

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”)

v2.9f [2026/05/28]

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

目次

1	はじめに	4
2	オプション	14
3	和文フォントの変更	44
4	フォントサイズ	54
5	レイアウト	61
5.1	ページレイアウト	62
6	改ページ (日本語 TeX 開発コミュニティ版のみ)	76
7	ページスタイル	78
8	文書のマークアップ	81
8.1	表題	81
8.2	章・節	86
8.3	リスト環境	99
8.4	パラメータの設定	106
8.5	フロート	107
8.6	キャプション	109
9	フォントコマンド	110


10	相互参照	113
10.1	目次の類	113
10.2	参考文献	118
10.3	索引	120
10.4	脚注	121
11	段落の頭へのグルー挿入禁止	124
12	いろいろなロゴ	128
13	amsmath との衝突の回避	128
14	初期設定	129
15	実験的コード	133
16	BXJS 独自の追加処理 	134
付録 A	和文ドライバの仕様 	139
付録 B	和文ドライバ：minimal 	140
B.1	準備	140
B.2	(u)pTeX 用の設定	143
B.3	pdfTeX 用の処理	148
B.4	X _g TeX 用の処理	148
B.5	後処理（エンジン共通）	149
付録 C	和文ドライバ：standard 	152
C.1	準備	152
C.2	和文ドライバパラメタ	153
C.3	共通処理 (1)	153
C.4	pTeX 用設定	160
C.5	pdfTeX 用設定：CJK + bxcjkatype	164
C.6	X _g TeX 用設定：xeCJK + zxjatype	166
C.7	LuaTeX 用設定：LuaTeX-ja	169
C.8	共通処理 (2)	173
付録 D	和文ドライバ：modern 	173
D.1	フォント設定	173
D.2	fixltx2e 読込	174
D.3	和文カテゴリコード	174
D.4	完了	174
付録 E	和文ドライバ：pandoc 	174

E.1	準備	175
E.2	和文ドライバパラメタ	176
E.3	dupload システム	177
E.4	lang 変数	178
E.5	geometry 変数	182
E.6	CJKmainfont 変数	182
E.7	Option clash 対策	182
E.8	レイアウト上書き禁止	182
E.9	paragraph のマーク	183
E.10	全角空白文字	184
E.11	hyperref 対策	184
E.12	Pandoc 要素に対する和文用の補正	185
E.13	ifPDFTeX スイッチ	186
E.14	完了	187
付録 F 補助パッケージ一覧		187
付録 G 補助パッケージ：bxjscompat		187
G.1	準備	188
G.2	8bit 欧文 TeX	188
G.3	X _g TeX	188
G.4	LuaTeX	190
G.5	完了	191
付録 H 補助パッケージ：bxjscjcat		191
H.1	準備	191
H.2	和文カテゴリコードの設定	192
H.3	ギリシャ・キリル文字の扱い	193
H.4	初期設定	200
H.5	完了	200
付録 I 補助パッケージ：bxjspandoc		200
I.1	準備	201
I.2	パッケージオプション	201
I.3	パッケージ読込の阻止	201
I.4	fixltx2e パッケージ	202
I.5	cmap パッケージ	202
I.6	microtype パッケージ	202
I.7	Unicode 文字変換対策	203
I.8	PandoLa モジュール	204
I.9	完了	204

1 はじめに

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

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

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

インストール時のモジュール指定は以下のものが用意されている。

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

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

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

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

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

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

[2009-02-22] 田中琢爾氏による $\mathrm{upIAT}_{\mathrm{E}}\mathrm{X}$ 対応パッチを取り込みました。

ここでは次のドキュメントクラス（スタイルファイル）を作ります。

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

<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、p \LaTeX 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 p \LaTeX が次第に対策コードをカーネル内に取り込むようになりました。そこで、新しい p \LaTeX カーネルと衝突しないように、日付が古い場合だけパッチをあてる場合があります。この処理に使用するフラグを定義します。

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

■環境検査

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

```

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

```

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

```

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

```

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

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

```

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

```

現状での処理系バージョン要件は以下の通りである。

- X_YT_EX： 0.997 版 (2007 年) 以上

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

3.0 版での **クラス本体** の処理系バージョン要件は以下の通りである。

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

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

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

T_EX 処理系のバージョンがサポート対象であるかを検査する。

```

41 \@tempwattrue

```

```

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

```

非サポートのバージョン場合は強制終了させる。

```

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

```

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

```

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

```

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

```

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

```

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

```

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

```

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

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

```

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

```

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

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

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

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

※ XeTeX と LuaTeX は ϵ -TeX 拡張をもつ版のみがあり、NTT JTeX はもたない版のみがある。その他のエンジンは両方の版がある。

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

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

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

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

※ 2020/02/02 版以降のカーネルには組み込まれている。

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

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

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

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

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

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

※ 2017/01/01 以降の L^AT_EX カーネルにおいて「Unicode を表す L^AT_EX 公式のフォントエンコーディング」である “TU” が導入され、これ以降の L^AT_EX を 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。

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

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

※カーネルの 2020/10/01 版で導入された。

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

`\bxjs@nclong` *無しの `\newcommand` で定義した引数無しマクロが `\long` 付になるならば `\long`、`\long` 無になるなら `\relax` と等価になる。

※カーネルの 2025/11/01 版において当該のマクロ定義が `\long` 無になるように仕様変更された。

```
94 \newcommand\bxjs@nclong{}\long\def\bxjs@tmpa{}
95 \ifx\bxjs@nclong\bxjs@tmpa \let\bxjs@nclong\long
96 \else \let\bxjs@nclong\relax
97 \fi
```

■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。

```
98 \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。

```
99 \RequirePackage{keyval}
```

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

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

```
100 \IfFileExists{iftex.sty}{%
101   \RequirePackage{iftex}
102 }{}
103 \begingroup\expandafter\endgroup
104 \expandafter\ifx\csname ifpdf\endcsname\@undefined\else
105   \expandafter\let\csname ifjsInPdfMode\expandafter\endcsname
106     \csname ifpdf\endcsname
107 \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。)

```
108 \def\jsAtEndOfClass{%
109   \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。

```
110 \IfFileExists{bxjscompat.sty}{%
111   \RequirePackage{bxjscompat}%
```

```
112 }{}

```

■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。

```
113 \ifx l\jsEngine
114   \directlua{ bxjs = {} }
115 \fi

```

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

```
116 \ifjsWithTeX \let\bxjs@protected\protected
117 \else \let\bxjs@protected\@empty
118 \fi

```

`\bxjs@robust@def` 無引数の頑強な命令を定義する。

```
119 \ifjsWithTeX
120   \def\bxjs@robust@def{\protected\def}
121 \else
122   \def\bxjs@robust@def{\DeclareRobustCommand*}
123 \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。

```
124 \bxjs@if@format@at@least{2021/11/15}{%
125   \def\bxjs@CGHN#1{#1}%
126 }{%else
127   \def\bxjs@CGHN#1{\bxjs@CGHN@a#1//}%
128   \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。

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

```

TODO:2.9 `\bxjs@expanded` を定義する。

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

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

```

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

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

```

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

```
137 \def\bxjs@catopt#1#2{%
138   #1\if\relax#1\relax\else\if\relax#2\relax\else,\fi\fi#2}
```

`\bxjs@ifplus \@ifstar` の + 版。

```
139 \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。

```
140 \def\bxjs@trim#1{\expandafter\bxjs@trim@a#1\@nil#1}
141 \def\bxjs@trim@a{\futurelet\bxjs@tmpb\bxjs@trim@b}
142 \def\bxjs@trim@b{\bxjs@cond\ifx\bxjs@tmpb\@sptoken\fi
143   {\bxjs@trim@c\bxjs@trim@a}{\bxjs@trim@d}}
144 \def\bxjs@trim@c#1 {#1}
145 \def\bxjs@trim@d#1\@nil{\bxjs@trim@e#1\@nil: \@nil\@nnil}
146 \def\bxjs@trim@e#1 \@nil#2\@nnil{\bxjs@cond\ifx\@nnil#2\@nnil\fi
147   {\bxjs@trim@f#1\@nnil}{\bxjs@trim@e#1\@nil: \@nil\@nnil}}
148 \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。

```
149 \def\bxjs@set@array@from@clist#1#2{%
150   \@tempcnta\z@
151   \@for\bxjs@tmpa:=\@empty#2\do{%
152     \bxjs@trim\bxjs@tmpa \bxjs@cslet{#1/\the\@tempcnta}\bxjs@tmpa
153     \advance\@tempcnta\@ne}
154   \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。

```
155 \let\c@bxjs@tempcnta\@tempcnta
156 \def\bxjs@gset@tempcnta{\setcounter{bxjs@tempcnta}}
```

`\bxjs@advance@qc \bxjs@advance@qc\CS{〈値〉}`： 擬似整数レジスタに値を加算する。

```
157 \def\bxjs@advance@qc#1#2{%
158   \begingroup
159     \@tempcnta=#1\relax \advance\@tempcnta by#2\relax
160     \global\chardef\bxjs@g@tmpa\@tempcnta
161   \endgroup \let#1\bxjs@g@tmpa}
```

`\bxjs@new@count` ε -TeX 拡張が有効なら通常の整数レジスタ、無効なら擬似整数レジスタを用いる。

```
\bxjs@advance@count 162 \ifjsWithTeX
163   \let\bxjs@new@count\newcount
164   \def\bxjs@advance@count#1#2{\advance#1by#2\relax}
165 \else
166   \def\bxjs@new@count#1{\chardef#1\z@}
167   \let\bxjs@advance@count\bxjs@advance@qc
168 \fi
```

`\bxjs@add@class@option` グローバルオプションを追加する。

※ 2021/06/01 版以降の L^AT_EX カーネルでは従来の `\@classoptionslist` に加えて「無加工のクラスオプションリスト」を格納するための `\@raw@classoptionslist` が用意されているので、その場合はそのリストにも追加する。

```
169 \def\bxjs@add@class@option#1{%
170   \edef\bxjs@args{{#1}}%
171   \expandafter\bxjs@add@class@option@a\bxjs@args}
172 \def\bxjs@add@class@option@a#1{%
173   \g@addto@macro\@classoptionslist{,#1}%
174   \ifx\@raw@classoptionslist\undefined\else
175     \g@addto@macro\@raw@classoptionslist{,#1}%
176   \fi}
```

`\jsSetQHLLength` `\jsSetQHLLength\CS{〈長さ式〉}`: `\setlength` の変種で、通常の `calc` の長さ式の代わりに、「 $Q/H/\text{true}Q/\text{true}H/\text{zw}/\text{zh}$ 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）。

```
177 \def\jsSetQHLLength#1#2{%
178   \begingroup
179   \bxjs@parse@qh{#2}%
180   \ifx\bxjs@tmpb\relax
181     \setlength\@tempdima{#2}%
182     \xdef\bxjs@g@tmpa{\the\@tempdima}%
183   \else \global\let\bxjs@g@tmpa\bxjs@tmpb
184   \fi
185   \endgroup
186   #1=\bxjs@g@tmpa\relax}
```

`\bxjs@parse@qh` #1 が $Q/H/\text{true}Q/\text{true}H/\text{zw}/\text{zh}$ 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。

```
187 \if j\jsEngine \def\bxjs@parse@qh@units{zw,zh}
188 \else \def\bxjs@parse@qh@units{trueQ,trueH,Q,H,zw,zh}
189 \fi
190 \def\bxjs@parse@qh#1{%
191   \let\bxjs@tmpb\relax
192   \@for\bxjs@tmpa:=\bxjs@parse@qh@units\do{%
193     \ifx\bxjs@tmpb\relax
194       \edef\bxjs@next{{\bxjs@tmpa}{#1}}%
195       \expandafter\bxjs@parse@qh@a\c@name \bxjs@unit@\bxjs@tmpa\expandafter
196       \endc@name\bxjs@next
197     \fi}}
198 \def\bxjs@parse@qh@a#1#2#3{%
199   \def\bxjs@next##1#2\@nil##2\@nnil{\bxjs@parse@qh@b{##1}{##2}#1}%
200   \bxjs@next#3\@nil#2\@nil\@nnil}
201 \def\bxjs@parse@qh@b#1#2#3{%
```

```

202 \ifx\@nnil#2\@nnil\else
203 \ifx#3\relax
204 \ClassError\bxjs@clsname
205 {You cannot use '\bxjs@tmpa' here}{\@ehc}%
206 \def\bxjs@tmpb{0pt}%
207 \else
208 \@tempdimb#3\relax \@tempdimb#1\@tempdimb
209 \edef\bxjs@tmpb{\the\@tempdimb}%
210 \fi
211 \fi}

```

今の段階では Q/H だけが使用可能。

```

212 \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。

```

213 \newif\ifbxjs@after@preamble

```

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

```

214 \@onlypreamble\bxjs@begin@document@hook
215 \def\bxjs@begin@document@hook{\bxjs@after@preambletrue}
216 \AtBeginDocument{\bxjs@begin@document@hook}

```

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

```

217 \@onlypreamble\bxjs@post@option@hook
218 \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。

```

219 \@onlypreamble\bxjs@pre@jadriver@hook
220 \let\bxjs@pre@jadriver@hook\@empty

```

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

※\AtEndPreamble が利用できない場合は無効になる。

```

221 \@onlypreamble\bxjs@endpreamble@hook
222 \let\bxjs@endpreamble@hook\@empty
223 \AtEndOfClass{%
224 \ifx\AtEndPreamble\@undefined\else
225 \AtEndPreamble{\bxjs@endpreamble@hook}%
226 \fi}

```

一時的な手続き用の制御綴。

```

227 \@onlypreamble\bxjs@tmpdo
228 \@onlypreamble\bxjs@tmpdo@a
229 \@onlypreamble\bxjs@tmpdo@b
230 \@onlypreamble\bxjs@tmpdo@c
231 \@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。

```

232 \bxjs@robust@def\jsInhibitGlue{%
233 \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。

```
234 \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。

```
235 \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。

```
236 %<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。

```
237 %<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。

```
238 %<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。

```
239 \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。

```

240 \onlypreamble\bxjs@setpaper
241 \def\bxjs@setpaper#1{\def\bxjs@param@paper{#1}}
242 \newif\ifbxjs@iso@bsize
243 \DeclareOption{iso-bsize}{\bxjs@iso@bsizetrue}
244 \onlypreamble\bxjs@setpaper@bsize
245 \def\bxjs@setpaper@bsize#1{\edef\bxjs@param@paper{%
246   b#1\ifbxjs@iso@bsize paper\else j\fi}}
247 \DeclareOption{a3paper}{\bxjs@setpaper{a3paper}}
248 \DeclareOption{a4paper}{\bxjs@setpaper{a4paper}}
249 \DeclareOption{a5paper}{\bxjs@setpaper{a5paper}}
250 \DeclareOption{a6paper}{\bxjs@setpaper{a6paper}}
251 \DeclareOption{b4paper}{\bxjs@setpaper@bsize{4}}
252 \DeclareOption{b5paper}{\bxjs@setpaper@bsize{5}}
253 \DeclareOption{b6paper}{\bxjs@setpaper@bsize{6}}
254 \DeclareOption{a4j}{\bxjs@setpaper{a4paper}}
255 \DeclareOption{a5j}{\bxjs@setpaper{a5paper}}
256 \DeclareOption{b4j}{\bxjs@setpaper{b4j}}
257 \DeclareOption{b5j}{\bxjs@setpaper{b5j}}
258 \DeclareOption{a4var}{\bxjs@setpaper{{210truemm}{283truemm}}}}
259 \DeclareOption{b5var}{\bxjs@setpaper{{182truemm}{230truemm}}}}
260 \DeclareOption{letterpaper}{\bxjs@setpaper{letterpaper}}
261 \DeclareOption{legalpaper}{\bxjs@setpaper{legalpaper}}
262 \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。

```

263 \@for\bxjs@tmpa:={%
264   a0,a1,a2,c0,c1,c2,c3,c4,c5,c6,ansia,ansib,ansic,ansid,ansie%
265 }\do{\edef\bxjs@next{%
266   \noexpand\DeclareOption{\bxjs@tmpa paper}%
267   {\noexpand\bxjs@setpaper{\bxjs@tmpa paper}}}%
268 }\bxjs@next}
269 \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。

```

270 \@for\bxjs@tmpa:={0,1,2,3}\do{\edef\bxjs@next{%
271   \noexpand\DeclareOption{b\bxjs@tmpa paper}%
272   {\noexpand\bxjs@setpaper@bsize{\bxjs@tmpa}}}%
273 }\bxjs@next}

```

Pandoc で用紙サイズを指定した場合は出力 \LaTeX 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。

```

274 \DeclareOption{a4varpaper}{\bxjs@setpaper{{210truemm}{283truemm}}}}
275 \DeclareOption{b5varpaper}{\bxjs@setpaper{{182truemm}{230truemm}}}}
276 \DeclareOption{screenpaper}{\bxjs@setpaper{screen}}

```

```
277 \DeclareOption{custompaper}{}  


---


```

■横置き 用紙の縦と横の長さを入れ換えます。

```
278 \newif\if@landscape  
279 \@landscapefalse  
280 \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。

```
281 \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。

```
282 %<!slide>\@slidefalse  
283 %<slide>\@slidetrue
```

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

[2003-03-22] 14Q オプションを追加しました。

[2003-04-18] 12Q オプションを追加しました。

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

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

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

```
284 \newcommand{\@ptsize}{0}  
285 \def\bxjs@param@basefontsize{10pt}  
286 \def\jsUnusualPtSize{-20}
```

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

