2\relax
2575 \refstepcounter{part}%
2576 \addcontentsline{toc}{part}{%
2577 \prepartname\thepart\postpartname\hspace{1\jsZw}#1}%
2578 \else
2579 \addcontentsline{toc}{part}{#1}%
2580 \fi
2581 \markboth{}{}%

```

```

2582 {\centering
2583 \interlinepenalty \@M
2584 \normalfont
2585 \ifnum \c@secnumdepth >-2\relax
2586 \huge\headfont \prepartname\thepart\postpartname
2587 \par\vskip20\p@?
2588 \fi
2589 \Huge \headfont #2\par}%
2590 \@endpart}
2591 %</book|report>

```

`\@spart` 番号を付けない部です。

```

2592 %<*!book&!report>
2593 \def\@spart#1{%
2594 \parindent \z@ \raggedright
2595 \interlinepenalty \@M
2596 \normalfont
2597 \huge \headfont #1\par}%
2598 \nobreak
2599 \vskip 3ex
2600 \@afterheading}
2601 %</!book&!report>
2602 %<*book|report>
2603 \def\@spart#1{%
2604 \centering
2605 \interlinepenalty \@M
2606 \normalfont
2607 \Huge \headfont #1\par}%
2608 \@endpart}
2609 %</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。

```

2610 %<*book|report>
2611 \def\@endpart{\vfil\newpage
2612 \if@twoside
2613 \if@openleft %% added (2017/02/24)
2614 \null\thispagestyle{empty}\newpage
2615 \else\if@openright %% added (2016/12/13)
2616 \null\thispagestyle{empty}\newpage
2617 \fi\fi %% added (2016/12/13, 2017/02/24)
2618 \fi
2619 \if@restonecol
2620 \twocolumn
2621 \fi}

```

2622 %</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。

```
2623 %<*book|report>
2624 \newcommand{\chapter}{%
2625   \ifopenleft\cleardoublepage\else
2626   \ifopenright\cleardoublepage\else\clearpage\fi\fi
2627   \plainifnotempty % 元: \thispagestyle{plain}
2628   \global\@topnum\z@
2629   \ifenglish \@afterindentfalse \else \@afterindenttrue \fi
2630   \secdef
2631     {\@omit@numberfalse\@chapter}%
2632     {\@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。

```
2633 \def\@chapter[#1]#2{%
2634   \ifnum \c@secnumdepth >\m@ne
2635     \if@mainmatter
2636       \refstepcounter{chapter}%
2637       \typeout{\@chapapp\thechapter\@chappos}%
2638       \addcontentsline{toc}{chapter}%
2639         {\protect\numberline
2640 %       %{\ifenglish\thechapter\else\@chapapp\thechapter\@chappos\fi}%
2641         {\@chapapp\thechapter\@chappos}%
2642         #1}%
2643     \else\addcontentsline{toc}{chapter}{#1}\fi
2644   \else
2645     \addcontentsline{toc}{chapter}{#1}%
2646   \fi
2647   \chaptermark{#1}%
2648   \addtocontents{lof}{\protect\addvspace{10\jsc@empt}}%
2649   \addtocontents{lot}{\protect\addvspace{10\jsc@empt}}%
2650   \if@twocolumn
2651     \@topnewpage[\@makechapterhead{#2}]%
2652   \else
2653     \@makechapterhead{#2}%
2654     \@afterheading
2655   \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。

```
2656 \def\@makechapterhead#1{%
2657   \vspace*{2\Cvs}% 欧文は 50pt
2658   {\parindent \z@ \raggedright \normalfont
2659     \ifnum \c@secnumdepth >\m@ne
```

```

2660      \if@mainmatter
2661      \huge\headfont \chapapp\thechapter\@chappos
2662      \par\nobreak
2663      \vskip \Cvs % 欧文は 20pt
2664      \fi
2665      \fi
2666      \interlinepenalty\@M
2667      \Huge \headfont #1\par\nobreak
2668      \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。

```

2669 \def\@schapter#1{%
2670   \chaptermark{#1}%
2671   \if@twocolumn
2672     \topnewpage[\@makeschapterhead{#1}]%
2673   \else
2674     \@makeschapterhead{#1}\@afterheading
2675   \fi}

```

`\@makeschapterhead` 番号なしの章見出しです。

```

2676 \def\@makeschapterhead#1{%
2677   \vspace*{2\Cvs}% 欧文は 50pt
2678   {\parindent \z@ \raggedright
2679     \normalfont
2680     \interlinepenalty\@M
2681     \Huge \headfont #1\par\nobreak
2682     \vskip 3\Cvs}} % 欧文は 40pt
2683 %</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。

```

2684 \if@twocolumn
2685   \newcommand{\section}{%
2686     %<jspf>\ifx\maketitle\relax\else\maketitle\fi
2687     \@startsection{section}{1}{\z@}%
2688     %<!kiyou> {0.6\Cvs}{0.4\Cvs}%
2689     %<kiyou> { \Cvs}{0.5\Cvs}%
2690     % {\normalfont\large\headfont\@secapp}}
2691     {\normalfont\large\headfont\raggedright}}
2692 \else
2693   \newcommand{\section}{%
2694     \if@slide\clearpage\fi
2695     \@startsection{section}{1}{\z@}%
2696     {\Cvs \@plus.5\Cdp \@minus.2\Cdp}% 前アキ
2697     {.5\Cvs \@plus.3\Cdp}% 後アキ

```

```

2698 %    {\normalfont\Large\headfont\@secapp}}
2699      {\normalfont\Large\headfont\raggedright}}
2700 \fi

```

`\subsection` 同上です。

```

2701 \if@twocolumn
2702   \newcommand{\subsection}{\@startsection{subsection}{2}{\z@}%
2703     {\z@}{\if@slide .4\Cvs \else \z@ \fi}%
2704     {\normalfont\normalsize\headfont}}
2705 \else
2706   \newcommand{\subsection}{\@startsection{subsection}{2}{\z@}%
2707     {\Cvs \@plus.5\Cdp \@minus.2\Cdp}% 前アキ
2708     {.5\Cvs \@plus.3\Cdp}% 後アキ
2709     {\normalfont\large\headfont}}
2710 \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)。

```

2711 \if@twocolumn
2712   \newcommand{\subsubsection}{\@startsection{subsubsection}{3}{\z@}%
2713     {\z@}{\if@slide .4\Cvs \else \z@ \fi}%
2714     {\normalfont\normalsize\headfont}}
2715 \else
2716   \newcommand{\subsubsection}{\@startsection{subsubsection}{3}{\z@}%
2717     {\Cvs \@plus.5\Cdp \@minus.2\Cdp}%
2718     {\if@slide .5\Cvs \@plus.3\Cdp \else \z@ \fi}%
2719     {\normalfont\normalsize\headfont}}
2720 \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。

```
2721 \let\jsJaChar\@empty
```

```

2722 \newcommand\jsParagraphMark{\relax\jsJaChar{■}}
2723 \let\bxjs@org@paragraph@mark\jsParagraphMark

```

```

2724 \ifx\bxjs@paragraph@mark\@empty
2725   \let\jsParagraphMark\@empty
2726 \else\ifx\bxjs@paragraph@mark\@undefined\else
2727   \long\edef\jsParagraphMark{\noexpand\jsJaChar{\bxjs@paragraph@mark}}
2728 \fi\fi
2729 \if@twocolumn
2730   \newcommand{\paragraph}{\@startsection{paragraph}{4}{\z@}%
2731     {\z@}{\if@slide .4\Cvs \else -1\jsZw\fi}% 改行せず 1\jsZw のアキ
2732 %<jspf>    {\normalfont\normalsize\headfont}}
2733 %<!jspf>    {\normalfont\normalsize\headfont\jsParagraphMark}}
2734 \else
2735   \newcommand{\paragraph}{\@startsection{paragraph}{4}{\z@}%
2736     {0.5\Cvs \@plus.5\Cdp \@minus.2\Cdp}%
2737     {\if@slide .5\Cvs \@plus.3\Cdp \else -1\jsZw\fi}% 改行せず 1\jsZw のアキ
2738 %<jspf>    {\normalfont\normalsize\headfont}}
2739 %<!jspf>    {\normalfont\normalsize\headfont\jsParagraphMark}}
2740 \fi

```

\subparagraph 見出しの後ろで改行されません。

```

2741 \if@twocolumn
2742   \newcommand{\subparagraph}{\@startsection{subparagraph}{5}{\z@}%
2743     {\z@}{\if@slide .4\Cvs \@plus.3\Cdp \else -1\jsZw\fi}%
2744     {\normalfont\normalsize\headfont}}
2745 \else
2746   \newcommand{\subparagraph}{\@startsection{subparagraph}{5}{\z@}%
2747     {\z@}{\if@slide .5\Cvs \@plus.3\Cdp \else -1\jsZw\fi}%
2748     {\normalfont\normalsize\headfont}}
2749 \fi

```

8.3 リスト環境

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

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

[2002-05-11] 3zw に変更しました。

[2005-03-19] 二段組は 2zw に戻しました。

```

2750 \if@slide
2751   \setlength\leftmargini{1\jsZw}
2752 \else
2753   \if@twocolumn
2754     \setlength\leftmargini{2\jsZw}
2755   \else
2756     \setlength\leftmargini{3\jsZw}
2757   \fi
2758 \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 2759 \if@slide
\leftmarginv 2760 \setlength\leftmarginii {1\jsZw}
2761 \setlength\leftmarginiii{1\jsZw}
\leftmarginvi 2762 \setlength\leftmarginiv {1\jsZw}
2763 \setlength\leftmarginv {1\jsZw}
2764 \setlength\leftmarginvi {1\jsZw}
2765 \else
2766 \setlength\leftmarginii {2\jsZw}
2767 \setlength\leftmarginiii{2\jsZw}
2768 \setlength\leftmarginiv {2\jsZw}
2769 \setlength\leftmarginv {1\jsZw}
2770 \setlength\leftmarginvi {1\jsZw}
2771 \fi
```

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

```
2772 \setlength \labelsep {0.5\jsZw} % .5em
2773 \setlength \labelwidth{\leftmargini}
2774 \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。

```
2775 \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 2776 \@beginparpenalty -\@lowpenalty
2777 \@endparpenalty -\@lowpenalty
\itempenalty 2778 \@itempenalty -\@lowpenalty
```

`\@listi` `\@listi` は `\leftmargin`, `\parsep`, `\topsep`, `\itemsep` などのトップレベルの定義を
`\@listI` します。この定義は、フォントサイズコマンドによって変更されます（たとえば `\small` の
中では小さい値に設定されます）。このため、`\normalsize` がすべてのパラメータを戻せる
ように、`\@listI` で `\@listi` のコピーを保存します。元の値はかなり複雑ですが、ここ
では簡素化してしまいました。特に最初と最後に行送りの半分の空きが入るようにしてありま
す。アスキーの標準スタイルではトップレベルの `itemize`, `enumerate` 環境でだけ最初と
最後に行送りの半分の空きが入るようになっていました。

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

```
2779 \def\@listi{\leftmargin\leftmarginI
2780 \parsep \z@
2781 \topsep 0.5\baselineskip
2782 \itemsep \z@ \relax}
2783 \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）。

```
2784 \@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 2785 \def\@listii{\leftmargin\leftmarginii
\@listiv 2786 \labelwidth\leftmarginii \advance\labelwidth-\labelsep
2787 \topsep \z@
\@listv 2788 \parsep \z@
\@listvi 2789 \itemsep\parsep}
2790 \def\@listiii{\leftmargin\leftmarginiii
2791 \labelwidth\leftmarginiii \advance\labelwidth-\labelsep
2792 \topsep \z@
2793 \parsep \z@
2794 \itemsep\parsep}
2795 \def\@listiv {\leftmargin\leftmarginiv
2796 \labelwidth\leftmarginiv
2797 \advance\labelwidth-\labelsep}
2798 \def\@listv {\leftmargin\leftmarginv
2799 \labelwidth\leftmarginv
2800 \advance\labelwidth-\labelsep}
2801 \def\@listvi {\leftmargin\leftmarginvi
2802 \labelwidth\leftmarginvi
2803 \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。

```
2804 \renewcommand{\theenumi}{\@arabic\c@enumi}
2805 \renewcommand{\theenumii}{\@alph\c@enumii}
2806 \renewcommand{\theenumiii}{\@roman\c@enumiii}
2807 \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。

```
2808 \newcommand*{\jsInJaParen}[1]{%
2809 \mbox{\jsInhibitGlue (#1) \jsInhibitGlue}}
2810 \newcommand{\labelenumi}{\theenumi.}
2811 \newcommand{\labelenumii}{\jsInJaParen{\theenumii}}
2812 \newcommand{\labelenumiii}{\theenumiii.}
2813 \newcommand{\labelenumiv}{\theenumiv.}
```

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

```
\p@enumiv 2814 \renewcommand{\p@enumii}{\theenumi}
2815 \renewcommand{\p@enumiii}{\theenumi\jsInhibitGlue (\theenumii )}
2816 \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 2817 \newcommand\labelitemi{\textbullet}
2818 \newcommand\labelitemii{\normalfont\bfseries \textendash}
\labelitemiii 2819 \newcommand\labelitemiii{\textasteriskcentered}
\labelitemiv 2820 \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でしまいます。これを解決した新しい `description` の実装です。

```
2821 \newenvironment{description}{%
2822   \list{}{%
2823     \labelwidth=\leftmargin
2824     \labelsep=1\jsZw
2825     \advance \labelwidth by -\labelsep
2826     \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。

```
2827 \newcommand*\descriptionlabel[1]{\normalfont\headfont #1\hfil}
```

■概要

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

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

`bxjsreport` クラスの `abstract` 環境は：

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

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

```

2828 %<*book|report>
2829 \newenvironment{chapterabstract}{%
2830   \begin{list}{}{%
2831     \listparindent=1\jsZw
2832     \itemindent=\listparindent
2833     \rightmargin=0pt
2834     \leftmargin=5\jsZw}\item[]{}\end{list}\vspace{\baselineskip}}
2835 %</book|report>

```

“普通の” abstract 環境の定義。

```

2836 %<*article|report|slide>
2837 \newbox\@abstractbox
2838 \if@titlepage
2839   \newenvironment{abstract}{%
2840     \titlepage
2841     \null\vfil
2842     \@beginparpenalty\@lowpenalty
2843     \begin{center}%
2844       \headfont \abstractname
2845       \@endparpenalty\@M
2846     \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。

```

2847   \par}%
2848   {\par\vfil\null\endtitlepage}
2849 \else
2850   \newenvironment{abstract}{%
2851     \if@twocolumn
2852       \ifx\maketitle\relax
2853         \section*{\abstractname}%
2854       \else
2855         \global\setbox\@abstractbox\hbox\bgroup
2856         \begin{minipage}[b]{\textwidth}
2857           \small\parindent1\jsZw
2858           \begin{center}%
2859             {\headfont \abstractname\vspace{-.5em}\vspace{\z@}}%
2860           \end{center}%
2861           \list{}{%
2862             \listparindent\parindent
2863             \itemindent \listparindent
2864             \rightmargin \leftmargin}%
2865           \item\relax
2866         \fi
2867       \else
2868         \small
2869         \begin{center}%
2870           {\headfont \abstractname\vspace{-.5em}\vspace{\z@}}%
2871         \end{center}%
2872         \list{}{%

```

```

2873         \listparindent\parindent
2874         \itemindent \listparindent
2875         \rightmargin \leftmargin}%
2876     \item\relax
2877 \fi}{\if@twocolumn
2878     \ifx\maketitle\relax
2879     \else
2880     \endlist\end{minipage}\egroup
2881 \fi
2882 \else
2883     \endlist
2884 \fi}
2885 \fi
2886 %</article|report|slide>
2887 %<*jspf>
2888 \newbox\@abstractbox
2889 \newenvironment{abstract}{%
2890     \global\setbox\@abstractbox\hbox\bgroup
2891     \begin{minipage}[b]{157\jsc@mmm}{\sffamily Abstract}\par
2892     \small
2893     \if@english \parindent6\jsc@mmm \else \parindent1\jsZw \fi}%
2894     {\end{minipage}\egroup}
2895 %</jspf>

```

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

```

2896 %<*book|report>
2897 \ifbxjs@force@chapterabstract
2898     \let\abstract\chapterabstract
2899     \let\endabstract\endchapterabstract
2900 \fi
2901 %</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。

```

2902 %<*jspf>
2903 %\newbox\@keywordsbox
2904 %\newenvironment{keywords}{%
2905 %     \global\setbox\@keywordsbox\hbox\bgroup
2906 %     \begin{minipage}[b]{1570\jsc@mmm}{\sffamily Keywords:}\par
2907 %     \small\parindent0\jsZw}%
2908 %     {\end{minipage}\egroup}
2909 %</jspf>

```

■verse 環境

`verse (env.)` 詩のための `verse` 環境です。

```
2910 \newenvironment{verse}{%
2911   \let \=\@centercr
2912   \list{}{%
2913     \itemsep \z@
2914     \itemindent -2\jsZw % 元: -1.5em
2915     \listparindent\itemindent
2916     \rightmargin \z@
2917     \advance\leftmargin 2\jsZw}% 元: 1.5em
2918   \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。

```
2919 \newenvironment{quotation}{%
2920   \list{}{%
2921     \listparindent\parindent
2922     \itemindent\listparindent
2923     \rightmargin \z@}%
2924   \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。

```
2925 \newenvironment{quote}%
2926   {\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。

```
2927 \def\@begintheorem#1#2{\trivlist\labelsep=1\jsZw
2928   \item[\hskip \labelsep{\headfont #1\ #2}]}
2929 \def\@opargbegintheorem#1#2#3{\trivlist\labelsep=1\jsZw
2930   \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。

```

2931 \newenvironment{titlepage}{%
2932 %<book> \pltx@cleartooddpage %% 2017-02-24
2933 \if@twocolumn
2934 \@restonecoltrue\onecolumn
2935 \else
2936 \@restonecolfalse\newpage
2937 \fi
2938 \thispagestyle{empty}%
2939 \ifodd\c@page\setcounter{page}\@ne\else\setcounter{page}\z@\fi %% 2017-
    02-24
2940 }%
2941 {\if@restonecol\twocolumn \else \newpage \fi
2942 \if@twoside\else
2943 \setcounter{page}\@ne
2944 \fi}

```

■付録

\appendix 本文と付録を分離するコマンドです。

```

2945 %<!*book&!report>
2946 \newcommand{\appendix}{\par
2947 \setcounter{section}{0}%
2948 \setcounter{subsection}{0}%
2949 \ifnum\bxjs@label@section=\bxjs@label@section@@compat
2950 \gdef\presectionname{\appendixname}%
2951 \gdef\postsectionname{}}%
2952 % \gdef\thesection{\@Alph\c@section}% [2003-03-02]
2953 \gdef\thesection{\presectionname\@Alph\c@section\postsectionname}%
2954 \gdef\thesubsection{\@Alph\c@section.\@arabic\c@subsection}%
2955 \else
2956 \gdef\@secapp{\appendixname}%
2957 \gdef\@secpos{}}%
2958 \gdef\thesection{\@Alph\c@section}%
2959 \fi}
2960 %</!*book&!report>
2961 %<*book|report>
2962 \newcommand{\appendix}{\par
2963 \setcounter{chapter}{0}%
2964 \setcounter{section}{0}%
2965 \gdef\@chapapp{\appendixname}%
2966 \gdef\@chappos{}}%
2967 \gdef\thechapter{\@Alph\c@chapter}}
2968 %</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。

```
2969 \setlength\arraycolsep{5\p@?}
```

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

```
2970 \setlength\tabcolsep{6\p@?}
```

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

```
2971 \setlength\arrayrulewidth{.4\p@}
```

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

```
2972 \setlength\doublerulesep{2\p@}
```

■tabbing 環境

`\tabbingsep` `\'` コマンドで入るアキです。

```
2973 \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。

```
2974 \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。

```
2975 \setlength\fboxsep{3\p@?}
```

```
2976 \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。

```
2977 %<!book&!report>\renewcommand \theequation {\@arabic\c@equation}
```

```
2978 %<*book|report>
```

```
2979 \@addtoreset{equation}{chapter}
```

```
2980 \renewcommand\theequation
```

```
2981 {\ifnum \c@chapter>\z@ \thechapter.\fi \@arabic\c@equation}
```

```
2982 %</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。

```
2983 % \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。

```
2984 % \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。

```
2985 % \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。

```
2986 %<!*book&!report>
2987 \newcounter{figure}
2988 \renewcommand \thefigure {\@arabic\c@figure}
2989 %<!/book&!report>
2990 %<*book|report>
2991 \newcounter{figure}[chapter]
2992 \renewcommand \thefigure
2993     {\ifnum \c@chapter>z@ \thechapter.\fi \@arabic\c@figure}
2994 %</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 2995 \def\fps@figure{tbp}
2996 \def\ftype@figure{1}
\fnum@figure 2997 \def\ext@figure{lof}
2998 \def\fnum@figure{\figurename\nobreak\thefigure}
```

`figure (env.)` * 形式は段抜きのフロートです。

```
figure* (env.) 2999 \newenvironment{figure}%
3000                 {\@float{figure}}%
3001                 {\end@float}
```

```

3002 \newenvironment{figure*}%
3003         {\@dblfloat{figure}}%
3004         {\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。

```

3005 %<!*book&!report>
3006 \newcounter{table}
3007 \renewcommand\thetable{\@arabic\c@table}
3008 %</!*book&!report>
3009 %<*book|report>
3010 \newcounter{table}[chapter]
3011 \renewcommand \thetable
3012         {\ifnum \c@chapter>z@ \thechapter.\fi \@arabic\c@table}
3013 %</book|report>

```

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

```

\ext@table 3014 \def\fps@table{tbp}
3015 \def\ftype@table{2}
\fnun@table 3016 \def\ext@table{lot}
3017 \def\fnun@table{\tablename\nobreak\thetable}

```

`table (env.)` * は段抜きのフロートです。

```

table* (env.) 3018 \newenvironment{table}%
3019         {\@float{table}}%
3020         {\end@float}
3021 \newenvironment{table*}%
3022         {\@dblfloat{table}}%
3023         {\end@dblfloat}

```

8.6 キャプション

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

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

```

3024 \newlength\abovecaptionskip
3025 \newlength\belowcaptionskip
3026 \setlength\abovecaptionskip{5\p@?} % 元: 10\p@
3027 \setlength\belowcaptionskip{5\p@?} % 元: 0\p@

```

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

[2003-11-05] ロジックを少し変えてみました。

[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。

```
3028 %<!*jspf>
3029 % \long\def\@makecaption#1#2{\small
3030 %   \advance\leftskip10\jsc@mmm
3031 %   \advance\rightskip10\jsc@mmm
3032 %   \vskip\abovcaptionskip
3033 %   \sbox\@tempboxa{#1\hskip1\jsZw\relax #2}%
3034 %   \ifdim \wd\@tempboxa >\hsize
3035 %     #1\hskip1\jsZw\relax #2\par
3036 %   \else
3037 %     \global \@minipagefalse
3038 %     \hb@xt@\hsize{\hfil\box\@tempboxa\hfil}%
3039 %   \fi
3040 %   \vskip\belowcaptionskip}}
3041 \long\def\@makecaption#1#2{\small
3042   \advance\leftskip .0628\linewidth
3043   \advance\rightskip .0628\linewidth
3044   \vskip\abovcaptionskip
3045   \sbox\@tempboxa{#1\zwspace#2}%
3046   \ifdim \wd\@tempboxa <\hsize \centering \fi
3047   #1\zwspace#2\par
3048   \vskip\belowcaptionskip}}
3049 %</!*jspf>
3050 %<*jspf>
3051 \long\def\@makecaption#1#2{%
3052   \vskip\abovcaptionskip
3053   \sbox\@tempboxa{\small\sffamily #1\quad #2}%
3054   \ifdim \wd\@tempboxa >\hsize
3055     {\small\sffamily
3056       \list{#1}{%
3057         \renewcommand{\makelabel}[1]{##1\hfil}
3058         \itemsep \z@
3059         \itemindent \z@
3060         \labelsep \z@
3061         \labelwidth 11\jsc@mmm
3062         \listparindent\z@
3063         \leftmargin 11\jsc@mmm}\item\relax #2\endlist}
3064   \else
3065     \global \@minipagefalse
3066     \hb@xt@\hsize{\hfil\box\@tempboxa\hfil}%
3067   \fi
3068   \vskip\belowcaptionskip}
3069 %</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
\if@jsc@warnoldfontcmdexception
\allow/disallowoldfontcommands
\jsc@DeclareOldFontCommand
\allowoldfontcommands
\disallowoldfontcommands
```

```

\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。

3070 \newif\if@jsc@warnoldfontcmd
3071 \@jsc@warnoldfontcmdtrue
3072 \newif\if@jsc@warnoldfontcmdexception
3073 \@jsc@warnoldfontcmdexceptionfalse

\jsc@DeclareOldFontCommand
3074 \newcommand*{\jsc@DeclareOldFontCommand}[3]{%
3075   \g@addto@macro\bxjs@oldfontcmd@list{\do#1}%
3076   \DeclareOldFontCommand{#1}{%
3077     \bxjs@oldfontcmd{#1}#2%
3078   }{%
3079     \bxjs@oldfontcmd{#1}#3%
3080   }%
3081 }
3082 \DeclareRobustCommand*{\jsc@warnoldfontcmd}[1]{%
3083   \ClassInfo\bxjs@clsname
3084   {Old font command '\string#1' is used!!\MessageBreak
3085     The first occurrence is}%
3086 }

\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。

3087 \newcommand*{\allowoldfontcommands}{%
3088   \@jsc@warnoldfontcmdexceptiontrue}
3089 \newcommand*{\disallowoldfontcommands}{%
3090   \@jsc@warnoldfontcmdexceptionfalse}