```
287 \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。

```

288 \jsSetQHLlength\@tempdima{#1}%
289 \edef\bxjs@param@basefontsize{\the\@tempdima}%
290 \ifdim\@tempdima=10pt \renewcommand\@ptsize{0}%
291 \else\ifdim\@tempdima=10.95pt \renewcommand\@ptsize{1}%
292 \else\ifdim\@tempdima=12pt \renewcommand\@ptsize{2}%
293 \else \bxjs@nclong\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。

```

294 \def\bxjs@setjbasefontsize#1{%
295 \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。

```

296 \newif\ifjsc@mag
297 \newif\ifjsc@mag@xreal
298 %\let\jsc@magscale\undefined

299 \DeclareOption{8pt}{\bxjs@setbasefontsize{8pt}}
300 \DeclareOption{9pt}{\bxjs@setbasefontsize{9pt}}
301 \DeclareOption{10pt}{\bxjs@setbasefontsize{10pt}}
302 \DeclareOption{11pt}{\bxjs@setbasefontsize{10.95pt}}
303 \DeclareOption{12pt}{\bxjs@setbasefontsize{12pt}}
304 \DeclareOption{14pt}{\bxjs@setbasefontsize{14.4pt}}
305 \DeclareOption{17pt}{\bxjs@setbasefontsize{17.28pt}}
306 \DeclareOption{20pt}{\bxjs@setbasefontsize{20pt}}
307 \DeclareOption{21pt}{\bxjs@setbasefontsize{20.74pt}}
308 \DeclareOption{25pt}{\bxjs@setbasefontsize{24.88pt}}
309 \DeclareOption{30pt}{\bxjs@setbasefontsize{29.86pt}}
310 \DeclareOption{36pt}{\bxjs@setbasefontsize{35.83pt}}
311 \DeclareOption{43pt}{\bxjs@setbasefontsize{43pt}}
312 \DeclareOption{12Q}{\bxjs@setjbasefontsize{3mm}}
313 \DeclareOption{14Q}{\bxjs@setjbasefontsize{3.5mm}}
314 \DeclareOption{10ptj}{\bxjs@setjbasefontsize{10pt}}
315 \DeclareOption{10.5ptj}{\bxjs@setjbasefontsize{10.5pt}}
316 \DeclareOption{11ptj}{\bxjs@setjbasefontsize{11pt}}
317 \DeclareOption{12ptj}{\bxjs@setjbasefontsize{12pt}}

```

JS クラス互換の magstyle 設定オプション。

```

318 \DeclareOption{usemag}{\let\bxjs@magstyle\bxjs@magstyle@@usemag}
319 \DeclareOption{nomag}{\let\bxjs@magstyle\bxjs@magstyle@@nomag}
320 \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 $\text{p\LaTeX} 2_{\epsilon}$ 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。

```
321 \if j\jsEngine
322 \hour\time \divide\hour by 60\relax
323 \@tempcnta\hour \multiply\@tempcnta 60\relax
324 \minute\time \advance\minute-\@tempcnta
325 \DeclareOption{tombow}{%
326   \tombowtrue \tombowdatetrue
327   \setlength{\@tombowwidth}{.1\p@}%
328   \@bannertoken{%
329     \jobname\space(\number\year-\two@digits\month-\two@digits\day
330     \space\two@digits\hour:\two@digits\minute)}%
331   \maketombowbox}
332 \DeclareOption{tombo}{%
333   \tombowtrue \tombowdatefalse
334   \setlength{\@tombowwidth}{.1\p@}%
335   \maketombowbox}
336 \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。

```
337 \if j\jsEngine
338 \DeclareOption{mentuke}{%
339   \tombowtrue \tombowdatefalse
340   \setlength{\@tombowwidth}{\z@}%
341   \maketombowbox}
342 \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。

```
343 \DeclareOption{oneside}{\@twosidefalse \@mparswitchfalse}
344 \DeclareOption{twoside}{\@twosidetrue \@mparswitchtrue}
345 \DeclareOption{vartwoside}{\@twosidetrue \@mparswitchfalse}
```

■二段組 `twocolumn` で二段組になります。

```
346 \DeclareOption{onecolumn}{\@twocolumnfalse}
347 \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。

```
348 \DeclareOption{titlepage}{\@titlepagetrue}
349 \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。

```
350 %<book|report>\DeclareOption{openright}{\@openrighttrue\@openleftfalse}
351 %<book|report>\DeclareOption{openleft}{\@openlefttrue\@openrightfalse}
352 %<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，追従します。

[2025-05-26] \LaTeX 2_ε 2025-06-01 に追従し，最初の `$$` を `\dollar\dollar@begin` に変更しました。 `\providecommand` で `\dollar\dollar@begin` を定義しているので，古い \LaTeX 2_ε カーネルでも問題ありません。

```
353 \providecommand\dollar\dollar@begin{$$}
354 \def\eqnarray{%
355   \stepcounter{equation}%
356   \def\@currentlabel{\p@equation\theequation}%
357   \def\@currentcounter{equation}%
358   \global\@eqnswtrue
359   \m@th
360   \global\@eqcnt\z@
361   \tabskip\@centering
362   \let\\\@eqncr
363   \dollar\dollar@begin\everycr{}\halign to\displaywidth\bgroup
364     \hskip\@centering$\displaystyle\tabskip\z@skip{##}$\@eqnse1
365     &\global\@eqcnt\@ne \hfil$\displaystyle{\{##\}}$\hfil
366     &\global\@eqcnt\tw@ $\displaystyle{##}$\hfil\tabskip\@centering
367     &\global\@eqcnt\thr@@ \hb@xt@\z@\bgroup\hss##\egroup
368     \tabskip\z@skip
369     \cr}
```

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

[2022-09-13] \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，追従します。

```
370 \DeclareOption{leqno}{\input{leqno.clo}}
```

```

371 \DeclareOption{fleqn}{\input{fleqn.clo}}
372 % fleqn用のeqnarray環境の再定義
373 \def\eqnarray{%
374   \stepcounter{equation}%
375   \def\@currentlabel{\p@equation\theequation}%
376   \def\@currentcounter{equation}%
377   \global\@eqnswtrue\m@th
378   \global\@eqcnt\z@
379   \tabskip\mathindent
380   \let\@=\@eqnocr
381   \setlength\abovedisplayskip{\topsep}%
382   \ifvmode
383     \addtolength\abovedisplayskip{\partopsep}%
384   \fi
385   \addtolength\abovedisplayskip{\parskip}%
386   \setlength\belowdisplayskip{\abovedisplayskip}%
387   \setlength\belowdisplayshortskip{\abovedisplayskip}%
388   \setlength\abovedisplayshortskip{\abovedisplayskip}%
389   \dollar\dollar@begin\everycr{}\halign to\linewidth% $$
390   \bgroup
391     \hskip\@centering$\displaystyle\tabskip\z@skip{##}$\@eqnsele
392     &\global\@eqcnt\@ne \hfil$\displaystyle{##}$\hfil
393     &\global\@eqcnt\tw@
394     $\displaystyle{##}$\hfil \tabskip\@centering
395     &\global\@eqcnt\thr@@ \hb@xt@\z@\bgroup\hss##\egroup
396   \tabskip\z@skip\cr
397   }}

```

■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。

```

398 % \DeclareOption{openbib}{%
399 %   \AtEndOfPackage{%
400 %     \renewcommand\@openbib@code{%
401 %       \advance\leftmargin\bibindent
402 %       \itemindent -\bibindent
403 %       \listparindent \itemindent
404 %       \parsep \z@}%
405 %     \renewcommand\newblock{\par}}

```

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

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

```
406 %\let\bxjs@enablejfam\@undefined
```

`enablejfam` オプションの処理。

```
407 \def\bxjs@kv@enablejfam@true{\let\bxjs@enablejfam=t}
408 \def\bxjs@kv@enablejfam@false{\let\bxjs@enablejfam=f}
409 \def\bxjs@kv@enablejfam@default{\let\bxjs@enablejfam\@undefined}
410 \define@key{bxjs}{enablejfam}[true]{%
411   \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。

```
412 \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。

```
413 \newif\ifjsDraft
414 \DeclareOption{draft}{\jsDrafttrue \overfullrule=5pt }
415 \DeclareOption{final}{\jsDraftfalse \overfullrule=0pt }
```

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

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

BXJS クラスではここは和文ドライバの管轄。

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

BXJS クラスでは `geometry` パッケージがこの処理を行う。

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

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

```
416 \newif\ifbxjs@papersize
417 \bxjs@papersizetrue
418 \DeclareOption{nopapersize}{\bxjs@papersizefalse}
419 \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。

```
420 \newif\if@english
421 \@englishfalse
422 \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 \LaTeX 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。

```
423 \newif\if@jslogo \@jslogofalse
424 \DeclareOption{jslogo}{\@jslogotrue}
425 \DeclareOption{nojslogo}{\@jslogofalse}
```

■複合設定オプション 

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

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

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

※元の長さが 128 pt 以上の場合でも動作するように修正した。

```
426 \mathchardef\bxjs@isc@ll=128
427 \mathchardef\bxjs@isc@sl=259
428 \def\bxjs@isc@sl@h{65539 }
429 \def\bxjs@invscale#1#2{%
430   \begingroup \@tempdima=#1\relax \@tempdimb#2\p@\relax
431   \ifdim\@tempdima<\bxjs@isc@ll\p@
432     \@tempcnta\@tempdima \multiply\@tempcnta\@cclvi
433     \divide\@tempcnta\@tempdimb \multiply\@tempcnta\@cclvi
434   \else
435     \@tempcnta\@tempdima \divide\@tempcnta\@tempdimb
436     \multiply\@tempcnta\p@ \let\bxjs@isc@sl\bxjs@isc@sl@h
437   \fi
438   \@tempcntb\p@ \divide\@tempcntb\@tempdimb
439   \advance\@tempcnta-\@tempcntb \advance\@tempcnta-\tw@
440   \@tempdimb\@tempcnta\@ne
441   \advance\@tempcnta\@tempcntb \advance\@tempcnta\@tempcntb
442   \advance\@tempcnta\bxjs@isc@sl \@tempdimc\@tempcnta\@ne
443   \@whiledim\@tempdimb<\@tempdimc\do{%
444     \@tempcntb\@tempdimb \advance\@tempcntb\@tempdimc
445     \advance\@tempcntb\@ne \divide\@tempcntb\tw@
446   \ifdim #2\@tempcntb>\@tempdima
447     \advance\@tempcntb\m@ne \@tempdimc=\@tempcntb\@ne
448   \else \@tempdimb=\@tempcntb\@ne \fi}%
449   \xdef\bxjs@gtmpa{\the\@tempdimb}%
450 \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 \LaTeX 用の既定テンプレートを用いて他形式から \LaTeX (および PDF) 形式に変換する用途に最適化した設定を与える。

```
451 \DeclareOption{pandoc}{%
452   \bxjs@apply@pandoc@opt}
453 \@onlypreamble\bxjs@apply@pandoc@opt
454 \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。

```
455 \g@addto@macro\bxjs@post@option@hook{%
456   \bxjs@oldfontcommandstrue
457   \setkeys{bxjs}{ja=pandoc}%
458   \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)。

```
459 \ifx\bxjs@driver@opt\undefined
```

```

460 \def\bxjs@driver@opt{dvipdfmx}%
461 \bxjs@dvi@opttrue
462 \fi
463 \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。

```

464 \DeclareOption{pandoc+}{%
465 \g@addto@macro\bxjs@post@option@hook{%
466 \edef\jsJaParam{\bxjs@catopt{_plus}\jsJaParam}}%
467 \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。

```
468 %\let\bxjs@engine@given\@undefined
```

`\bxjs@engine@opt` 明示されたエンジンのオプション名。

```
469 %\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。

```

470 \DeclareOption{autodetect-engine}{%
471 \let\bxjs@engine@given=*}
472 \DeclareOption{latex}{%
473 \def\bxjs@engine@opt{latex}%
474 \let\bxjs@engine@given=n}
475 \DeclareOption{platex}{%
476 \def\bxjs@engine@opt{platex}%
477 \let\bxjs@engine@given=j}
478 \DeclareOption{uplatex}{%
479 \def\bxjs@engine@opt{uplatex}%
480 \let\bxjs@engine@given=u}
481 \DeclareOption{xelatex}{%
482 \def\bxjs@engine@opt{xelatex}%
483 \let\bxjs@engine@given=x}
484 \DeclareOption{pdfplatex}{%
485 \def\bxjs@engine@opt{pdfplatex}%
486 \let\bxjs@engine@given=p}
487 \DeclareOption{lualatex}{%
488 \def\bxjs@engine@opt{lualatex}%
489 \let\bxjs@engine@given=l}
490 \DeclareOption{platex-ng}{%
491 \def\bxjs@engine@opt{platex-ng}%
492 \let\bxjs@engine@given=g}
493 \DeclareOption{platex-ng*}{%

```



```

494 \def\bxjs@engine@opt{platex-ng*}%
495 \let\bxjs@platexng@nodrv=t%
496 \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。

```

497 %\let\bxjs@driver@given\@undefined
498 \let\bxjs@driver@@dvimode=0
499 \let\bxjs@driver@@dvipdfmx=1
500 \let\bxjs@driver@@pdfmode=2
501 \let\bxjs@driver@@xetex=3
502 \let\bxjs@driver@@dvips=4
503 \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。

```

504 %\let\bxjs@driver@opt\@undefined

```

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

```

505 \DeclareOption{dvips}{%
506   \def\bxjs@driver@opt{dvips}%
507   \let\bxjs@driver@given\bxjs@driver@@dvips}
508 \DeclareOption{dviout}{%
509   \def\bxjs@driver@opt{dviout}%
510   \let\bxjs@driver@given\bxjs@driver@@dvimode}
511 \DeclareOption{xdvi}{%
512   \def\bxjs@driver@opt{xdvi}%
513   \let\bxjs@driver@given\bxjs@driver@@dvimode}
514 \DeclareOption{dvipdfmx}{%
515   \def\bxjs@driver@opt{dvipdfmx}%
516   \let\bxjs@driver@given\bxjs@driver@@dvipdfmx}
517 \DeclareOption{nodvidriver}{%
518   \def\bxjs@driver@opt{nodvidriver}%
519   \let\bxjs@driver@given\bxjs@driver@@none}
520 \DeclareOption{class-nodvidriver}{%
521   \def\bxjs@driver@opt{class-nodvidriver}%
522   \let\bxjs@driver@given\bxjs@driver@@none}
523 \DeclareOption{pdftex}{%
524   \def\bxjs@driver@opt{pdftex}%
525   \let\bxjs@driver@given\bxjs@driver@@pdfmode}
526 \DeclareOption{luatex}{%
527   \def\bxjs@driver@opt{luatex}%
528   \let\bxjs@driver@given\bxjs@driver@@pdfmode}
529 \DeclareOption{xetex}{%
530   \def\bxjs@driver@opt{xetex}%
531   \let\bxjs@driver@given\bxjs@driver@@xetex}

```

`dvipdfmx-if-dvi` は 2.0 版より非推奨となった。

```

532 \DeclareOption{dvipdfmx-if-dvi}{\bxjs@depreopt@do{dvipdfmx-if-dvi}{dvi=dvipdfmx}}

```

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

`\bxjs@depre@opt` 非推奨のオプションについて警告を出す。

```
\bxjs@depre@opt@do 533 \@onlypreamble\bxjs@depre@opt
534 \def\bxjs@depre@opt#1#2{%
535   \ClassWarningNoLine\bxjs@clsname
536   {The old option '#1' is DEPRECATED\MessageBreak
537     and may be abolished in future!\MessageBreak
538     You should instead write:\MessageBreak
539     \space\space #2}}
540 \@onlypreamble\bxjs@depre@opt@do
541 \def\bxjs@depre@opt@do#1#2{%
542   \bxjs@depre@opt{#1}{#2}%
543   \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。

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

`nobigcode` / `bigcode` オプションの定義。

```
545 \DeclareOption{nobigcode}{%
546   \bxjs@bigcodefalse}
547 \DeclareOption{bigcode}{%
548   \bxjs@bigcodetrue}
```

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

```
549 \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。

```
550 \DeclareOption{nooldfontcommands}{%
551   \bxjs@oldfontcommandsfalse}
552 \DeclareOption{oldfontcommands}{%
553   \bxjs@oldfontcommandstrue}
```

■無効および廃止されたオプション 🐛

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

```
554 \ifbxjs@brace@safe
555   \@onlypreamble\bxjs@register@badopt
556   \def\bxjs@register@badopt#1{%
557     \expandafter\onlypreamble\csname bxjs@badopt/#1\endcsname
```

```

558 \namedef{bxjs@badopt/#1}{
559 \g@addto@macro{bxjs@begin@document@hook}{%
560 \for{bxjs@tmpa:=\@unusedoptionlist\do{%
561 \nameuse{bxjs@badopt/\bxjs@tmpa}}}}
562 \fi

```

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

```

563 \onlypreamble{bxjs@invalid@opt
564 \ifbxjs@brace@safe
565 \def{bxjs@invalid@opt#1#2}{%
566 \bxjs@register@badopt{#1}{\ClassError{bxjs@clsname{#2}}\@ehc}}
567 \else
568 \def{bxjs@invalid@opt#1#2}{%
569 \DeclareOption{#1}{\ClassWarningNoLine{bxjs@clsname{#2}}}}
570 \fi