3091 \let\bxjs@oldfontcmd@list\@empty
```

```

3092 \def\bxjs@oldfontcmd#1{%
3093   \expandafter\bxjs@oldfontcmd@a\csname bxjs@ofc/\string#1\endcsname#1}
3094 \def\bxjs@oldfontcmd@a#1#2{%
3095   \if@jsc@warnoldfontcmdexception\else
3096     \global\@jsc@warnoldfontcmdfalse
3097     \ifx#1\relax
3098       \global\let#1=t%
3099       \jsc@warnoldfontcmd{#2}%
3100     \fi
3101   \fi}
3102 \def\bxjs@warnoldfontcmd@final{%
3103 % \par
3104   \global\let\bxjs@warnoldfontcmd@final\@empty
3105   \let\@tempa\@empty
3106   \def\do##1{%
3107     \expandafter\ifx\csname bxjs@ofc/\string##1\endcsname\relax\else
3108       \edef\@tempa{\@tempa \space\string##1}\fi}
3109   \bxjs@oldfontcmd@list
3110   \ifx\@tempa\@empty\else
3111     \ClassWarningNoLine\bxjs@clsname
3112     {Some old font commands were used in text:\MessageBreak
3113     \space\@tempa\MessageBreak
3114     You should note, that since 1994 LaTeX2e provides a\MessageBreak
3115     new font selection scheme called NFSS2 with several\MessageBreak
3116     new, combinable font commands. The
3117     class provides\MessageBreak
3118     the old font commands only for compatibility}
3119   \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。

```

3120 \ifbxjs@old@hook@system
3121   \AtEndDocument{%
3122     \g@addto@macro\clearpage{\bxjs@warnoldfontcmd@final}}
3123 \else
3124   \AddToHook{enddocument/afterlastpage}{\bxjs@warnoldfontcmd@final}
3125 \fi

```

`\mc` フォントファミリーを変更します。

```

\gt 3126 \jsc@DeclareOldFontCommand{\mc}{\normalfont\mcfamily}{\mathmc}
\gt 3127 \jsc@DeclareOldFontCommand{\gt}{\normalfont\gtfamily}{\mathgt}
\rm 3128 \jsc@DeclareOldFontCommand{\rm}{\normalfont\rmfamily}{\mathrm}
\sff 3129 \jsc@DeclareOldFontCommand{\sf}{\normalfont\sffamily}{\mathsf}
\tt 3130 \jsc@DeclareOldFontCommand{\tt}{\normalfont\ttfamily}{\mathtt}

```

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

```
3131 \jsc@DeclareOldFontCommand{\bf}{\normalfont\bfseries}{\mathbf}
```

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

```
3132 \jsc@DeclareOldFontCommand{\it}{\normalfont\itshape}{\mathit}
```

```
3133 \jsc@DeclareOldFontCommand{\sl}{\normalfont\slshape}{\@nomath\sl}
```

```
3134 \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 3135 \DeclareRobustCommand*{\cal}{\@fontswitch\relax\mathcal}
```

```
3136 \DeclareRobustCommand*{\mit}{\@fontswitch\relax\mathnormal}
```

10 相互参照

10.1 目次の類

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

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

たとえば `\section` に見出し番号が付く場合、上の「タイトル」は

```
\numberline{番号}{見出し}
```

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

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

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

この「番号」は `\thefigure` コマンドで生成された図番号です。

`table` 環境も同様です。

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

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

という書式です。

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

インデント 左側の字下げ量です。

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

`\@pnumwidth` ページ番号の入る箱の幅です。

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

`\@dotsep` 点の間隔です (単位 mu)。

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

```
3137 \newcommand\@pnumwidth{1.55em}
3138 \newcommand\@tocrmarg{2.55em}
3139 \newcommand\@dotsep{4.5}
3140 %<!\book&!report>\setcounter{tocdepth}{2}
3141 %<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)

```
3142 \newdimen\jsc@tocl@width
3143 \newcommand{\tableofcontents}{%
3144 %<*book|report>
3145   \settowidth\jsc@tocl@width{\headfont\prechaptername\postchaptername}%
3146   \settowidth\@tempdima{\headfont\appendixname}%
3147   \ifdim\jsc@tocl@width<\@tempdima \setlength\jsc@tocl@width{\@tempdima}\fi
3148   \ifdim\jsc@tocl@width<2\jsZw \divide\jsc@tocl@width by 2 \advance\jsc@tocl@width 1\jsZw\fi
3149   \if@twocolumn
3150     \@restonecoltrue\onecolumn
3151   \else
3152     \@restonecolfalse
3153   \fi
3154   \chapter*{\contentsname}%
3155   \@mkboth{\contentsname}{}%
3156 %</book|report>
3157 %<*\book&!report>
3158   \settowidth\jsc@tocl@width{\headfont\presectionname\postsectionname}%
3159   \settowidth\@tempdima{\headfont\appendixname}%
3160   \ifdim\jsc@tocl@width<\@tempdima\relax\setlength\jsc@tocl@width{\@tempdima}\fi
3161   \ifdim\jsc@tocl@width<2\jsZw \divide\jsc@tocl@width by 2 \advance\jsc@tocl@width 1\jsZw\fi
3162   \section*{\contentsname}%
3163   \@mkboth{\contentsname}{\contentsname}%
3164 %</!\book&!report>
3165   \@starttoc{toc}%
3166 %<book|report> \if@restonecol\twocolumn\fi
3167 }
```

`\l@part` 部の目次です。

```
3168 \newcommand*{\l@part}[2]{%
3169   \ifnum \c@tocdepth >-2\relax
```

```

3170 %<!book&!report> \addpenalty\@secpenalty
3171 %<book|report> \addpenalty{-\@highpenalty}%
3172 \addvspace{2.25em \@plus\p@?}%
3173 \begingroup
3174 \parindent \z@
3175 % \@pnumwidth should be \@tocrmarg
3176 % \rightskip \@pnumwidth
3177 \rightskip \@tocrmarg
3178 \parfillskip -\rightskip
3179 {\leavevmode
3180 \large \headfont
3181 \setlength\@lnumwidth{4\jsZw}%
3182 #1\hfil \hb@xt@\@pnumwidth{\hss #2}}\par
3183 \nobreak
3184 %<book|report> \global\@nobreaktrue
3185 %<book|report> \everypar{\global\@nobreakfalse\everypar{}}%
3186 \endgroup
3187 \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)

```

3188 %<*book|report>
3189 \newcommand*{\l@chapter}[2]{%
3190 \ifnum \c@tocdepth >\m@ne
3191 \addpenalty{-\@highpenalty}%
3192 \addvspace{1.0em \@plus\p@?}
3193 % \vskip 1.0em \@plus\p@ % book.cls では↑がこうなっている
3194 \begingroup
3195 \parindent\z@
3196 % \rightskip\@pnumwidth
3197 \rightskip\@tocrmarg
3198 \parfillskip-\rightskip
3199 \leavevmode\headfont
3200 % % \if@english\setlength\@lnumwidth{5.5em}\else\setlength\@lnumwidth{4.683\jsZw}\fi
3201 \setlength\@lnumwidth{\jsc@tocl@width}\advance\@lnumwidth 2.683\jsZw
3202 \advance\leftskip\@lnumwidth \hskip-\leftskip
3203 #1\nobreak\hfil\nobreak\hbox to\@pnumwidth{\hss#2}\par
3204 \penalty\@highpenalty
3205 \endgroup
3206 \fi}
3207 %</book|report>

```

\l@section 節の目次です。

```

3208 %<*!book&!report>
3209 \newcommand*{\l@section}[2]{%
3210 \ifnum \c@tocdepth >\z@
3211 \addpenalty{\@secpenalty}%
3212 \addvspace{1.0em \@plus\p@?}%

```



```

3213 \begingroup
3214 \parindent\z@
3215 % \rightskip\@pnumwidth
3216 \rightskip\@tocrmarg
3217 \parfillskip-\rightskip
3218 \leavevmode\headfont
3219 % % \setlength\@lnumwidth{4\jsZw}% 元 1.5em [2003-03-02]
3220 \setlength\@lnumwidth{\jsc@tocl@width}\advance\@lnumwidth 2\jsZw
3221 \advance\leftskip\@lnumwidth \hskip-\leftskip
3222 #1\nobreak\hfil\nobreak\hbox to\@pnumwidth{\hss#2}\par
3223 \endgroup
3224 \fi}
3225 %<book&!report>

```

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

```

3226 %<book|report> % \newcommand*{\l@section}{\@dottedtocline{1}{1\jsZw}{3.683\jsZw}}

```

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

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

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

```

\l@subparagraph 3227 %<*!book&!report>
3228 % \newcommand*{\l@section} {\@dottedtocline{2}{1.5em}{2.3em}}
3229 % \newcommand*{\l@subsection}{\@dottedtocline{3}{3.8em}{3.2em}}
3230 % \newcommand*{\l@paragraph} {\@dottedtocline{4}{7.0em}{4.1em}}
3231 % \newcommand*{\l@subparagraph} {\@dottedtocline{5}{10em}{5em}}
3232 %
3233 % \newcommand*{\l@section} {\@dottedtocline{2}{1zw}{3zw}}
3234 % \newcommand*{\l@subsection}{\@dottedtocline{3}{2\jsZw}{3\jsZw}}
3235 % \newcommand*{\l@paragraph} {\@dottedtocline{4}{3\jsZw}{3\jsZw}}
3236 % \newcommand*{\l@subparagraph} {\@dottedtocline{5}{4\jsZw}{3\jsZw}}
3237 %
3238 \newcommand*{\l@section}{%
3239 \tempdima\jsc@tocl@width \advance\tempdima -1\jsZw
3240 \@dottedtocline{2}{\tempdima}{3\jsZw}}
3241 \newcommand*{\l@subsection}{%
3242 \tempdima\jsc@tocl@width \advance\tempdima 0\jsZw
3243 \@dottedtocline{3}{\tempdima}{4\jsZw}}
3244 \newcommand*{\l@paragraph}{%
3245 \tempdima\jsc@tocl@width \advance\tempdima 1\jsZw
3246 \@dottedtocline{4}{\tempdima}{5\jsZw}}
3247 \newcommand*{\l@subparagraph}{%
3248 \tempdima\jsc@tocl@width \advance\tempdima 2\jsZw
3249 \@dottedtocline{5}{\tempdima}{6\jsZw}}
3250 %<book&!report>
3251 %<*book|report>
3252 % \newcommand*{\l@section} {\@dottedtocline{2}{3.8em}{3.2em}}
3253 % \newcommand*{\l@subsection}{\@dottedtocline{3}{7.0em}{4.1em}}

```

```

3254 % \newcommand*{\l@paragraph}    {\@dottedtocline{4}{10em}{5em}}
3255 % \newcommand*{\l@subparagraph} {\@dottedtocline{5}{12em}{6em}}
3256 \newcommand*{\l@section}{%
3257     \@tempdima\jsc@tocl@width \advance\@tempdima -1\jsZw
3258     \@dottedtocline{1}{\@tempdima}{3.683\jsZw}}
3259 \newcommand*{\l@subsection}{%
3260     \@tempdima\jsc@tocl@width \advance\@tempdima 2.683\jsZw
3261     \@dottedtocline{2}{\@tempdima}{3.5\jsZw}}
3262 \newcommand*{\l@subsubsection}{%
3263     \@tempdima\jsc@tocl@width \advance\@tempdima 6.183\jsZw
3264     \@dottedtocline{3}{\@tempdima}{4.5\jsZw}}
3265 \newcommand*{\l@paragraph}{%
3266     \@tempdima\jsc@tocl@width \advance\@tempdima 10.683\jsZw
3267     \@dottedtocline{4}{\@tempdima}{5.5\jsZw}}
3268 \newcommand*{\l@subparagraph}{%
3269     \@tempdima\jsc@tocl@width \advance\@tempdima 16.183\jsZw
3270     \@dottedtocline{5}{\@tempdima}{6.5\jsZw}}
3271 %</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。

```

3272 \newdimen\@lnumwidth
3273 \def\numberline#1{\hb@xt@\@lnumwidth{#1\hfil}\hspace{0pt}}

```

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

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

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

とします。

```

3274 \def\jsTocLine{\leaders\hbox{%
3275     $\m@th \mkern \@dotsep mu\hbox{.}\mkern \@dotsep mu$\}\hfill}
3276 \def\@dottedtocline#1#2#3#4#5{\ifnum #1>\c@tocdepth \else
3277     \vskip \z@ \@plus.2\p@?
3278     {\leftskip #2\relax \rightskip \@tocrmarg \parfillskip -\rightskip
3279     \parindent #2\relax\@afterindenttrue
3280     \interlinepenalty\@M
3281     \leavevmode
3282     \@lnumwidth #3\relax
3283     \advance\leftskip \@lnumwidth \null\nobreak\hskip -\leftskip
3284     {#4}\nobreak
3285     \jsTocLine \nobreak\hb@xt@\@pnumwidth{%
3286         \hfil\normalfont \normalcolor #5}\par}\fi}

```

■ 図目次と表目次

`\listoffigures` 図目次を出力します。

```
3287 \newcommand{\listoffigures}{%
3288 %<*book|report>
3289 \if@twocolumn\@restonecoltrue\onecolumn
3290 \else\@restonecolfalse\fi
3291 \chapter*{\listfigurename}%
3292 \mkboth{\listfigurename}{}%
3293 %</book|report>
3294 %<!*book&!report>
3295 \section*{\listfigurename}%
3296 \mkboth{\listfigurename}{\listfigurename}%
3297 %</!book&!report>
3298 \@starttoc{lof}%
3299 %<book|report> \if@restonecol\twocolumn\fi
3300 }
```

`\l@figure` 図目次の項目を出力します。

```
3301 \newcommand*{\l@figure}{\@dottedtocline{1}{1\jsZw}{3.683\jsZw}}
```

`\listoftables` 表目次を出力します。

```
3302 \newcommand{\listoftables}{%
3303 %<*book|report>
3304 \if@twocolumn\@restonecoltrue\onecolumn
3305 \else\@restonecolfalse\fi
3306 \chapter*{\listtablename}%
3307 \mkboth{\listtablename}{}%
3308 %</book|report>
3309 %<!*book&!report>
3310 \section*{\listtablename}%
3311 \mkboth{\listtablename}{\listtablename}%
3312 %</!book&!report>
3313 \@starttoc{lot}%
3314 %<book|report> \if@restonecol\twocolumn\fi
3315 }
```

`\l@table` 表目次は図目次と同じです。

```
3316 \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。

```
3317 \newdimen\bibindent
3318 \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。

```

3319 \newenvironment{thebibliography}[1]{%
3320   \@jsc@warnoldfontcmdexceptiontrue
3321   \global\let\presectionname\relax
3322   \global\let\postsectionname\relax
3323   %<article|slide> \section*{\refname}\mkboth{\refname}{\refname}%
3324   %<*kiyou>
3325   \vspace{1.5\baselineskip}
3326   \subsubsection*{\refname}\mkboth{\refname}{\refname}%
3327   \vspace{0.5\baselineskip}
3328   %</kiyou>
3329   %<book|report> \chapter*{\bibname}\mkboth{\bibname}{}%
3330   %<book|report> \addcontentsline{toc}{chapter}{\bibname}%
3331   \list{\@biblabel{\@arabic\c@enumiv}}%
3332     {\settowidth\labelwidth{\@biblabel{#1}}%
3333      \leftmargin\labelwidth
3334      \advance\leftmargin\labelsep
3335      \@openbib@code
3336      \usecounter{enumiv}%
3337      \let\p@enumiv\@empty
3338      \renewcommand\theenumiv{\@arabic\c@enumiv}}%
3339   %<kiyou> \small
3340   \sloppy
3341   \clubpenalty4000
3342   \@clubpenalty\clubpenalty
3343   \widowpenalty4000%
3344   \sfcode`\.\@m}
3345   {\def\@noitemerr
3346     {\@latex@warning{Empty `thebibliography' environment}}}%
3347   \endlist}

```

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

```

3348 \newcommand{\newblock}{\hspace .11em\@plus.33em\@minus.07em}

```

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

```

3349 \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。

```

3350 % \def\@biblabel#1{\jsInhibitGlue [#1] \jsInhibitGlue}

```

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

`\cite` フォントにするには次のようにします。とりあえずコメントアウトしておきましたので、必

```

\@citex

```

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

```

3351 % \def\@citex[#1]#2{\leavevmode
3352 %   \let\@citea\@empty
3353 %   \@cite{\@for\@citeb:=#2\do
3354 %     {\@citea\def\@citea{, \inhibitglue\penalty\@m\ }%
3355 %     \edef\@citeb{\expandafter\@firstofone\@citeb\@empty}}%
3356 %     \if@filesw\immediate\write\@auxout{\string\citation{\@citeb}}\fi
3357 %     \ifundefined{b@\@citeb}{\mbox{\normalfont\bfseries ?}%
3358 %       \G@refundefinedtrue
3359 %       \@latex@warning
3360 %         {Citation '\@citeb' on page \thepage \space undefined}}%
3361 %       {\@cite@ofmt{\csname b@\@citeb\endcsname}}}{#1}}
3362 % \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。

```

3363 % \DeclareRobustCommand\cite{\unskip
3364 %   \@ifnextchar [{\@tempswattrue\@citex}{\@tempswafalse\@citex[]}]%
3365 % \def\@cite#1#2{${\hbox{\scriptsize{#1\if@tempswa
3366 %   , \jsInhibitGlue\ #2\fi}})}$}

```

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)。

```

3367 \newenvironment{theindex}{% 索引を 3 段組で出力する環境
3368   \if@twocolumn
3369     \onecolumn\@restonecolfalse
3370   \else
3371     \clearpage\@restonecoltrue
3372   \fi
3373   \columnseprule.4pt \columnsep 2\jsZw
3374   \ifx\multicols\@undefined
3375 %<book|report>      \twocolumn[\@makeschapterhead{\indexname}%
3376 %<book|report>      \addcontentsline{toc}{chapter}{\indexname}}%
3377 %<!book&!report>    \def\presectionname{}\def\postsectionname{}%
3378 %<!book&!report>    \twocolumn[\section*{\indexname}}%
3379   \else
3380     \ifdim\textwidth<\fullwidth
3381       \setlength{\evensidemargin}{\oddsidemargin}
3382       \setlength{\textwidth}{\fullwidth}
3383       \setlength{\linewidth}{\fullwidth}
3384 %<book|report>      \begin{multicols}{3}[\chapter*{\indexname}%
3385 %<book|report>      \addcontentsline{toc}{chapter}{\indexname}}%
3386 %<!book&!report>    \def\presectionname{}\def\postsectionname{}%
3387 %<!book&!report>    \begin{multicols}{3}[\section*{\indexname}}%

```

```

3388      \else
3389 %<book|report>      \begin{multicols}{2}[\chapter*{\indexname}]%
3390 %<book|report>      \addcontentsline{toc}{chapter}{\indexname}}%
3391 %<!book&!report>    \def\presectionname{}\def\postsectionname{}%
3392 %<!book&!report>    \begin{multicols}{2}[\section*{\indexname}]%
3393      \fi
3394      \fi
3395 %<book|report>      \@mkboth{\indexname}{}%
3396 %<!book&!report>    \@mkboth{\indexname}{\indexname}%
3397      \plainifnotempty % \thispagestyle{plain}
3398      \parindent\z@
3399      \parskip\z@ \@plus .3\p@?\relax
3400      \let\item\@idxitem
3401      \raggedright
3402      \footnotesize\narrowbaselines
3403    }{
3404      \ifx\multicols\@undefined
3405        \if@restonecol\onecolumn\fi
3406      \else
3407        \end{multicols}
3408      \fi
3409      \clearpage
3410    }

```

`\@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 3411 \newcommand{\@idxitem}{\par\hangindent 4\jsZw} % 元 40pt
\subsubitem 3412 \newcommand{\subitem}{\@idxitem \hspace*{2\jsZw}} % 元 20pt
3413 \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。

```

3414 \newcommand{\indexspace}{\par \vskip 10\p@? \@plus5\p@? \@minus3\p@?\relax}

```

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

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

```

3415 \newcommand\seename{\if@english see\else →\fi}
3416 \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。

```

3417 \ifx\pltx@foot@penalty\@undefined
3418   \let\footnotes@ve=\footnote
3419   \def\footnote{\jsInhibitGlue\footnotes@ve}
3420   \let\footnotemarks@ve=\footnotemark
3421   \def\footnotemark{\jsInhibitGlue\footnotemarks@ve}
3422 \fi

```

`\@makefnmark` 脚注番号を付ける命令です。ここでは脚注番号の前に記号 * を付けています。「注 1」の形式にするには `\textasteriskcentered` を `注\kern0.1em` にしてください。`\@xfootnotenext` と合わせて、もし脚注番号が空なら記号も出力しないようにしてあります。

[2002-04-09] インプリメントの仕方を変えたため消しました。

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

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

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

pTeX 依存のコードなので、minimal 和文ドライバ実装に移動。

`\thefootnote` 脚注番号に * 印が付くようにしました。ただし、番号がゼロのときは * 印も脚注番号も付きません。

[2003-08-15] `\textasteriskcentered` ではフォントによって下がりすぎるので変更しました。

[2016-10-08] TODO: 脚注番号が `newtxtext` や `newpctext` の使用時におかしくなってしまう。これらのパッケージは内部で `\thefootnote` を再定義していますので、気になる場合はパッケージを読み込むときに `defaultsups` オプションを付けてください (qa:57284, qa:57287)。

```

3423 \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。

```

3424 % \def\thefootnote{\ifnum\c@footnote>\z@ 注\kern0.1\jsZw\@arabic\c@footnote\fi}

```

`\footnoterule` 本文と脚注の間の罫線です。

```

3425 \renewcommand{\footnoterule}{%
3426   \kern-2.6\p@? \kern-.4\p@
3427   \hrule width .4\columnwidth
3428   \kern 2.6\p@?}

```

`\c@footnote` 脚注番号は章ごとにリセットされます。

```

3429 %<book|report>\@addtoreset{footnote}{chapter}

```

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

[2016-08-25] コミュニティ版 p \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。

```
3430 \long\def\@footnotetext{%
3431   \insert\footins\bgroup
3432     \normalfont\footnotesize
3433     \interlinepenalty\interfootnotelinepenalty
3434     \splittopskip\footnotesep
3435     \splitmaxdepth \dp\strutbox \floatingpenalty \MM
3436     \hsize\columnwidth \@parboxrestore
3437     \def\@currentcounter{footnote}%
3438     \protected@edef\@currentlabel{%
3439       \csname p@footnote\endcsname\@thefnmark
3440     }%
3441     \color@begingroup
3442       \@makefnmark{%
3443         \rule{z@}{\footnotesep\ignorespaces}%
3444         \futurelet\jsc@next\jsc@fo@t}
3445 \def\jsc@fo@t{\ifcat\bgroup\noexpand\jsc@next \let\jsc@next\jsc@f@t
3446                                     \else \let\jsc@next\jsc@f@t\fi \jsc@next}
3447 \def\jsc@f@t{\bgroup\aftergroup\jsc@@foot\let\jsc@next}
3448 \def\jsc@f@t#1{#1\jsc@@foot}
3449 \def\jsc@@foot{\@finalstrut\strutbox\color@endgroup\egroup
3450   \ifx\pltx@foot@penalty\@undefined\else
3451     \ifhmode\null\fi
3452     \ifnum\pltx@foot@penalty=z@\else
3453       \penalty\pltx@foot@penalty
3454       \pltx@foot@penalty\z@
3455     \fi
3456   \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。

```
3457 \newcommand\@makefnmark[1]{%
3458   \advance\leftskip 3\jsZw
3459   \parindent 1\jsZw
3460   \noindent
3461   \llap{\@makefnmark\hskip0.3\jsZw}#1}
```


`\@footnotenext` 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。

```
3462 % \def\@footnotenext[#1]{%
3463 %   \begingroup
3464 %     \ifnum#1>\z@
3465 %       \csname c@\mpfn\endcsname #1\relax
3466 %       \unrestored@protected@xdef\@thefnmark{\thempfn}%
3467 %     \else
3468 %       \unrestored@protected@xdef\@thefnmark{}%
3469 %     \fi
3470 %   \endgroup
3471 % \@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。

```
3472 \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。

```
3473 \def\everyparhook{\jsInhibitGlueAtParTop}
3474 \ifnum\bxjs@everyparhook=\bxjs@everyparhook@@compat
3475 \g@addto@macro\bxjs@begin@document@hook{\everypar{\everyparhook}}
3476 \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。

```
3477 \ifx j\jsEngine
3478 \def\@inhibitglue{%
3479   \futurelet\@let@token\@inhibitglue}
3480 \begingroup
3481 \let\GDEF=\gdef
3482 \let\CATCODE=\catcode
3483 \let\ENDGROUP=\endgroup
3484 \CATCODE`k=12
3485 \CATCODE`a=12
3486 \CATCODE`n=12
3487 \CATCODE`j=12
3488 \CATCODE`i=12
3489 \CATCODE`c=12
3490 \CATCODE`h=12
3491 \CATCODE`r=12
3492 \CATCODE`t=12
3493 \CATCODE`e=12
3494 \GDEF\KANJI@CHARACTER{kanji character }
3495 \ENDGROUP
3496 \def\@inhibitglue{%
3497   \expandafter\expandafter\expandafter\jsc@inhibitglue\expandafter\meaning\expandafter\@let@
3498   \expandafter\def\expandafter\jsc@inhibitglue\expandafter#\expandafter1\KANJI@CHARACTER#2#3\j
3499   \def\jsc@ig@temp{#1}%
3500   \ifx\jsc@ig@temp\@empty
3501     \ifnum\the\inhibitxspcode`#2=2\relax
3502       \inhibitglue
3503     \fi
3504   \fi}
3505 \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。

```
3506 \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。

```
3507 \def\@doendpe{%
3508   \@endpetrue
```

```

3509 \def\par{%
3510   \@restorepar\clubpenalty\@clubpenalty\everypar{\everyparhook}\par\@endpefalse}%
3511 \everypar{\setbox\z@\lastbox}\everypar{\everyparhook}\@endpefalse\everyparhook}}

```

[2017-08-31] minipage 環境にも対策します。

```

3512 \def\@setminipage{%
3513   \@minipagetrue
3514   \everypar{\@minipagefalse\everypar{\everyparhook}}%
3515 }

```

\item 命令の直後です。

```

3516 \def\@item[#1]{%
3517   \if@nparitem
3518     \@donoparitem
3519   \else
3520     \if@inlabel
3521       \indent \par
3522     \fi
3523     \ifhmode
3524       \unskip\unskip \par
3525     \fi
3526     \if@newlist
3527       \if@nobreak
3528         \@nbitem
3529       \else
3530         \addpenalty\@beginparpenalty
3531         \addvspace\@topsep
3532         \addvspace{-\parskip}%
3533       \fi
3534     \else
3535       \addpenalty\@itempenalty
3536       \addvspace\itemsep
3537     \fi
3538     \global\@inlabeltrue
3539   \fi
3540   \everypar{%
3541     \@minipagefalse
3542     \global\@newlistfalse
3543     \if@inlabel
3544       \global\@inlabelfalse
3545       {\setbox\z@\lastbox
3546        \ifvoid\z@
3547          \kern-\itemindent
3548        \fi}%
3549     \box\@labels
3550     \penalty\z@
3551   \fi
3552   \if@nobreak
3553     \@nobreakfalse

```

```

3554     \clubpenalty \@M
3555     \else
3556         \clubpenalty \@clubpenalty
3557         \everypar{\everyparhook}%
3558     \fi\everyparhook}%
3559 \if@noitemarg
3560     \@noitemargfalse
3561     \if@nmbbrlist
3562         \refstepcounter\@listctr
3563     \fi
3564 \fi
3565 \sbox\@tempboxa{\makelabel{#1}}%
3566 \global\setbox\@labels\hbox{%
3567     \unhbox\@labels
3568     \hskip \itemindent
3569     \hskip -\labelwidth
3570     \hskip -\labelsep
3571     \ifdim \wd\@tempboxa >\labelwidth
3572         \box\@tempboxa
3573     \else
3574         \hbox to\labelwidth {\unhbox\@tempboxa}%
3575     \fi
3576     \hskip \labelsep}%
3577 \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。

```

3578 \def\@afterheading{%
3579     \@nbreaktrue
3580     \everypar{%
3581         \if@nbreak
3582             \@nbreakfalse
3583             \clubpenalty \@M
3584             \if@afterindent \else
3585                 {\setbox\z@\lastbox}%
3586             \fi
3587         \else
3588             \clubpenalty \@clubpenalty
3589             \everypar{\everyparhook}%
3590         \fi\everyparhook}}

```

「`\everyparhook` 用のパッチ処理」はここまで。

```

3591 \fi

```

`\@gnewline` についてはちょっと複雑な心境です。もともとの $\text{p}\text{I}\text{A}\text{T}\text{E}\text{X}\,2_{\epsilon}$ は段落の頭にグルーが入る方で統一されていました。しかし `\` の直後にはグルーが入らず、不統一でした。そこで `\` の直後にもグルーを入れるように直していただいた経緯があります。しかし、こ

こ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。

```
3592 \ifnum\bxjs@everyparhook=\bxjs@everyparhook@@none\else
3593 \AtEndOfPackage{%
3594 \def\@gnewline #1{%
3595   \ifvmode
3596     \@nolnerr
3597   \else
3598     \unskip \reserved@a {\reserved@f#1}\nobreak \hfil \break \null
3599     \jsInhibitGlue \ignorespaces
3600   \fi}
3601 }
3602 \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。

```
3603 \if@jslogo
3604   \IfFileExists{jslogo.sty}{%
3605     \RequirePackage{jslogo}%
3606   }{%
3607     \ClassWarningNoLine\bxjs@clsname
3608       {The package 'jslogo' is not installed.\MessageBreak
3609         It is included in the recent release of\MessageBreak
3610         the 'jsclasses' bundle}
3611   }
3612 \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: しっぱ愛好家さん)。

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

```
3613 %\let\ltx@ifnextchar\@ifnextchar
3614 %\def\ProvidesFile#1{%
3615 %  \begingroup
3616 %    \catcode`\ 10 %
3617 %    \ifnum \endlinechar<256 %
3618 %        \ifnum \endlinechar>\m@ne
3619 %            \catcode\endlinechar 10 %
3620 %        \fi
3621 %    \fi
3622 %    \@makeother\/%
3623 %    \@makeother\&%
3624 %    \ltx@ifnextchar[{\@providesfile{#1}}{\@providesfile{#1}[]}]}
```

14 初期設定

■いろいろな語

```
\prepartname
\postpartname 3625 \newcommand{\prepartname}{\if@english Part~\else 第\fi}
\prechaptername 3626 \newcommand{\postpartname}{\if@english\else 部\fi}
3627 %<book|report>\newcommand{\prechaptername}{\if@english Chapter~\else 第\fi}
\postchaptername 3628 %<book|report>\newcommand{\postchaptername}{\if@english\else 章\fi}
\presectionname 3629 \newcommand{\presectionname}{}% 第
\postsectionname 3630 \newcommand{\postsectionname}{}% 節

\contentsname

\listfigurename 3631 \newcommand{\contentsname}{\if@english Contents\else 目次\fi}
\listtablename 3632 \newcommand{\listfigurename}{\if@english List of Figures\else 図目次\fi}
3633 \newcommand{\listtablename}{\if@english List of Tables\else 表目次\fi}

\refname
\bibname 3634 \newcommand{\refname}{\if@english References\else 参考文献\fi}
3635 \newcommand{\bibname}{\if@english Bibliography\else 参考文献\fi}
\indexname 3636 \newcommand{\indexname}{\if@english Index\else 索引\fi}
```

`\figurename`

```
\tablename 3637 %<!jspf>\newcommand{\figurename}{\if@english Fig.~\else 図\fi}
3638 %<jspf>\newcommand{\figurename}{Fig.~}
3639 %<!jspf>\newcommand{\tablename}{\if@english Table~\else 表\fi}
3640 %<jspf>\newcommand{\tablename}{Table~}
```

`\appendixname`

```
\abstractname 3641 % \newcommand{\appendixname}{\if@english Appendix~\else 付録\fi}
3642 \newcommand{\appendixname}{\if@english \else 付録\fi}
3643 %<!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 \TeX ではそもそも非 ASCII の制御綴は使えないのであるが、JS クラスのユーザ命令である `\西暦`/`\和暦` だけは擬似的に使えるようにする。欧文 \TeX では

- `\西暦=\^^e8^^a5^^bf^^e6^^9a^^a6`
- `\和暦=\^^e5^^92^^8c^^e6^^9a^^a6`

と扱われるため、`\^^e8` と `\^^e5` を「固定の引数付のマクロ」として定義すればよい。もちろん、同じバイトで始まる他の名前（例えば `\西暦 true`）とは共存できないので、この 2 つのユーザ命令以外の非 ASCII の制御綴は使わないようにする。

\TeX エンジンの種類により処理を分ける。

```
3644 \@onlypreamble\bxjs@decl@Seireki@cmds
3645 \@tempswafalse
3646 \if p\jsEngine \@tempwattrue \fi
3647 \if n\jsEngine \@tempwattrue \fi
3648 \bxjs@cond\if@tempswa\fi{%
```

8 ビット欧文 \TeX の場合。

`\ifjsSeireki` [スイッチ] `西暦` スイッチ (`\if 西暦`) の代わりに用いる。

```
3649 \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。

```
3650 \def\bxjs@decl@Seireki@cmds{%
3651   \def\西暦{\jsSeirekitrue}%
3652   \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`

```

3653 \def\Seireki{\jsSeirekitrue}
3654 \def\Wareki{\jsSeirekifalse\bxjs@wareki@used}

3655 \def\bxjs@if@use@seireki{\bxjs@cond\ifjsSeireki\fi}
3656 \def\bxjs@iai{\noexpand~}
3657 }{%

```

8 ビット欧文 TeX ではない場合。ここでは JS クラスと合わせるため 西暦 スイッチを使う。

```

3658 \newif\if 西暦 \西暦 true
3659 \def\bxjs@decl@Seireki@cmds{%
3660   \def\西暦{\西暦 true}%
3661   \def\和暦{\西暦 false\bxjs@wareki@used}}
3662 \def\Seireki{\西暦 true}
3663 \def\Wareki{\西暦 false\bxjs@wareki@used}
3664 \def\bxjs@if@use@seireki{\bxjs@cond\if 西暦\fi}
3665 \let\bxjs@iai\@empty
3666 }
3667 \bxjs@decl@Seireki@cmds
3668 \let\bxjs@unxp\@firstofone \let\bxjs@onxp\@firstofone
3669 \bxjs@test@engine\unexpanded{%
3670   \let\bxjs@unxp\unexpanded \def\bxjs@onxp{\unexpanded\expandafter}}

```

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

```

3671 \newif\ifbxjs@bxwareki@avail
3672 \IfFileExists{bxwareki.sty}{%
3673   \RequirePackage{bxwareki}[]%
3674   \bxjs@bxwareki@availtrue}{%

```

\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。

```

3675 \ifbxjs@bxwareki@avail \let\bxjs@wareki@used\@empty
3676 \else
3677   \bxjs@robust@def\bxjs@wareki@used{%
3678     \global\let\bxjs@wareki@used\@empty
3679     \ClassWarning\bxjs@clsname
3680       {Wareki mode is not supported, since\MessageBreak
3681         'bxwareki' is unavailable, reported}}
3682     \g@addto@macro\bxjs@begin@document@hook{%
3683       \let\bxjs@wareki@used\@empty}
3684 \fi

```

\jayear 和暦における年の表記の「年」以前の部分（元号＋年数）。

※\heisei の代替となる機能（だから常に和暦を扱う）。

\heisei 年数を表す整数レジスタで、元号が「平成」である場合にのみ定義される。

※ JS クラスと互換の機能。

```

3685 \ifbxjs@bxwareki@avail
3686   \let\jayear\warekiyear
3687   \def\bxjs@tmpa{H}\ifx\bxjs@tmpa\warekigengoinitial
3688     \newcount\heisei \heisei=\value{warekiyear}
3689   \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。

```
3690 \else
3691   \edef\jyear{\the\year \bxjs@iai}
3692 \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。

```
3693 %<jsclasses>
3694 \newif\if 西暦 \西暦 true
3695 \def\西暦{\西暦 true}
3696 \def\和暦{\西暦 false}
3697 \newcount\heisei \heisei\year \advance\heisei-1988\relax
3698 \def\plt@today@year@#1{%
3699   \ifnum\numexpr\year-#1=1 元\else
3700     \ifnum1=\iftdir\ifmdir0\else1\fi\else0\fi
3701     \kansuji\numexpr\year-#1\relax
3702   \else
3703     \number\numexpr\year-#1\relax\nobreak
3704   \fi
3705 \fi 年
3706 }
3707 \def\plt@today@year{%
3708   \ifnum\numexpr\year*10000+\month*100+\day<19890108
3709     昭和\plt@today@year@{1925}%
3710   \else\ifnum\numexpr\year*10000+\month*100+\day<20190501
3711     平成\plt@today@year@{1988}%
3712   \else
3713     令和\plt@today@year@{2018}%
3714   \fi\fi}
3715 %</jsclasses>
3716 \let\bxjs@next\relax
3717 \ifbxjs@bxwareki@avail \ifx\warekigengo@\empty\else
3718   \def\bxjs@next{\bxjs@onxp{\warekitoday}}
3719 \fi\fi
3720 \edef\bxjs@today{%
3721   \if@english
3722     \ifcase\month\or
3723       January\or February\or March\or April\or May\or June\or
3724       July\or August\or September\or October\or November\or December\fi
3725     \space\number\day, \number\year
3726   \else
3727     \ifx\bxjs@next\relax \expandafter\@firstoftwo
3728     \else \noexpand\bxjs@if@use@seireki
3729     \fi {%
3730       \number\year\bxjs@iai\bxjs@unxp{年}%
3731       \bxjs@iai\number\month\bxjs@iai\bxjs@unxp{月}%
3732       \bxjs@iai\number\day\bxjs@iai\bxjs@unxp{日}%
```

```

3733     }\bxjs@next}%
3734 \fi}
3735 \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。

```

3736 \g@addto@macro\bxjs@begin@document@hook{%
3737   \ifx\bb1\jpn@maybekansuji\@undefined\else
3738     \bxjs@decl@Seireki@cmds
3739     \g@addto@macro\datejapanese{%
3740       \let\today\bxjs@today}%
3741   \fi}

```

■ハイフネーション例外 $\mathrm{T}_{\mathrm{E}}\mathrm{X}$ のハイフネーションルールの補足です (ペンディング: eng-lish)

```

3742 \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。

```

3743 %<slide>\pagestyle{empty}%
3744 %<article|report>\pagestyle{plain}%
3745 %<book>\pagestyle{headings}%
3746 \pagenumbering{arabic}
3747 \if@twocolumn
3748   \twocolumn
3749   \sloppy
3750   \flushbottom
3751 \else
3752   \onecolumn
3753   \raggedbottom
3754 \fi
3755 %<*>slide>
3756 \renewcommand\familydefault{\sfdefault}
3757 \raggedright
3758 %</slide>

```

■BXJS 独自の追加処理 🍷

フックを実行する。

```

3759 \bxjs@pre@jadriver@hook

```

和文ドライバのファイルを読み込む。

```

3760 \input{bxjsja-\bxjs@jadriver.def}

```

おしまい。

15 実験的コード

この節は JS クラスの話で、BXJS クラスには当てはまらない。

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

3761 %</class>

以上です。

付録 A 和文ドライバの仕様

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

- `\jsDocClass` [文字トークンの `let`] 文書クラスの種類を示し、次のいずれかと一致する (`\if` で判定可能)。
 - `\jsArticle` `bxjsarticle` クラス
 - `\jsBook` `bxjsbook` クラス
 - `\jsReport` `bxjsreport` クラス
 - `\jsSlide` `bxjsslide` クラス
- `\jsEngine` [文字トークンの `let`] 使用されているエンジンの種別。 (`\if` で判定可能)。
 - `p` `pdfTeX` (DVI モードも含む)
 - `l` `LuaTeX` (〃)
 - `x` `XYTeX`
 - `j` `pTeX` または `upTeX`
 - `n` 以上の何れでもない
- `\ifjsWithupTeX` [スイッチ] 使用されているエンジンが `upTeX` であるか。
- `\ifjsWitheTeX` [スイッチ] 使用されているエンジンが 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」はこの設定と辻褄が合うように和文フォントサイズを調整する必要がある。ほとんどの場合、和文フォントを NFSS で規定する際に `\jsScale` の値をスケール値として与えれば上手くいく。

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

以下で標準で用意されている和文ドライバの実装を示す。

```
3762 %<*>
```

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-jatex 等) を自分で読み込んで適切な設定を行うという使用状況を想定している。

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

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

B.1 補助マクロ

```
3763 %<*>
```

```
3764 %% このファイルは日本語文字を含みます
```

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

```
3765 \def\DeclareJaTextFontCommand#1#2{%
3766   \DeclareRobustCommand#1[1]{%
3767     \relax
3768     \ifmmode \expandafter\nfss@text \fi
3769     {#2##1}}%
3770 }
```

`\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。

```
3771 \def\DeclareJaMathFontCommand#1#2{%
3772   \DeclareRobustCommand#1[1]{%
3773     \relax
3774     \ifmmode\else \non@alpherr{#1\space}\fi
```

```

3775 \nfss@text{\fontfamily\familydefault
3776 \fontseries{m}\fontshape{n}\selectfont\relax
3777 #2##1}%
3778 }%
3779 }

```

`\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。

```

3780 \long\def\bxjs@@CSsfdefault{\sfdefault}%
3781 \@onlypreamble\bxjs@if@sf@default
3782 \def\bxjs@if@sf@default#1{%
3783 \ifx\familydefault\bxjs@@CSsfdefault#1\fi
3784 \g@addto@macro\bxjs@begin@document@hook{%
3785 \ifx\familydefault\bxjs@@CSsfdefault#1\fi}%
3786 }

```

`\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。

```

3787 \@tempdima\p@ \bxjs@invscale\@tempdima\jsScale
3788 \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。

```

3789 \def\jsLetHeadChar#1#2{%
3790 \begingroup
3791 \escapechar=`\ %
3792 \let\bxjs@tmpa={% brace-match-hack
3793 \bxjs@let@hchar@exp#2}%
3794 \endgroup
3795 \let#1\bxjs@g@tmpa}
3796 \def\bxjs@let@hchar@exp{%
3797 \futurelet\@let@token\bxjs@let@hchar@exp@a}
3798 \def\bxjs@let@hchar@exp@a{%
3799 \bxjs@cond@ifcat\noexpand\@let@token\bgroup\fi{% 波括弧
3800 \bxjs@let@hchar@out\let\relax
3801 }{\bxjs@cond@ifcat\noexpand\@let@token\@sptoken\fi{% 空白
3802 \bxjs@let@hchar@out\let\space%
3803 }{\bxjs@cond@if\noexpand\@let@token\@backslashchar\fi{% バックスラッシュ
3804 \bxjs@let@hchar@out\let\@backslashchar
3805 }{\bxjs@let@hchar@exp@b}}}}
3806 \def\bxjs@let@hchar@exp@b#1{%
3807 \expandafter\bxjs@let@hchar@exp@c\string#1?\@nil#1}

```

```

3808 \def\bxjs@let@hchar@exp@c#1#2\@nil{%
3809 %\message{<#1#2>}%
3810 \bxjs@cond\if#1\@backslashchar\fi{% 制御綴
3811 \bxjs@cond\expandafter\ifx\noexpand\@let@token\@let@token\fi{%
3812 \bxjs@let@hchar@out\let\relax
3813 }{%else
3814 \expandafter\bxjs@let@hchar@exp
3815 }%
3816 }{%else
3817 \bxjs@let@hchar@chr#1%
3818 }}
3819 \def\bxjs@let@hchar@chr#1{%
3820 \bxjs@let@hchar@out\def{{#1}}}
3821 \def\bxjs@let@hchar@out#1#2{%
3822 \global#1\bxjs@g@tmpa#2\relax
3823 \toks@\bgroup}% skip to right brace

```

UTF-8 のバイト列を扱うコード。

```

3824 \chardef\bxjs@let@hchar@csta=128
3825 \chardef\bxjs@let@hchar@cstb=192
3826 \chardef\bxjs@let@hchar@cstc=224
3827 \chardef\bxjs@let@hchar@cstd=240
3828 \chardef\bxjs@let@hchar@cste=248
3829 \let\bxjs@let@hchar@chr@ue@a\bxjs@let@hchar@chr
3830 \def\bxjs@let@hchar@chr@ue#1{%
3831 \@tempcnta=`#1\relax
3832 %\message{\the\@tempcnta}%
3833 \bxjs@cond\ifnum\@tempcnta<\bxjs@let@hchar@csta\fi{%
3834 \bxjs@let@hchar@chr@ue@a#1%
3835 }{\bxjs@cond\ifnum\@tempcnta<\bxjs@let@hchar@cstb\fi{%
3836 \bxjs@let@hchar@out\let\relax
3837 }{\bxjs@cond\ifnum\@tempcnta<\bxjs@let@hchar@cstc\fi{%
3838 \bxjs@let@hchar@chr@ue@b
3839 }{\bxjs@cond\ifnum\@tempcnta<\bxjs@let@hchar@cstd\fi{%
3840 \bxjs@let@hchar@chr@ue@c
3841 }{\bxjs@cond\ifnum\@tempcnta<\bxjs@let@hchar@cste\fi{%
3842 \bxjs@let@hchar@chr@ue@d
3843 }{%else
3844 \bxjs@let@hchar@out\let\relax
3845 }}}}}
3846 \def\bxjs@let@hchar@chr@ue@a#1{%
3847 \bxjs@let@hchar@out\def{{#1}}}
3848 \def\bxjs@let@hchar@chr@ue@b#1#2{%
3849 \bxjs@let@hchar@out\def{{#1#2}}}
3850 \def\bxjs@let@hchar@chr@ue@c#1#2#3{%
3851 \bxjs@let@hchar@out\def{{#1#2#3}}}
3852 \def\bxjs@let@hchar@chr@ue@d#1#2#3#4{%
3853 \bxjs@let@hchar@out\def{{#1#2#3#4}}}

```

B.2 (u)pTeX 用の設定

```
3854 \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。

```
3855 \def\bxjs@let@hchar@chr@pp#1#2{%
3856   \expandafter\bxjs@let@hchar@chr@pp@a\meaning#2\relax#1#2}
3857 \def\bxjs@let@hchar@chr@pp@a#1#2\relax#3#4{%
3858   %\message{(\meaning#3:\meaning#4)}%
3859   \bxjs@cond@if#1k\fi{%
3860     \bxjs@let@hchar@out\def{{#4}}%
3861   }{%else
3862     \bxjs@let@hchar@chr@ue#3#4%
3863   }}
3864 \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。

```
3865 \edef\jsc@JYn{\ifjsWithupTeX JY2\else JY1\fi}
3866 \edef\jsc@JTn{\ifjsWithupTeX JT2\else JT1\fi}
3867 \edef\jsc@pfx@{\ifjsWithupTeX u\fi}
```

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

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

```
3868 \@onlypreamble\bxjs@declarefontshape
3869 \ifjsWithupTeX
3870 \def\bxjs@declarefontshape{%
3871   \DeclareFontShape{JY2}{mc}{m}{n}{<->s*[\bxjs@scale]upjpnrm-h}{}%
3872   \DeclareFontShape{JY2}{gt}{m}{n}{<->s*[\bxjs@scale]upjpngt-h}{}%
3873   \DeclareFontShape{JT2}{mc}{m}{n}{<->s*[\bxjs@scale]upjpnrm-v}{}%
3874   \DeclareFontShape{JT2}{gt}{m}{n}{<->s*[\bxjs@scale]upjpngt-v}{}%
3875 }
3876 \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。

```
3877 \else
3878 \def\bxjs@declarefontshape{%
```



```

3879 \DeclareFontShape{JY1}{mc}{m}{n}{<->s*[\bxjs@scale]jis}{}%
3880 \DeclareFontShape{JY1}{gt}{m}{n}{<->s*[\bxjs@scale]jisg}{}%
3881 \DeclareFontShape{JT1}{mc}{m}{n}{<->s*[\bxjs@scale]tmin10}{}%
3882 \DeclareFontShape{JT1}{gt}{m}{n}{<->s*[\bxjs@scale]tgoth10}{}%
3883 }
3884 \def\bxjs@sizereference{jis}
3885 \fi

```

既に使用されている標準和文フォント定義がもしあれば取り消す。

```

3886 \def\bxjs@next#1/#2/#3/#4/#5\relax{%
3887   \def\bxjs@tmpb{#5}}
3888 \ifjsWithpTeXng \def\bxjs@tmpb{10}%
3889 \else
3890 \expandafter\expandafter\expandafter\bxjs@next
3891 \expandafter\string\the\jfont\relax
3892 \fi
3893 \@for\bxjs@tmpa:={\jsc@JYn/mc/m/n,\jsc@JYn/gt/m/n,%
3894                  \jsc@JTn/mc/m/n,\jsc@JTn/gt/m/n}\do
3895   {\expandafter\let\csname\bxjs@tmpa/10\endcsname=\@undefined
3896    \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。

```

3897 \begingroup
3898 % 参照用フォント (\bxjs@sizereference) の全角空白の幅を取得
3899 \font\bxjs@tmpa=\bxjs@sizereference\space at 10pt
3900 \setbox\z@\hbox{\bxjs@tmpa\char\jis"2121\relax}
3901 % 幅が丁度 10pt なら補正は不要
3902 \ifdim\wd\z@=10pt
3903   \global\let\bxjs@scale\jsScale
3904 \else
3905 %  $(10*s)/(10*f)$  として計算、\bxjs@invscale は BXJS で定義
3906   \edef\bxjs@tmpa{\strip@pt\wd\z@}
3907   \@tempdima=10pt \@tempdima=\jsScale\@tempdima
3908   \bxjs@invscale\@tempdima\bxjs@tmpa
3909   \xdef\bxjs@scale{\strip@pt\@tempdima}
3910 \fi
3911 \endgroup
3912 %\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。