```

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

※ `ltjsclasses` クラスでも警告を出している。

```

571 \bxjs@invalid@opt{winjis}{%
572 This class does not support 'winjis' option}
573 \bxjs@invalid@opt{mingoth}{%
574 This class does not support 'mingoth' option}
575 \bxjs@invalid@opt{jis}{%
576 This class does not support 'jis' option}
577 \if j{jsEngine}\else
578 \bxjs@invalid@opt{tombo}{%
579 Option 'tombo' can be used only on (u)pLaTeX}
580 \bxjs@invalid@opt{tombow}{%
581 Option 'tombow' can be used only on (u)pLaTeX}
582 \bxjs@invalid@opt{mentuke}{%
583 Option 'mentuke' can be used only on (u)pLaTeX}
584 \fi

```

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

```

585 \DeclareOption*{%
586 \bxjs@check@ja@prefix \ifx{bxjs@next}\relax
587 \def{bxjs@next}{\bxjs@cls@setkeys{bxjs}}}%
588 \expandafter{bxjs@next}\expandafter{\CurrentOption}%
589 \else

```

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

```

590 \edef{bxjs@next}{%
591 \noexpand\setkeys{bxjs}{japaram={\bxjs@next}}}%
592 }{bxjs@next
593 \fi}

```

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

```
594 \def\bxjs@check@ja@prefix{%
595   \let\bxjs@next\relax
596   \expandafter\bxjs@check@ja@prefix@a\CurrentOption\@nil ja:\@nil\@nnil}
597 \def\bxjs@check@ja@prefix@a#1ja:#2\@nil#3\@nnil{%
598   \ifx\@nil#1\@nil \def\bxjs@next{#2}\fi}
```

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

```
599 \def\bxjs@safe@setkeys#1#2{%
600   \let\bxjs@save@KV@errx\KV@errx \let\KV@errx\@gobble
601   \setkeys{#1}{#2}%
602   \let\KV@errx\bxjs@save@KV@errx}
```

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

```
603 \def\bxjs@cls@setkeys#1#2{%
604   \let\bxjs@save@KV@errx\KV@errx
605   \def\KV@errx##1{\OptionNotUsed}%
606   \setkeys{#1}{#2}%
607   \let\KV@errx\bxjs@save@KV@errx}
608 \ifbxjs@brace@safe\else
609   \let\bxjs@cls@setkeys\bxjs@safe@setkeys
610 \fi
```

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

```
611 \@onlypreamble\bxjs@declare@enum@option
612 \def\bxjs@declare@enum@option#1#2#3{%
613   \bxjs@csletcs{bxjs@#2}{bxjs@#2@@#3}%
614   \define@key{bxjs}{#1}{%
615     \@ifundefined{bxjs@#2@@##1}{%
616       \bxjs@error@keyval{#1}{##1}%
617     }{\bxjs@csletcs{bxjs@#2}{bxjs@#2@@##1}}}
```

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

```
618 \@onlypreamble\bxjs@declare@bool@option
619 \def\bxjs@declare@bool@option#1#2#3{%
620   \csname newif\expandafter\endcsname\csname ifbxjs@#2\endcsname
621   \@nameuse{bxjs@#2#3}%
622   \define@key{bxjs}{#1}[true]{%
623     \@ifundefined{bxjs@#2##1}{%
624       \bxjs@error@keyval{#1}{##1}%
625     }
```

```

625     }\@nameuse{bxjs@#2#1}}}}

\bxjs@set@keyval \bxjs@set@keyval{\<key>}{\<value>}{\<error>}{
    \bxjs@kv@{\<key>@{\<value>} が定義済ならそれを実行し、未定義ならエラーを出す。
626 \def\bxjs@set@keyval#1#2#3{%
627   \bxjs@csletcs{bxjs@next}{bxjs@kv@#1@#2}%
628   \ifx\bxjs@next\relax
629     \bxjs@error@keyval{#1}{#2}%
630     #3%
631   \else \bxjs@next
632   \fi}
633 \@onlypreamble\bxjs@error@keyval
634 \def\bxjs@error@keyval#1#2{%
635   \ClassError\bxjs@clsname
636   {Invalid value '#2' for option #1}\@ehc}

\jsScale [実数値マクロ] 和文スケール値。
637 \def\jsScale{0.924715}

\bxjs@base@opt 明示された base オプションの値。
638 %\let\bxjs@base@opt\@undefined

    base オプションの処理。
639 \define@key{bxjs}{base}{%
640   \edef\bxjs@base@opt{#1}%
641   \bxjs@setbasefontsize{#1}}
642 \define@key{bxjs}{fontsize}{\setkeys{bxjs}{base=#1}}

\bxjs@jbase@opt 明示された jbase オプションの値。
643 %\let\bxjs@jbase@opt\@undefined

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

\bxjs@scale@opt 明示された scale オプションの値。
646 %\let\bxjs@scale@opt\@undefined

    scale オプションの処理。
647 \define@key{bxjs}{scale}{%
648   \edef\bxjs@scale@opt{#1}%
649   \let\jsScale\bxjs@scale@opt}
650 \define@key{bxjs}{jafontscale}{\setkeys{bxjs}{scale=#1}}

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

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

```

mag オプションの処理。

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

paper オプションの処理。

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

`\bxjs@jadriver` 和文ドライバの名前。

```
655 \let\bxjs@jadriver\relax
```

`\bxjs@jadriver@opt` 明示された和文ドライバの名前。

```
656 %\let\bxjs@jadriver@opt\undefined
```

ja オプションの処理。

※ `jadriver` は 0.9 版で用いられた旧称。

TODO:3.0 `jadriver` は廃止の予定。

※単なる `ja` という指定は無視される (Pandoc 対策)。

```
657 \define@key{bxjs}{jadriver}{%
658   \bxjs@depre@opt{jadriver}{ja=#1}\edef\bxjs@jadriver@opt{#1}}
659 \define@key{bxjs}{ja}[\relax]{%
660   \ifx\relax#1\else\edef\bxjs@jadriver@opt{#1}\fi}
```

`\jsJaFont` 和文フォント設定の名前。

```
661 \let\jsJaFont\@empty
```

jafont オプションの処理。

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

`\jsJaParam` 和文ドライバパラメタの文字列。

```
663 \let\jsJaParam\@empty
```

japaram オプションの処理。

```
664 \define@key{bxjs}{japaram}{%
665   \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。

```
666 \define@key{bxjs}{pandoc}[]{%
667   \ExecuteOptions{pandoc}%
668   \edef\jsJaParam{\bxjs@catopt\jsJaParam{#1}}}%
669 \define@key{bxjs}{pandoc+}[]{%
670   \ExecuteOptions{pandoc+}%
671   \edef\jsJaParam{\bxjs@catopt\jsJaParam{#1}}}%
```

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

```
672 \let\bxjs@magstyle@mag=m
673 \let\bxjs@magstyle@real=r
674 \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。

```
675 \let\bxjs@magstyle@@usemag\bxjs@magstyle@@mag
676 \let\bxjs@magstyle@@nomag\bxjs@magstyle@@real
677 \bxjs@cslet\bxjs@magstyle@@nomag*\bxjs@magstyle@@xreal
```

`\bxjs@magstyle@@default` は既定の値を表す。

```
678 \let\bxjs@magstyle@@default\bxjs@magstyle@@usemag
679 \ifx l\jsEngine \ifnum\luatexversion>86
680   \let\bxjs@magstyle@@default\bxjs@magstyle@@xreal
681 \fi\fi
682 \ifjsWithpTeXng
683   \let\bxjs@magstyle@@default\bxjs@magstyle@@xreal
684 \fi
685 \let\bxjs@magstyle\bxjs@magstyle@@default
```

`magstyle` オプションの処理。

```
686 \define@key{bxjs}{magstyle}{%
687   \bxjs@csletcs{bxjs@magstyle}{bxjs@magstyle@@#1}%
688   \ifx\bxjs@magstyle\relax
689     \bxjs@error@keyval{magstyle}{#1}%
690     \let\bxjs@magstyle\bxjs@magstyle@@default
691   \fi}
```

`\bxjs@geometry geometry` オプションの指定値。

```
692 \let\bxjs@geometry@@class=c
693 \let\bxjs@geometry@@user=u
694 \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。

```
695 \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。

```
696 \newif\ifbxjs@dvi@opt
```

DVI モードのドライバとドライバ種別との対応。

```
697 \let\bxjs@dvidriver@@dvipdfmx=\bxjs@driver@@dvipdfmx
698 \let\bxjs@dvidriver@@dvips=\bxjs@driver@@dvips
699 \let\bxjs@dvidriver@@dviout=\bxjs@driver@@dvimode
700 \let\bxjs@dvidriver@@xdvi=\bxjs@driver@@dvimode
701 \let\bxjs@dvidriver@@nodvidriver=\bxjs@driver@@none
702 \bxjs@cslet\bxjs@dvidriver@@class-nodvidriver\bxjs@driver@@none
```

`dvi` オプションの処理。

```
703 \define@key{bxjs}{dvi}{%
704   \bxjs@csletcs{bxjs@tmpa}{bxjs@dvidriver@@#1}%
705   \ifx\bxjs@tmpa\relax
706     \bxjs@error@keyval{dvi}{#1}%
707   \else
```

`\bxjs@driver@given` を未定義にしていることに注意。

```
708 \def\bxjs@driver@opt{#1}%  
709 \let\bxjs@driver@given\@undefined  
710 \bxjs@dvi@opttrue  
711 \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。

```
712 \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。

```
713 \newif\ifbxjs@force@chapterabstract  
714 %<book>\bxjs@force@chapterabstracttrue
```

`layout` オプションの処理。

```
715 \@namedef{bxjs@kv@layout@v1}{%  
716 %<book>\bxjs@layout@buggyhmargintrue  
717 %<report>\bxjs@force@chapterabstracttrue  
718 }  
719 \@namedef{bxjs@kv@layout@v2}{%  
720 %<book>\bxjs@layout@buggyhmarginfalse  
721 %<report>\bxjs@force@chapterabstractfalse  
722 }  
723 \define@key{bxjs}{layout}{%  
724 \bxjs@set@keyval{layout}{#1}{}}
```

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

```
725 %\let\bxjs@textwidth@limit@opt\@undefined  
726 \define@key{bxjs}{textwidth-limit}{%  
727 \bxjs@depre@opt{textwidth-limit}{textwidth=#1zw}%  
728 \edef\bxjs@textwidth@limit@opt{#1}}
```

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

```
729 %\let\bxjs@textwidth@opt\@undefined  
730 \define@key{bxjs}{textwidth}{\edef\bxjs@textwidth@opt{#1}}  
731 \define@key{bxjs}{line_length}{\setkeys{bxjs}{textwidth=#1}}
```

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

```
732 %\let\bxjs@number@of@lines@opt\@undefined  
733 \define@key{bxjs}{number-of-lines}{\edef\bxjs@number@of@lines@opt{#1}}  
734 \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。

```
735 %\let\bxjs@paragraph@mark\@undefined  
736 \define@key{bxjs}{paragraph-mark}{%  
737 \edef\bxjs@paragraph@mark{#1}}
```


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

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

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

```
739 \bxjs@declare@bool@option{jaspace-cmd}{jaspace@cmd}{true}
740 \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。

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

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

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

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

```
743 \chardef\bxjs@everyparhook@@none=0
744 \chardef\bxjs@everyparhook@@compat=1
745 \chardef\bxjs@everyparhook@@modern=2
746 \bxjs@declare@enum@option{everyparhook}{everyparhook}{%
747   \if j\jsEngine compat\else modern\fi}
```

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

```
748 \chardef\bxjs@label@section@@none=0
749 \chardef\bxjs@label@section@@compat=1
750 \chardef\bxjs@label@section@@modern=2
751 \bxjs@declare@enum@option{label-section}{label@section}{compat}
```

`\ifbxjs@usezw` [スイッチ] `use-zw` の指定値。

TODO:3.0 `zw/nozw` は廃止の予定。

```
752 \bxjs@declare@bool@option{use-zw}{usezw}{true}
753 \DeclareOption{nozw}{\bxjs@depre@opt@do{nozw}{use-zw=false}}
754 \DeclareOption{zw}{\bxjs@depre@opt@do{zw}{use-zw=true}}
```

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

TODO:3.0 `js/nojs` は廃止の予定。

```
755 \bxjs@declare@bool@option{disguise-js}{disguise@js}{true}
756 \DeclareOption{nojs}{\bxjs@depre@opt@do{nojs}{disguise-js=false}}
757 \DeclareOption{js}{\bxjs@depre@opt@do{js}{disguise-js=true}}
```

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

```
758 \bxjs@declare@bool@option{precise-text}{precisetext}{false}
759 \DeclareOption{noprecisetext}{\bxjs@depre@opt@do{noprecisetext}{precise-
  text=false}}
760 \DeclareOption{precisetext}{\bxjs@depre@opt@do{precisetext}{precise-
  text=true}}
```

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

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

```
763 \DeclareOption{simplejasetup}{\bxjs@depre@opt@do{simplejasetup}{simple-ja-
  setup=true}}
```

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

```
764 \bxjs@declare@bool@option{plautopatch}{plautopatch}{false}
765 \g@addto@macro\bxjs@plautopatchtrue{\let\bxjs@plautopatch@given\@undefined}
766 \g@addto@macro\bxjs@plautopatchfalse{\def\bxjs@plautopatch@given{false}}
```

■オプションの実行

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

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

```
767 \ifbxjs@brace@safe\else
768 \let\bxjs@org@removeelement\@removeelement
769 \def\@removeelement#1#2#3{%
770   \def\reserved@a{#2}%
771   \ifx\reserved@a\@empty \let#3\@empty
772   \else \bxjs@org@removeelement{#1}{#2}{#3}%
773   \fi}
774 \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。

```
775 %<article>\ExecuteOptions{a4paper,oneside,onecolumn,notitlepage,final}
776 %<book>\ExecuteOptions{a4paper,twoside,onecolumn,titlepage,openright,final}
777 %<report>\ExecuteOptions{a4paper,oneside,onecolumn,titlepage,openany,final}
778 %<slide>\ExecuteOptions{36pt,a4paper,landscape,oneside,onecolumn,titlepage,final}
779 \ProcessOptions\relax
780 \bxjs@post@option@hook
```

後処理

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

```
781 \if@slide
782   \def\maybeblue{\@ifundefined{ver@color.sty}{\color{blue}}{}}
783 \fi
784 %<*jsclasses>
785 \if@landscape
786   \setlength\@tempdima {\paperheight}
787   \setlength\paperheight{\paperwidth}
788   \setlength\paperwidth {\@tempdima}
789 \fi
790 %</jsclasses>
```

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

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

```

791 \def\bxjs@tmpdo{%
792   \def\bxjs@tmpa{\@gobble}%
793   \expandafter\bxjs@tmpdo@a\@classoptionslist,\@nil,%
794   \let\@classoptionslist\bxjs@tmpa}
795 \def\bxjs@tmpdo@a#1,{%
796   \ifx\@nil#1\relax\else
797     \bxjs@tmpdo@b#1{}\@nil
798     \if@tempswa \edef\bxjs@tmpa{\bxjs@tmpa,#1}\fi
799     \expandafter\bxjs@tmpdo@a
800   \fi}
801 \def\bxjs@tmpdo@b#1#\bxjs@tmpdo@c}
802 \def\bxjs@tmpdo@c#1\@nil{%
803   \ifx\@nil#1\@nil \@tempwattrue \else \@tempwafalse \fi}
804 \bxjs@tmpdo

```

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

TODO:3.0 noscale オプションは廃止予定。

```

805 \@expandtwoargs\@removeelement
806   {papersize}\@classoptionslist\@classoptionslist
807 \@expandtwoargs\@removeelement
808   {10pt}\@classoptionslist\@classoptionslist
809 \@expandtwoargs\@removeelement
810   {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。

```

811 \ifx\bxjs@jadriver@opt\@undefined\else

```

```
812 \let\bxjs@jadriver\bxjs@jadriver@opt
813 \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。

```
814 \let\bxjs@tmpb\jsEngine
815 \ifx j\bxjs@tmpb\ifjsWithpTeXng
816 \let\bxjs@tmpb=g
817 \fi\fi
818 \ifx j\bxjs@tmpb\ifjsWithupTeX
819 \let\bxjs@tmpb=u
820 \fi\fi
821 \ifx p\bxjs@tmpb\ifjsInPdfMode\else
822 \let\bxjs@tmpb=n
823 \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。)

```
824 \ifx *\bxjs@engine@given
825 \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。

```
826 \ifx j\bxjs@engine@given
827 \bxjs@add@class@option{platex}
828 \else\ifx u\bxjs@engine@given
829 \bxjs@add@class@option{uplatex}
830 \fi\fi
831 \fi
832 \ifx\bxjs@engine@given\@undefined\else
833 \ifx\bxjs@engine@given\bxjs@tmpb\else
834 \ClassError\bxjs@clsname
835 {Option '\bxjs@engine@opt' used on wrong engine}\@ehc
836 \fi
837 \fi
```

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

```
838 \ifjsWithpTeXng
839 \bxjs@add@class@option{uplatex}
840 \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。

```
841 \@tempwattrue
842 \ifx \bxjs@driver@given\@undefined\else
843 \ifjsInPdfMode
844 \ifx\bxjs@driver@given\bxjs@driver@@pdfmode\else
845 \@tempwafalse
```

```

846 \fi
847 \else\ifx x\jsEngine
848 \ifx\bxjs@driver@given\bxjs@driver@@xetex\else
849 \@tempswafalse
850 \fi
851 \else
852 \ifx\bxjs@driver@given\bxjs@driver@@pdfmode
853 \@tempswafalse
854 \else\ifx\bxjs@driver@given\bxjs@driver@@xetex
855 \@tempswafalse
856 \fi\fi
857 \ifjsWithpTeXng\ifx\bxjs@driver@given\bxjs@driver@@dvipdfmx\else
858 \@tempswafalse
859 \fi\fi
860 \fi\fi
861 \fi
862 \if@tempswa\else
863 \ClassError\bxjs@clsname
864 {Option '\bxjs@driver@opt' used on wrong engine}\@ehc
865 \fi