```

3913 \bxjs@declarefontshape

```

フォント代替の明示的定義。

```

3914 \DeclareFontShape{\jsc@JYn}{mc}{m}{it}{<->ssub*mc/m/n}{ }
3915 \DeclareFontShape{\jsc@JYn}{mc}{m}{sl}{<->ssub*mc/m/n}{ }
3916 \DeclareFontShape{\jsc@JYn}{mc}{m}{sc}{<->ssub*mc/m/n}{ }
3917 \DeclareFontShape{\jsc@JYn}{gt}{m}{it}{<->ssub*gt/m/n}{ }
3918 \DeclareFontShape{\jsc@JYn}{gt}{m}{sl}{<->ssub*gt/m/n}{ }
3919 \DeclareFontShape{\jsc@JYn}{mc}{bx}{it}{<->ssub*gt/m/n}{ }
3920 \DeclareFontShape{\jsc@JYn}{mc}{bx}{sl}{<->ssub*gt/m/n}{ }
3921 \DeclareFontShape{\jsc@JYn}{gt}{bx}{it}{<->ssub*gt/m/n}{ }
3922 \DeclareFontShape{\jsc@JYn}{gt}{bx}{sl}{<->ssub*gt/m/n}{ }
3923 \DeclareFontShape{\jsc@JYn}{mc}{b}{n}{<->ssub*mc/bx/n}{ }
3924 \DeclareFontShape{\jsc@JYn}{mc}{b}{it}{<->ssub*mc/bx/n}{ }
3925 \DeclareFontShape{\jsc@JYn}{mc}{b}{sl}{<->ssub*mc/bx/n}{ }
3926 \DeclareFontShape{\jsc@JYn}{gt}{b}{n}{<->ssub*gt/bx/n}{ }
3927 \DeclareFontShape{\jsc@JYn}{gt}{b}{it}{<->ssub*gt/bx/n}{ }
3928 \DeclareFontShape{\jsc@JYn}{gt}{b}{sl}{<->ssub*gt/bx/n}{ }
3929 \DeclareFontShape{\jsc@JTn}{mc}{m}{it}{<->ssub*mc/m/n}{ }
3930 \DeclareFontShape{\jsc@JTn}{mc}{m}{sl}{<->ssub*mc/m/n}{ }
3931 \DeclareFontShape{\jsc@JTn}{mc}{m}{sc}{<->ssub*mc/m/n}{ }
3932 \DeclareFontShape{\jsc@JTn}{gt}{m}{it}{<->ssub*gt/m/n}{ }
3933 \DeclareFontShape{\jsc@JTn}{gt}{m}{sl}{<->ssub*gt/m/n}{ }
3934 \DeclareFontShape{\jsc@JTn}{mc}{bx}{it}{<->ssub*gt/m/n}{ }
3935 \DeclareFontShape{\jsc@JTn}{mc}{bx}{sl}{<->ssub*gt/m/n}{ }
3936 \DeclareFontShape{\jsc@JTn}{gt}{bx}{it}{<->ssub*gt/m/n}{ }
3937 \DeclareFontShape{\jsc@JTn}{gt}{bx}{sl}{<->ssub*gt/m/n}{ }
3938 \DeclareFontShape{\jsc@JTn}{mc}{b}{n}{<->ssub*mc/bx/n}{ }
3939 \DeclareFontShape{\jsc@JTn}{mc}{b}{it}{<->ssub*mc/bx/n}{ }
3940 \DeclareFontShape{\jsc@JTn}{mc}{b}{sl}{<->ssub*mc/bx/n}{ }
3941 \DeclareFontShape{\jsc@JTn}{gt}{b}{n}{<->ssub*gt/bx/n}{ }
3942 \DeclareFontShape{\jsc@JTn}{gt}{b}{it}{<->ssub*gt/bx/n}{ }
3943 \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。

```

3944 \@ifl@t@r\fmtversion{2020/10/01}
3945   {\jsc@needspace@tchfalse}{\jsc@needspace@tchtrue}
3946 \ifjsc@needspace@tch % --- for 2020-02-02 or older BEGIN
3947 \ifx\@rmfamilyhook\@undefined % old
3948 \DeclareRobustCommand\rmfamily
3949   {\not@math@alphabet\rmfamily\mathrm
3950    \romanfamily\rmdefault\kanjifamily\mcdefault\selectfont}
3951 \DeclareRobustCommand\sffamily
3952   {\not@math@alphabet\sffamily\mathsf
3953    \romanfamily\sfddefault\kanjifamily\gtdefault\selectfont}
3954 \DeclareRobustCommand\ttfamily
3955   {\not@math@alphabet\ttfamily\mathtt
3956    \romanfamily\ttdefault\kanjifamily\gtdefault\selectfont}
3957 \g@addto@macro\bxjs@begin@document@hook{%
3958   \ifx\mweights@init\@undefined\else % mweights.sty is loaded

```

```

3959 % my definitions above should have been overwritten, recover it!
3960 % \selectfont is executed twice but I don't care about speed...
3961 \expandafter\g@addto@macro\csname rmfamily \endcsname
3962   {\kanjifamily\mcdefault\selectfont}%
3963 \expandafter\g@addto@macro\csname sffamily \endcsname
3964   {\kanjifamily\gtdefault\selectfont}%
3965 \expandafter\g@addto@macro\csname ttfamily \endcsname
3966   {\kanjifamily\gtdefault\selectfont}%
3967 \fi}
3968 \else % 2020-02-02
3969 \g@addto@macro\@rmfamilyhook
3970   {\prepare@family@series@update@kanji{mc}\mcdefault}
3971 \g@addto@macro\@sffamilyhook
3972   {\prepare@family@series@update@kanji{gt}\gtdefault}
3973 \g@addto@macro\@ttfamilyhook
3974   {\prepare@family@series@update@kanji{gt}\gtdefault}
3975 \fi
3976 \else % --- for 2020-02-02 or older END & for 2020-10-01 BEGIN
3977 \AddToHook{rmfamily}%
3978   {\prepare@family@series@update@kanji{mc}\mcdefault}
3979 \AddToHook{sffamily}%
3980   {\prepare@family@series@update@kanji{gt}\gtdefault}
3981 \AddToHook{ttfamily}%
3982   {\prepare@family@series@update@kanji{gt}\gtdefault}
3983 \fi % --- for 2020-10-01 END
3984 \ifx\DeclareFixJFMCJKTextFontCommand\@undefined
3985 \DeclareJaTextFontCommand{\textmc}{\mcfamily}
3986 \DeclareJaTextFontCommand{\textgt}{\gtfamily}
3987 \fi
3988 \bxjs@if@sf@default{%
3989   \renewcommand\kanjifamilydefault{\gtdefault}}

```

念のため。

```

3990 \selectfont

```

これ以降では、`\bxjs@parse@qh` の処理は p_TE_X 系では不要になるので無効化する（つまり `\jsSetQHLength` は `\setlength` と等価になる）。

```

3991 \def\bxjs@parse@qh#1{\let\bxjs@tmpb\relax}
3992 \let\bxjs@parse@qh@a\@undefined
3993 \let\bxjs@parse@qh@b\@undefined

```

■パラメタの設定

```

3994 \prebreakpenalty\jis"2147=10000
3995 \postbreakpenalty\jis"2148=10000
3996 \prebreakpenalty\jis"2149=10000
3997 \inhibitxspcode`!=1
3998 \inhibitxspcode`〒=2
3999 \xspcode`+=3
4000 \xspcode`%=3

```

"80～"FF の範囲の \spcode を 3 に変更。

```
4001 \@tempcnta="80 \@whilenum\@tempcnta<"100 \do{%  
4002 \xspcode\@tempcnta=3\advance\@tempcnta\@ne}
```

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

```
4003 \let\jsInhibitGlueAtParTop\@inhibitglue
```

\jsResetDimen は空のままでよい。

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

```
4004 \begingroup  
4005 \catcode`\!=0
```

\bxjs@ptex@dir 現在の組方向：t=縦、y=横、?=その他。

```
4006 \gdef\bxjs@ptex@dir{%  
4007 !iftdir t%  
4008 !else!ifydir y%  
4009 !else ?%  
4010 !fi!fi}
```

新版の pTeX で脚注番号の周囲の空きが過大になる現象への対処。

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

```
4011 % 古い \@makefnmark の定義  
4012 \long\def\bxjs@tmpa{\hbox{%  
4013 !ifydir \@textsuperscript{\normalfont\@thefnmark}}%  
4014 !else\hbox{\yoko\@textsuperscript{\normalfont\@thefnmark}}!fi}}  
4015 \ifx\@makefnmark\bxjs@tmpa  
4016 \long\gdef\@makefnmark{%  
4017 !ifydir \hbox{\hbox{\@textsuperscript{\normalfont\@thefnmark}}\hbox{}}%  
4018 !else\hbox{\yoko\@textsuperscript{\normalfont\@thefnmark}}!fi}  
4019 \fi
```

エスケープ文字の変更はここまで。

```
4020 \endgroup
```

■minijs パッケージのブロック やっておく。

```
4021 \@namedef{ver@minijs.sty}{}
```

B.3 pdfTeX 用の処理

```
4022 \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。

```
4023 \let\bxjs@let@hchar@chr\bxjs@let@hchar@chr@ue
```

ムニャムニャ。

```

4024 \@onlypreamble\bxjs@CJKloaded
4025 \def\bxjs@CJKloaded{%
4026   \def\@footnotemark{%
4027     \leavevmode
4028     \ifhmode
4029       \edef\x@sf{\the\spacefactor}%
4030       \ifdim\lastkern>\z@\ifdim\lastkern<5sp\relax
4031         \unkern\unkern
4032         \ifdim\lastskip>\z@ \unskip \fi
4033       \fi\fi
4034       \nobreak
4035     \fi
4036     \@makefnmark
4037     \ifhmode \spacefactor\x@sf \fi
4038   \relax}%
4039 \let\bxjs@CJKloaded\relax
4040 }
4041 \g@addto@macro\bxjs@begin@document@hook{%
4042   \@ifpackageloaded{CJK}{%
4043     \bxjs@CJKloaded
4044   }{}%
4045 }

```

B.4 X₃TeX 用の処理

```

4046 \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。

```

4047 \def\bxjs@let@hchar@chr#1{%
4048   \@tempcnta`#1\relax \divide\@tempcnta"800\relax
4049   \bxjs@cond\ifnum\@tempcnta=27 \fi{%
4050     \bxjs@let@hchar@chr@xe
4051   }\bxjs@let@hchar@out\def{{#1}}}}
4052 \def\bxjs@let@hchar@chr@xe#1{%
4053   \lccode`0=`#1\relax
4054   \lowercase{\bxjs@let@hchar@out\def{{0}}}}

```

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

```

4055 \@onlypreamble\bxjs@do@precisetext
4056 \ifx\XeTeXgenerateactualtext\undefined\else
4057   \def\bxjs@do@precisetext{%
4058     \XeTeXgenerateactualtext=\@one}
4059 \fi

```

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

TODO: バージョン要件を見直して暫定措置を解除する。

```

4060 \@onlypreamble\bxjs@do@simplejasetup
4061 \def\bxjs@do@simplejasetup{%
4062   \@namedef{bxjs@zeroglue/0.0pt}{T}}%

```

```

4063 \ifnum\XeTeXinterchartokenstate>\z@
4064 \else\expandafter\ifx\csname bxjs@zeroglue/\the\XeTeXlinebreakskip\endcsname\relax\else
4065 \jsSimpleJaSetup
4066 \ClassInfo\bxjs@clsname
4067 {\string\jsSimpleJaSetup' is applied\@gobble}%
4068 \fi\fi}

```

`\jsSimpleJaSetup` 日本語出力用の超簡易的な設定。

```

4069 \newcommand*{\jsSimpleJaSetup}{%
4070 \XeTeXlinebreaklocale "ja"\relax
4071 \XeTeXlinebreakskip=0pt plus 1pt minus 0.1pt\relax
4072 \XeTeXlinebreakpenalty=0\relax}

```

B.5 後処理（エンジン共通）

```

4073 \fi\fi\fi

```

`simplejasetup` オプションの処理。

```

4074 \ifx\bxjs@do@simplejasetup\@undefined\else
4075 \g@addto@macro\bxjs@begin@document@hook{%
4076 \ifbxjs@simplejasetup
4077 \bxjs@do@simplejasetup
4078 \fi}
4079 \fi

```

`precisetext` オプションの処理。

```

4080 \ifbxjs@precisetext
4081 \ifx\bxjs@do@precisetext\@undefined
4082 \ClassWarning\bxjs@clsname
4083 {The current engine does not support the\MessageBreak
4084 'precise-text' option\@gobble}
4085 \else
4086 \bxjs@do@precisetext
4087 \fi
4088 \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。

```

4089 \g@addto@macro\bxjs@begin@document@hook{%
4090 \ifx\jsInhibitGlueAtParTop\@empty
4091 \def\bxjs@tmpa{\jsInhibitGlueAtParTop}%
4092 \ifx\everyparhook\bxjs@tmpa
4093 \let\everyparhook\@empty
4094 \fi
4095 \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。

```

4096 \ifnum\bxjs@everyparhook=\bxjs@everyparhook@@modern

```

まず `\everypar` を“乗っ取る”処理を行う。

```
4097 \let\bxjs@everypar\everypar
4098 \newtoks\everypar
4099 \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。

```
4100 \bxjs@everypar{\the\expandafter\everypar\everyparhook}%
4101 \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。

```
4102 \ifbxjs@fancyhdr
```

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

```
4103 \@onlypreamble\bxjs@adjust@fancyhdr
4104 \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。

```
4105 \def\bxjs@tmpa{\fancyplain{}{\sl\rightmark}\strut}%
4106 \def\bxjs@tmpb{\fancyplain{}{\rightmark}\strut}%
4107 \ifx\f@ncyelh\bxjs@tmpa \global\let\f@ncyelh\bxjs@tmpb \fi
4108 \ifx\f@ncyerh\bxjs@tmpa \global\let\f@ncyerh\bxjs@tmpb \fi
4109 \ifx\f@ncyolh\bxjs@tmpa \global\let\f@ncyolh\bxjs@tmpb \fi
4110 \ifx\f@ncyorh\bxjs@tmpa \global\let\f@ncyorh\bxjs@tmpb \fi
4111 \def\bxjs@tmpa{\fancyplain{}{\sl\leftmark}\strut}%
4112 \def\bxjs@tmpb{\fancyplain{}{\leftmark}\strut}%
4113 \ifx\f@ncyelh\bxjs@tmpa \global\let\f@ncyelh\bxjs@tmpb \fi
4114 \ifx\f@ncyerh\bxjs@tmpa \global\let\f@ncyerh\bxjs@tmpb \fi
4115 \ifx\f@ncyolh\bxjs@tmpa \global\let\f@ncyolh\bxjs@tmpb \fi
4116 \ifx\f@ncyorh\bxjs@tmpa \global\let\f@ncyorh\bxjs@tmpb \fi
4117 \def\bxjs@tmpa{\rm\thepage\strut}%
4118 \def\bxjs@tmpb{\thepage\strut}%
4119 \ifx\f@ncyecf\bxjs@tmpa \global\let\f@ncyecf\bxjs@tmpb \fi
4120 \ifx\f@ncyocf\bxjs@tmpa \global\let\f@ncyocf\bxjs@tmpb \fi
```

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

```
4121 \ifx\fullwidth\undefined\else \ifdim\textwidth<\fullwidth
4122 \setlength{\@tempdima}{\fullwidth-\textwidth}%
4123 \edef\bxjs@tmpa{\noexpand\fancyhfoffset[EL,OR]{\the\@tempdima}%
4124 }\bxjs@tmpa
4125 \fi\fi
```

```

4126 \PackageInfo\bxjs@clsname
4127 {Patch to fancyhdr is applied\@gobble}}

```

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

```

4128 \def\bxjs@pagestyle@hook{%
4129   \@ifpackageloaded{fancyhdr}{%
4130     \bxjs@adjust@fancyhdr
4131     \global\let\bxjs@adjust@fancyhdr\relax
4132   }{}}

```

`\pagestyle` にフックを入れ込む。

```

4133 \let\bxjs@org@pagestyle\pagestyle
4134 \def\pagestyle{%
4135   \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。

```

4136 \g@addto@macro\bxjs@begin@document@hook{%
4137   \bxjs@pagestyle@hook
4138   \global\let\bxjs@pagestyle@hook\relax}

```

終わり。

```

4139 \fi

```

■和文空白命令

```

4140 \ifbxjs@jaspace@cmd

```

`\jaenspace` 半角幅の水平空き。

```

4141 \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。

```

4142 \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。

```

4143 \def\_{\zwspace}

```

`\jaspace` `jlreq` クラスと互換の命令。

```

4144 \DeclareRobustCommand*{\jaspace}[1]{%
4145   \expandafter\ifx\csname bxjs@jaspace@@#1\endcsname\relax
4146     \ClassError\bxjs@clsname
4147     {Unknown jaspac: #1}{\@eha}%
4148   \else
4149     \csname bxjs@jaspace@@#1\endcsname
4150   \fi}
4151 \def\bxjs@jaspace@@zenkaku{\hskip 1\jsZw\relax}
4152 \def\bxjs@jaspace@@nibu{\hskip .5\jsZw\relax}
4153 \def\bxjs@jaspace@@shibu{\hskip .25\jsZw\relax}

```


終わり。

```
4154 \fi
```

以上で終わり。

```
4155 %</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

■和文フォント指定の扱い

C.1 準備

まず minimal ドライバを読み込む。

```
4156 %<*standard>
```

```
4157 %% このファイルは日本語文字を含みます
```

```
4158 \input{bxjsja-minimal.def}
```

simplejasetup は standard では無効になる。

```
4159 \bxjs@simplejasetupfalse
```

C.2 和文ドライバパラメタ

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

\ifbxjs@jp@jismmiv 2004JIS 字形を優先させるか。

```
4160 \newif\ifbxjs@jp@jismmiv
```

jis2004 オプションの処理。

```
4161 \bxjs@cslet{bxjs@kv@jis2004@true}\bxjs@jp@jismmivtrue
```

```
4162 \bxjs@cslet{bxjs@kv@jis2004@false}\bxjs@jp@jismmivfalse
```

```
4163 \define@key{bxjsStd}{jis2004}[true]{%
```

```
4164 \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。

```
4165 \newif\ifbxjs@jp@units
```

`units` オプションの処理。

```
4166 \let\bxjs@kv@units@true\bxjs@jp@unitstrue
4167 \let\bxjs@kv@units@false\bxjs@jp@unitsfalse
4168 \define@key{bxjsStd}{units}[true]{%
4169   \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。

```
4170 \let\bxjs@jp@font\@empty
```

`font` オプションの処理。

```
4171 \define@key{bxjsStd}{font}{%
4172   \edef\bxjs@jp@font{#1}}
```

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

```
4173 \newif\ifbxjs@jp@strong@cmd \bxjs@jp@strong@cmdtrue
```

`strong-cmd` オプションの処理。

```
4174 \let\bxjs@kv@strongcmd@true\bxjs@jp@strong@cmdtrue
4175 \let\bxjs@kv@strongcmd@false\bxjs@jp@strong@cmdfalse
4176 \define@key{bxjs}{strong-cmd}[true]{\bxjs@set@keyval{strongcmd}{#1}{}}
```

実際の `japaram` の値を適用する。

```
4177 \def\bxjs@next#1{\bxjs@safe@setkeys{bxjsStd}{#1}}
4178 \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。

```
4179 \@onlypreamble\bxjs@apply@mmiv
4180 \def\bxjs@apply@mmiv{%
4181   \g@addto@macro\@classoptionslist{jis2004}
4182   % \ifpackagewith 判定への対策
4183   \PassOptionsToPackage{jis2004}{otf}
4184   \global\let\bxjs@apply@mmiv\relax}
4185 \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。

```
4186 \if j\jsEngine
4187   \bxjs@jp@unitsfalse
4188 \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。

```

4189 \ifbxjs@jp@units
4190   \IfFileExists{bxcalc.sty}{%
4191     \RequirePackage{bxcalc}[2018/01/28]%v1.0a
4192     \ifx\usepTeXunits\@undefined
4193       \PackageWarningNoLine{bxjs@clsname}
4194         {Cannot support pTeX units (zw etc.), since\MessageBreak
4195           the package 'bxcalc' is too old}%
4196       \bxjs@jp@unitsfalse
4197     \else \usepTeXunits
4198     \fi
4199   }{%else
4200     \PackageWarningNoLine{bxjs@clsname}
4201       {Cannot support pTeX units (zw etc.), since\MessageBreak
4202         the package 'bxcalc' is unavailable}%
4203     \bxjs@jp@unitsfalse
4204   }
4205 \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。

```

4206 \ifbxjs@jp@units
4207 \def\bxjs@parse@qh#1{\let\bxjs@tmpb\relax}
4208 \let\bxjs@parse@qh@a\@undefined
4209 \let\bxjs@parse@qh@b\@undefined
4210 \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。

```

4211 \ifbxjs@jp@units
4212   \def\bxjs@let@lenexpr#1#2{%
4213     \edef#1{#2}%
4214     \expandafter\CUXParseExpr\expandafter#1\expandafter{#1}}
4215   \else
4216     \def\bxjs@let@lenexpr{\edef}
4217   \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。

```

4218 \ifbxjs@jp@strong@cmd\jsAtEndOfClass{%
4219   \ifx\strong\@undefined\ifx\strongenv\@undefined
4220     \DeclareRobustCommand{\strongenv}{\bxjs@strong@font}%
4221     \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。

```
4222 \newcommand*{\strongfontdeclare}{\bxjs@strongfontdeclare}%
4223 \newcount\bxjs@strong@level
4224 \bxjs@protected\def\bxjs@strongfontdeclare#1{%
4225   \bxjs@set@array@from@clist{\bxjs@strong}{#1}%
4226   \bxjs@strong@level\z@}%
4227 \bxjs@strongfontdeclare{\bfseries}%
4228 \def\bxjs@strong@font{%
4229   \bxjs@csletcs{\bxjs@tmpa}{\bxjs@strong/\the\bxjs@strong@level}%
4230   \ifx\bxjs@tmpa\relax
4231     \advance\bxjs@strong@level\m@ne \bxjs@strong@font
4232   \else \advance\bxjs@strong@level\@ne \bxjs@tmpa
4233   \fi}%
4234 \fi\fi
4235 }\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。

```
4236 \ifjsWithTeX
```

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 4237 \@tempdima=0.25mm
4238 \protected\edef\jQ{\dimexpr\the\@tempdima\relax}
4239 \let\jH\jQ
```

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

```
\trueH 4240 \ifjsc@mag
4241   \@tempdimb=\jsBaseFontSize\relax
4242   \edef\bxjs@tmpa{\strip@pt\@tempdimb}%
4243   \@tempdima=2.5mm
4244   \bxjs@invscale\@tempdima\bxjs@tmpa
4245   \protected\edef\trueQ{\dimexpr\the\@tempdima\relax}
4246   \@tempdima=10pt
4247   \bxjs@invscale\@tempdima\bxjs@tmpa
4248   \protected\edef\bxjs@truept{\dimexpr\the\@tempdima\relax}
4249 \else \let\trueQ\jQ \let\bxjs@truept\p@
4250 \fi
4251 \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。

```
4252 \@tempdima\trueQ \bxjs@invscale\@tempdima\jsScale
4253 \protected\edef\ascQ{\dimexpr\the\@tempdima\relax}
4254 \@tempdima\bxjs@truept \bxjs@invscale\@tempdima\jsScale
```

```

4255 \protected\edef\ascpt{\dimexpr\the\@tempdima\relax}
4256 \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。

```

4257 \def\jafontsize#1#2{%
4258   \begingroup
4259     \bxjs@jafontsize@a{#1}%
4260     \@tempdimb\jsInverseScale\@tempdima
4261     \bxjs@jafontsize@a{#2}%
4262     \xdef\bxjs@g@tmpa{%
4263       \noexpand\fontsize{\the\@tempdimb}{\the\@tempdima}}%
4264   \endgroup\bxjs@g@tmpa}
4265 \def\bxjs@jafontsize@a#1{%
4266   \bxjs@parse@qh{#1}%
4267   \ifx\bxjs@tmpb\relax \def\bxjs@tmpb{#1}\fi
4268   \@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。

```

4269 \def\bxjs@kanjiskip{0pt}

```

`\setkanjiskip` 和文間空白の量を設定する。

```

4270 \newcommand*\setkanjiskip[1]{%
4271   \bxjs@let@lenexpr\bxjs@kanjiskip{#1}%
4272   \bxjs@reset@kanjiskip}

```

`\getkanjiskip` 和文間空白の量を表すテキストに展開する。

```

4273 \newcommand*\getkanjiskip{%
4274   \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。

```

4275 \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 4276 \bxjs@robust@def\bxjs@enable@kanjiskip{%
4277   \bxjs@kanjiskip@enabledtrue
4278   \bxjs@reset@kanjiskip}
4279 \bxjs@robust@def\bxjs@disable@kanjiskip{%
4280   \bxjs@kanjiskip@enabledfalse
4281   \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。

```

4282 \bxjs@robust@def\bxjs@reset@kanjiskip{%
4283   \ifbxjs@kanjiskip@enabled
4284     \setlength{\@tempskipa}{\bxjs@kanjiskip}%

```

```

4285 \else \@tempskipa\z@
4286 \fi
4287 \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 4288 \def\bxjs@xkanjiskip{0pt}
\getxkanjiskip 4289 \newcommand*\setxkanjiskip[1]{%
\ifbxjs@xkanjiskip@enabled 4290 \bxjs@let@lenexpr\bxjs@xkanjiskip{#1}%
4291 \bxjs@reset@xkanjiskip}
\bxjs@enable@xkanjiskip 4292 \newcommand*\getxkanjiskip{%
4293 \bxjs@xkanjiskip}
\bxjs@disable@xkanjiskip 4294 \newif\ifbxjs@xkanjiskip@enabled \bxjs@xkanjiskip@enabledtrue
\bxjs@reset@xkanjiskip 4295 \bxjs@robust@def\bxjs@enable@xkanjiskip{%
4296 \bxjs@xkanjiskip@enabledtrue
4297 \bxjs@reset@xkanjiskip}
4298 \bxjs@robust@def\bxjs@disable@xkanjiskip{%
4299 \bxjs@xkanjiskip@enabledfalse
4300 \bxjs@reset@xkanjiskip}
4301 \bxjs@robust@def\bxjs@reset@xkanjiskip{%
4302 \ifbxjs@xkanjiskip@enabled
4303 \setlength{\@tempskipa}{\bxjs@xkanjiskip}%
4304 \else \@tempskipa\z@
4305 \fi
4306 \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。

```

4307 \g@addto@macro\jsResetDimen{%
4308 \bxjs@reset@kanjiskip
4309 \bxjs@reset@xkanjiskip}
4310 \let\bxjs@apply@kanjiskip\relax
4311 \let\bxjs@apply@xkanjiskip\relax

```

■和文フォント指定の扱い standard 和文ドライバでは `\jsJaFont` の値を和文フォントの“プリセット”の指定として用いる。プリセットの値は、 $\mathrm{T}_{\mathrm{E}}\mathrm{X}$ Live の `kanji-config-updmap` コマンドで使う“ファミリ”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。

```

4312 \@onlypreamble\bxjs@adjust@jafont
4313 \def\bxjs@adjust@jafont#1{%
4314 \ifx\jsJaFont\bxjs@@auto
4315 \bxjs@get@kanjiEmbed
4316 \ifx\bxjs@jaEmbed\relax
4317 \let\bxjs@tmpa\@empty
4318 \else

```

```

4319      \let\bxjs@tmpa\bxjs@jaEmbed
4320      \ifx\bxjs@jaVariant\bxjs@@hziv
4321        \bxjs@apply@mmiv
4322      \fi
4323    \fi
4324  \else
4325    \let\bxjs@tmpa\jsJaFont
4326  \fi
4327  \if f#1\ifx\bxjs@tmpa\bxjs@@noEmbed
4328    \ClassWarningNoLine\bxjs@clsname
4329      {Option 'jafont=noEmbed' is ignored, because it is\MessageBreak
4330        not available on the current situation}%
4331    \let\bxjs@tmpa\@empty
4332  \fi\fi
4333 }
4334 \def\bxjs@@auto{auto}
4335 \def\bxjs@@noEmbed{noEmbed}
4336 \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。

```

4337 \let\bxjs@jaEmbed\relax
4338 \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。

```

4339 \@onlypreamble\bxjs@get@kanjiEmbed
4340 \def\bxjs@get@kanjiEmbed{%
4341   \begingroup\setbox\z@=\hbox{%
4342     \global\let\bxjs@tmpdo\@empty
4343     \def\bxjs@next##1##2##3{%
4344       \def##1####1##3 #####2\@nil####3\@nnil{%
4345         \ifx$####1$\gdef##2{####2}\fi}%
4346       \g@addto@macro\bxjs@tmpdo{%
4347         \expandafter##1\bxjs@tmpa\@nil##3 \@nil\@nnil}}%
4348       \bxjs@next\bxjs@tmpdo@a\bxjs@g@tmpa{kanjiEmbed}%
4349       \bxjs@next\bxjs@tmpdo@b\bxjs@g@tmpa{jaEmbed}%
4350       \bxjs@next\bxjs@tmpdo@c\bxjs@g@tmpb{kanjiVariant}%
4351       \bxjs@next\bxjs@tmpdo@d\bxjs@g@tmpb{jaVariant}%
4352   %
4353   \global\let\bxjs@g@tmpa\relax
4354   \global\let\bxjs@g@tmpb\relax
4355   \endlinechar\m@ne
4356   \let\do\@makeother\dospecials
4357   \catcode32=10 \catcode12=10 %form-feed
4358   \let\bxjs@tmpa\@empty
4359   \openin\@inputcheck="|kpsewhich updmap.cfg"\relax
4360   \ifeof\@inputcheck\else
4361     \read\@inputcheck to\bxjs@tmpa

```

```

4362     \closein\@inputcheck
4363     \fi
4364     \ifx\bxjs@tmpa\@empty\else
4365         \openin\@inputcheck="\bxjs@tmpa"\relax
4366         \@tempswatrue
4367         \loop\if@tempswa
4368             \read\@inputcheck to\bxjs@tmpa
4369             \bxjs@tmpdo
4370             \ifeof\@inputcheck \@tempswafalse \fi
4371         \repeat
4372     \fi
4373 } \endgroup
4374 \let\bxjs@jaEmbed\bxjs@g@tmpa
4375 \let\bxjs@jaVariant\bxjs@g@tmpb
4376 }

```

`\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。

```

4377 \@onlypreamble\bxjs@resolve@jafont@paren
4378 \def\bxjs@resolve@jafont@paren#1{%
4379     \def\bxjs@tmpb{\let#1}%
4380     \expandafter\bxjs@resolve@jafont@paren@a#1\@nil()\@nil\@nnil#1}
4381 \@onlypreamble\bxjs@resolve@jafont@paren@a
4382 \def\bxjs@resolve@jafont@paren@a#1(#2)#3\@nil#4\@nnil#5{%
4383     \ifx\relax#4\relax \bxjs@tmpb#5%
4384     \else
4385         \edef\bxjs@tmpa{#1\bxjs@jafont@paren{#2}#3}%
4386         \bxjs@tmpb\bxjs@tmpa
4387     \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。

```

4388 \newcommand*\jachar[1]{%
4389     \begingroup

```

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

```

4390     \jsLetHeadChar\bxjs@tmpa{#1}%
4391     \ifx\bxjs@tmpa\relax
4392         \ClassWarningNoLine\bxjs@clsname
4393             {Illegal argument given to \string\jachar}%
4394     \else
4395         \expandafter\bxjs@jachar\expandafter{\bxjs@tmpa}%
4396     \fi
4397 \endgroup}

```

`\jsJaChar` を `\jachar` と等価にする。

```

4398 \let\jsJaChar\jachar

```


下請けの \bxjs@jachar の実装はエンジンにより異なる。

```
4399 \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。

```
4400 \PassOptionsToPackage{setpagesize=false}{hyperref}
```

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

```
4401 \@onlypreamble\bxjs@fix@hyperref@unicode
4402 \def\bxjs@fix@hyperref@unicode#1{%
4403   \PassOptionsToPackage{bxjs/hook=#1}{hyperref}%
4404   \@namedef{KV@Hyp@bxjs/hook}##1{%
4405     \KV@Hyp@unicode{##1}%
4406     \def\KV@Hyp@unicode####1{%
4407       \expandafter\ifx\csname if##1\expandafter\endcsname
4408       \csname if####1\endcsname\else
4409       \ClassWarningNoLine{bxjs@clsname
4410       {Blcked hyperref option 'unicode=####1'}}%
4411     \fi
4412   }%
4413 }%
4414 }
```

\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。

```
4415 \@onlypreamble\jsCheckHyperrefUnicode
4416 \let\jsCheckHyperrefUnicode\@empty
4417 \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。

```
4418 \@onlypreamble\bxjs@check@hyperref@unicode
4419 \def\bxjs@check@hyperref@unicode#1{%
4420   \g@addto@macro\jsCheckHyperrefUnicode{%
4421     \@tempwafalse
4422     \begingroup
4423     \expandafter\ifx\csname ifHy@unicode\endcsname\relax
4424     \aftergroup\@tempwattrue \fi
4425     \expandafter\ifx\csname ifHy@unicode\expandafter\endcsname
4426     \csname if#1\endcsname
4427     \aftergroup\@tempwattrue \fi
4428   \endgroup
4429   \if@tempwa\else
4430     \ClassError{bxjs@clsname
4431     {The value of hyperref 'unicode' key is not suitable\MessageBreak
4432     for the present engine (must be #1)}}%
4433     {\@ehc}%
4434   \fi}}
```

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

```
4435 \@onlypreamble\bxjs@urgent@special
4436 \@onlypreamble\bxjs@urgent@special@a
```

L^AT_EX 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。

```
4437 \ifbxjs@old@hook@system
4438 \def\bxjs@urgent@special#1{%
4439   \AtBeginDvi{\special{#1}}%
4440   \g@addto@macro\bxjs@begin@document@hook{%
4441     \ifpackageloaded{atbegshi}{%
4442       \begingroup
4443         \toks\z@{\special{#1}}%
4444         \toks\tw@\expandafter{\AtBegShi@HookFirst}%
4445         \xdef\AtBegShi@HookFirst{\the\toks@the\toks\tw@}%
4446       \endgroup
4447     }{}%
4448   }%
4449 }
```

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。

```
4450 \else
4451   \def\bxjs@urgent@special#1{%
4452     \bxjs@urgent@special@a
4453     \AddToHook{shipout/firstpage}[pxjahyper/enc]{\special{#1}}
4454   \def\bxjs@urgent@special@a{%
4455     \DeclareHookRule{shipout/firstpage}{pxjahyper/enc}{<}{hyperref}%
4456     \global\let\bxjs@urgent@special@a\relax
4457   \fi
```

C.4 pT_EX 用設定

```
4458 \if j\jsEngine
```

■ 共通命令の実装

```
4459 \def\bxjs@apply@kanjiskip{%
4460   \kanjiskip\@tempskipa}
4461 \def\bxjs@apply@xkanjiskip{%
4462   \xkanjiskip\@tempskipa}
```

`\jaJaChar` のサブマクロ。

```
4463 \def\bxjs@jachar#1{%
4464   \bxjs@jachar@a#1...\@nil}
4465 \def\bxjs@jachar@a#1#2#3#4#5\@nil{%
```

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

```
4466   \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。

```

4467 \else\ifx.#3%
4468   \@tempcnta`#1 \multiply\@tempcnta64
4469   \advance\@tempcnta`#2 \advance\@tempcnta-"3080
4470   \bxjs@jachar@b
4471 \else\ifx.#4%
4472   \@tempcnta`#1 \multiply\@tempcnta64
4473   \advance\@tempcnta`#2 \multiply\@tempcnta64
4474   \advance\@tempcnta`#3 \advance\@tempcnta-"E2080
4475   \bxjs@jachar@b
4476 \else
4477   \@tempcnta`#1 \multiply\@tempcnta64
4478   \advance\@tempcnta`#2 \multiply\@tempcnta64
4479   \advance\@tempcnta`#3 \multiply\@tempcnta64
4480   \advance\@tempcnta`#4 \advance\@tempcnta-"3C82080
4481   \bxjs@jachar@b
4482 \fi\fi\fi}

```

符号値が `\@tempcnta` の和文文字を出力する処理。

```

4483 \ifjsWithupTeX
4484 \def\bxjs@jachar@b{\kchar\@tempcnta}
4485 \else
4486 \def\bxjs@jachar@b{%
4487   \ifx\bxUInt\@undefined\else
4488     \bxUInt{\@tempcnta}%
4489   \fi}
4490 \fi

```

和欧文間空白の命令 `\jathinspace` の実装。

```

4491 \ifbxjs@jaspace@cmd
4492 \def\jathinspace{\hskip\xkanjiskip}
4493 \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。

```

4494 \ifbxjs@jp@jismmiv
4495 \PassOptionsToPackage{prefer2004jis}{pxchfon}
4496 \PassOptionsToPackage{prefer2004jis}{pxbabel}
4497 \fi

```

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

```

4498 \let\bxjs@jafont@paren\@firstofone
4499 \let\bxjs@tmpa\jsJaFont
4500 \ifx\bxjs@tmpa\bxjs@@auto

```

```

4501 \let\bxjs@tmpa\@empty
4502 \else\ifx\bxjs@tmpa\bxjs@noEmbed
4503 \def\bxjs@tmpa{noembed}
4504 \fi\fi
4505 \bxjs@resolve@jafont@paren\bxjs@tmpa
4506 \edef\bxjs@tmpa{\bxjs@catopt\bxjs@tmpa\bxjs@jp@font}
4507 \ifx\bxjs@tmpa\@empty\else
4508 \edef\bxjs@next{%
4509 \noexpand\RequirePackage[\bxjs@tmpa]{pxchfon}[2010/05/12]% v0.5
4510 }\bxjs@next
4511 \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……。)

```

4512 \begingroup
4513 \global\let\bxjs@g@tmpa\relax
4514 \catcode`\|=0 \catcode`\|=12
4515 |def|bxjs@tmpdo#a#1|@nil{%
4516 |bxjs@tmpdo@a#1|@nil\RequirePackage|@nnil}%
4517 |def|bxjs@tmpdo@a#1\RequirePackage#2|@nnil{%
4518 |ifx$#1$|bxjs@tmpdo@b#2|@nil keyval|@nnil |fi}%
4519 |catcode`\|=0 \catcode`\|=12
4520 \def\bxjs@tmpdo@b#1keyval#2\@nnil{%
4521 \ifx$#2$\else
4522 \xdef\bxjs@g@tmpa{%
4523 \noexpand\PassOptionsToPackage{scale=\jsScale}{otf}}%
4524 \fi}
4525 \@firstofone{%
4526 \catcode10=12 \endlinechar\m@ne
4527 \let\do\@makeother \dospecials \catcode32=10
4528 \openin\@inputcheck=otf.sty\relax
4529 \@tempswatrue
4530 \loop\if@tempswa
4531 \ifeof\@inputcheck \@tempswafalse \fi
4532 \if@tempswa
4533 \read\@inputcheck to\bxjs@next
4534 \expandafter\bxjs@tmpdo\bxjs@next\@nil
4535 \fi
4536 \repeat
4537 \closein\@inputcheck
4538 \endgroup}
4539 \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。

```
4540 \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。

```
4541 \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。

```
4542 \ifpackagelater{plautopatch}{2020/05/25}{% v0.9g
```

```
4543 \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。

```
4544 \let\bxjs@avail@hy@unicode=f
```

```
4545 \if \ifx t\bxjs@plautopatch@new T%
```

```
4546 \else\ifbxjs@old@hook@system F\else T\fi\fi T%
```

```
4547 \IfFileExists{pxjahyper-uni.def}{\let\bxjs@avail@hy@unicode=t}{}
```

```
4548 \fi
```

```
4549 \if t\bxjs@avail@hy@unicode
```

```
4550 \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。

```
4551 \ifbxjs@old@hook@system\else
```

```
4552 \AddToHook{\bxjs@CGHN{package/hyperref/after}}{%
```

```
4553 \input{pxjahyper-uni.def}}
```

```
4554 \fi
```

```
4555 \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 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。

```

4556      \let\bxjs@avail@hy@unicode=f
4557      \ifx t\bxjs@plautopatch@new
4558        \@ifl@t@r\fmtversion{2023/06/01}{\let\bxjs@avail@hy@unicode=t}{ }
4559      \fi
4560    \fi
4561  \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。

```

4562  \if f\bxjs@avail@hy@unicode
4563    \PassOptionsToPackage{unicode=false}{hyperref}
4564  \fi
4565 \fi

```

tunicode special 命令を出力する。

```

4566 \if \ifx\bxjs@driver@given\bxjs@driver@@dvipdfmx T%
4567   \else\ifjsWithpTeXng T\else F\fi\fi T%
4568   \IfFileExists{pxjahyper-enc.sty}{\@tempwattrue}{\@tempwafalse}
4569   \if@tempswa
4570     \RequirePackage{pxjahyper-enc}[2020/10/05]%v0.6
4571     \ifbxjs@bigcode\else \suppressbigcode \fi
4572   \else
4573     \ifnum\jis"2121="A1A1 %euc
4574       \bxjs@urgent@special{pdf:tunicode EUC-UCS2}
4575     \else\ifnum\jis"2121="8140 %sjis
4576       \bxjs@urgent@special{pdf:tunicode 90ms-RKSJ-UCS2}
4577     \else\ifnum\jis"2121="3000 %uptex
4578       \ifbxjs@bigcode
4579         \bxjs@urgent@special{pdf:tunicode UTF8-UTF16}
4580         \PassOptionsToPackage{bigcode}{pxjahyper}
4581       \else
4582         \bxjs@urgent@special{pdf:tunicode UTF8-UCS2}
4583         \PassOptionsToPackage{nobigcode}{pxjahyper}
4584       \fi
4585     \fi\fi\fi
4586     \let\bxToUnicodeSpecialDone=t
4587   \fi
4588 \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。

```

4589 \ifx f\bxjs@enablejfam\else
4590   \@enablejfamtrue
4591 \fi

```

実際に和文用の数式ファミリの設定を行う。

```

4592 \if@enablejfam
4593   \DeclareSymbolFont{mincho}{\jsc@JYn}{mc}{m}{n}
4594   \DeclareSymbolFontAlphabet{\mathmc}{mincho}

```

```

4595 \SetSymbolFont{mincho}{bold}{\jsc@JYn}{gt}{m}{n}
4596 \jfam\symmincho
4597 \DeclareMathAlphabet{\mathgt}{\jsc@JYn}{gt}{m}{n}
4598 \g@addto@macro\bxjs@begin@document@hook{%
4599   \ifx\reDeclareMathAlphabet\@undefined\else
4600     \reDeclareMathAlphabet{\mathrm}{\@mathrm}{\@mathmc}%
4601     \reDeclareMathAlphabet{\mathbf}{\@mathbf}{\@mathgt}%
4602     \reDeclareMathAlphabet{\mathsf}{\@mathsf}{\@mathgt}%
4603   \fi}
4604 \fi

```

C.5 pdfTeX 用設定：CJK + bxcjkatype

```

4605 \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。

```

4606 \bxjs@adjust@jafont{f}
4607 \let\bxjs@jafont@paren\@firstofone
4608 \bxjs@resolve@jafont@paren\bxjs@tmpa
4609 \edef\bxjs@tmpa{\bxjs@catopt\bxjs@tmpa\bxjs@jp@font}
4610 \edef\bxjs@tmpa{\bxjs@catopt\bxjs@tmpa{whole}}
4611 \ifx\bxjs@jadriver\bxjs@@pandoc\else
4612   \edef\bxjs@tmpa{\bxjs@catopt\bxjs@tmpa{autotilde}}
4613 \fi
4614 \edef\bxjs@next{%
4615   \noexpand\RequirePackage[\bxjs@tmpa]{bxcjkatype}[2013/10/15]% v0.2c
4616 }\bxjs@next
4617 \bxjs@cjk@loaded

```

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

```

4618 \ifbxjs@hyperref@enc
4619   \PassOptionsToPackage{unicode}{hyperref}
4620 \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。

```

4621 \ifx\bxcjkatypeHyperrefPatchDone\@undefined
4622 \begingroup
4623   \CJK@input{UTF8.bdg}
4624 \endgroup

```

```

4625 \g@addto@macro\pdfstringdefPreHook{%
4626   \@nameuse{CJK@UTF8Binding}%
4627 }
4628 \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。

```

4629 \ifx\bxckjatypeHyperrefPatchDone\@undefined
4630 \g@addto@macro\pdfstringdefPreHook{%
4631   \ifx~\bxjs@CJKtilde
4632     \let\bxjs@org@LetUnexpandableSpace\HyPsd@LetUnexpandableSpace
4633     \let\HyPsd@LetUnexpandableSpace\bxjs@LetUnexpandableSpace
4634     \let~\@empty
4635   \fi
4636 }
4637 \def\bxjs@CJKtilde{\CJKecglue\ignorespaces}
4638 \def\bxjs@tildecmd{~}
4639 \def\bxjs@LetUnexpandableSpace#1{%
4640   \def\bxjs@tmpa{#1}\ifx\bxjs@tmpa\bxjs@tildecmd\else
4641     \bxjs@org@LetUnexpandableSpace#1%
4642   \fi}
4643 \fi

```

■ 共通命令の実装

```

4644 \newskip\jsKanjiSkip
4645 \newskip\jsXKanjiSkip
4646 \ifx\CJKecglue\@undefined
4647   \def\CJKtilde{\CJK@global\def~{\CJKecglue\ignorespaces}}
4648 \fi
4649 \let\autospacing\bxjs@enable@kanjiskip
4650 \let\noautospacing\bxjs@disable@kanjiskip
4651 \protected\def\bxjs@CJKglue{\hskip\jsKanjiSkip}
4652 \def\bxjs@apply@kanjiskip{%
4653   \jsKanjiSkip\@tempkipa
4654   \let\CJKglue\bxjs@CJKglue}
4655 \let\autoxspacing\bxjs@enable@xkanjiskip
4656 \let\noautoxspacing\bxjs@disable@xkanjiskip
4657 \protected\def\bxjs@CJKecglue{\hskip\jsXKanjiSkip}
4658 \def\bxjs@apply@xkanjiskip{%
4659   \jsXKanjiSkip\@tempkipa
4660   \let\CJKecglue\bxjs@CJKecglue}

```

\jachar のサブマクロの実装。

```

4661 \def\bxjs@jachar#1{%
4662   \CJKforced{#1}}

```

和欧文間空白の命令 \jathinspace の実装。

```

4663 \ifbxjs@jaspace@cmd
4664   \protected\def\jathinspace{\CJKecglue}
4665 \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。

```
4666 \ifx t\bxjs@enablejfam
4667   \ClassWarningNoLine\bxjs@clsname
4668     {You cannot use 'enablejfam=true', since the\MessageBreak
4669       CJK package does not support Japanese math}
4670 \fi
```

C.6 X_YT_EX 用設定：xeCJK + zxjatype

```
4671 \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。

```
4672 \RequirePackage{zxjatype}
4673 \PassOptionsToPackage{no-math}{fontspec}%!
4674 \PassOptionsToPackage{xetex}{graphicx}%!
4675 \PassOptionsToPackage{xetex}{graphics}%!
4676 \ifx\zxJaFamilyName\@undefined
4677   \ClassError\bxjs@clsname
4678     {xeCJK or zxjatype is too old}\@ehc
4679 \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。

```
4680 \bxjs@adjust@jafont{f}
4681 \let\bxjs@jafont@paren@gobble
4682 \bxjs@resolve@jafont@paren\bxjs@tmpa
4683 \edef\bxjs@tmpa{\bxjs@catopt\bxjs@tmpa\bxjs@jp@font}
4684 \ifx\bxjs@tmpa\@empty
4685   \setCJKmainfont[BoldFont=HaranoAjiGothic-Medium.otf]{HaranoAjiMincho-
     Regular.otf}
4686   \setCJKsansfont[BoldFont=HaranoAjiGothic-Medium.otf]{HaranoAjiGothic-
     Medium.otf}
4687 \else
4688   \edef\bxjs@next{%
4689     \noexpand\RequirePackage[\bxjs@tmpa]{zxjafont}[2013/01/28]% v0.2a
4690   }\bxjs@next
4691 \fi
```

■hyperref 対策 unicode オプションの指定に関する話。

X_YT_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_YT_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

と見なしていて警告が発生する。

これを踏まえて、ここでは、「Xe_qTeX のバージョンが 0.99992 以上の場合に unicode を既定で有効にする」ことにする。

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

TODO: バージョン要件を見直して暫定措置を解除する。

```
4692 \ifx\strcmp@\undefined\else % 未定義なら条件を満たさない
4693 \ifnum\strcmp{\the\XeTeXversion\XeTeXrevision}{0.99992}>\m@ne
4694   \ifbxjs@hyperref@enc
4695     \PassOptionsToPackage{unicode}{hyperref}
4696   \fi
4697 \fi
4698 \fi
```

■段落頭でのグルー挿入禁止 どうやら、`zxjatype` の `\inhibitglue` の実装が極めて杜撰なため、1.0 版での実装では全く期待通りの動作をしていないし、そもそも (少なくとも現状の) `xeCJK` では、段落頭での `\inhibitglue` は実行しないほうが JS クラスの出力に近いものが得られるらしい。

従って、`\jsInhibitGlueAtParTop` は結局何もしないことにする。

強制改行直後のグルー禁止処理、のような怪しげな何か。

```
4699 \AtEndOfPackage{%
4700 \def\@newline #1{%
4701   \ifvmode \@nolnerr
4702   \else
4703     \unskip \reserved@e {\reserved@f#1}\nobreak \hfil \break \null
4704     \nobreak \hskip-1sp\hskip1sp\relax
4705     \ignorespaces
4706   \fi}
4707 }
```

■共通命令の実装

```
4708 \newskip\jsKanjiSkip
4709 \newskip\jsXKanjiSkip
4710 \ifx\CJKecglue@\undefined
4711   \def\CJKtilde{\CJK@global\def~{\CJKecglue\ignorespaces}}
4712 \fi
4713 \let\autospacing\bxjs@enable@kanjiskip
4714 \let\noautospacing\bxjs@disable@kanjiskip
4715 \protected\def\bxjs@CJKglue{\hskip\jsKanjiSkip}
4716 \def\bxjs@apply@kanjiskip{%
4717   \jsKanjiSkip\@tempskipa
4718   \xeCJKsetup{CJKglue={\bxjs@CJKglue}}}
4719 \let\autoxspacing\bxjs@enable@xkanjiskip
4720 \let\noautoxspacing\bxjs@disable@xkanjiskip
4721 \protected\def\bxjs@CJKecglue{\hskip\jsXKanjiSkip}
```

```

4722 \def\bxjs@apply@xkanjiskip{%
4723   \jsXKanjiSkip\@tempskipa
4724   \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。

```

4725 \ifx\mcfamily\undefined
4726   \protected\def\mcfamily{\CJKfamily{\CJKrmdefault}}
4727   \protected\def\gtfamily{\CJKfamily{\CJKsfdefault}}
4728 \fi

```

`\jachar` のサブマクロの実装。

```

4729 \def\bxjs@jachar#1{%
4730   \xeCJKDeclareCharClass{CJK}{`#1}\relax
4731   #1}

```

`\jathinspace` の実装。

```

4732 \ifbxjs@jaspace@cmd
4733   \protected\def\jathinspace{\CJKecglue}
4734 \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。

```

4735 \ifx t\bxjs@enablejfam
4736   \@enablejfamtrue
4737 \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。

```

4738 \if@enablejfam
4739   \xeCJKsetup{CJKmath=true}
4740 \fi

```

C.7 LuaTeX 用設定：LuaTeX-ja

```

4741 \else\if l\jsEngine

```

■LuaTeX-ja パッケージの読込 `luatexja` とともに `luatexja-fontspec` パッケージを読み込む。

`luatexja` は自前の `\zw` (これは実際の現在和文フォントに基づく値を返す) を定義するので、`\zw` の定義を消しておく。なお、レイアウト定義の「全角幅」は「規定」に基づく `\jsZw` であることに注意が必要。

※ 1.0b 版から「graphics パッケージに `pdftex` オプションを渡す」処理を行っていたが、1.4 版で廃止された。

```

4742 \let\zw\undefined
4743 \RequirePackage{luatexja}
4744 \edef\bxjs@next{%
4745   \noexpand\RequirePackage[scale=\jsScale]{luatexja-fontspec}[2015/08/26]%

```

4746 } \bxjs@next

フォント代替の明示的定義。