```

DVI 出力のエンジンである場合の追加処理。

```

866 \ifjsInPdfMode \@tempswafalse
867 \else\ifx x\jsEngine \@tempswafalse
868 \else\ifjsWithpTeXng \@tempswafalse
869 \else \@tempwattrue
870 \fi\fi\fi
871 \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)。

```

872 \ifx\bxjs@driver@opt\@undefined
873 \if \ifbxjs@explIII T\else\ifx\bxjs@jadriver@opt\@undefined F\else T\fi\fi T%
874 \ClassWarningNoLine\bxjs@clsname
875 {A driver option is MISSING!!\MessageBreak
876 You should properly specify one of the valid\MessageBreak
877 driver options according to the DVI driver\MessageBreak
878 that is in use:\MessageBreak
879 \@spaces dvips, dvipdfmx, dviout, xdvi,\MessageBreak
880 \@spaces nodvidriver}
881 \fi
882 \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。)

```

883 \ifbxjs@dvi@opt
884 \bxjs@cslctcs{bxjs@driver@given}{bxjs@dvidriver@@\bxjs@driver@opt}
885 \bxjs@add@class@option{\bxjs@driver@opt}
886 \fi

```

```
887 \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。

```
888 \ifjsWithpTeXng
889   \ifx\bxjs@driver@given\bxjs@driver@@dvipdfmx
890     \let\bxjs@platexng@nodrv\undefined
891   \else\ifx t\bxjs@platexng@nodrv\else
892     \bxjs@add@class@option{dvipdfmx}
893   \fi\fi
894 \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。

```
895 \ifx\bxjs@driver@given\bxjs@driver@@none
896   \bxjs@papersizefalse
897 \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。

```
898 \@ifpackageloaded{plautopatch}{%
899   \bxjs@plautopatchtrue
900 }{}
```

標準の和文ドライバの名前の定数。

```
901 \def\bxjs@@minimal{minimal}
902 \def\bxjs@@standard{standard}
903 \def\bxjs@@pandoc{pandoc}
904 \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。

```
905 \ifx\bxjs@jadriver\relax
906   \ifx j\jsEngine
907     \let\bxjs@jadriver\bxjs@@standard
908   \else
909     \ClassWarningNoLine\bxjs@clsname
910     {The option 'ja' is MISSING!!\MessageBreak
911     So 'ja=minimal' is assumed as fallback, but\MessageBreak
912     such implicit setting is now DEPRECATED!\MessageBreak
913     You should write 'ja=minimal' explicitly,\MessageBreak
914     if it is intended}
915     \let\bxjs@jadriver\bxjs@@minimal
916   \fi
917 \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。

```
918 \ifx\bxjs@jadriver\bxjs@pandoc \ifx\bxjs@plautopatch@given\undefined
919   \ifjsWithTeX
920   \bxjs@plautopatchtrue
921 \fi\fi\fi
922 \ifx j\jsEngine \ifbxjs@plautopatch
923   \RequirePackage{plautopatch}[2018/08/22]%v0.3
924 \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。

```
925 \ifx\bxjs@jadriver@opt\undefined\else
926   \ifx\bxjs@engine@given\undefined
927     \ClassError\bxjs@clsname
928     {An engine option must be explicitly given}%
929     {When you use a Japanese-driver you must specify a correct\MessageBreak
930       engine option.\MessageBreak \@ehc}
931 \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。)

```
932 \ifx\bxjs@magstyle@@default\bxjs@magstyle@mag\else
933   \ifx\bxjs@magstyle\bxjs@magstyle@mag
934     \let\bxjs@magstyle\bxjs@magstyle@@default
935     \ClassError\bxjs@clsname
936     {The engine does not support 'magstyle=usemag'}%
937     {LuaTeX v0.87 or later no longer supports the "mag" feature of TeX.\MessageBreak
938       The default value 'nomag*' is used instead.\MessageBreak \@ehc}
939   \fi
940 \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。

```
941 \ifx\bxjs@jbase@opt\undefined\else
942   \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。

```
943   \jsSetQHLLength\@tempdima{\bxjs@jbase@opt}%
944   \bxjs@invscale\@tempdima\jsScale
945   \bxjs@setbasefontsize{\@tempdima}%
946 \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。

```
947   \ifx\bxjs@scale@opt\undefined\else
948     \ClassWarningNoLine\bxjs@clsname
949     {Redundant 'scale' option is ignored}%
```

```

950 \fi
951 \jsSetQHLlength\@tempdima{\bxjs@jbase@opt}%
952 \@tempdimb=\bxjs@param@basefontsize\relax
953 \edef\jsScale{\strip@pt\@tempdimb}%
954 \bxjs@invscale\@tempdima\jsScale
955 \edef\jsScale{\strip@pt\@tempdima}%
956 \fi
957 \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。

```

958 \let\Cjascale\jsScale

```

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

```

959 \ifbxjs@disguise@js
960 %<book|report>\def\bxjs@js@clsname{jsbook}
961 %<!book&!report>\def\bxjs@js@clsname{jsarticle}
962 \@namedef{ver@\bxjs@js@clsname.cls}{2001/01/01 (bxjs)}
963 \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。

```

964 \bxjs@add@class@option{nosetpagesize}

```

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

```

965 \ifbxjs@oldfontcommands
966 \AtEndOfClass{\allowoldfontcommands}
967 \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 `pLATEX 2ε` はトンボ出力幅を両側に 1 インチとっていますので、`dvips` 使用時に

```

-O -0.5in,-0.5in

```

というオプションを与えて両側 0.5 インチのトンボにするといいでしょう。

[2003-05-17] トンボをプレビューに使うことを考えて 1 インチを 2 インチにしました。

[2016-07-11] `memoir` クラスのマニュアルによると、トンボを含めた用紙の寸法は `\stockwidth`, `\stockheight` と呼ぶようですので、これを使うことにしました。

[2017-01-11] トンボオプションが指定されているとき「だけ」`\stockwidth`, `\stockheight` を定義するようにしました。

[2020-10-04] 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。

```
968 \ifx\stockwidth\@undefined\newdimen\stockwidth\fi
969 \ifx\stockheight\@undefined\newdimen\stockheight\fi
970 \begingroup\expandafter\expandafter\expandafter\endgroup
971 \expandafter\ifx\csname iftombow\expandafter\endcsname\csname iftrue\endcsname
972   \setlength{\stockwidth}{\paperwidth}
973   \setlength{\stockheight}{\paperheight}
974   \advance \stockwidth 2in
975   \advance \stockheight 2in
976 \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。

```
977 %<slide>\def\n@baseline{13}%
978 %<!slide>\ifdim\bxjs@magstyle\bxjs@magstyle@magfontsize<10pt \def\n@baseline{15}%
979 %<!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。

```
980 \ifx\bxjs@magstyle\bxjs@magstyle@mag
981   \jsc@magtrue
982 \else\ifx\bxjs@magstyle\bxjs@magstyle@xreal
983   \jsc@mag@xrealtrue
984 \fi\fi
```

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

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

- `geometry` 側でオプション `truedimen` を指定してください。

- geometry 側でオプション mag は使えません。

設定すべき \mag 値を (基底サイズ)/(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。

```

985 \ifx\bxjs@param@mag\relax
986   \@tempdima=\bxjs@param@basefontsize
987   \advance\@tempdima.001pt \multiply\@tempdima25
988   \divide\@tempdima16384\relax \@tempcnta\@tempdima\relax
989   \edef\bxjs@param@mag{\the\@tempcnta}
990 \else
991 % mag値が直接指定された場合
992   \bxjs@gset@tempcnta{\bxjs@param@mag}
993   \ifnum\@tempcnta<\z@ \@tempcnta=\z@ \fi
994 % 有効なmag値の範囲は1--32768
995   \edef\bxjs@param@mag{\the\@tempcnta}
996   \advance\@tempcnta100000
997   \def\bxjs@tmpa#1#2#3#4#5\@nil{\@tempdima=#2#3#4.#5\p@}
998   \expandafter\bxjs@tmpa\the\@tempcnta\@nil
999   \edef\bxjs@param@basefontsize{\the\@tempdima}
1000 \fi
1001 \@tempcnta\bxjs@param@mag \advance\@tempcnta100000
1002 \def\bxjs@tmpa#1#2#3#4\@nil{\@tempdima=#2#3.#4\p@}
1003 \expandafter\bxjs@tmpa\the\@tempcnta\@nil
1004 \edef\jsc@magscale{\strip@pt\@tempdima}
1005 \let\jsBaseFontSize\bxjs@param@basefontsize

```

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

※ 2.9 版において \p@? 表記を廃止。

```

1006 \newdimen\jsc@mpt
1007 \newdimen\jsc@mmm
1008 \ifjsc@mag
1009   \jsc@mpt=1\p@
1010   \jsc@mmm=1mm
1011 \else
1012   \jsc@mpt=\jsc@magscale\p@
1013   \jsc@mmm=\jsc@magscale mm
1014 \fi

```

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

use-zw が真の時は \zw を \jsZw と同義にする。

```

1015 \newdimen\jsZw
1016 \jsZw=10\jsc@mpt \jsZw=\jsScale\jsZw
1017 \ifbxjs@usezw

```

```

1018 \providecommand*\zw{\jsZw}
1019 \fi

```

`\zwspace` 全角幅の水平空き。

```

1020 \def\zwspace{\hskip\jsZw\relax}

```

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

```

1021 \ifjsc@mag@xreal
1022 \RequirePackage{type1cm}
1023 \let\jsc@invscale\bxjs@invscale

```

```

1024 \ifbxjs@TUenc
1025 \expandafter\let\csname TU/lmr/m/n/10\endcsname\relax
1026 \else
1027 \expandafter\let\csname OT1/cmr/m/n/10\endcsname\relax
1028 \fi
1029 \expandafter\let\csname OMX/cmex/m/n/10\endcsname\relax
1030 \let\jsc@get@external@font\get@external@font
1031 \def\get@external@font{%
1032 \jsc@preadjust@extract@font
1033 \jsc@get@external@font}
1034 \def\jsc@fstrunc#1{%
1035 \edef\jsc@tmpa{\strip@pt#1}%
1036 \expandafter\jsc@fstrunc@a\jsc@tmpa.****\@nil}
1037 \def\jsc@fstrunc@a#1.#2#3#4#5#6\@nil{%
1038 \if#5*\else
1039 \edef\jsc@tmpa{#1%
1040 \ifnum#2#3>\z@ .#2\ifnum#3>\z@ #3\fi\fi}%
1041 \fi}
1042 \def\jsc@preadjust@extract@font{%
1043 \let\jsc@req@size\f@size
1044 \dimen@f@size\p@ \jsc@invscale\dimen@\jsc@magscale
1045 \advance\dimen@.005pt\relax \jsc@fstrunc\dimen@
1046 \let\jsc@ref@size\jsc@tmpa
1047 \let\f@size\jsc@ref@size}
1048 \def\execute@size@function#1{%
1049 \let\jsc@cref@size\f@size
1050 \let\f@size\jsc@req@size
1051 \csname s@fct@#1\endcsname}
1052 \let\jsc@DeclareErrorFont\DeclareErrorFont
1053 \def\DeclareErrorFont#1#2#3#4#5{%
1054 \@tempdimc#5\p@ \@tempdimc\jsc@magscale\@tempdimc
1055 \edef\jsc@tmpa{#{1}-{#2}-{#3}-{#4}-{strip@pt\@tempdimc}}
1056 \expandafter\jsc@DeclareErrorFont\jsc@tmpa}
1057 \def\gen@sfcnt{%
1058 \edef\mandatory@arg{\mandatory@arg\jsc@cref@size}%
1059 \empty@sfcnt}
1060 \def\genb@sfcnt{%

```

```

1061 \edef\mandatory@arg{%
1062 \mandatory@arg\expandafter\genb@x\jsc@cref@size..\@@}%
1063 \empty@sfcnt}
1064 \ifbxjs@TUenc\else
1065 \DeclareErrorFont{OT1}{cmr}{m}{n}{10}
1066 \fi
1067 \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 1068 \def\jsc@smallskip{\vspace\jsc@smallskipamount}
\jsc@bigskip 1069 %\def\jsc@medskip{\vspace\jsc@medskipamount}
1070 %\def\jsc@bigskip{\vspace\jsc@bigskipamount}

```

```

\jsc@smallskipamount
\jsc@medskipamount 1071 \newskip\jsc@smallskipamount
\jsc@bigskipamount 1072 \jsc@smallskipamount=3\jsc@empt plus 1\jsc@empt minus 1\jsc@empt
1073 %\newskip\jsc@medskipamount
1074 %\jsc@medskipamount =6\jsc@empt plus 2\jsc@empt minus 2\jsc@empt
1075 %\newskip\jsc@bigskipamount
1076 %\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] トンボオブションが指定されているとき「だけ」`\stockwidth`, `\stockheight` が定義されています。

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

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

```

1077 % \ifpapersize
1078 % \setlength{\@tempdima}{\paperwidth}
1079 % \setlength{\@tempdimb}{\paperheight}
1080 % \iftombow
1081 % \advance \@tempdima 2truein
1082 % \advance \@tempdimb 2truein
1083 % \fi
1084 % \AtBeginDvi{\special{pdf: pagesize width \the\@tempdima space height \the\@tempdimb}}
1085 % \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。

```
1086 %</class>
1087 %<*jsclasses>
1088 %<*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。

```
1089 %</class>
1090 %<*minijs>
1091 %% min/goth -> jis/jisg (for pLaTeX only)
1092 \ifnum\jis"2121="3000 \else
1093 \@for\@tempa:=5,6,7,8,9,10,10.95,12,14.4,17.28,20.74,24.88\do{%
1094   \expandafter\let\csname JY1/mc/m/n/\@tempa\endcsname\relax
1095   \expandafter\let\csname JY1/gt/m/n/\@tempa\endcsname\relax
1096   \expandafter\let\csname JT1/mc/m/n/\@tempa\endcsname\relax
1097   \expandafter\let\csname JT1/gt/m/n/\@tempa\endcsname\relax
1098 }
1099 \def\Cjascale{0.924690}
1100 \DeclareFontShape{JY1}{mc}{m}{n}{<-> s * [0.961] jis}{ }
1101 \DeclareFontShape{JY1}{gt}{m}{n}{<-> s * [0.961] jisg}{ }
1102 \DeclareFontShape{JT1}{mc}{m}{n}{<-> s * [0.961] tmin10}{ }
1103 \DeclareFontShape{JT1}{gt}{m}{n}{<-> s * [0.961] tgoth10}{ }
1104 \fi
1105 %</minijs>
1106 %<*class>
```

```

1107 %<!*jspf>
1108 \def\Cjascale{0.924690}
1109 \ifmingoth
1110 \DeclareFontShape{\jsc@JYn}{mc}{m}{n}{<-> s * [0.961] \jsc@pfx@ min10}{-}
1111 \DeclareFontShape{\jsc@JYn}{gt}{m}{n}{<-> s * [0.961] \jsc@pfx@ goth10}{-}
1112 \DeclareFontShape{\jsc@JTn}{mc}{m}{n}{<-> s * [0.961] \jsc@pfx@ tmin10}{-}
1113 \DeclareFontShape{\jsc@JTn}{gt}{m}{n}{<-> s * [0.961] \jsc@pfx@ tgoth10}{-}
1114 \else
1115 \ifjisfont
1116 \DeclareFontShape{\jsc@JYn}{mc}{m}{n}{<-> s * [0.961] \jsc@pfx@ jis}{-}
1117 \DeclareFontShape{\jsc@JYn}{gt}{m}{n}{<-> s * [0.961] \jsc@pfx@ jisg}{-}
1118 \DeclareFontShape{\jsc@JTn}{mc}{m}{n}{<-> s * [0.961] \jsc@pfx@ tmin10}{-}
1119 \DeclareFontShape{\jsc@JTn}{gt}{m}{n}{<-> s * [0.961] \jsc@pfx@ tgoth10}{-}
1120 \else
1121 \if@jsc@uplatex
1122 \DeclareFontShape{JY2}{mc}{m}{n}{<-> s * [0.924690] upjisr-h}{-}
1123 \DeclareFontShape{JY2}{gt}{m}{n}{<-> s * [0.924690] upjisg-h}{-}
1124 \DeclareFontShape{JT2}{mc}{m}{n}{<-> s * [0.924690] upjisr-v}{-}
1125 \DeclareFontShape{JT2}{gt}{m}{n}{<-> s * [0.924690] upjisg-v}{-}
1126 \else
1127 \DeclareFontShape{\jsc@JYn}{mc}{m}{n}{<-> s * [0.961] \jsc@pfx@ jis}{-}
1128 \DeclareFontShape{\jsc@JYn}{gt}{m}{n}{<-> s * [0.961] \jsc@pfx@ jisg}{-}
1129 \DeclareFontShape{\jsc@JTn}{mc}{m}{n}{<-> s * [0.961] \jsc@pfx@ tmin10}{-}
1130 \DeclareFontShape{\jsc@JTn}{gt}{m}{n}{<-> s * [0.961] \jsc@pfx@ tgoth10}{-}
1131 \fi
1132 \fi
1133 \fi
1134 %</!*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。