```
4747 \DeclareFontShape{JY3}{mc}{m}{it}{<->ssub*mc/m/n}{}
4748 \DeclareFontShape{JY3}{mc}{m}{sl}{<->ssub*mc/m/n}{}
4749 \DeclareFontShape{JY3}{mc}{m}{sc}{<->ssub*mc/m/n}{}
4750 \DeclareFontShape{JY3}{gt}{m}{it}{<->ssub*gt/m/n}{}
4751 \DeclareFontShape{JY3}{gt}{m}{sl}{<->ssub*gt/m/n}{}
4752 \DeclareFontShape{JY3}{mc}{bx}{it}{<->ssub*gt/m/n}{}
4753 \DeclareFontShape{JY3}{mc}{bx}{sl}{<->ssub*gt/m/n}{}
4754 \DeclareFontShape{JY3}{gt}{bx}{it}{<->ssub*gt/m/n}{}
4755 \DeclareFontShape{JY3}{gt}{bx}{sl}{<->ssub*gt/m/n}{}
4756 \DeclareFontShape{JY3}{mc}{b}{n}{<->ssub*mc/bx/n}{}
4757 \DeclareFontShape{JY3}{mc}{b}{it}{<->ssub*mc/bx/n}{}
4758 \DeclareFontShape{JY3}{mc}{b}{sl}{<->ssub*mc/bx/n}{}
4759 \DeclareFontShape{JY3}{gt}{b}{n}{<->ssub*gt/bx/n}{}
4760 \DeclareFontShape{JY3}{gt}{b}{it}{<->ssub*gt/bx/n}{}
4761 \DeclareFontShape{JY3}{gt}{b}{sl}{<->ssub*gt/bx/n}{}
4762 \DeclareFontShape{JT3}{mc}{m}{it}{<->ssub*mc/m/n}{}
4763 \DeclareFontShape{JT3}{mc}{m}{sl}{<->ssub*mc/m/n}{}
4764 \DeclareFontShape{JT3}{mc}{m}{sc}{<->ssub*mc/m/n}{}
4765 \DeclareFontShape{JT3}{gt}{m}{it}{<->ssub*gt/m/n}{}
4766 \DeclareFontShape{JT3}{gt}{m}{sl}{<->ssub*gt/m/n}{}
4767 \DeclareFontShape{JT3}{mc}{bx}{it}{<->ssub*gt/m/n}{}
4768 \DeclareFontShape{JT3}{mc}{bx}{sl}{<->ssub*gt/m/n}{}
4769 \DeclareFontShape{JT3}{gt}{bx}{it}{<->ssub*gt/m/n}{}
4770 \DeclareFontShape{JT3}{gt}{bx}{sl}{<->ssub*gt/m/n}{}
4771 \DeclareFontShape{JT3}{mc}{b}{n}{<->ssub*mc/bx/n}{}
4772 \DeclareFontShape{JT3}{mc}{b}{it}{<->ssub*mc/bx/n}{}
4773 \DeclareFontShape{JT3}{mc}{b}{sl}{<->ssub*mc/bx/n}{}
4774 \DeclareFontShape{JT3}{gt}{b}{n}{<->ssub*gt/bx/n}{}
4775 \DeclareFontShape{JT3}{gt}{b}{it}{<->ssub*gt/bx/n}{}
4776 \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。

```
4777 \bxjs@adjust@jafont{t}
4778 \ifx\bxjs@tmpa\bxjs@noEmbed
4779   \def\bxjs@tmpa{noembed}
4780 \fi
4781 \let\bxjs@jafont@paren\gobble
4782 \bxjs@resolve@jafont@paren\bxjs@tmpa
4783 \edef\bxjs@tmpa{\bxjs@catopt\bxjs@tmpa\bxjs@jp@font}
4784 \ifx\bxjs@tmpa@empty
4785   \defaultjfontfeatures{ Kerning=Off }
4786   \setmainjfont[BoldFont=HaranoAjiGothic-Medium.otf,JFM=ujis]{HaranoAjiMincho-

```

```

Regular.otf}
4787 \setsansfont[BoldFont=HaranoAjiGothic-Medium.otf,JFM=ujis]{HaranoAjiGothic-
Medium.otf}
4788 \else
4789 \edef\bxjs@next{%
4790 \noexpand\RequirePackage[\bxjs@tmpa]{luatexja-preset}%
4791 }\bxjs@next
4792 \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。

```

4793 \ifpackagelater{luatexja}{2016/03/31}{\else
4794 \DeclareRobustCommand\rmfamily
4795 {\not@math@alphabet\rmfamily\mathrm
4796 \romanfamily\rmdefault\kanjifamily\mcdefault\selectfont}
4797 \DeclareRobustCommand\sffamily
4798 {\not@math@alphabet\sffamily\mathsf
4799 \romanfamily\sfdefault\kanjifamily\gtdefault\selectfont}
4800 \DeclareRobustCommand\ttfamily
4801 {\not@math@alphabet\ttfamily\mathtt
4802 \romanfamily\ttdefault\kanjifamily\gtdefault\selectfont}
4803 }
4804 \long\def\jttdefault{\gtdefault}
4805 \unless\ifx\@ltj@match@familytrue\@undefined
4806 \@ltj@match@familytrue
4807 \fi
4808 \g@addto@macro\bxjs@begin@document@hook{%
4809 \reDeclareMathAlphabet{\mathrm}{\mathrm}{\mathmc}%
4810 \reDeclareMathAlphabet{\mathbf}{\mathbf}{\mathgt}%
4811 \reDeclareMathAlphabet{\mathsf}{\mathsf}{\mathgt}}%
4812 \bxjs@if@sf@default{%
4813 \renewcommand\kanjifamilydefault{\gtdefault}}

```

■和文パラメタの設定

```

4814 % 次の3つは既定値の通り
4815 %\ltjsetparameter{prebreakpenalty={` ,10000}}
4816 %\ltjsetparameter{postbreakpenalty={` ",10000}}
4817 %\ltjsetparameter{prebreakpenalty={` ”,10000}}
4818 \ltjsetparameter{jaxspmode={` !,1}}
4819 \ltjsetparameter{jaxspmode={` 〒,2}}
4820 \ltjsetparameter{alxspmode={` +,3}}
4821 \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。

```

4822 \ifx\ltjfakeparbegin\@undefined
4823 \protected\def\ltjfakeparbegin{%
4824   \ifhmode
4825     \relax\directlua{%
4826       luatexja.jfmglue.create_beginpar_node()}}
4827 \fi}
4828 \fi

    ltjs* クラスの定義と同等になるようにパッチを当てる。
4829 \unless\ifnum\bxjs@everyparhook=\bxjs@everyparhook@@none
4830 \begingroup
4831 \let\%\@percentchar \def\@#1{[[\detokenize{#1}]]}
4832 \@gobble\if\def\bxjs@tmpa{\@{\everypar{}}\fi}
4833 \ifnum\bxjs@everyparhook=\bxjs@everyparhook@@compat
4834 \@gobble\if\def\bxjs@tmpa{\@{\everypar{\everyparhook}}\fi}}\fi
4835 \directlua{
4836   local function patchcmd(cs, code, from, to)
4837     tex.sprint(code:gsub(from:gsub("\%W", "\%\%\%0"), "\%0"..to)
4838       :gsub("macro:", \@gdef..cs, 1):gsub("->", "{", 1)..")
4839   end
4840   patchcmd(\@@xsect, [[\meaning\@xsect]],
4841     \@{\hskip-\@tempskipa}, \@{\ltjfakeparbegin})
4842   patchcmd(\@@item, [[\meaning\@item]],
4843     \bxjs@tmpa, \@{\ltjfakeparbegin})}
4844 \endgroup
4845 \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。

```

4846 \ifbxjs@hyperref@enc
4847 \PassOptionsToPackage{unicode}{hyperref}
4848 \bxjs@check@hyperref@unicode{true}
4849 \fi

```

■共通命令の実装

```

4850 \protected\def\autospacing{%
4851 \ltjsetparameter{autospacing=true}}
4852 \protected\def\noautospacing{%
4853 \ltjsetparameter{autospacing=false}}
4854 \protected\def\autoxspacing{%
4855 \ltjsetparameter{autoxspacing=true}}
4856 \protected\def\noautoxspacing{%
4857 \ltjsetparameter{autoxspacing=false}}
4858 \def\bxjs@apply@kanjiskip{%
4859 \ltjsetparameter{kanjiskip={\@tempskipa}}}
4860 \def\bxjs@apply@xkanjiskip{%
4861 \ltjsetparameter{xkanjiskip={\@tempskipa}}}

```

\jachar のサブマクロの実装。

```

4862 \def\bxjs@jachar#1{%
4863   \ltjjachar`#1\relax}

\jathinspace の実装。
4864 \ifbxjs@jaspace@cmd
4865   \protected\def\jathinspace{%
4866     \hskip\ltjgetparameter{xkanjiskip}\relax}
4867 \fi

```

■和文数式ファミリー Lua_T_EX-j_a では和文数式ファミリーは常に有効で、既にこの時点で必要な設定は済んでいる。従って @enablej_fam は常に真になる。

```

4868 \ifx f\bxjs@enablejfam
4869   \ClassWarningNoLine\bxjs@clsname
4870   {You cannot use 'enablejfam=false', since the\MessageBreak
4871     LuaTEX-ja always provides Japanese math families}
4872 \fi

```

C.8 共通処理 (2)

```

4873 \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 4874 \ifx\DeclareFixJFMCJKTextFontCommand\@undefined
4875   \DeclareJaTextFontCommand{\textmc}{\mcfamily}
4876   \DeclareJaTextFontCommand{\textgt}{\gtfamily}
4877 \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。

```

4878 \ifx\mathmc\@undefined
4879   \DeclareJaMathFontCommand{\mathmc}{\mcfamily}
4880   \DeclareJaMathFontCommand{\mathgt}{\gtfamily}
4881 \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)。

```

4882 \ifbxjs@jaspace@cmd
4883   \bxjs@protected\def\bxjs@choice@jathinspace{%
4884     \relax\ifmmode \mskip\medmuskip
4885     \else \jathinspace\ignorespaces
4886     \fi}
4887   \jsAtEndOfClass{%
4888     \ifjsWitheTeX \let\>\bxjs@choice@jathinspace
4889     \else \def\>{\protect\bxjs@choice@jathinspace}%

```

```
4890 \fi}
4891 \fi
```

■和文・和欧文間空白の初期値

```
4892 \setkanjiskip{0pt plus.1\jsZw minus.01\jsZw}
4893 \ifx\jsDocClass\jsSlide \setxkanjiskip{0.1em}
4894 \else \setxkanjiskip{0.25em plus 0.15em minus 0.06em}
4895 \fi
```

以上で終わり。

```
4896 %</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。

```
4897 %<*modern>
4898 \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。

```
4899 \ifnum0\if x\jsEngine1\fi\if 1\jsEngine1\fi=\z@
4900 \def\encodingdefault{T1}%
4901 \input{t1enc.def}%
4902 \fontencoding\encodingdefault\selectfont
4903 \fi
```

基本フォントを Latin Modern フォントファミリーに変更する。

※以下は `\usepackage[noamth]{lmodern}` と同じ。ユーザは後で `lmodern` を好きなオプションを付けて読み込むことができる。

```
4904 \ifnum0\if x\jsEngine1\fi\if 1\jsEngine1\fi=\z@
4905 \renewcommand{\rmdefault}{lrm}
4906 \renewcommand{\sfdefault}{lss}
4907 \renewcommand{\ttdefault}{lmtt}
4908 \fi
```

大型演算子用の数式フォントの設定。

※ `amsfonts` パッケージと同等にする。

```
4909 \DeclareFontShape{OMX}{cmex}{m}{n}{%
4910 <-7.5>cmex7<7.5-8.5>cmex8%
4911 <8.5-9.5>cmex9<9.5->cmex10}{}%
4912 \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。

```
4913 \def\cmex@opt{10}
```


D.2 fixltx2e 読込

※ fixltx2e 廃止前の L^AT_EX カーネルの場合。

```
4914 \ifx\@IncludeInRelease\@undefined
4915 \RequirePackage{fixltx2e}
4916 \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。

```
4917 \RequirePackage{bxjscjkat}
```

D.4 完了

おしまい。

```
4918 %</modern>
```

付録 E 和文ドライバ：pandoc 🐼

「Pandoc モード」で使用される和文ドライバ。standard ドライバの機能を継承するが、「Pandoc の既定の latex テンプレート」が使われることを前提として、それと BXJS の設定を整合させるための措置を加えている。

E.1 準備

standard ドライバの設定を引き継ぐ。

```
4919 %<*pandoc>
```

```
4920 \input{bxjsja-standard.def}
```

bxjspandoc パッケージを読み込む。

```
4921 \RequirePackage{bxjspandoc}
```

ε- $\text{T}_{\text{E}}\text{X}$ ではない場合に警告を出す。

※近い将来に ε- $\text{T}_{\text{E}}\text{X}$ 拡張を必須にする予定。

```
4922 \ifjsWitheTeX\else
```

```
4923 \ClassWarningNoLine{bxjs}{clsname}
```

```
4924 {!!!!!!! WARNING !!!!!!!\MessageBreak
```

```
4925 This engine does not support e-TeX extension!\MessageBreak
```

```
4926 Some feature might not work properly}
```

```
4927 \fi
```

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

```
4928 \newif\ifbxjs@bxghost@available
```

```
4929 \ifjsWitheTeX
```

```

4930 \RequirePackage{pdfcmds}[2009/09/22]% v0.5
4931 \IfFileExists{bxghost.sty}{%
4932   \bxjs@bxghost@availabletrue
4933   \@namedef{bxjs@bgbv/79E70A0991967E27981832C84DB5DF99}{1}%v0.2.0
4934   \ifx\pdf@filemdfivesum\undefined\else
4935     \expandafter\ifx\csname bxjs@bgbv/\pdf@filemdfivesum{bxghost.sty}%
4936       \endcsname\relax\else \bxjs@bxghost@availablefalse \fi
4937   \fi
4938 }{}
4939 \fi

```

`\bxjs@endpreamble@hook` etoolbox の `\AtEndPreamble` で実行される BXJS クラス用のフック。

※ ϵ -TeX 以外では無効になる。(将来 pandoc の外に出す可能性あり。)

```

4940 \@onlypreamble\bxjs@endpreamble@hook
4941 \let\bxjs@endpreamble@hook\empty

```

パッケージ読込。

```

4942 \RequirePackage{iftex}[2013/04/04]% v0.2
4943 \ifjsWitheTeX
4944   \RequirePackage{etoolbox}[2010/08/21]% v2.0
4945   \AtEndPreamble{\bxjs@endpreamble@hook}
4946   \RequirePackage{filehook}[2011/10/12]% v0.5d
4947 \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。

```

4948 \newif\ifbxjs@jp@fix@strong \bxjs@jp@fix@strongtrue

```

`fix-strong` オプションの処理。

```

4949 \let\bxjs@kv@fixstrong@true\bxjs@jp@fix@strongtrue
4950 \let\bxjs@kv@fixstrong@false\bxjs@jp@fix@strongfalse
4951 \define@key{bxjsPan}{fix-strong}[true]{%
4952   \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。

```

4953 \newif\ifbxjs@jp@fix@code \bxjs@jp@fix@codetrue

```

`fix-code` オプションの処理。

```

4954 \let\bxjs@kv@fixcode@true\bxjs@jp@fix@codetrue
4955 \let\bxjs@kv@fixcode@false\bxjs@jp@fix@codefalse
4956 \define@key{bxjsPan}{fix-code}[true]{%
4957   \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。

```

4958 \chardef\bxjs@jp@strong=0

```

`strong` オプションの処理。

```
4959 \def\bxjs@kv@strong@bold{\chardef\bxjs@jp@strong=0 }
4960 \def\bxjs@kv@strong@sans{\chardef\bxjs@jp@strong=1 }
4961 \def\bxjs@kv@strong@boldsans{\chardef\bxjs@jp@strong=2 }
4962 \define@key{bxjsPan}{strong}{%
4963   \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 4964 \newif\ifbxjs@jp@or@indent \bxjs@jp@or@indenttrue
\ifbxjs@jp@or@block@heading 4965 \newif\ifbxjs@jp@or@secnumdepth \bxjs@jp@or@secnumdepthtrue
4966 \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。

```
4967 \define@key{bxjsPan}{_plus}[]{%
4968   \bxjs@jp@or@indentfalse
4969   \bxjs@jp@or@secnumdepthfalse
4970   \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。

```
4971 \let\bxjs@kv@orindent@true\bxjs@jp@or@indenttrue
4972 \let\bxjs@kv@orindent@false\bxjs@jp@or@indentfalse
4973 \define@key{bxjsPan}{or-indent}[true]{%
4974   \bxjs@set@keyval{orindent}{#1}{}}
4975 \let\bxjs@kv@orsecnumdepth@true\bxjs@jp@or@secnumdepthtrue
4976 \let\bxjs@kv@orsecnumdepth@false\bxjs@jp@or@secnumdepthfalse
4977 \define@key{bxjsPan}{or-secnumdepth}[true]{%
4978   \bxjs@set@keyval{orsecnumdepth}{#1}{}}
4979 \let\bxjs@kv@orblockheading@true\bxjs@jp@or@block@headingtrue
4980 \let\bxjs@kv@orblockheading@false\bxjs@jp@or@block@headingfalse
4981 \define@key{bxjsPan}{or-block-heading}[true]{%
4982   \bxjs@set@keyval{orblockheading}{#1}{}}
```

実際の `japaram` の値を適用する。

```
4983 \def\bxjs@next#1{\bxjs@safe@setkeys{bxjsPan}{#1}}
4984 \expandafter\bxjs@next\expandafter{\jsJaParam}
```

E.3 dupload システム

パッケージが重複して読み込まれたときに “option clash” の検査をスキップする。この時に何らかのコードを実行させることができる。

`\bxjs@set@dupload@proc` `\bxjs@set@dupload@proc{〈ファイル名〉}{〈定義本体〉}`： 指定の名前の特定のファイルの読み込みが `\@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。

```

4985 \@onlypreamble\bxjs@set@dupload@proc
4986 \def\bxjs@set@dupload@proc#1{%
4987   \expandafter\bxjs@set@dupload@proc@a\csname bxjs@dlp/#1\endcsname}
4988 \@onlypreamble\bxjs@set@dupload@proc@a
4989 \def\bxjs@set@dupload@proc@a#1{%
4990   \@onlypreamble#1\def#1##1}
4991 \def\bxjs@unset@dupload@proc#1{%
4992   \bxjs@cslet{bxjs@dlp/#1}\@undefined}

```

\@if@options \@if@options の再定義。

```

4993 \@onlypreamble\bxjs@org@if@options
4994 \let\bxjs@org@if@options\@if@options
4995 \@onlypreamble\bxjs@org@reset@options
4996 \let\bxjs@org@reset@options\relax
4997 \def\@if@options#1#2#3{%
4998   \let\bxjs@next\@secondoftwo
4999   \def\bxjs@tmpa{#1}\def\bxjs@tmpb{\@currentx}%
5000   \ifx\bxjs@tmpa\bxjs@tmpb
5001     \expandafter\ifx\csname bxjs@dlp/#2.#1\endcsname\relax\else
5002       \let\bxjs@next\@firstoftwo \fi
5003   \fi
5004   \bxjs@next\bxjs@do@dupload@proc\bxjs@org@if@options{#1}{#2}{#3}}
5005 \g@addto@macro\bxjs@begin@document@hook{%
5006   \let\@if@options\bxjs@org@if@options}
5007 \@onlypreamble\bxjs@do@dupload@proc
5008 \def\bxjs@do@dupload@proc#1#2#3{%
5009   \ifx\bxjs@org@reset@options\relax
5010     \let\bxjs@org@reset@options\@reset@options
5011   \fi
5012   \bxjs@csletcs{bxjs@next}{bxjs@dlp/#2.#1}%
5013   \def\@reset@options{%
5014     \let\@reset@options\bxjs@org@reset@options
5015     \@reset@options
5016     \bxjs@next{#3}}%
5017   \@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、これの読込を避ける必要がある。そのため、メイン言語が `japanese` である場合（古い Pandoc ではこの場合に引数が空の `\setmainlanguage{}` が実行されるがこのパターンも同様に扱う）には、Polyglossia の処理を無効化してしまうことにする。つまり、Polyglossia が提供する命令について、何もしないダミーの定義を与える。※ Polyglossia は古い Pandoc のテンプレートにおいて、エンジンが \TeX か \LaTeX 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。

```
5018 \let\bxjs@polyglossia@options\relax
```

エンジンが \TeX か \LaTeX 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。

```
5019 \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。

```
5020 \pandocSkipLoadPackage{polyglossia}
5021 \bxjs@set@dupload@proc{polyglossia.sty}{%
5022   \bxjs@unset@dupload@proc{polyglossia.sty}%
5023   \ClassWarning\bxjs@clsname
5024     {Package polyglossia is requested}%
5025   \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。

```
5026 \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。

```
5027   \ifboolexpr{test{\ifblank{##2}}or test{\ifstrequal{##2}{japanese}}}{%
5028     \ClassWarning\bxjs@clsname
5029       {Main language is 'japanese', thus fallback\MessageBreak
5030         definitions will be employed}%
5031     \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。

```
5032   }{%else
5033     \ClassWarning\bxjs@clsname
5034       {Main language is '##2',\MessageBreak
5035         thus polyglossia will be loaded}%
5036     \csundef{ver@polyglossia.sty}%
5037     \edef\bxjs@next{%
5038       \noexpand\RequirePackage[\bxjs@polyglossia@options]{polyglossia}[]%
5039     }\bxjs@next
5040     \setmainlanguage[##1]{##2}%
```

```
5041   }}
```

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。

```
5042 \g@addto@macro\bxjs@endpreamble@hook{%
5043   \ifx\bxjs@polyglossia@options\relax
5044     \csundef{ver@polyglossia.sty}%
5045   \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。

```
5046 \@onlypreamble\bxjs@pandoc@polyglossia@ja
5047 \def\bxjs@pandoc@polyglossia@ja{%
5048   \renewcommand*\setmainlanguage[2] []{%
5049     \newcommand*\setotherlanguage[2] []{%
5050       \ifblank{##2}{\}%else
5051         \cslet{##2}\@empty \cslet{end##2}\@empty
5052         \cslet{text##2}\@firstofone}}%
5053   \newcommand*\setotherlanguages[2] []{%
5054     \@for\bxjs@tmpa:={##2}\do{%
5055       \setotherlangauge{\bxjs@tmpa}}}%
```

Polyglossia の読込済マークは外れるようにしておく。

```
5056   \let\bxjs@polyglossia@options\relax}%
5057 \fi
```

■Babel について 現在の Pandoc では、テンプレートで用いられる多言語パッケージとしてエンジンの種別によらずに Babel が使われる。

※ Xe_{La}TeX では 2.15 版で、Lua_{La}TeX は 2.6 版で Polyglossia から Babel に変更されている。

\bxjs@babel@options Babel のオプション列のテキスト。“実際には読み込まれていない” 場合は \relax になる。

```
5058 \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。

```
5059 \pandocSkipLoadPackage{babel}
5060 \bxjs@set@dupload@proc{babel.sty}{%
5061   \bxjs@unset@dupload@proc{babel.sty}%
5062   \ClassWarning\bxjs@clsname
5063     {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。

```
5064 \@tempswafalse \let\bxjs@babel@options\@empty
5065 \def\bxjs@tmpb{main=}%
5066 \@for\bxjs@tmpa:=#1\do{%
5067   \ifx\bxjs@tmpa\bxjs@tmpb \def\bxjs@tmpa{main=japanese}\fi
5068   \edef\bxjs@babel@options{\bxjs@babel@options,\bxjs@tmpa}}%
```

```

5069 \bxjs@cslet{ver@babel.sty}\@undefined
5070 \edef\bxjs@next{%
5071   \noexpand\RequirePackage[\bxjs@babel@options]{babel}\relax
5072 } \bxjs@next
5073 \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。

```

5074 \g@addto@macro\bxjs@endpreamble@hook{%
5075   \ifx\bxjs@babel@options\relax
5076     \bxjs@cslet{ver@babel.sty}\@undefined
5077   \fi}

```

3.0 版より前の japanese.ldf はサポート対象エンジンが限られていた。ここでは、エンジンの種類を問わず、「japanese.ldf が古い場合は読込を回避してダミー定義で代替する」という対策を入れる。実は japanese.ldf で行う定義は bxorigcapt の機能等により実質的に全て無効化されている。最新の環境においては「japanese 指定の Babel + bxorigrcapt パッケージ」の状態にしておきたい。

```

5078 \ifjsWitheTeX

```

filehook の機能を用いて japanese.ldf の読込にフックを仕込む。

```

5079 \AtBeginOfFile{japanese.ldf}{\bxjs@begin@japanese@ldf@hook}
5080 \def\bxjs@begin@japanese@ldf@hook{%
5081   \let\bxjs@begin@japanese@ldf@hook\relax
5082   \let\bxjs@save@ProvidesLanguage\ProvidesLanguage
5083   \let\bxjs@save@LdfInit\LdfInit
5084   \def\ProvidesLanguage##1[##2]{\bxjs@do@japanese@ldf{##2}}%
5085   \def\LdfInit##1##2{\bxjs@do@japanese@ldf{0000/00/00}}

```

バージョンを判定する部分。

※\LdfInit にも細工を入れている理由は、初期の japanese.ldf には \ProvidesLanguage が記述されていないため。

```

5086 \def\bxjs@do@japanese@ldf#1{\bxjs@do@japanese@ldf@a#1\@nil}
5087 \def\bxjs@do@japanese@ldf@a#1/#2/#3#4#5\@nil{%
5088   \let\LdfInit\bxjs@save@LdfInit
5089   \ClassInfo\bxjs@clsname
5090   {Release date of japanese.ldf is:\MessageBreak
5091     \@spaces #1/#2/#3#4\@gobble}%
5092   \ifnum#1#2#3#4<20201206 % v3.0
5093     \let\bxjs@japanese@ldf@skipped=t\csuse{endingput}%
5094   \fi}
5095 \AtEndOfFile{japanese.ldf}{\bxjs@end@japanese@ldf@hook}
5096 \def\bxjs@end@japanese@ldf@hook{%
5097   \let\bxjs@end@japanese@ldf@hook\relax
5098   \let\ProvidesLanguage\bxjs@save@ProvidesLanguage
5099   \let\LdfInit\bxjs@save@LdfInit
5100   \ifx t\bxjs@japanese@ldf@skipped
5101     \ClassWarningNoLine\bxjs@clsname
5102     {Loading japanese.ldf is skipped}%

```

ダミーの言語定義。

```
5103 \ifundef\l@japanese{\chardef\l@japanese\z@}{}%
5104 \let\datejapanese\@empty\let\captionjapanese\@empty
5105 \let\extrajapanese\@empty\let\noextrajapanese\@empty
5106 \main@language{japanese}%
5107 \fi}
5108 \g@addto@macro\bxjs@begin@document@hook{%
5109 \let\bxjs@begin@japanese\ldf@hook\relax
5110 \let\bxjs@end@japanese\ldf@hook\relax}
5111 \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。

```
5112 \bxjs@set@dupload@proc{geometry.sty}{%
5113 \setpagelayout*{#1}}
```

E.6 CJKmainfont 変数

LuaTeX (+ LuaTeX-jan) の場合に CJKmainfont 変数が指定された場合は \setmainfont の指定にまわす。

```
5114 \if 1\jsEngine
5115 \pandocSkipLoadPackage{xeCJK}
5116 \providecommand*\setCJKmainfont{}\setmainfont}
5117 \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。

```
5118 \if x\jsEngine
5119 \expandafter\g@addto@macro\csname opt@xeCJK.sty\endcsname{%
5120 ,space}
5121 \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。

```
5122 \ifjsWithTeX
5123 \@onlypreamble\bxjs@info@or@ban
5124 \def\bxjs@info@or@ban#1{%
```



```

5125 \PackageInfo{bxjs@clsname
5126   {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。

```

5127 \unless\ifbxjs@jp@or@indent
5128   \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。

```

5129 \IfFileExists{parskip.sty}{%
5130   \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。

```

5131 }{%else
5132   \eappto\bxjs@endpreamble@hook{%
5133     \parindent=\the\parindent\relax
5134     \parskip=\the\parskip\relax}}
5135 \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。

```

5136 \ifbxjs@jp@or@secnumdepth\else
5137   \bxjs@info@or@ban{secnumdepth}
5138   \eappto\bxjs@endpreamble@hook{%
5139     \c@secnumdepth=\the\c@secnumdepth\relax}
5140 \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。

```

5141 \ifbxjs@jp@or@block@heading\else
5142   \let\bxjs@frozen@paragraph\paragraph
5143   \let\bxjs@frozen@subparagraph\subparagraph
5144   \bxjs@info@or@ban{block-heading}
5145   \appto\bxjs@endpreamble@hook{%
5146     \let\oldparagraph\@undefined
5147     \let\paragraph\bxjs@frozen@paragraph
5148     \let\subparagraph\bxjs@frozen@subparagraph}
5149 \fi

```

以上。

```

5150 \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。

```
5151 \g@addto@macro\bxjs@begin@document@hook{%
5152   \@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。

```
5153   \ifx\oldparagraph\@undefined\else
5154     \@tempswatrue
5155   \fi
```

`\paragraph` が番号付きの場合は、マーク変更が必要である。

```
5156   \ifnum\c@secnumdepth>3
5157     \@tempswatrue
5158   \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。

```
5159   \if@tempswa\ifx\jsParagraphMark\bxjs@org@paragraph@mark
5160     \let\jsParagraphMark\@empty
5161   \fi\fi}
```

E.10 全角空白文字

LaTeX でない入力では、全角空きを入れるために全角空白文字（U+3000）が使われる可能性があるので、全角空白文字を和文文字でなく空きとして扱うようにしておく。

※ (u)pLaTeX では対応できないので対象外。

`\pandocZWSpace` 全角空白文字の入力で実行されるコード。

```
5162 \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。

```
5163 \ifnum0\if x\jsEngine1\fi\if 1\jsEngine1\fi>\z@
5164   \catcode"3000=\active
5165   \begingroup \catcode`\!=7
5166   \protected\gdef!!!!3000{\pandocZWSpace}
5167   \endgroup
5168 \else\ifx\DeclareUnicodeCharacter\@undefined\else
5169   \DeclareUnicodeCharacter{3000}{\bxjs@zsp@char}
5170   \bxjs@protected\def\bxjs@zsp@char{\pandocZWSpace}
5171 \fi\fi
```

E.11 hyperref 対策

hyperref の unicode オプションの固定を行う。

TODO: hyperref の開発状況を鑑みる限り、unicode オプションの固定処理は“危険”だと思われるので、可能ならば廃止したい。

```
5172 \if j\jsEngine
5173   \bxjs@fix@hyperref@unicode{false}
5174 \else
5175   \bxjs@fix@hyperref@unicode{true}
5176 \fi
```

E.12 Pandoc 要素に対する和文用の補正

■**重要要素** 重要 (Strong) 要素に対する L^AT_EX 出力は `\textbf` となるが、代わりに `\strong` を使いたいため、`\textbf` を書き換えてしまう (うわあ)。

```
5177 \ifbxjs@jp@fix@strong\ifbxjs@jp@strong@cmd
5178   \let\orgtextbf\textbf
5179   \DeclareRobustCommand\pandocTextbf[1]{%
5180     \begingroup
5181       \let\textbf\orgtextbf
5182       \strong{#1}%
5183     \endgroup}%
5184   \g@addto@macro\bxjs@begin@document@hook{%
5185     \let\textbf\pandocTextbf}
5186 \fi\fi
```

`\strong` の書体を設定する。

```
5187 \jsAtEndOfClass{%
5188   \ifx\strongfontdeclare\undefined\else
5189     \ifcase\bxjs@jp@strong
5190     \or \strongfontdeclare{\sffamily}%
5191     \or \strongfontdeclare{\sffamily\bfseries}%
5192   \fi
5193 \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。

```
5194 \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。

```
5195   \ifbxjs@bxghost@available
5196     \RequirePackage[verb]{bxghost}[2020/01/31]% v0.3.0
5197     \let\bxjs@eghostguarded\eghostguarded
5198   \else
```

```

5199 \chardef\bxjs@eghost@c=23
5200 \ifx j\jsEngine \xspcode\bxjs@eghost@c=3
5201 \else\ifx l\jsEngine \ltjsetparameter{alxspmode={\bxjs@eghost@c,3}}
5202 \else\ifx x\jsEngine %no-op
5203 \else \let\bxjs@eghost@c\@undefined
5204 \fi\fi\fi
5205 \ifx\bxjs@eghost@c\@undefined\else
5206 \font\bxjs@eghost@f=ec-lmr10 at 1.23456pt
5207 \def\bxjs@pan@eghost{\bgroup\bxjs@eghost@f\bxjs@eghost@c\egroup}
5208 \def\bxjs@eghostguarded#1{%
5209 \bxjs@pan@eghost\null#1\null\bxjs@pan@eghost}
5210 \fi
5211 \fi
5212 \ifx\bxjs@eghostguarded\@undefined\else
5213 \let\orgtexttt\texttt
5214 \DeclareRobustCommand\pandocTexttt[1]{%
5215 \ifmmode \nfss@text{\ttfamily #1}%
5216 \else
5217 \ifvmode \leavevmode \fi
5218 \bxjs@eghostguarded{\begingroup\ttfamily#1\endgroup}%
5219 \fi}
5220 \g@addto@macro\bxjs@begin@document@hook{%
5221 \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。

```

5222 \ifbxjs@bxghost@available\else
5223 \expandafter\def\expandafter\verb\expandafter{%
5224 \expandafter\bxjs@pan@eghost\verb}
5225 \g@addto@macro\verb@egroup{\bxjs@pan@eghost}
5226 \fi
5227 \fi
5228 \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。

```
5229 \if j\jsEngine
```

```
\bxjs@check@frontier \bxjs@check@frontier\CS は現在のパッケージ読込ネストレベルが丁度 1 であるときにのみ \CS を実行する。
```

```
5230 \def\bxjs@check@frontier{%
5231   \expandafter\bxjs@check@frontier@a\@currnamestack\noindent...\@nil}
5232 \def\bxjs@check@frontier@a#1#2#3#4#5\@nil#6{%
5233   \ifx\noindent#4#6\fi}
```

```
\bxjs@unforge@ifPDFTeX \ifPDFTeX を偽（正しい値）にする。
```

```
5234 \@onlypreamble\bxjs@unforge@ifPDFTeX
5235 \def\bxjs@unforge@ifPDFTeX{\global\bxjs@csletcs{ifPDFTeX}{iffalse}}
```

```
\bxjs@forge@ifPDFTeX \ifPDFTeX を真（偽装した値）にする。
```

```
5236 \@onlypreamble\bxjs@forge@ifPDFTeX
5237 \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。
```

```
5238 \def\bxjs@unload@forge@ifPDFTeX{%
5239   \bxjs@unforge@ifPDFTeX
5240   \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。

```
5241 \jsAtEndOfClass{\bxjs@forge@ifPDFTeX}
5242 \ifjsWithTeX
5243   \AtBeginOfFile{\bxjs@check@frontier\bxjs@unforge@ifPDFTeX}%
5244   \AtEndOfFile{\bxjs@check@frontier\bxjs@forge@ifPDFTeX}%
5245   \g@addto@macro\bxjs@endpreamble@hook{\bxjs@unload@forge@ifPDFTeX}
5246 \else
5247   \g@addto@macro\bxjs@begin@document@hook{\bxjs@unload@forge@ifPDFTeX}
5248 \fi
5249 \fi
```

E.14 完了

おしまい。

```
5250 %</pandoc>
```

和文ドライバ実装はここまで。

```
5251 %</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。

5252 %<*anc>

付録 G 補助パッケージ：bxjscompat 🤖

古いやつをどうにかするためのムニャムニャ。

G.1 準備

5253 %<*compat>

5254 \def\bxac@pkgname{bxjscompat}

\bxjx@engine エンジンの種別。

5255 \let\bxac@engine=n

5256 \def\bxac@do#1#2{%

5257 \edef\bxac@tmpa{\string#1}%

5258 \edef\bxac@tmpb{\meaning#1}%

5259 \ifx\bxac@tmpa\bxac@tmpb #2\fi}

5260 \bxac@do\XeTeXversion{\let\bxac@engine=x}

5261 \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。

5262 \ifx\jsAtEndOfClass\undefined

5263 \let\bxac@delayed@if@bxjs\@firstofone

5264 \else \let\bxac@delayed@if@bxjs\jsAtEndOfClass

5265 \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。

5266 \newif\ifbxac@in@old@behavior

5267 \let\ImposeOldLuaTeXBehavior\relax

5268 \let\RevokeOldLuaTeXBehavior\relax

G.2 X_YTeX 部分

5269 \ifx x\bxac@engine

■文字クラスの設定 X_YTeX の文字クラス (\XeTeXcharclass) の Unicode 規定に基づく設定は、初期の版ではフォーマットに組み込まれていたが、2016/02/01 以降の L^AT_EX カーネルでは「必要に応じて後から設定用のファイルを読み込む」方式に変更された。ここでは「設定されている状態」を担保する。

※ちなみに、Xe_{La}TeX 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。

```
5270 \ifx\XeTeXcharclass\undefined\else
5271 \bxac@delayed@if@bxjs{%
5272   \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。

```
5273   \ifx\XeTeXalloc@intercharclass\undefined\else
5274     \ifnum\XeTeXalloc@intercharclass=\z@
5275       \PackageInfo\bxac@pkgname
5276         {Setting up interchar class for CJK...\@gobble}%
5277       \InputIfFileExists{load-unicode-xetex-classes.tex}{%
5278         \XeTeXalloc@intercharclass=3
5279       }{\%else
5280       \PackageWarning\bxac@pkgname
5281         {Cannot find file 'load-unicode-xetex-classes.tex'%
5282         \@gobble}%
5283     }%
5284   \fi\fi
```

フォーマット組込んだ時代の設定は不完全なところがあるので補正する。

```
5285   \ifnum\XeTeXcharclass"3041=\z@
5286     \PackageInfo\bxac@pkgname
5287       {Adjusting interchar class for CJK...\@gobble}%
5288     \@for\bxac@tmpb:={%
5289       3041,3043,3045,3047,3049,3063,3083,3085,3087,308E,%
5290       3095,3096,30A1,30A3,30A5,30A7,30A9,30C3,30E3,30E5,%
5291       30E7,30EE,30F5,30F6,30FC,31F0,31F1,31F2,31F3,31F4,%
5292       31F5,31F6,31F7,31F8,31F9,31FA,31FB,31FC,31FD,31FE,%
5293       31FF%
5294     }\do{\XeTeXcharclass"\bxac@tmpb=\@ne}%
5295   \fi
5296 }%
5297 }
5298 \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。

```
5299 \chardef\bxac@tmpb=11
5300 \def\bxac@do#1#2{%
5301   \@tempcnta=#1\relax
5302   \unless\ifnum\catcode\@tempcnta=\bxac@tmpb
5303     \chardef\bxac@tmpa=#2\relax
5304     \@whilenum{\@tempcnta<\bxac@tmpa}\do{%
5305       \catcode\@tempcnta\bxac@tmpb \advance\@tempcnta\@ne}%
5306     \fi}
5307 \bxac@do{"4E00}{ "9FCD}
```

以上。

```
5308 \fi
```

G.3 LuaTeX 部分

```
5309 \ifx 1\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。

```
5310 %\unless\ifnum\luatexversion<80 \ifnum\luatexversion<85
5311 % \chardef\pdftexversion=200
5312 % \def\pdftexrevision{0}
5313 % \let\pdftexbanner\luatexbanner
5314 %\fi\fi
```

`\ImposeOldLuaTeXBehavior` 0.85 版以降であるかを確認する。

```
\RevokeOldLuaTeXBehavior 5315 \begingroup\expandafter\expandafter\expandafter\endgroup
5316 \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。

```
5317 \def\bxac@ob@list{%
5318   \do{\let}\pdfoutput{\outputmode}%
5319   \do{\let}\pdfpagewidth{\pagewidth}%
5320   \do{\let}\pdfpageheight{\pageheight}%
5321   \do{\protected\edef}\pdfhorigin{{\pdfvariable horigin}}%
5322   \do{\protected\edef}\pdfvorigin{{\pdfvariable vorigin}}%
5323 \def\bxac@ob@do#1#2{\begingroup
5324   \expandafter\bxac@ob@do@a\csname bxac@\string#2\endcsname{#1}#2}
5325 \def\bxac@ob@do@a#1#2#3#4{\endgroup
5326   \ifbxac@in@old@behavior \let#1#3\relax #2#3#4\relax
5327   \else \let#3#1\relax \let#1\@undefined
5328   \fi}
5329 \protected\def\ImposeOldLuaTeXBehavior{%
5330   \unless\ifbxac@in@old@behavior
5331     \bxac@in@old@behaviortrue
5332     \let\do\bxac@ob@do \bxac@ob@list
5333   \fi}
5334 \protected\def\RevokeOldLuaTeXBehavior{%
5335   \ifbxac@in@old@behavior
5336     \bxac@in@old@behaviorfalse
5337     \let\do\bxac@ob@do \bxac@ob@list
5338   \fi}
5339 \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。

```
5340 \directlua{
5341   local function range(cs, ce, cc, ff)
5342     if ff or not tex.getcatcode(cs) == cc then
5343       local setcc = tex.setcatcode
5344       for c = cs, ce do setcc(c, cc) end
5345     end
```



```

5346     end
5347     range(0x3400, 0x4DB5, 11, false)
5348     \ifnum\luatexversion>64
5349     range(0x4DB5, 0x4DBF, 11, true)
5350     range(0x4E00, 0x9FCC, 11, false)
5351     range(0x9FCD, 0x9FFF, 11, true)
5352     range(0xAC00, 0xD7A3, 11, false)
5353     range(0x20000, 0x2A6D6, 11, false)
5354     range(0x2A6D7, 0x2A6FF, 11, true)
5355     range(0x2A700, 0x2B734, 11, false)
5356     range(0x2B735, 0x2B73F, 11, true)
5357     range(0x2B740, 0x2B81D, 11, false)
5358     range(0x2B81E, 0x2B81F, 11, true)
5359     range(0x2B820, 0x2CEA1, 11, false)
5360     range(0x2CEA2, 0x2FFFD, 11, true)
5361     \fi
5362 }

```

以上。

```
5363 \fi
```

G.4 完了

おしまい。

```
5364 %</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

```

5365 %<*cjkcat>
5366 \def\bxjx@pkgname{bxjscjkat}
5367 \newcount\bxjx@canta
5368 \@onlypreamble\bxjx@tmpdo
5369 \@onlypreamble\bxjx@tmpdo@a
5370 \@onlypreamble\bxjx@tmpdo@b

```

\bxjx@engine エンジンの種別。

```

5371 \let\bxjx@engine=n
5372 \def\bxjx@tmpdo#1#2{%
5373   \edef\bxjx@tmpa{\string#1}%
5374   \edef\bxjx@tmpb{\meaning#1}%
5375   \ifx\bxjx@tmpa\bxjx@tmpb #2\fi}
5376 \bxjx@tmpdo\kanjiskip{\let\bxjx@engine=j}
5377 \bxjx@tmpdo\enablecjktoken{%
5378   \ifx\ucs\@undefined\else \ifnum\ucs"3000="3000

```

```

5379 \let\bxjx@engine=u\fi\fi}
5380 \bxjx@tmpdo\XeTeXversion{\let\bxjx@engine=x}
5381 \bxjx@tmpdo\pdfTeXversion{\let\bxjx@engine=p}
5382 \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。

```

5383 \def\bxjx@tmpdo#1#2{%
5384   \if#1\bxjx@engine
5385     \@ifpackageloaded{#2}{\fi}%else
5386     \PackageError\bxjx@pkgname
5387       {Package '#2' must be loaded}%
5388       {Package loading is aborted.\MessageBreak\@ehc}%
5389   \endinput}
5390 \fi}
5391 \bxjx@tmpdo{p}{bxcjkatype}
5392 \bxjx@tmpdo{x}{xeCJK}
5393 \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。

```

5394 \ifx\TextOrMath\@undefined
5395   \RequirePackage{fixltx2e}
5396 \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。

```

5397 \if u\bxjx@engine
5398 \@for\bxjx@tmpa:={%
5399 0080,0100,0180,0250,02B0,0300,0500,0530,0590,0600,%
5400 0700,0750,0780,07C0,0800,0840,0860,08A0,0900,0980,%
5401 0A00,0A80,0B00,0B80,0C00,0C80,0D00,0D80,0E00,0E80,%
5402 0F00,1000,10A0,1200,1380,13A0,1400,1680,16A0,1700,%
5403 1720,1740,1760,1780,1800,18B0,1900,1950,1980,19E0,%
5404 1A00,1A20,1AB0,1B00,1B80,1BC0,1C00,1C50,1C80,1CC0,%
5405 1CD0,1D00,1D80,1DC0,1E00,1F00,2440,27C0,27F0,2800,%
5406 2A00,2C00,2C60,2C80,2D00,2D30,2D80,2DE0,2E00,4DC0,%
5407 A4D0,A500,A640,A6A0,A700,A720,A800,A830,A840,A880,%
5408 A8E0,A900,A930,A980,A9E0,AA00,AA60,AA80,AAE0,AB00,%
5409 AB30,AB70,ABC0,D800,DB80,DC00,E000,FB00,FB50,FE00,%
5410 FE70,FF00,%
5411 10000,10080,10100,10140,10190,101D0,10280,102A0,%
5412 102E0,10300,10330,10350,10380,103A0,10400,10450,%
5413 10480,104B0,10500,10530,10600,10800,10840,10860,%
5414 10880,108E0,10900,10920,10980,109A0,10A00,10A60,%
5415 10A80,10AC0,10B00,10B40,10B60,10B80,10C00,10C80,%

```

```

5416 10E60,11000,11080,110D0,11100,11150,11180,111E0,%
5417 11200,11280,112B0,11300,11400,11480,11580,11600,%
5418 11660,11680,11700,118A0,11A00,11A50,11AC0,11C00,%
5419 11C70,11D00,12000,12400,12480,13000,14400,16800,%
5420 16A40,16AD0,16B00,16F00,1BC00,1BCA0,1D000,1D100,%
5421 1D200,1D300,1D360,1D400,1D800,1E000,1E800,1E900,%
5422 1EE00,1F000,1F030,1FOA0,1F300,1F600,1F650,1F680,%
5423 1F700,1F780,1F800,1F900,E0000,E0100,F0000,100000,%
5424 00C0%
5425 }\do{%
5426 \@tempcnta="\bxjx@tmpa\relax
5427 \@tempcntb\@tempcnta\advance\@tempcntb\m@ne
5428 \chardef\bxjx@tmpb\kcatcode\@tempcntb
5429 \kcatcode\@tempcnta=15 \kcatcode\@tempcntb\bxjx@tmpb}
5430 \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。