```

1135 %<!*jspf>
1136 \def\Cjascale{0.903375}
1137 \ifmingoth
1138 \DeclareFontShape{\jsc@JYn}{mc}{m}{n}{<-> s * [0.93885] \jsc@pfx@ min10}{-}
1139 \DeclareFontShape{\jsc@JYn}{gt}{m}{n}{<-> s * [0.93885] \jsc@pfx@ goth10}{-}
1140 \DeclareFontShape{\jsc@JTn}{mc}{m}{n}{<-> s * [0.93885] \jsc@pfx@ tmin10}{-}
1141 \DeclareFontShape{\jsc@JTn}{gt}{m}{n}{<-> s * [0.93885] \jsc@pfx@ tgoth10}{-}
1142 \else
1143 \ifjisfont
1144 \DeclareFontShape{\jsc@JYn}{mc}{m}{n}{<-> s * [0.93885] \jsc@pfx@ jis}{-}
1145 \DeclareFontShape{\jsc@JYn}{gt}{m}{n}{<-> s * [0.93885] \jsc@pfx@ jisg}{-}
1146 \DeclareFontShape{\jsc@JTn}{mc}{m}{n}{<-> s * [0.93885] \jsc@pfx@ tmin10}{-}
1147 \DeclareFontShape{\jsc@JTn}{gt}{m}{n}{<-> s * [0.93885] \jsc@pfx@ tgoth10}{-}
1148 \else
1149 \if@jsc@uplatex
1150 \DeclareFontShape{JY2}{mc}{m}{n}{<-> s * [0.903375] upjisr-h}{-}
1151 \DeclareFontShape{JY2}{gt}{m}{n}{<-> s * [0.903375] upjisg-h}{-}

```

```

1152 \DeclareFontShape{JT2}{mc}{m}{n}{<-> s * [0.903375] upjisr-v}{ }
1153 \DeclareFontShape{JT2}{gt}{m}{n}{<-> s * [0.903375] upjisg-v}{ }
1154 \else
1155 \DeclareFontShape{\jsc@JYn}{mc}{m}{n}{<-> s * [0.93885] \jsc@pfx@ jis}{ }
1156 \DeclareFontShape{\jsc@JYn}{gt}{m}{n}{<-> s * [0.93885] \jsc@pfx@ jisg}{ }
1157 \DeclareFontShape{\jsc@JTn}{mc}{m}{n}{<-> s * [0.93885] \jsc@pfx@ tmin10}{ }
1158 \DeclareFontShape{\jsc@JTn}{gt}{m}{n}{<-> s * [0.93885] \jsc@pfx@ tgoth10}{ }
1159 \fi
1160 \fi
1161 \fi
1162 %</jspf>

```

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

[2003-03-16] イタリック体, 斜体について, 和文でゴシックを当てていましたが, 数学の定理環境などで多量のイタリック体を使うことがあり, ゴシックにすると黒々となってしまうという弊害がありました。amsthm を使わない場合は定理の本文が明朝になるように `\newtheorem` 環境を手直ししてしのいでいましたが, $\mathrm{T}_{\mathrm{E}}\mathrm{X}$ 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。

```

1163 % \DeclareFontShape{\jsc@JYn}{mc}{bx}{n}{<->ssub*gt/m/n}{ } % in \jsc@JYnmc
1164 % \DeclareFontShape{\jsc@JYn}{gt}{bx}{n}{<->ssub*gt/m/n}{ } % in \jsc@JYngt
1165 \DeclareFontShape{\jsc@JYn}{mc}{m}{it}{<->ssub*mc/m/n}{ }
1166 \DeclareFontShape{\jsc@JYn}{mc}{m}{sl}{<->ssub*mc/m/n}{ }
1167 \DeclareFontShape{\jsc@JYn}{mc}{m}{sc}{<->ssub*mc/m/n}{ }
1168 \DeclareFontShape{\jsc@JYn}{gt}{m}{it}{<->ssub*gt/m/n}{ }
1169 \DeclareFontShape{\jsc@JYn}{gt}{m}{sl}{<->ssub*gt/m/n}{ }
1170 \DeclareFontShape{\jsc@JYn}{mc}{bx}{it}{<->ssub*gt/m/n}{ }
1171 \DeclareFontShape{\jsc@JYn}{mc}{bx}{sl}{<->ssub*gt/m/n}{ }
1172 % \DeclareFontShape{\jsc@JTn}{mc}{bx}{n}{<->ssub*gt/m/n}{ } % in \jsc@JTnmc
1173 % \DeclareFontShape{\jsc@JTn}{gt}{bx}{n}{<->ssub*gt/m/n}{ } % in \jsc@JTngt
1174 \DeclareFontShape{\jsc@JTn}{mc}{m}{it}{<->ssub*mc/m/n}{ }
1175 \DeclareFontShape{\jsc@JTn}{mc}{m}{sl}{<->ssub*mc/m/n}{ }
1176 \DeclareFontShape{\jsc@JTn}{mc}{m}{sc}{<->ssub*mc/m/n}{ }
1177 \DeclareFontShape{\jsc@JTn}{gt}{m}{it}{<->ssub*gt/m/n}{ }
1178 \DeclareFontShape{\jsc@JTn}{gt}{m}{sl}{<->ssub*gt/m/n}{ }
1179 \DeclareFontShape{\jsc@JTn}{mc}{bx}{it}{<->ssub*gt/m/n}{ }
1180 \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。

```

1181 %</class>
1182 %<*class|minijs>
1183 %% ad-hoc "relation font"
1184 \@ifl@t@r\fmtversion{2020/10/01}
1185     {\jsc@needspace@tchfalse}{\jsc@needspace@tchtrue}
1186 \ifjsc@needspace@tch      % --- for 2020-02-02 or older BEGIN
1187 \ifx\@rmfamilyhook\@undefined % old
1188 \DeclareRobustCommand\rmfamily
1189     {\not@math@alphabet\rmfamily\mathrm
1190      \romanfamily\rmdefault\kanjifamily\mcdefault\selectfont}
1191 \DeclareRobustCommand\sffamily
1192     {\not@math@alphabet\sffamily\mathsf
1193      \romanfamily\sfdefault\kanjifamily\gtdefault\selectfont}
1194 \DeclareRobustCommand\ttfamily
1195     {\not@math@alphabet\ttfamily\mathtt
1196      \romanfamily\ttdefault\kanjifamily\gtdefault\selectfont}
1197 \AtBeginDocument{%
1198   \ifx\mweights@init\@undefined\else % mweights.sty is loaded
1199     % my definitions above should have been overwritten, recover it!
1200     % \selectfont is executed twice but I don't care about speed...
1201     \expandafter\g@addto@macro\csname rmfamily \endcsname
1202       {\kanjifamily\mcdefault\selectfont}%
1203     \expandafter\g@addto@macro\csname sffamily \endcsname
1204       {\kanjifamily\gtdefault\selectfont}%
1205     \expandafter\g@addto@macro\csname ttfamily \endcsname
1206       {\kanjifamily\gtdefault\selectfont}%
1207   \fi}
1208 \else % 2020-02-02
1209 \g@addto@macro\@rmfamilyhook
1210   {\prepare@family@series@update@kanji{mc}\mcdefault}
1211 \g@addto@macro\@sffamilyhook
1212   {\prepare@family@series@update@kanji{gt}\gtdefault}
1213 \g@addto@macro\@ttfamilyhook
1214   {\prepare@family@series@update@kanji{gt}\gtdefault}
1215 \fi
1216 \else % --- for 2020-02-02 or older END & for 2020-10-01 BEGIN
1217 \AddToHook{rmfamily}%
1218   {\prepare@family@series@update@kanji{mc}\mcdefault}
1219 \AddToHook{sffamily}%
1220   {\prepare@family@series@update@kanji{gt}\gtdefault}
1221 \AddToHook{ttfamily}%
1222   {\prepare@family@series@update@kanji{gt}\gtdefault}
1223 \fi % --- for 2020-10-01 END
1224 %</class|minijs>
1225 %<*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。

```
1226 \ifx\DeclareFixJFMCJKTextFontCommand\undefined
1227 \DeclareRobustCommand\textmc[1]{%
1228   \relax\ifmmode \expandafter\nfss@text \fi{\mcfamily #1}}
1229 \DeclareRobustCommand\textgt[1]{%
1230   \relax\ifmmode \expandafter\nfss@text \fi{\gtfamily #1}}
1231 \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。

```
1232 \def\reDeclareMathAlphabet#1#2#3{%
1233   \edef\@tempa{\expandafter\@gobble\string#2}%
1234   \edef\@tempb{\expandafter\@gobble\string#3}%
1235   \edef\@tempc{\string \@ \expandafter\@gobbletwo\string#2}%
1236   \ifx\@tempc\@tempa%
1237     \edef\@tempa{\expandafter\@gobbletwo\string#2}%
1238     \edef\@tempb{\expandafter\@gobbletwo\string#3}%
1239   \fi
1240   \begingroup
1241     \let\protect\noexpand
1242     \def\@tempaa{\relax}%
1243     \expandafter\ifx\csname RDMAorg@\@tempa\endcsname\relax
1244       \edef\@tempaa{\expandafter\def\expandafter\noexpand%
1245         \csname RDMAorg@\@tempa\endcsname{%
1246           \expandafter\noexpand\csname\@tempa\endcsname}}%
1247     \fi
1248     \def\@tempbb{\relax}%
1249     \expandafter\ifx\csname RDMAorg@\@tempb\endcsname\relax
1250       \edef\@tempbb{\expandafter\def\expandafter\noexpand%
1251         \csname RDMAorg@\@tempb\endcsname{%
```

```

1252         \expandafter\noexpand\csname\@tempb\endcsname}}%
1253     \fi
1254     \edef\@tempc{\@tempaa\@tempbb}%
1255     \expandafter\endgroup\@tempc%
1256     \edef#1{\noexpand\protect\expandafter\noexpand\csname%
1257         \expandafter\@gobble\string#1\space\space\endcsname}%
1258     \expandafter\edef\csname\expandafter\@gobble\string#1\space\space\endcsname%
1259         {\noexpand\DualLang@mathalph@bet%
1260             {\expandafter\noexpand\csname RDMAorg@\@tempa\endcsname}%
1261             {\expandafter\noexpand\csname RDMAorg@\@tempb\endcsname}%
1262         }%
1263 }
1264 \@onlypreamble\reDeclareMathAlphabet
1265 \def\DualLang@mathalph@bet#1#2{%
1266     \relax\ifmmode
1267         \ifx\math@bgroup\bgroup%      2e normal style      (\mathrm{...})
1268             \bgroup\let\DualLang@Mfontsw\DLMfontsw@standard
1269         \else
1270             \ifx\math@bgroup\relax%    2e two letter style (\rm->\mathrm)
1271                 \let\DualLang@Mfontsw\DLMfontsw@oldstyle
1272             \else
1273                 \ifx\math@bgroup\@empty% 2.09 oldfont style ({\mathrm ...})
1274                     \let\DualLang@Mfontsw\DLMfontsw@oldfont
1275                 \else%
1276                     panic! assume 2e normal style
1277                     \bgroup\let\DualLang@Mfontsw\DLMfontsw@standard
1278                 \fi
1279             \fi
1280         \else
1281             \let\DualLang@Mfontsw\@firstoftwo
1282         \fi
1283     \DualLang@Mfontsw{#1}{#2}%
1284 }
1285 \def\DLMfontsw@standard#1#2#3{#1{#2{#3}}\egroup}
1286 \def\DLMfontsw@oldstyle#1#2{#1\relax\@fontswitch\relax{#2}}
1287 \def\DLMfontsw@oldfont#1#2{#1\relax#2\relax}
1288 \if@enablejfam
1289     \DeclareSymbolFont{mincho}{\jsc@JYn}{mc}{m}{n}
1290     \DeclareSymbolFontAlphabet{\mathmc}{mincho}
1291     \SetSymbolFont{mincho}{bold}{\jsc@JYn}{gt}{m}{n}
1292     \jfam\symmincho
1293     \DeclareMathAlphabet{\mathgt}{\jsc@JYn}{gt}{m}{n}
1294     \AtBeginDocument{%
1295         \reDeclareMathAlphabet{\mathrm}{\@mathrm}{\@mathmc}
1296         \reDeclareMathAlphabet{\mathbf}{\@mathbf}{\@mathgt}}
1297 \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。

```
1298 % \DeclareTextCommand{\textsterling}{OT1}{\itshape\char`\$}}
```

禁則パラメータも若干修正します。

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

```
1299 \prebreakpenalty\jis"2147=10000      % 5000  '
1300 \postbreakpenalty\jis"2148=10000      % 5000  "
1301 \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。

```
1302 \inhibitxspcode`!=1
1303 \inhibitxspcode`π=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

```
1304 % \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。

```
1305 \xspcode`+=3
1306 \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。

```
1307 \xspcode`^^80=3
1308 \xspcode`^^81=3
1309 \xspcode`^^82=3
1310 \xspcode`^^83=3
1311 \xspcode`^^84=3
1312 \xspcode`^^85=3
1313 \xspcode`^^86=3
1314 \xspcode`^^87=3
1315 \xspcode`^^88=3
1316 \xspcode`^^89=3
1317 \xspcode`^^8a=3
1318 \xspcode`^^8b=3
1319 \xspcode`^^8c=3
1320 \xspcode`^^8d=3
1321 \xspcode`^^8e=3
```

1322 \xspcode`^^8f=3
1323 \xspcode`^^90=3
1324 \xspcode`^^91=3
1325 \xspcode`^^92=3
1326 \xspcode`^^93=3
1327 \xspcode`^^94=3
1328 \xspcode`^^95=3
1329 \xspcode`^^96=3
1330 \xspcode`^^97=3
1331 \xspcode`^^98=3
1332 \xspcode`^^99=3
1333 \xspcode`^^9a=3
1334 \xspcode`^^9b=3
1335 \xspcode`^^9c=3
1336 \xspcode`^^9d=3
1337 \xspcode`^^9e=3
1338 \xspcode`^^9f=3
1339 \xspcode`^^a0=3
1340 \xspcode`^^a1=3
1341 \xspcode`^^a2=3
1342 \xspcode`^^a3=3
1343 \xspcode`^^a4=3
1344 \xspcode`^^a5=3
1345 \xspcode`^^a6=3
1346 \xspcode`^^a7=3
1347 \xspcode`^^a8=3
1348 \xspcode`^^a9=3
1349 \xspcode`^^aa=3
1350 \xspcode`^^ab=3
1351 \xspcode`^^ac=3
1352 \xspcode`^^ad=3
1353 \xspcode`^^ae=3
1354 \xspcode`^^af=3
1355 \xspcode`^^b0=3
1356 \xspcode`^^b1=3
1357 \xspcode`^^b2=3
1358 \xspcode`^^b3=3
1359 \xspcode`^^b4=3
1360 \xspcode`^^b5=3
1361 \xspcode`^^b6=3
1362 \xspcode`^^b7=3
1363 \xspcode`^^b8=3
1364 \xspcode`^^b9=3
1365 \xspcode`^^ba=3
1366 \xspcode`^^bb=3
1367 \xspcode`^^bc=3
1368 \xspcode`^^bd=3
1369 \xspcode`^^be=3
1370 \xspcode`^^bf=3

1371 \xspcode`^^c0=3
1372 \xspcode`^^c1=3
1373 \xspcode`^^c2=3
1374 \xspcode`^^c3=3
1375 \xspcode`^^c4=3
1376 \xspcode`^^c5=3
1377 \xspcode`^^c6=3
1378 \xspcode`^^c7=3
1379 \xspcode`^^c8=3
1380 \xspcode`^^c9=3
1381 \xspcode`^^ca=3
1382 \xspcode`^^cb=3
1383 \xspcode`^^cc=3
1384 \xspcode`^^cd=3
1385 \xspcode`^^ce=3
1386 \xspcode`^^cf=3
1387 \xspcode`^^d0=3
1388 \xspcode`^^d1=3
1389 \xspcode`^^d2=3
1390 \xspcode`^^d3=3
1391 \xspcode`^^d4=3
1392 \xspcode`^^d5=3
1393 \xspcode`^^d6=3
1394 \xspcode`^^d7=3
1395 \xspcode`^^d8=3
1396 \xspcode`^^d9=3
1397 \xspcode`^^da=3
1398 \xspcode`^^db=3
1399 \xspcode`^^dc=3
1400 \xspcode`^^dd=3
1401 \xspcode`^^de=3
1402 \xspcode`^^df=3
1403 \xspcode`^^e0=3
1404 \xspcode`^^e1=3
1405 \xspcode`^^e2=3
1406 \xspcode`^^e3=3
1407 \xspcode`^^e4=3
1408 \xspcode`^^e5=3
1409 \xspcode`^^e6=3
1410 \xspcode`^^e7=3
1411 \xspcode`^^e8=3
1412 \xspcode`^^e9=3
1413 \xspcode`^^ea=3
1414 \xspcode`^^eb=3
1415 \xspcode`^^ec=3
1416 \xspcode`^^ed=3
1417 \xspcode`^^ee=3
1418 \xspcode`^^ef=3
1419 \xspcode`^^f0=3

```

1420 \xspcode^^f1=3
1421 \xspcode^^f2=3
1422 \xspcode^^f3=3
1423 \xspcode^^f4=3
1424 \xspcode^^f5=3
1425 \xspcode^^f6=3
1426 \xspcode^^f7=3
1427 \xspcode^^f8=3
1428 \xspcode^^f9=3
1429 \xspcode^^fa=3
1430 \xspcode^^fb=3
1431 \xspcode^^fc=3
1432 \xspcode^^fd=3
1433 \xspcode^^fe=3
1434 \xspcode^^ff=3

1435 %</class>
1436 %</jsclasses>
1437 %<*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 \fi 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。

```

1438 \chardef\bxjs@periodchar=\.
1439 \bxjs@robust@def\bxjs@SE{%
1440   \ifnum\spacefactor<\@m \spacefactor\@m
1441   \else \spacefactor\sfcode\bxjs@periodchar
1442   \fi}
1443 \ifbxjs@fix@at@cmd
1444   \def\@{\bxjs@SE{}}
1445 \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。

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

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

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

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

```
1446 %\let\bxjs@prev@set@fontsize\@undefined
1447 \@onlypreamble\bxjs@patch@set@fontsize
1448 \def\bxjs@patch@set@fontsize{%
1449   \ifx\bxjs@prev@set@fontsize\set@fontsize\else
1450     \def\bxjs@tmpa{\def\set@fontsize####1####2####3}%
1451   \expandafter\bxjs@tmpa\expandafter{%
1452     \set@fontsize{##1}{##2}{##3}%
1453 % 末尾にコードを追加
```

```

1454 \expandafter\def\expandafter\size@update\expandafter{%
1455   \size@update
1456   \jsFontSizeChanged}%
1457 }
1458 \let\bxjs@prev@set@fontsize\set@fontsize
1459 \fi}

```

この場とパッケージ末尾で作動させる。

```

1460 \bxjs@patch@set@fontsize
1461 \AtEndOfClass{\bxjs@patch@set@fontsize}

```

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

```

1462 \newcommand*\jsFontSizeChanged{%
1463   \jsZw=\f@size\p@
1464   \jsZw=\jsScale \jsZw
1465   \ifdim\parindent>\z@
1466     \if@english \parindent=1em
1467     \else       \parindent=1\jsZw
1468   \fi
1469   \fi\relax
1470   \jsResetDimen}

```

`\jsResetDimen` ユーザ定義用のフック。

```

1471 \providecommand*\jsResetDimen{}

```

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

```

1472 \ifjsc@mag
1473   \let\jsc@setfontsize\@setfontsize
1474 \else
1475   \def\jsc@setfontsize#1#2#3{%
1476     \@setfontsize#1{#2\jsc@mpt}{#3\jsc@mpt}}
1477 % microtype対策
1478 \ifjsWithTeX\if j\jsEngine\else
1479   \def\jsc@setfontsize#1#2#3{%
1480     \edef\bxjs@sfs@next{%
1481       \unexpanded{\@setfontsize#1}%
1482       {\the\dimexpr#2\jsc@mpt\relax}{\the\dimexpr#3\jsc@mpt\relax}%
1483     }\bxjs@sfs@next}
1484   \fi\fi
1485 \fi

```

これらのグルーをもっ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。

```
1486 \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、しっぱ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。

```
1487 \newif\ifnarrowbaselines
1488 \if@english
1489   \narrowbaselinestrue
1490 \fi
1491 \def\narrowbaselines{%
1492   \narrowbaselinestrue
1493   \skip0=\abovedisplayskip
1494   \skip2=\abovedisplayshortskip
1495   \skip4=\belowdisplayskip
1496   \skip6=\belowdisplayshortskip
1497 % 一時的に警告を無効化する
1498   \let\bxjs@save@nomath\@nomath
1499   \let\@nomath\@gobble
1500   \@currsize\selectfont
1501   \let\@nomath\bxjs@save@nomath
1502   \abovedisplayskip=\skip0
1503   \abovedisplayshortskip=\skip2
1504   \belowdisplayskip=\skip4
1505   \belowdisplayshortskip=\skip6\relax}
1506 \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@ifnarrowbaselines` スイッチ `narrowbaselines` を L^AT_EX 式条件文にしたもの。

```

1507 \def\bxjs@if@narrowbaselines{%
1508   \ifnarrowbaselines\expandafter\@firstoftwo
1509   \else \expandafter\@secondoftwo
1510   \fi
1511 }

```

`\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。

```

1512 \renewcommand{\normalsize}{%
1513   \bxjs@if@narrowbaselines{%
1514     \jsc@setfontsize\normalsize\@xpt\@xiipt
1515   }{%else
1516     \jsc@setfontsize\normalsize\@xpt{\n@baseline}%
1517   }%

```

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。

```

1518 \abovedisplayskip 11\jsc@mpt \@plus3\jsc@mpt \@minus4\jsc@mpt
1519 \abovedisplayshortskip \z@ \@plus3\jsc@mpt
1520 \belowdisplayskip 9\jsc@mpt \@plus3\jsc@mpt \@minus4\jsc@mpt
1521 \belowdisplayshortskip \belowdisplayskip

```

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

```

1522 \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。

```

1523 %</class>
1524 %<*class|minijs>
1525 %% initialize
1526 \normalsize
1527 %</class|minijs>
1528 %<*class>

```

\Cht 基準となる長さの設定をします。p_{La}T_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 p_TE_X 系以外では未定義なのでここで定義する。

```
1529 \ifx\Cht\undefined \newdimen\Cht \fi
1530 \ifx\Cdp\undefined \newdimen\Cdp \fi
1531 \ifx\Cwd\undefined \newdimen\Cwd \fi
1532 \ifx\Cvs\undefined \newdimen\Cvs \fi
1533 \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。

```
1534 \setlength\Cht{0.88\jsZw}
1535 \setlength\Cdp{0.12\jsZw}
1536 \setlength\Cwd{1\jsZw}
1537 \setlength\Cvs{\baselineskip}
1538 \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。

```
1539 \newcommand{\small}{%
1540   \bxjs@if@narrowbaselines{%
1541     %<!kiyou>   \jsc@setfontsize\small\@ixpt{11}%
1542     %<kiyou>    \jsc@setfontsize\small{8.8888}{11}%
1543   }{%else
1544     %<!kiyou>   \jsc@setfontsize\small\@ixpt{13}%
1545     %<kiyou>    \jsc@setfontsize\small{8.8888}{13.2418}%
1546   }%
1547   \abovedisplayskip 9\jsc@mpt \@plus3\jsc@mpt \@minus4\jsc@mpt
1548   \abovedisplayshortskip \z@ \@plus3\jsc@mpt
1549   \belowdisplayskip \abovedisplayskip
1550   \belowdisplayshortskip \belowdisplayskip
1551   \def\@listif\leftmargin\leftmargini
1552     \topsep \z@
1553     \parsep \z@
1554     \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。

```
1555 \newcommand{\footnotesize}{%
1556   \bxjs@if@narrowbaselines{%
1557     %<!kiyou>   \jsc@setfontsize\footnotesize\@viipt{9.5}%
```

```

1558 %<kiyou> \jsc@setfontsize\footnotesize{8.8888}{11}%
1559 }{%else
1560 %<!kiyou> \jsc@setfontsize\footnotesize\@viipt{11}%
1561 %<kiyou> \jsc@setfontsize\footnotesize{8.8888}{13.2418}%
1562 }%
1563 \abovedisplayskip 6\jsc@empt \@plus2\jsc@empt \@minus3\jsc@empt
1564 \abovedisplayshortskip \z@ \@plus2\jsc@empt
1565 \belowdisplayskip \abovedisplayskip
1566 \belowdisplayshortskip \belowdisplayskip
1567 \def\@listi{\leftmargin\leftmargin
1568 \topsep \z@
1569 \parsep \z@
1570 \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 1571 \newcommand{\scriptsize}{\jsc@setfontsize\scriptsize\@viipt\@viipt}
1572 \newcommand{\tiny}{\jsc@setfontsize\tiny\@vpt\@vpt}
\LARGE 1573 \if@twocolumn
1574 %<!kiyou> \newcommand{\large}{\jsc@setfontsize\large\@xipt{\n@baseline}}
1575 %<kiyou> \newcommand{\large}{\jsc@setfontsize\large{11.111}{\n@baseline}}
1576 \else
1577 %<!kiyou> \newcommand{\large}{\jsc@setfontsize\large\@xipt{17}}
1578 %<kiyou> \newcommand{\large}{\jsc@setfontsize\large{11.111}{17}}
1579 \fi
1580 %<!kiyou>\newcommand{\Large}{\jsc@setfontsize\Large\@xivpt{21}}
1581 %<kiyou>\newcommand{\Large}{\jsc@setfontsize\Large{12.222}{21}}
1582 \newcommand{\LARGE}{\jsc@setfontsize\LARGE\@xvipt{25}}
1583 \newcommand{\huge}{\jsc@setfontsize\huge\@xxpt{28}}
1584 \newcommand{\Huge}{\jsc@setfontsize\Huge\@xxvpt{33}}
1585 \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。

```

1586 \everydisplay=\expandafter{\the\everydisplay \narrowbaselines}

```

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

見出し用のフォントは `\bfseries` 固定ではなく、`\headfont` という命令で定めることにします。これは太ゴシックが使えるときは `\sffamily \bfseries` でいいと思いますが、

通常の中ゴシックでは単に `\sffamily` だけのほうがよさそうです。『pLATEX 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。

```
1587 % \newcommand{\headfont}{\bfseries}
1588 \newcommand{\headfont}{\sffamily}
1589 % \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。

```
1590 %<!kiyou>\setlength\columnsep{2\Cwd}
1591 %<kiyou>\setlength\columnsep{28truebp}
1592 \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 1593 \setlength\lineskip{1\jsc@mp}
1594 \setlength\normallineskip{1\jsc@mp}
\normallineskiplimit 1595 \setlength\lineskiplimit{1\jsc@mp}
1596 \setlength\normallineskiplimit{1\jsc@mp}
```

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

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

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

```
1597 \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。

```
1598 \setlength\parskip{\z@}
1599 \if@slide
1600 \setlength\parindent{0\p@}
1601 \else
1602 \setlength\parindent{1\Cwd}
1603 \fi
```

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

```

1604 \@lowpenalty 51
1605 \@medpenalty 151
1606 \@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。

```
1607 % \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。

```
1608 % \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。

```

1609 \@onlypreamble\bxjs@bd@pre@geometry@hook
1610 \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。

```

1611 \ifjsc@mag
1612 \mag=\bxjs@param@mag
1613 \fi
1614 \setlength{\topskip}{10\jsc@empt}

```

`\jsSetQHLength` のための和文単位の定義。

```

1615 \def\bxjs@unit@trueQ{0.25trueem}\let\bxjs@unit@trueH\bxjs@unit@trueQ
1616 \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。

```

1617 \@tempwafalse
1618 \def\bxjs@tmpdo{\@ifnextchar\bgroup\bxjs@tmpdo@a\remove@to@nnil}
1619 \def\bxjs@tmpdo@a#1{\edef\bxjs@tmpa{#1}%
1620   \@ifnextchar\bgroup\bxjs@tmpdo@b\remove@to@nnil}
1621 \def\bxjs@tmpdo@b#1{\edef\bxjs@tmpa{\bxjs@tmpa,#1}%
1622   \@ifnextchar\@nnil\bxjs@tmpdo@c\remove@to@nnil}
1623 \def\bxjs@tmpdo@c\@nnil{\@tempwattrue
1624   \edef\bxjs@param@paper{papersize={\bxjs@tmpa}}}
1625 \expandafter\bxjs@tmpdo\bxjs@param@paper\@nnil

```

`W,H` の形式について。

```
1626 \if@tempwa\else
```

```

1627 \def\bxjs@tmpa{\@nil,\@nil}
1628 \def\bxjs@tmpdo#1,#2,#3\@nnil{%
1629   \def\bxjs@tmpb{#3}\ifx\bxjs@tmpa\bxjs@tmpb
1630     \@tempswatrue\edef\bxjs@param@paper{papersize={#1,#2}}\fi}
1631 \expandafter\bxjs@tmpdo\bxjs@param@paper,\@nil,\@nil\@nnil
1632 \fi

```

W*H の形式について。

```

1633 \if@tempswa\else
1634   \def\bxjs@tmpa{\@nil*\@nil}
1635   \def\bxjs@tmpdo#1*#2*#3\@nnil{%
1636     \def\bxjs@tmpb{#3}\ifx\bxjs@tmpa\bxjs@tmpb
1637       \@tempswatrue\edef\bxjs@param@paper{papersize={#1,#2}}\fi}
1638   \expandafter\bxjs@tmpdo\bxjs@param@paper*\@nil*\@nil\@nnil
1639 \fi

```

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

```

1640 \edef\bxjs@layout@paper{%
1641   \ifjsc@mag truedimen,\fi
1642   \if@landscape landscape,\fi
1643   \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。

```

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

```

```

1670 \fi %---
1671 %</book>
1672 %<*slide>
1673 \def\bxjs@layout@base{%
1674   noheadfoot,%
1675 }
1676 \edef\bxjs@layout{\bxjs@layout@base
1677   hscale=0.9,hmarginratio=1:1,%
1678   vscale=0.944,vmarginratio=1:1,%
1679 }
1680 %</slide>

```

textwidth オプションの設定を反映する。

```

1681 %<*!book>
1682 \ifx\bxjs@textwidth@opt\undefined\else
1683   \jsSetQHLlength\@tempdima{\bxjs@textwidth@opt}
1684   \edef\bxjs@layout{\bxjs@layout width=\the\@tempdima,}
1685 \fi
1686 %</!book>
1687 \ifx\bxjs@number@of@lines@opt\undefined\else
1688   \bxjs@gset@tempcnta{\bxjs@number@of@lines@opt}
1689   \edef\bxjs@layout{\bxjs@layout lines=\the\@tempcnta,}
1690 \fi

```

\fullwidth [寸法レジスタ] ヘッダ・フッタ領域の横幅。

```
1691 \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。

```

1692 %<*book>
1693 \newcommand\jsTextWidthLimit{40}
1694 \@tempdima=\jsTextWidthLimit\Cwd
1695 \ifx\bxjs@textwidth@limit@opt\undefined\else
1696   \bxjs@gset@tempcnta{\bxjs@textwidth@limit@opt}
1697   \@tempdima=\@tempcnta\Cwd
1698 \fi
1699 \ifx\bxjs@textwidth@opt\undefined\else
1700   \jsSetQHLlength\@tempdima{\bxjs@textwidth@opt}
1701 \fi
1702 \edef\bxjs@textwidth@limit{\the\@tempdima}
1703 \ifdim\@tempdima=\jsTextWidthLimit\Cwd\else
1704   \bxjs@invscale\@tempdima{\strip@pt\Cwd}
1705   \bxjs@nclong\edef\jsTextWidthLimit{\strip@pt\@tempdima}
1706 \fi
1707 %</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。

```
1708 \def\bxjs@preproc@layout{%
1709   \edef\bxjs@save@ht@strutbox{\the\ht\strutbox}\ht\strutbox=10\jsc@empt}
```

\bxjs@postproc@layout geometry の後処理。

```
1710 \def\bxjs@postproc@layout{%
```

geometry のドライバを再設定する。

```
1711 \ifx\bxjs@geometry@driver\relax\else
1712   \let\Gm@driver\bxjs@geometry@driver
1713 \fi
```

\ht\strutbox の値を元に戻す。

```
1714 \ht\strutbox=\bxjs@save@ht@strutbox\relax
```

\textwidth の値を補正する。

```
1715 \ifbxjs@whole@zw@lines
1716   \@tempdima=\textwidth
1717   \if@twocolumn \@tempdima=2\Cwd \else \@tempdima=1\Cwd \fi
1718   \advance\textwidth.005pt\relax
1719   \divide\textwidth\@tempdima \multiply\textwidth\@tempdima
1720   \advance\@tempdima-\textwidth
1721   \advance\oddsidemargin 0.5\@tempdima
1722   \advance\evensidemargin 0.5\@tempdima
1723 \fi
1724 \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。

```
1725 %<*book>
1726 \@tempdima=\bxjs@textwidth@limit\relax
1727 \ifbxjs@whole@zw@lines
1728   \advance\@tempdima.005pt\relax
1729   \divide\@tempdima\Cwd \multiply\@tempdima\Cwd
1730 \fi
1731 \ifdim\textwidth>\@tempdima
1732   \textwidth=\@tempdima
1733   \addtolength\evensidemargin{\fullwidth-\textwidth}
1734 \fi
1735 %</book>
```

\textheight 関連の調整。

```
1736 \@tempdima=\textheight
1737 \advance\textheight-\topskip
1738 \advance\textheight.005pt\relax
1739 \divide\textheight\baselineskip \multiply\textheight\baselineskip
1740 \advance\textheight\topskip
1741 \advance\@tempdima-\textheight
1742 \advance\topmargin0.5\@tempdima
```

`\headheight` 関連の調整。

```
1743 \@tempdima=\topskip
1744 \advance\headheight\@tempdima
1745 \advance\topmargin-\@tempdima
```

`marginpar` 関連の調整。

```
1746 \setlength\marginparsep{\columnsep}
1747 \setlength\marginparpush{\baselineskip}
1748 \setlength\marginparwidth{\paperwidth-\oddsidemargin-1truein%
1749   -\textwidth-10\jsc@mmm-\marginparsep}
1750 \ifbxjs@whole@zw@lines
1751   \divide\marginparwidth\Cwd \multiply\marginparwidth\Cwd
1752 \fi
```

連動する変数。