```

5431 \@onlypreamble\bxjx@grkcyr@list
5432 \def\bxjx@grkcyr@list{%
5433 \do{0391}{LGR}{\textAlpha}{A}%           % GR. C. L. ALPHA
5434 \do{0392}{LGR}{\textBeta}{B}%           % GR. C. L. BETA
5435 \do{0393}{LGR}{\textGamma}{\Gamma}%     % GR. C. L. GAMMA
5436 \do{0394}{LGR}{\textDelta}{\Delta}%     % GR. C. L. DELTA
5437 \do{0395}{LGR}{\textEpsilon}{E}%        % GR. C. L. EPSILON
5438 \do{0396}{LGR}{\textZeta}{Z}%           % GR. C. L. ZETA
5439 \do{0397}{LGR}{\textEta}{H}%            % GR. C. L. ETA
5440 \do{0398}{LGR}{\textTheta}{\Theta}%     % GR. C. L. THETA
5441 \do{0399}{LGR}{\textIota}{I}%           % GR. C. L. IOTA
5442 \do{039A}{LGR}{\textKappa}{K}%          % GR. C. L. KAPPA
5443 \do{039B}{LGR}{\textLambda}{\Lambda}%   % GR. C. L. LAMDA
5444 \do{039C}{LGR}{\textMu}{M}%             % GR. C. L. MU
5445 \do{039D}{LGR}{\textNu}{N}%            % GR. C. L. NU
5446 \do{039E}{LGR}{\textXi}{\Xi}%          % GR. C. L. XI
5447 \do{039F}{LGR}{\textOmicron}{O}%      % GR. C. L. OMICRON
5448 \do{03A0}{LGR}{\textPi}{\Pi}%          % GR. C. L. PI
5449 \do{03A1}{LGR}{\textRho}{P}%           % GR. C. L. RHO

```

5450 \do{03A3}{LGR}{\textSigma}{\Sigma}%	% GR. C. L. SIGMA
5451 \do{03A4}{LGR}{\textTau}{T}%	% GR. C. L. TAU
5452 \do{03A5}{LGR}{\textUpsilon}{\Upsilon}%	% GR. C. L. UPSILON
5453 \do{03A6}{LGR}{\textPhi}{\Phi}%	% GR. C. L. PHI
5454 \do{03A7}{LGR}{\textChi}{X}%	% GR. C. L. CHI
5455 \do{03A8}{LGR}{\textPsi}{\Psi}%	% GR. C. L. PSI
5456 \do{03A9}{LGR}{\textOmega}{\Omega}%	% GR. C. L. OMEGA
5457 \do{03B1}{LGR}{\textalpha}{\alpha}%	% GR. S. L. ALPHA
5458 \do{03B2}{LGR}{\textbeta}{\beta}%	% GR. S. L. BETA
5459 \do{03B3}{LGR}{\textgamma}{\gamma}%	% GR. S. L. GAMMA
5460 \do{03B4}{LGR}{\textdelta}{\delta}%	% GR. S. L. DELTA
5461 \do{03B5}{LGR}{\textepsilon}{\epsilon}%	% GR. S. L. EPSILON
5462 \do{03B6}{LGR}{\textzeta}{\zeta}%	% GR. S. L. ZETA
5463 \do{03B7}{LGR}{\texteta}{\eta}%	% GR. S. L. ETA
5464 \do{03B8}{LGR}{\texttheta}{\theta}%	% GR. S. L. THETA
5465 \do{03B9}{LGR}{\textiota}{\iota}%	% GR. S. L. IOTA
5466 \do{03BA}{LGR}{\textkappa}{\kappa}%	% GR. S. L. KAPPA
5467 \do{03BB}{LGR}{\textlambda}{\lambda}%	% GR. S. L. LAMDA
5468 \do{03BC}{LGR}{\textmu}{\mu}%	% GR. S. L. MU
5469 \do{03BD}{LGR}{\textnu}{\nu}%	% GR. S. L. NU
5470 \do{03BE}{LGR}{\textxi}{\xi}%	% GR. S. L. XI
5471 \do{03BF}{LGR}{\textomicron}{o}%	% GR. S. L. OMICRON
5472 \do{03C0}{LGR}{\textpi}{\pi}%	% GR. S. L. PI
5473 \do{03C1}{LGR}{\textrho}{\rho}%	% GR. S. L. RHO
5474 \do{03C2}{LGR}{\textvarsigma}{\varsigma}%	% GR. S. L. FINAL SIGMA
5475 \do{03C3}{LGR}{\textsigma}{\sigma}%	% GR. S. L. SIGMA
5476 \do{03C4}{LGR}{\texttau}{\tau}%	% GR. S. L. TAU
5477 \do{03C5}{LGR}{\textupsilon}{\upsilon}%	% GR. S. L. UPSILON
5478 \do{03C6}{LGR}{\textphi}{\phi}%	% GR. S. L. PHI
5479 \do{03C7}{LGR}{\textchi}{\chi}%	% GR. S. L. CHI
5480 \do{03C8}{LGR}{\textpsi}{\psi}%	% GR. S. L. PSI
5481 \do{03C9}{LGR}{\textomega}{\omega}%	% GR. S. L. OMEGA
5482 \do{0401}{T2A}{\CYR YO}{}	% CY. C. L. IO
5483 \do{0410}{T2A}{\CYR A}{}	% CY. C. L. A
5484 \do{0411}{T2A}{\CYR B}{}	% CY. C. L. BE
5485 \do{0412}{T2A}{\CYR V}{}	% CY. C. L. VE
5486 \do{0413}{T2A}{\CYR G}{}	% CY. C. L. GHE
5487 \do{0414}{T2A}{\CYR D}{}	% CY. C. L. DE
5488 \do{0415}{T2A}{\CYR E}{}	% CY. C. L. IE
5489 \do{0416}{T2A}{\CYR ZH}{}	% CY. C. L. ZHE
5490 \do{0417}{T2A}{\CYR Z}{}	% CY. C. L. ZE
5491 \do{0418}{T2A}{\CYR I}{}	% CY. C. L. I
5492 \do{0419}{T2A}{\CYRISHRT}{}	% CY. C. L. SHORT I
5493 \do{041A}{T2A}{\CYR K}{}	% CY. C. L. KA
5494 \do{041B}{T2A}{\CYR L}{}	% CY. C. L. EL
5495 \do{041C}{T2A}{\CYR M}{}	% CY. C. L. EM
5496 \do{041D}{T2A}{\CYR N}{}	% CY. C. L. EN
5497 \do{041E}{T2A}{\CYR O}{}	% CY. C. L. O
5498 \do{041F}{T2A}{\CYR P}{}	% CY. C. L. PE

5499 \do{0420}{T2A}{\CYRR}{}%	% CY. C. L. ER
5500 \do{0421}{T2A}{\CYRS}{}%	% CY. C. L. ES
5501 \do{0422}{T2A}{\CYRT}{}%	% CY. C. L. TE
5502 \do{0423}{T2A}{\CYRU}{}%	% CY. C. L. U
5503 \do{0424}{T2A}{\CYRF}{}%	% CY. C. L. EF
5504 \do{0425}{T2A}{\CYRH}{}%	% CY. C. L. HA
5505 \do{0426}{T2A}{\CYRC}{}%	% CY. C. L. TSE
5506 \do{0427}{T2A}{\CYRCH}{}%	% CY. C. L. CHE
5507 \do{0428}{T2A}{\CYRSH}{}%	% CY. C. L. SHA
5508 \do{0429}{T2A}{\CYRSHCH}{}%	% CY. C. L. SHCHA
5509 \do{042A}{T2A}{\CYRHRDSN}{}%	% CY. C. L. HARD SIGN
5510 \do{042B}{T2A}{\CYRERY}{}%	% CY. C. L. YERU
5511 \do{042C}{T2A}{\CYRSFTSN}{}%	% CY. C. L. SOFT SIGN
5512 \do{042D}{T2A}{\CYREREV}{}%	% CY. C. L. E
5513 \do{042E}{T2A}{\CYRYU}{}%	% CY. C. L. YU
5514 \do{042F}{T2A}{\CYRYA}{}%	% CY. C. L. YA
5515 \do{0430}{T2A}{\cyra}{}%	% CY. S. L. A
5516 \do{0431}{T2A}{\cyrb}{}%	% CY. S. L. BE
5517 \do{0432}{T2A}{\cyrv}{}%	% CY. S. L. VE
5518 \do{0433}{T2A}{\cyrg}{}%	% CY. S. L. GHE
5519 \do{0434}{T2A}{\cyrd}{}%	% CY. S. L. DE
5520 \do{0435}{T2A}{\cyre}{}%	% CY. S. L. IE
5521 \do{0436}{T2A}{\cyrzh}{}%	% CY. S. L. ZHE
5522 \do{0437}{T2A}{\cyrz}{}%	% CY. S. L. ZE
5523 \do{0438}{T2A}{\cyri}{}%	% CY. S. L. I
5524 \do{0439}{T2A}{\cyrishrt}{}%	% CY. S. L. SHORT I
5525 \do{043A}{T2A}{\cyrk}{}%	% CY. S. L. KA
5526 \do{043B}{T2A}{\cyr1}{}%	% CY. S. L. EL
5527 \do{043C}{T2A}{\cyr2}{}%	% CY. S. L. EM
5528 \do{043D}{T2A}{\cyrn}{}%	% CY. S. L. EN
5529 \do{043E}{T2A}{\cyro}{}%	% CY. S. L. O
5530 \do{043F}{T2A}{\cyrp}{}%	% CY. S. L. PE
5531 \do{0440}{T2A}{\cyrr}{}%	% CY. S. L. ER
5532 \do{0441}{T2A}{\cyrs}{}%	% CY. S. L. ES
5533 \do{0442}{T2A}{\cyrt}{}%	% CY. S. L. TE
5534 \do{0443}{T2A}{\cyru}{}%	% CY. S. L. U
5535 \do{0444}{T2A}{\cyrf}{}%	% CY. S. L. EF
5536 \do{0445}{T2A}{\cyrh}{}%	% CY. S. L. HA
5537 \do{0446}{T2A}{\cyrc}{}%	% CY. S. L. TSE
5538 \do{0447}{T2A}{\cyrch}{}%	% CY. S. L. CHE
5539 \do{0448}{T2A}{\cyrsh}{}%	% CY. S. L. SHA
5540 \do{0449}{T2A}{\cyrshch}{}%	% CY. S. L. SHCHA
5541 \do{044A}{T2A}{\cyrhrdsn}{}%	% CY. S. L. HARD SIGN
5542 \do{044B}{T2A}{\cyrery}{}%	% CY. S. L. YERU
5543 \do{044C}{T2A}{\cyrsftsn}{}%	% CY. S. L. SOFT SIGN
5544 \do{044D}{T2A}{\cyrerev}{}%	% CY. S. L. E
5545 \do{044E}{T2A}{\cyryu}{}%	% CY. S. L. YU
5546 \do{044F}{T2A}{\cyrya}{}%	% CY. S. L. YA
5547 \do{0451}{T2A}{\cyryo}{}%	% CY. S. L. IO

```

5548 \do{00A7}{TS1}{\textsection}{\mathsection}% SECTION SYMBOL
5549 \do{00A8}{TS1}{\textasciidieresis}{}% % DIAERESIS
5550 \do{00B0}{TS1}{\textdegree}{\mathdegree}% % DEGREE SIGN
5551 \do{00B1}{TS1}{\textpm}{\pm}% % PLUS-MINUS SIGN
5552 \do{00B4}{TS1}{\textasciicircum}{}% % ACUTE ACCENT
5553 \do{00B6}{TS1}{\textparagraph}{\mathparagraph}% PILCROW SIGN
5554 \do{00D7}{TS1}{\texttimes}{\times}% % MULTIPLICATION SIGN
5555 \do{00F7}{TS1}{\textdiv}{\div}% % DIVISION SIGN
5556 }

```

`\mathdegree` 面倒なので補っておく。

```
5557 \providecommand*{\mathdegree}{\circ}
```

`\ifbxjx@gcc@cjkl` [スイッチ]「特定 CJK 曖昧文字」を和文扱いにするか。

```
5558 \newif\ifbxjx@gcc@cjkl
```

`\greekasCJK` [公開命令]「特定 CJK 曖昧文字」を和文扱いにする。

```

5559 \newcommand*\greekasCJK{%
5560   \bxjx@gcc@cjkltrue}

```

`\nogreekasCJK` [公開命令]「特定 CJK 曖昧文字」を欧文扱いにする。

```

5561 \newcommand*\nogreekasCJK{%
5562   \bxjx@gcc@cjklfalse}

```

`\bxjx@fake@grkl` `\bxjx@fake@grkl{〈出力文字〉}{〈基準文字〉}`： ラテン文字で代用される数式ギリシャ文字の出力を行う。〈基準文字〉(`mathchardef`の制御綴)の数式クラスと数式ファミリーを引き継いで、〈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@grkl{B}{\Pi}` は `\mathchar"7042` を実行する。

※フォントパッケージ使用時の再定義を考慮して、〈基準文字〉が `mathchardef` であるかを検査し、そうでない場合はフォールバックとして単に〈出力文字〉を実行する。

```

5563 \def\bxjx@tmpdo#1\relax{%
5564   \def\bxjx@fake@grkl##1##2{%
5565     \expandafter\bxjx@fake@grkl@a\meaning##2#1\@nil{##1}{##2}}%
5566   \def\bxjx@fake@grkl@a##1##2\@nil##3##4{%
5567     \ifx\##1\%
5568       \bxjx@canta##4\divide\bxjx@canta\@cclvi
5569       \multiply\bxjx@canta\@cclvi \advance\bxjx@canta`##3\relax
5570       \mathchar\bxjx@canta
5571     \else ##3\fi}
5572 } \expandafter\bxjx@tmpdo\string\mathchar\relax

```

■pdfTeX・upTeX の場合

```
5573 \ifnum0\if p\bxjx@engine1\fi\if u\bxjx@engine1\fi>0
```

- `\[bxjx@KC/〈符号値〉]`： その文字が「特定曖昧 CJK 文字」に該当する場合に定義済になる。

まず `inputenc` を読み込んで入力エンコーディングを `utf8` に変更する。

※「既定 UTF-8 化」後の \LaTeX においても、必ず「`inputenc` が明示的に読み込まれた」状態になる。

```
5574 \ifpackageloaded{inputenc}{\else
5575   \RequirePackage[utf8]{inputenc}}
5576 \def\bxjx@tmpa{utf8}
5577 \ifx\bxjx@tmpa\inputencdoingname
5578   \PackageWarningNoLine\bxjx@pkgname
5579     {Input encoding changed to utf8}%
5580   \inputencoding{utf8}%
5581 \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。

```
5582 \if u\bxjx@engine
5583 \kcatcode"0370=15
5584 \kcatcode"0400=15
5585 \kcatcode"0500=15
5586 \fi
```

各文字について `\DeclareUnicodeCharacter` を実行する。

```
5587 \def\bxjx@tmpdo#1{%
5588   \@tempcnta="#1\relax
5589   \expandafter\bxjx@tmpdo@a\csname bxjx@KC/\the\@tempcnta\endcsname{#1}}
5590 \def\bxjx@tmpdo@a#1#2#3#4#5{%
```

引数 = $\backslash\text{[bxjx@KC/}\langle\text{符号値}\rangle\text{]}{\langle\text{符号値}\rangle}{\langle\text{fontenc}\rangle}{\langle\text{LICR}\rangle}{\langle\text{数式 LICR}\rangle}$

“数式中の動作”を決定する。 $\langle\text{数式 LICR}\rangle$ が空（数式非対応）なら警告を出す。

```
5591 \ifx\#5\%
5592   \def\bxjx@tmpa{\inmathwarn#4}%
```

$\langle\text{数式 LICR}\rangle$ 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。

```
5593 \else\ifcat A\noexpand#5%
5594   \edef\bxjx@tmpa{\noexpand\bxjx@fake@grk{#5}%
5595     {\ifnum\uccode`#5=`#5\noexpand\Pi\else\noexpand\pi\fi}}%
```

それ以外は $\langle\text{数式 LICR}\rangle$ をそのまま実行する。

```
5596 \else \def\bxjx@tmpa{#5}%
5597 \fi\fi
5598 \def\bxjx@tmpb{\bxjx@tmpdo@b{#1}{#2}{#3}{#4}}%
5599 \expandafter\bxjx@tmpb\expandafter{\bxjx@tmpa}}
```

以降はエンジン種別で分岐する。 upTeX の場合。

```
5600 \if u\bxjx@engine
5601 \def\bxjx@tmpdo@b#1#2#3#4#5{%
```

引数 = $\backslash\text{[bxjx@KC/}\langle\text{符号値}\rangle\text{]}{\langle\text{符号値}\rangle}{\langle\text{fontenc}\rangle}{\langle\text{LICR}\rangle}{\langle\text{数式中の動作}\rangle}$

当該の Unicode 文字の動作は「テキストでは $\langle\text{LICR}\rangle$ 、数式では $\langle\text{数式中の動作}\rangle$ 」となる。 LICR は現在エンコーディングで有効な定義がある場合はそれが実行されるはずである。（つ

まり、現在が LGR である場合はギリシャ文字は常に欧文扱いになる。) それ以外の場合は LICR を `\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。

```
5602 \kchardef#1=\@tempcnta
5603 \DeclareTextCommandDefault{#4}{\bxjx@ja@or@not{#1}{#3}{#4}}%
5604 \DeclareUnicodeCharacter{#2}{\TextOrMath{#4}{#5}}}
```

pdfTeX の場合も処理はほとんど同じ。ただし、和文用の定義として `\UTF{⟨符号値⟩}` を使う (`\UTF` は `bxckjatype` の命令)。`\[bxjx@KC/...` は使わないが定義済にする必要がある。

```
5605 \else\if p\bxjx@engine
5606 \def\bxjx@tmpdo@b#1#2#3#4#5{%
5607   \mathchardef#1=\@tempcnta
5608   \DeclareTextCommandDefault{#4}{\bxjx@ja@or@not{\UTF{#2}}{#3}{#4}}%
5609   \DeclareUnicodeCharacter{#2}{\TextOrMath{#4}{#5}}}%
5610 \fi\fi
```

以上の処理を「特定 CJK 曖昧文字」の各々に適用する。

```
5611 \let\do\bxjx@tmpdo \bxjx@grkcyr@list
```

`\bxjx@DeclareUnicodeCharacter` `\bxjx@DeclareUnicodeCharacter` を改変して、「特定 CJK 曖昧文字」の場合に再定義を抑止したもの。

```
5612 \@onlypreamble\bxjx@org@DeclareUnicodeCharacter
5613 \let\bxjx@org@DeclareUnicodeCharacter\DeclareUnicodeCharacter
5614 \@onlypreamble\bxjx@DeclareUnicodeCharacter
5615 \def\bxjx@DeclareUnicodeCharacter#1#2{%
5616   \count@=#1\relax
5617   \expandafter\ifx\csname bxjx@KC/\the\count@\endcsname\relax
5618     \bxjx@org@DeclareUnicodeCharacter{#1}{#2}%
5619   \else
5620     \wlog{ \space\space skipped defining Unicode char U+#1}%
5621   \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。

```
5622 \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。

```
5623   \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。

```
5624   \else\expandafter\ifx\csname T@#2\endcsname\relax #1%
5625   \else \UseTextSymbol{#2}{#3}%
5626   \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。

```
5627 \begingroup
```



```

5628 \toks@\expandafter{\DeclareFontEncoding@{#1}{#2}{#3}}
5629 \xdef\next{\def\noexpand\DeclareFontEncoding###1##2###3{%
5630   \noexpand\bxjx@swap@DUC@cmd
5631   \the\toks@
5632   \noexpand\bxjx@swap@DUC@cmd}}
5633 \endgroup\next
5634 \def\bxjx@swap@DUC@cmd{%
5635   \let\bxjx@tmpa\DeclareUnicodeCharacter
5636   \let\DeclareUnicodeCharacter\bxjx@DeclareUnicodeCharacter
5637   \let\bxjx@DeclareUnicodeCharacter\bxjx@tmpa
5638   \let\bxjx@tmpa\relax}

```

以上。

■X_YTeX・LuaTeX の場合

```

5639 \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。

```

5640 \def\bxjx@tmpdo#1{%
5641   \bxjx@cna="#1\relax
5642   \begingroup
5643     \lccode`~=\bxjx@cna
5644     \lowercase{\endgroup
5645       \bxjx@tmpdo@a{~}}{#1}}
5646 \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。

```

5647   \ifx\\#5\\\let\bxjx@tmpa\relax
5648   \else\ifcat A\noexpand#5%
5649     \edef\bxjx@tmpa{\noexpand\bxjx@fake@grk{#5}%
5650       {\ifnum\uccode`#5=#5\noexpand\Pi\else\noexpand\pi\fi}}%
5651   \else \def\bxjx@tmpa{#5}%
5652   \fi\fi
5653   \ifx\bxjx@tmpa\relax\else
5654     \mathcode\bxjx@cna"8000 \let#1\bxjx@tmpa
5655   \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。

```

5656 \mathchardef\bxjx@tmpa="119
5657 \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。

```

5658 \if l\bxjx@engine
5659   \protected\def\greekasCJK{%

```

```

5660 \bxjx@gcc@cjctrue
5661 \ltjsetparameter{jacharrange={+2, +8}}
5662 \protected\def\nogreekasCJK{%
5663 \bxjx@gcc@cjcfalse
5664 \ltjsetparameter{jacharrange={-2, -8}}
5665 \fi

```

X_ƎT_ƎX の場合、xeCJK は X_ƎT_ƎX 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。

```

5666 \if x\bxjx@engine
5667 \let\bxjx@gcc@cjkl@list\@empty
5668 \def\do#1#2#3#4{%
5669 \edef\bxjx@gcc@cjkl@list{\bxjx@gcc@cjkl@list
5670 \noexpand\XeTeXcharclass"#1\bxjx@c@nta}}
5671 \bxjx@grkcy@list
5672 \protected\def\greekasCJK{%
5673 \bxjx@gcc@cjctrue
5674 \bxjx@c@nta=\@ne \bxjx@gcc@cjkl@list}
5675 \protected\def\nogreekasCJK{%
5676 \bxjx@gcc@cjcfalse
5677 \bxjx@c@nta=\z@ \bxjx@gcc@cjkl@list}
5678 \fi

```

以上。

```
5679 \fi\fi
```

H.4 初期設定

「特定 CJK 曖昧文字」を欧文扱いにする。

```
5680 \nogreekasCJK
```

H.5 完了

おしまい。

```
5681 %</cjkat>
```

付録 I 補助パッケージ：bxjspandoc 🐼

Pandoc の L^AT_ƎX 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_ƎX 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

```

5682 %<*ancpandoc>
5683 %% このファイルは日本語文字を含みます.
5684 \def\bxjsp@pkgname{bxjscjkat}

```

`\bxjsp@engine` エンジンの種別。

```

5685 \let\bxjsp@engine=n
5686 \@onlypreamble\bxjsp@do
5687 \def\bxjsp@do#1#2{%
5688   \edef\bxjsp@tmpa{\string#1}%
5689   \edef\bxjsp@tmpb{\meaning#1}%
5690   \ifx\bxjsp@tmpa\bxjsp@tmpb #2\fi}
5691 \bxjsp@do\kanjiskip{\let\bxjsp@engine=j}
5692 \bxjsp@do\XeTeXversion{\let\bxjsp@engine=x}
5693 \bxjsp@do\pdftexversion{\let\bxjsp@engine=p}
5694 \bxjsp@do\luatexversion{\let\bxjsp@engine=l}

```

`\bxjsp@begin@document@hook` 文書本体開始時フック。

```

5695 \@onlypreamble\bxjsp@begin@document@hook
5696 \let\bxjsp@begin@document@hook\@empty
5697 \AtBeginDocument{\bxjsp@begin@document@hook}

```

`\ifbxjsp@babel@used` [スイッチ] Babel が読み込まれたか。

```

5698 \newif\ifbxjsp@babel@used
5699 \g@addto@macro\bxjsp@begin@document@hook{%
5700   \@ifpackageloaded{babel}{\bxjsp@babel@usedtrue}{}}

```

1.2 パッケージオプション

`english` オプションが指定されている場合、`\ldots` の調整を抑止する。

※つまり、「グローバルの `english` オプション」が指定されている場合も抑止の対象になる。BXJS クラスの英語モードを想定しているが、それ以外の場合でも、一般的な L^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。

```

5701 \newif\ifbxjsp@english
5702 \DeclareOption{english}{\bxjsp@englishtrue}

```

オプション定義はおしまい。

```

5703 \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。

```

5704 \@onlypreamble\pandocSkipLoadFile
5705 \newcommand*\pandocSkipLoadFile[1]{%
5706   \expandafter\bxjsp@skip@load@file@a\csname ver@#1\endcsname{#1}}
5707 \def\bxjsp@skip@load@file@a#1#2{%
5708   \ifx#1\relax

```

```

5709 \def#1{2001/01/01}%
5710 \PackageInfo{bxjisp@pkgname
5711 {File '#2' marked as loaded\@gobble}%
5712 \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。

```

5713 \@onlypreamble\pandocSkipLoadPackage
5714 \newcommand*\pandocSkipLoadPackage[1]{%
5715 \pandocSkipLoadFile{#1.sty}}

```

I.4 fixltx2e パッケージ

テンプレートでは `fixltx2e` パッケージを読み込むが、最近（2015 年版以降）の `LATEX` ではこれで警告が出る。これを抑止する。

`LATEX` カーネルが新しい場合は `fixltx2e` を読込済にする。

```

5716 \ifx\@IncludeInRelease\@undefined\else
5717 \pandocSkipLoadPackage{fixltx2e}
5718 \fi

```

I.5 cmap パッケージ

エンジンが `(u)pLATEX` のときに `cmap` パッケージが読み込まれるのを阻止する。（実際は警告が出るだけで無害であるが。）

```

5719 \if j\bxjisp@engine
5720 \pandocSkipLoadPackage{cmap}
5721 \fi

```

I.6 microtype パッケージ

警告が多すぎなので消す。

```

5722 \if j\bxjisp@engine \else
5723 \PassOptionsToPackage{verbose=silent}{microtype}
5724 \fi

```

エンジンが `(u)pLATEX` のときに `microtype` パッケージが読み込まれるのを阻止し、さらにテンプレートで使われている命令を通すためにダミーの定義を行う。

※昔は `standard` ドライバでこの処理を行っていたが、元来は `Pandoc` 用の処理なので、1.5 版で `pandoc` に移動。

```

5725 \if j\bxjisp@engine
5726 \pandocSkipLoadPackage{microtype}
5727 \newcommand*\UseMicrotypeSet[2][{}]{%
5728 \fi

```

I.7 Unicode 文字変換対策

Pandoc で \LaTeX 形式に書き出す場合は、元データ中の一部の Unicode 文字を「 \LaTeX の表記」に置き換える。その中には日本語文書で問題になるものが含まれる。

```
…→\ldots{} ‘→` ’→' “→` ”→''
```

日本語 \LaTeX では「 \LaTeX の表記」は欧文扱い、Unicode 文字は和文扱いとして使い分ける習慣があるので、このような置換が行われるのは好ましくない。

これらの置換のうち、後の 4 つは Pandoc の `--no-tex-ligatures` オプションを指定すれば抑止できるが、「…」の置換を抑止する機能はないようである。そこで、「`\ldots` を『…』に戻す」という処置を行う。

`\pandocLdots` Pandoc 用の `\ldots` の実装。非数式である場合は代わりに … を実行する。

※以前は「Pandoc が必ず `\ldots{}` の形で書き出す」ことを利用して後続に `{}` があるかで「元が … であるか」を判断していた。ところが、Pandoc 2.7 版で `{}` を必ずしも付けなくなったため、1.9f 版で非数式の `\ldots` を全て … に戻す動作に変更した。

```
5729 \DeclareRobustCommand{\pandocLdots}{%
5730   \let\bxjsp@do\bxjsp@ja@ellipsis
5731   \ifmmode \let\bxjsp@do\bxjsp@org@ldots
5732   \else\ifbxjsp@babel@used
5733     \expandafter\ifx\csname bxjsp@ld/\language\endcsname\relax
5734     \let\bxjsp@do\bxjsp@org@ldots \fi
5735   \fi\fi \bxjsp@do}
5736 \@namedef{bxjsp@ld/japanese}{1}
5737 \def\bxjsp@ja@ellipsis{…}
5738 \let\bxjsp@org@ldots\ldots
```

`\ldots` の実装を `\pandocLdots` に置き換える。

```
5739 \g@addto@macro\bxjsp@begin@document@hook{%
5740   \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。

```
5741 \long\def\bxjsp@tmpa{\ldots}%
5742 \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。

```
5743   \ifnum0\ifbxjsp@english\ifbxjsp@babel@used\else1\fi\fi=0
5744   \let\ldots\pandocLdots
5745   \fi
5746 \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_{\LaTeX} ・ $\text{Lua}\text{\LaTeX}$ 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。

```
5747 \chardef\bxjsp@cc@other=12
5748 \@onlypreamble\bxjsp@makeother@range
5749 \def\bxjsp@makeother@range#1#2{%
5750   \@tempcnta"#1\relax \@tempcntb"#2\relax
5751   \loop\ifnum\@tempcnta<\@tempcntb
5752     \catcode\@tempcnta\bxjsp@cc@other
5753     \advance\@tempcnta\@ne
5754   \repeat}
5755 \ifnum0\if x\bxjsp@engine1\fi\if 1\bxjsp@engine1\fi>0
5756   \catcode"1F23B=\bxjsp@cc@other
5757   \bxjsp@makeother@range{9FCD}{A000}
5758   \bxjsp@makeother@range{1B002}{1B170}
5759   \bxjsp@makeother@range{2B820}{2EBF0}
5760 \fi
```

1.8 PandoLa モジュール

インストール済であれば読み込む。

```
5761 \IfFileExists{bxpandola.sty}{%
5762   \RequirePackage{bxpandola}\relax
5763   \PackageInfo{bxjsp@pkgname
5764     {PandoLa module is loaded\@gobble}
5765 }{}
```

1.9 完了

おしまい。

```
5766 %</ancpandoc>
```

補助パッケージ実装はここまで。

```
5767 %</anc>
```