```
1753 \maxdepth=.5\topskip
1754 \stockwidth=\paperwidth
1755 \stockheight=\paperheight
1756 }
```

`\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。

```
1757 \edef\jsGeometryOptions{%
1758   \bxjs@layout@paper,\bxjs@layout}
```

■`geometry` パッケージを読み込む 🐼

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

※ \LaTeX のフックシステムがある場合はムニャムニャ。

```
1759 \def\bxjs@geometry@name{geometry}
1760 \ifbxjs@old@hook@system
1761   \let\bxjs@apply@bd@pre@geometry@hook\AtBeginDocument
1762 \else
1763   \def\bxjs@apply@bd@pre@geometry@hook{%
1764     \AddToHook{begindocument}{\bxjs@geometry@name}}
1765 \fi
```

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

```
1766 \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。

```
1767 \ifbxjs@papersize
1768   \ifx\bxjs@driver@given\bxjs@driver@@dvipdfmx
1769     \PassOptionsToPackage{dvipdfm}{geometry}
1770   \else\ifx\bxjs@driver@given\bxjs@driver@@dvimode
```

```

1771 \PassOptionsToPackage{dvipdfm}{geometry}
1772 \fi\fi
1773 \let\bxPapersizeSpecialDone=t
1774 \else
1775 \PassOptionsToPackage{driver=none}{geometry}
1776 \fi

```

ここで geometry を読み込む。

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

```

1777 \bxjs@apply@bd@pre@geometry@hook{%
1778 \bxjs@bd@pre@geometry@hook
1779 \@nameuse{ImposeOldLuaTeXBehavior}}
1780 \bxjs@preproc@layout
1781 \edef\bxjs@next{%
1782 \noexpand\RequirePackage[\bxjs@layout@paper,\bxjs@layout]{geometry}%
1783 }\bxjs@next
1784 \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。

```

1785 \let\bxjs@geometry@driver\Gm@driver
1786 \bxjs@postproc@layout

```

geometry のドライバ自動判別に対する前処理。

```

1787 \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。

```

1788 \@ifpackagelater{geometry}{2010/02/12}{\else
1789 \PackageError\bxjs@clsname
1790 {Your 'geometry' package is too old (< v5.0)}}%
1791 {\@ehc}%
1792 \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。

```

1793 \ifjsWithpTeXng
1794 \ifx\Gm@driver\@empty
1795 \def\Gm@driver{pdftex}%
1796 \fi
1797 \fi}

```

`\setpagelayout` ページレイアウト設定のためのユーザ命令。

```

1798 \def\setpagelayout{%
1799 \bxjs@ifplus{\bxjs@setpagelayout@a\tw@}{\else
1800 \@ifstar{\bxjs@setpagelayout@a\@ne}{\bxjs@setpagelayout@a\@z}}
1801 \def\bxjs@setpagelayout@a#1#2{%
1802 \ifcase#1% modify
1803 \def\bxjs@next{\ifjsc@mag truedimen,\fi #2}%
1804 \or% reset(*)

```

```

1805 \def\bxjs@next{reset,\bxjs@layout@paper,#2}%
1806 \or% semireset(+)
1807 \def\bxjs@next{reset,\bxjs@layout@paper,\bxjs@layout@base,#2}%
1808 \fi
1809 \bxjs@preproc@layout
1810 \edef\bxjs@next{%
1811 \noexpand\geometry{\bxjs@next}%
1812 }\bxjs@next
1813 \bxjs@postproc@layout}

```

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

```
1814 \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。

```

1815 \g@addto@macro\bxjs@begin@document@hook{%
1816 \ifdim\textwidth=.5\maxdimen
1817 \ClassError\bxjs@clsname
1818 {Page layout is not properly set}%
1819 {\@ehd}%
1820 \fi}
1821 \def\jsUseMinimalPageLayout{%
1822 \setlength{\textwidth}{6.5in}%
1823 \setlength{\textheight}{8in}}

```

\setpagelayout はとりあえず無効にしておく。

```

1824 \let\bxjs@geometry@driver\relax
1825 \def\setpagelayout{%
1826 \bxjs@ifplus{\bxjs@pagelayout@a}{\%else
1827 \@ifstar{\bxjs@pagelayout@a}{\bxjs@pagelayout@a}}
1828 \def\bxjs@pagelayout@a#1{%
1829 \ClassError\bxjs@clsname
1830 {Command '\string\setpagelayout' is not supported,\MessageBreak
1831 because 'geometry' value is not 'class'}\@eha}
1832 %
1833 \fi\fi

```

■縦方向のスペース

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

```
1834 %<*jsclasses>
```

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

にすると、本文中に \int のような高い文字が入ったときに 1 行目のベースラインが他のページより下がってしまいます。ここでは本文の公称フォントサイズ (10pt) にします。

[2003-06-26] `\headheight` はヘッダの高さで、元は 12pt ですが、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。

```
1835 \setlength\topskip{1.38zw}%% from 10\jsc@mpt (2016-08-17)
1836 \if@slide
1837   \setlength\headheight{0\jsc@mpt}
1838 \else
1839   \setlength\headheight{20\jsc@mpt}%% from 2\topskip (2016-08-17); from \topskip (2003-
      06-26)
1840 \fi
```

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

```
1841 %<*article|kiyou>
1842 \if@slide
1843   \setlength\footskip{0pt}
1844 \else
1845   \setlength\footskip{0.03367\paperheight}
1846   \ifdim\footskip<\baselineskip
1847     \setlength\footskip{\baselineskip}
1848   \fi
1849 \fi
1850 %</article|kiyou>
1851 %<jspf>\setlength\footskip{9\jsc@mmm}
1852 %<*book>
1853 \if@report
1854   \setlength\footskip{0.03367\paperheight}
1855   \ifdim\footskip<\baselineskip
1856     \setlength\footskip{\baselineskip}
1857   \fi
1858 \else
1859   \setlength\footskip{0pt}
1860 \fi
1861 %</book>
1862 %<*report>
1863 \setlength\footskip{0.03367\paperheight}
1864 \ifdim\footskip<\baselineskip
1865   \setlength\footskip{\baselineskip}
1866 \fi
```

1867 %</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)。

```
1868 %<*article>
1869 \if@slide
1870   \setlength\headsep{0\jsc@empt}
1871   \addtolength\headsep{-\topskip}%% added (2016-10-08)
1872   \addtolength\headsep{10\jsc@empt}%% added (2016-10-08)
1873 \else
1874   \setlength\headsep{\footskip}
1875   \addtolength\headsep{-\topskip}
1876 \fi
1877 %</article>
1878 %<*book>
1879 \if@report
1880   \setlength\headsep{\footskip}
1881   \addtolength\headsep{-\topskip}
1882 \else
1883   \setlength\headsep{6\jsc@mmm}
1884   \addtolength\headsep{-\topskip}%% added (2016-10-08)
1885   \addtolength\headsep{10\jsc@empt}%% added (2016-10-08)
1886 \fi
1887 %</book>
1888 %<*report>
1889 \setlength\headsep{\footskip}
1890 \addtolength\headsep{-\topskip}
1891 %</report>
1892 %<*jspf>
1893 \setlength\headsep{9\jsc@mmm}
1894 \addtolength\headsep{-\topskip}
1895 %</jspf>
1896 %<*kiyou>
1897 \setlength\headheight{0\jsc@empt}
1898 \setlength\headsep{0\jsc@empt}
1899 \addtolength\headsep{-\topskip}%% added (2016-10-08)
1900 \addtolength\headsep{10\jsc@empt}%% added (2016-10-08)
1901 %</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_EX 2e では `\maxdepth + \topskip` を本文フォントサイズの 1.5 倍にしたいのですが, `\topskip` は本文フォントサイズ (ここでは 10pt) に等しいので, 結局 `\maxdepth` は `\topskip` の半分の値 (具体的には 5pt) にします。

```
1902 \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。

```
1903 \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。

```
1904 %<*article>
1905 \if@slide
1906   \setlength\fullwidth{0.9\paperwidth}
1907 \else
1908   \setlength\fullwidth{0.76\paperwidth}
1909 \fi
1910 \if@twocolumn \@tempdima=2zw \else \@tempdima=1zw \fi
1911 \divide\fullwidth\@tempdima \multiply\fullwidth\@tempdima
1912 \setlength\textwidth{\fullwidth}
1913 %</article>
1914 %<*book>
1915 \if@report
1916   \setlength\fullwidth{0.76\paperwidth}
1917 \else
1918   \setlength\fullwidth{\paperwidth}
1919   \addtolength\fullwidth{-36\jsc@mmm}
1920 \fi
1921 \if@twocolumn \@tempdima=2zw \else \@tempdima=1zw \fi
1922 \divide\fullwidth\@tempdima \multiply\fullwidth\@tempdima
1923 \setlength\textwidth{\fullwidth}
1924 \if@report \else
1925   \if@twocolumn \else
1926     \ifdim \fullwidth>40zw
1927       \setlength\textwidth{40zw}
1928     \fi
1929   \fi
1930 \fi
1931 %</book>
1932 %<*report>
1933 \setlength\fullwidth{0.76\paperwidth}
1934 \if@twocolumn \@tempdima=2zw \else \@tempdima=1zw \fi
```

```

1935 \divide\fullwidth\@tempdima \multiply\fullwidth\@tempdima
1936 \setlength\textwidth{\fullwidth}
1937 %</report>
1938 %<*jspf>
1939 \setlength\fullwidth{50zw}
1940 \addtolength\fullwidth{8\jsc@mmm}
1941 \setlength\textwidth{\fullwidth}
1942 %</jspf>
1943 %<*kiyou>
1944 \setlength\fullwidth{48zw}
1945 \addtolength\fullwidth{\columnsep}
1946 \setlength\textwidth{\fullwidth}
1947 %</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つたので、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)。

```

1948 %<*article|book|report>
1949 \if@slide
1950 \setlength{\textheight}{0.95\paperheight}
1951 \else
1952 \setlength{\textheight}{0.83\paperheight}
1953 \fi
1954 \addtolength{\textheight}{-10\jsc@mpt}%% from -\topskip (2016-10-08); from -
\headheight (2003-06-26)
1955 \addtolength{\textheight}{-\headsep}
1956 \addtolength{\textheight}{-\footskip}
1957 \addtolength{\textheight}{-\topskip}
1958 \divide\textheight\baselineskip
1959 \multiply\textheight\baselineskip
1960 %</article|book|report>
1961 %<jspf>\setlength{\textheight}{51\baselineskip}
1962 %<kiyou>\setlength{\textheight}{47\baselineskip}
1963 \addtolength{\textheight}{\topskip}
1964 \addtolength{\textheight}{0.1\jsc@mpt}

```



```
1965 %<jspf>\setlength{\mathindent}{10\jsc@mmm}
```

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

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

ですが、次のようにします。

```
1966 \def\flushbottom{%
1967   \def\@textbottom{\vskip \z@ \@plus.1\jsc@empt}%
1968   \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。

```
1969 \setlength\marginparsep{\columnsep}
1970 \setlength\marginparpush{\baselineskip}
```

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

```
1971 \setlength{\oddsidemargin}{\paperwidth}
1972 \addtolength{\oddsidemargin}{-\fullwidth}
1973 \setlength{\oddsidemargin}{.5\oddsidemargin}
1974 \iftombow
1975   \addtolength{\oddsidemargin}{-1in}
1976 \else
1977   \addtolength{\oddsidemargin}{-\inv@mag in}
1978 \fi
1979 \setlength{\evensidemargin}{\oddsidemargin}
1980 \if@mparswitch
1981   \addtolength{\evensidemargin}{\fullwidth}
1982   \addtolength{\evensidemargin}{-\textwidth}
1983 \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。

```
1984 \setlength\marginparwidth{\paperwidth}
1985 \addtolength\marginparwidth{-\oddsidemargin}
1986 \addtolength\marginparwidth{-\inv@mag in}
1987 \addtolength\marginparwidth{-\textwidth}
1988 \addtolength\marginparwidth{-10\jsc@mmm}
1989 \addtolength\marginparwidth{-\marginparsep}
1990 \@tempdima=1zw
1991 \divide\marginparwidth\@tempdima
```

```
1992 \multiply\marginparwidth\@tempdima
```

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

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

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

```
1993 \setlength\topmargin{\paperheight}
1994 \addtolength\topmargin{-\textheight}
1995 \if@slide
1996   \addtolength\topmargin{-\headheight}
1997 \else
1998   \addtolength\topmargin{-10\jsc@empt}%% from -\topskip (2016-10-08); from -
      \headheight (2003-06-26)
1999 \fi
2000 \addtolength\topmargin{-\headsep}
2001 \addtolength\topmargin{-\footskip}
2002 \setlength\topmargin{0.5\topmargin}
2003 %<kiyou>\setlength\topmargin{81truebp}
2004 \iftombow
2005   \addtolength\topmargin{-1in}
2006 \else
2007   \addtolength\topmargin{-\inv@mag in}
2008 \fi
2009 %</jsclasses>
```

■脚注

ここからのコードは以下の点を除いて JS クラスのものを踏襲する。

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

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

ここは元々は

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

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

2010 `\footnotesep=11\jsc@mp \footnotesep=0.7\footnotesep`

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

2011 `\setlength{\skip\footins}{16\jsc@mp \@plus 5\jsc@mp \@minus 2\jsc@mp}`

■**フロート関連** フロート（図、表）関連のパラメータは L^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。

2012 `\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。

2013 `\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。

2014 `\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。

2015 `\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。

2016 `\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。

2017 `\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。

2018 `\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。

2019 `\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。

2020 `\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。

```
2021 \renewcommand{\dblfloatpagefraction}{.8}
```

`\floatsep` `\floatsep` はページ上部・下部のフロート間の距離です。`\textfloatsep` はページ上部・
`\textfloatsep` 下部のフロートと本文との距離です。`\intextsep` は本文の途中に出力されるフロートと本
`\intextsep` 文との距離です。

```
2022 \setlength\floatsep {12\jsc@empt \@plus 2\jsc@empt \@minus 2\jsc@empt}
```

```
2023 \setlength\textfloatsep{20\jsc@empt \@plus 2\jsc@empt \@minus 4\jsc@empt}
```

```
2024 \setlength\intextsep {12\jsc@empt \@plus 2\jsc@empt \@minus 2\jsc@empt}
```

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

```
\dbltextfloatsep 2025 \setlength\dblfloatsep {12\jsc@empt \@plus 2\jsc@empt \@minus 2\jsc@empt}
```

```
2026 \setlength\dbltextfloatsep{20\jsc@empt \@plus 2\jsc@empt \@minus 4\jsc@empt}
```

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

`\fpsep` `\fpsep` はフロート間に入ります。

```
\fpbot 2027 \setlength\fpsep{0\jsc@empt \@plus 1fil}
```

```
2028 \setlength\fpsep{8\jsc@empt \@plus 2fil}
```

```
2029 \setlength\fpbot{0\jsc@empt \@plus 1fil}
```

`\dblfpsep` 段抜きフロートについての値です。

```
\dblfpsep 2030 \setlength\dblfpsep{0\jsc@empt \@plus 1fil}
```

```
\dblfpbot 2031 \setlength\dblfpsep{8\jsc@empt \@plus 2fil}
```

```
2032 \setlength\dblfpbot{0\jsc@empt \@plus 1fil}
```

6 改ページ（日本語 T_EX 開発コミュニティ版のみ）

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

`\pltx@cleartooddpage`

`\pltx@cleartoevenpage`

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

となっています。

```
2033 %\def\pltx@cleartorightpage{\clearpage\if@twoside
```

```
2034 % \ifodd\c@page
```

```
2035 % \iftdir
```

```
2036 % \hbox{}\thispagestyle{empty}\newpage
```

```
2037 % \if@twocolumn\hbox{}\newpage\fi
```

```
2038 % \fi
```

```
2039 % \else
```

```
2040 % \ifydir
```

```
2041 % \hbox{}\thispagestyle{empty}\newpage
```

```

2042 %      \if@twocolumn\hbox{}\newpage\fi
2043 %      \fi
2044 % \fi\fi}
2045 %\def\pltx@cleartoleftpage{\clearpage\if@twoside
2046 % \ifodd\c@page
2047 % \ifydir
2048 % \hbox{}\thispagestyle{empty}\newpage
2049 % \if@twocolumn\hbox{}\newpage\fi
2050 % \fi
2051 % \else
2052 % \iftdir
2053 % \hbox{}\thispagestyle{empty}\newpage
2054 % \if@twocolumn\hbox{}\newpage\fi
2055 % \fi
2056 % \fi\fi}
2057 \def\pltx@cleartooddpage{\clearpage\if@twoside
2058 \ifodd\c@page\else
2059 \hbox{}\thispagestyle{empty}\newpage
2060 \if@twocolumn\hbox{}\newpage\fi
2061 \fi\fi}
2062 \def\pltx@cleartoevenpage{\clearpage\if@twoside
2063 \ifodd\c@page
2064 \hbox{}\thispagestyle{empty}\newpage
2065 \if@twocolumn\hbox{}\newpage\fi
2066 \fi\fi}

```

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

```

2067 \let\pltx@cleartorightpage\pltx@cleartooddpage
2068 \let\pltx@cleartoleftpage\pltx@cleartoevenpage

\size の値がアレな場合は本体開始まで \clearpage を無効にする。
2069 \ifdim\size=\z@
2070 \begingroup
2071 \toks@{\expandafter{\clearpage}}
2072 \xdef\clearpage{\noexpand\ifbxjs@after@preamble\the\toks@\noexpand\fi}
2073 \endgroup
2074 \fi

```

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

```

2075 %<*book|report>
2076 \if@openleft
2077 \let\cleardoublepage\pltx@cleartoleftpage
2078 \else\if@openright
2079 \let\cleardoublepage\pltx@cleartorightpage
2080 \fi\fi
2081 %</book|report>

```

7 ページスタイル

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

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

ページスタイルは `\ps@...` の形のマクロで定義されています。

`\@evenhead` `\@oddhead`, `\@oddfoot`, `\@evenhead`, `\@evenfoot` は偶数・奇数ページの柱 (ヘッダ, フッタ) を出力する命令です。これらは `\fullwidth` 幅の `\hbox` の中で呼び出されます。
`\@evenfoot` `\ps@...` の中で定義しておきます。

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

<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。 \LaTeX 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。

```
2082 % \def\ps@empty{%
2083 %   \let\mkboth\gobbletwo
2084 %   \let\@oddhead\@empty
2085 %   \let\@oddfoot\@empty
2086 %   \let\@evenhead\@empty
2087 %   \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。

```
2088 \def\ps@plainfoot{%
2089   \let\mkboth\gobbletwo
2090   \let\@oddhead\@empty
2091   \def\@oddfoot{\normalfont\hfil\thepage\hfil}%
2092   \let\@evenhead\@empty
```

```

2093 \let\@evenfoot\@oddfoot}
2094 \def\ps@plainhead{%
2095 \let\@mkboth\@gobbletwo
2096 \let\@oddfoot\@empty
2097 \let\@evenfoot\@empty
2098 \def\@evenhead{%
2099 \if@mparswitch \hss \fi
2100 \hbox to \fullwidth{\textbf{\thepage}\hfil}%
2101 \if@mparswitch\else \hss \fi}%
2102 \def\@oddhead{%
2103 \hbox to \fullwidth{\hfil\textbf{\thepage}}\hss}%
2104 %<book>\let\ps@plain\ps@plainhead
2105 %<!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。

```

2106 %<*article|slide>
2107 \if@twoside
2108 \def\ps@headings{%
2109 \let\@oddfoot\@empty
2110 \let\@evenfoot\@empty
2111 \def\@evenhead{\if@mparswitch \hss \fi
2112 \underline{\hbox to \fullwidth{\textbf{\thepage}\hfil\leftmark}}}%
2113 \if@mparswitch\else \hss \fi}%
2114 \def\@oddhead{%
2115 \underline{%
2116 \hbox to \fullwidth{\rightmark}\hfil\textbf{\thepage}}\hss}%
2117 \let\@mkboth\markboth
2118 \def\sectionmark##1{\markboth{%
2119 \ifnum \c@secnumdepth >\z@ \bxjs@label@sect{section}\hskip1\jsZw\fi
2120 ##1}}}%
2121 \def\subsectionmark##1{\markright{%
2122 \ifnum \c@secnumdepth >\@ne \bxjs@label@sect{subsection}\hskip1\jsZw\fi
2123 ##1}}}%
2124 }
2125 \else % if not twoside
2126 \def\ps@headings{%
2127 \let\@oddfoot\@empty
2128 \def\@oddhead{%
2129 \underline{%
2130 \hbox to \fullwidth{\rightmark}\hfil\textbf{\thepage}}\hss}%
2131 \let\@mkboth\markboth
2132 \def\sectionmark##1{\markright{%
2133 \ifnum \c@secnumdepth >\z@ \bxjs@label@sect{section}\hskip1\jsZw\fi
2134 ##1}}}%
2135 \fi
2136 %</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）。

2137 %<*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。

2138 `\def\bxjs@maybe@autoxspacing{%`

2139 `\ifx\autoxspacing\undefined\else \autoxspacing \fi}`

2140 `\newif\if@omit@number`

2141 `\def\ps@headings{%`

2142 `\let\@oddfoot\@empty`

2143 `\let\@evenfoot\@empty`

2144 `\def\@evenhead{%`

2145 `\if@mparswitch \hss \fi`

2146 `\underline{\hbox to \fullwidth{\bxjs@maybe@autoxspacing`

2147 `\textbf{\thepage}\hfil\leftmark}}%`

2148 `\if@mparswitch\else \hss \fi}%`

2149 `\def\@oddhead{\underline{\hbox to \fullwidth{\bxjs@maybe@autoxspacing`

2150 `{\if@twoside\rightmark\else\leftmark\fi}\hfil\textbf{\thepage}}}\hss}%`

2151 `\let\@mkboth\markboth`

2152 `\def\chaptermark##1{\markboth{%`

2153 `\ifnum \c@secnumdepth >\m@ne`

2154 `\if@mainmatter`

2155 `\if@omit@number\else`

2156 `\@chapapp\thechapter\@chappos\hskip1\jsZw`

2157 `\fi`

2158 `\fi`

2159 `\fi`

2160 `##1}}}%`

2161 `\def\sectionmark##1{\markright{%`

2162 `\ifnum \c@secnumdepth >\z@ \bxjs@label@sect{section}\hskip1\jsZw\fi`

2163 `##1}}}%`

2164 %</book|report>

最後は学会誌の場合です。

2165 %<*jspf>

2166 `\def\ps@headings{%`

2167 `\def\@oddfoot{\normalfont\hfil\thepage\hfil}`

2168 `\def\@evenfoot{\normalfont\hfil\thepage\hfil}`

2169 `\def\@oddhead{\normalfont\hfil \@title \hfil}`

2170 `\def\@evenhead{\normalfont\hfil プラズマ・核融合学会誌\hfil}`

2171 %</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。

```
2172 \def\ps@myheadings{%
2173   \let\@oddfoot\@empty\let\@evenfoot\@empty
2174   \def\@evenhead{%
2175     \if@mparswitch \hss \fi%
2176     \hbox to \fullwidth{\thepage\hfil\leftmark}%
2177     \if@mparswitch\else \hss \fi}%
2178   \def\@oddhead{%
2179     \hbox to \fullwidth{\rightmark\hfil\thepage}\hss}%
2180   \let\@mkboth\@gobbletwo
2181   %<book|report> \let\chaptermark\@gobble
2182   \let\sectionmark\@gobble
2183   %<!book&!report> \let\subsectionmark\@gobble
2184 }
```

8 文書のマークアップ

`\bxjs@phantomsection` `hyperref` が読み込まれている場合に `\phantomsection` を実行する。

```
2185 \let\bxjs@phantomsection\relax
2186 \g@addto@macro\bxjs@begin@document@hook{%
2187   \@ifpackageloaded{hyperref}{%
2188     \let\bxjs@phantomsection\phantomsection
2189   }{}%
2190 }
```

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 2191 % \newcommand*{\title}[1]{\gdef\@title{#1}}
2192 % \newcommand*{\author}[1]{\gdef\@author{#1}}
2193 % \newcommand*{\date}[1]{\gdef\@date{#1}}
2194 % \date{\today}
```

`\subtitle` 副題を設定する。

`\jsSubtitle` ※プレアンブルにおいて `\newcommand*{\subtitle}{...}` が行われることへの対策として、`\subtitle` の定義を `\title` の実行まで遅延させることにする。もしどうしても主題より前に副題を設定したい場合は、`\jsSubtitle` 命令を直接用いばよい。

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

本体を `\jsSubtitle` として定義する。

```

2195 \newcommand*{\jsSubtitle}[1]{\gdef\bxjs@subtitle{#1}}
2196 %\let\bxjs@subtitle\@undefined

\title にフックを入れる。

2197 \renewcommand*{\title}[1]{\bxjs@decl@subtitle\gdef\@title{#1}}
2198 \g@addto@macro\bxjs@begin@document@hook{\bxjs@decl@subtitle}
2199 \def\bxjs@decl@subtitle{%
2200   \global\let\bxjs@decl@subtitle\relax
2201   \ifx\subtitle\@undefined
2202     \global\let\subtitle\jsSubtitle
2203   \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。

```

2204 \def\bxjs@annihilate@subtitle{%
2205   \ifx\subtitle\jsSubtitle \global\let\subtitle\relax \fi
2206   \global\let\jsSubtitle\relax}

```

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

```

\author 2207 %<*jspf>
\keywords 2208 \newcommand*{\etitle}[1]{\gdef\@etitle{#1}}
2209 \newcommand*{\eauthor}[1]{\gdef\@eauthor{#1}}
2210 \newcommand*{\keywords}[1]{\gdef\@keywords{#1}}
2211 \newcommand*{\email}[1]{\gdef\authors@mail{#1}}
2212 \newcommand*{\AuthorsEmail}[1]{\gdef\authors@mail{author's e-mail:\ #1}}
2213 %</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。

```

2214 \def\plainifnotempty{%
2215   \ifx \@oddhead \@empty
2216     \ifx \@oddfoot \@empty
2217       \else
2218         \thispagestyle{plainfoot}%
2219       \fi
2220   \else
2221     \thispagestyle{plainhead}%
2222   \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。

```
2223 %<*article|book|report|slide>
2224 \if@titlepage
2225   \newcommand{\maketitle}{%
2226     \begin{titlepage}%
2227       \let\footnotesize\small
2228       \let\footnoterule\relax
2229       \let\footnote\thanks
2230       \null\vfil
2231       \if@slide
2232         {\footnotesize \@date}%
2233         \begin{center}
2234           \mbox{} \\\[1\jsZw]
2235           \large
2236           {\maybeblue\hrule height0\jsc@mpt depth2\jsc@mpt\relax}\par
2237           \jsc@smallskip
2238           \@title
2239           \ifx\bxjs@subtitle\@undefined\else
2240             \par\vskip\z@
2241             {\small \bxjs@subtitle\par}
2242           \fi
2243           \jsc@smallskip
2244           {\maybeblue\hrule height0\jsc@mpt depth2\jsc@mpt\relax}\par
2245           \vfill
2246           {\small \@author}%
2247         \end{center}
2248       \else
2249         \vskip 60\jsc@mpt
2250         \begin{center}%
2251           {\LARGE \@title \par}%
2252           \ifx\bxjs@subtitle\@undefined\else
2253             \vskip5\jsc@mpt
2254             {\normalsize \bxjs@subtitle\par}
2255           \fi
2256           \vskip 3em%
2257           {\large
2258             \lineskip .75em
2259             \begin{tabular}[t]{c}%
2260               \@author
2261             \end{tabular}\par}%
2262           \vskip 1.5em
2263           {\large \@date \par}%
2264         \end{center}%
2265       \fi
2266     \par
2267     \@thanks\vfil\null
2268   \end{titlepage}%
2269   \setcounter{footnote}{0}%
```

```

2270 \global\let\thanks\relax
2271 \global\let\maketitle\relax
2272 \global\let\@thanks\@empty
2273 \global\let\@author\@empty
2274 \global\let\@date\@empty
2275 \global\let\@title\@empty
2276 \global\let\title\relax
2277 \global\let\author\relax
2278 \global\let\date\relax
2279 \global\let\and\relax
2280 \bxjs@annihilate@subtitle
2281 }%
2282 \else
2283 \newcommand{\maketitle}{\par
2284 \begin{group}
2285 \renewcommand\thefootnote{\@fnsymbol\c@footnote}%
2286 \def\@makefnmark{\rlap{\@textsuperscript{\normalfont\@thefnmark}}}%
2287 \long\def\@makefntext##1{\advance\leftskip 3\jsZw
2288 \parindent 1\jsZw\noindent
2289 \llap{\@textsuperscript{\normalfont\@thefnmark}\hskip0.3\jsZw}##1}%
2290 \if@twocolumn
2291 \ifnum \col@number=\@one
2292 \@maketitle
2293 \else
2294 \twocolumn[\@maketitle]%
2295 \fi
2296 \else
2297 \newpage
2298 \global\@topnum\z@ % Prevents figures from going at top of page.
2299 \@maketitle
2300 \fi
2301 \plainifnotempty
2302 \@thanks
2303 \endgroup
2304 \setcounter{footnote}{0}%
2305 \global\let\thanks\relax
2306 \global\let\maketitle\relax
2307 \global\let\@thanks\@empty
2308 \global\let\@author\@empty
2309 \global\let\@date\@empty
2310 \global\let\@title\@empty
2311 \global\let\title\relax
2312 \global\let\author\relax
2313 \global\let\date\relax
2314 \global\let\and\relax
2315 \bxjs@annihilate@subtitle
2316 }

```

`\@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。

```

2317 \def\@maketitle{%
2318   \newpage\null
2319   \vskip 2em
2320   \begin{center}%
2321     \let\footnote\thanks
2322     {\LARGE \@title \par}%
2323     \ifx\bxjs@subtitle\@undefined\else
2324       \vskip3\jsc@mpt
2325       {\normalsize \bxjs@subtitle\par}
2326     \fi
2327     \vskip 1.5em
2328     {\large
2329       \lineskip .5em
2330       \begin{tabular}[t]{c}%
2331         \@author
2332       \end{tabular}\par}%
2333     \vskip 1em
2334     {\large \@date}%
2335   \end{center}%
2336   \par\vskip 1.5em
2337 %<article|slide> \ifvoid\@abstractbox\else\centerline{\box\@abstractbox}\vskip1.5em\fi
2338 }
2339 \fi
2340 %</article|book|report|slide>
2341 %<*jspf>
2342 \newcommand{\maketitle}{\par
2343   \begingroup
2344     \renewcommand\thefootnote{\@fnsymbol\c@footnote}%
2345     \def\@makefnmark{\rlap{\@textsuperscript{\normalfont\@thefnmark}}}%
2346     \long\def\@makefntext##1{\advance\leftskip 3\jsZw
2347       \parindent 1\jsZw\noindent
2348       \llap{\@textsuperscript{\normalfont\@thefnmark}\hskip0.3\jsZw}##1}%
2349     \twocolumn[\@maketitle]%
2350     \plainifnotempty
2351     \@thanks
2352   \endgroup
2353   \setcounter{footnote}{0}%
2354   \global\let\thanks\relax
2355   \global\let\maketitle\relax
2356   \global\let\@thanks\@empty
2357   \global\let\@author\@empty
2358   \global\let\@date\@empty
2359 % \global\let\@title\@empty % \@title は柱に使う
2360   \global\let\title\relax
2361   \global\let\author\relax
2362   \global\let\date\relax
2363   \global\let\and\relax

```

```

2364 \ifx\authors@mail\@undefined\else{%
2365 \def\@makefntext{\advance\leftskip 3\jsZw \parindent -3\jsZw}%
2366 \footnotetext[0]{\itshape\authors@mail}%
2367 }\fi
2368 \global\let\authors@mail\@undefined}
2369 \def\@maketitle{%
2370 \newpage\null
2371 \vskip 6em % used to be 2em
2372 \begin{center}
2373 \let\footnote\thanks
2374 \ifx\@title\@undefined\else{\LARGE\headfont\@title\par}\fi
2375 \lineskip .5em
2376 \ifx\@author\@undefined\else
2377 \vskip 1em
2378 \begin{tabular}[t]{c}%
2379 \@author
2380 \end{tabular}\par
2381 \fi
2382 \ifx\@etitle\@undefined\else
2383 \vskip 1em
2384 {\large \@etitle \par}%
2385 \fi
2386 \ifx\@eauthor\@undefined\else
2387 \vskip 1em
2388 \begin{tabular}[t]{c}%
2389 \@eauthor
2390 \end{tabular}\par
2391 \fi
2392 \vskip 1em
2393 \@date
2394 \end{center}
2395 \vskip 1.5em
2396 \centerline{\box\@abstractbox}
2397 \ifx\@keywords\@undefined\else
2398 \vskip 1.5em
2399 \centerline{\parbox{157\jsc@mmm}{\textsf{Keywords:}}\small\@keywords}}
2400 \fi
2401 \vskip 1.5em}
2402 %</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。

```

2403 \def\bxjs@label@sect#1{%

```

```

2404 \ifundefined{label#1}{\@nameuse{the#1}}{\@nameuse{label#1}}
2405 \def\@seccntformat#1{\bxjs@label@sect{#1}\quad}

```

\@secapp 節番号の接頭辞。

\@secpos 節番号の接尾辞。

```

2406 \ifnum\bxjs@label@section=\bxjs@label@section@compat\else
2407 \def\@secapp{\presectionname}
2408 \def\@secpos{\postsectionname}
2409 \fi

```

\labelsection 節番号の出力書式。

```

2410 \ifnum\bxjs@label@section=\bxjs@label@section@modern
2411 \def\labelsection{\@secapp\thesection\@secpos}
2412 \fi

```

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

```

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

```

それぞれの引数の意味は次の通りです。

名 ユーザレベルコマンドの名前です (例: section)。

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

字下げ 見出しの字下げ量です。

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

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

スタイル 見出しの文字スタイルの設定です。

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

別見出し 目次や柱に出力する見出しです。

見出し 見出しです。

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

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

```

2413 \def\@startsection#1#2#3#4#5#6{%
2414 \ifnoskipsec \leavevmode \fi
2415 \par

```

```

2416 % 見出し上の空きを \@tempskipa にセットする
2417 \@tempskipa #4\relax
2418 % \@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
2419 \if@english \@afterindentfalse \else \@afterindenttrue \fi
2420 % 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
2421 \ifdim \@tempskipa <\z@
2422   \@tempskipa -\@tempskipa \@afterindentfalse
2423 \fi
2424 \if@nobreak
2425 % \everypar{\everyparhook}% これは間違い
2426 \everypar{}%
2427 \else
2428 \addpenalty\@secpenalty
2429 % 次の行は削除
2430 % \addvspace\@tempskipa
2431 % 次の \noindent まで追加
2432 \ifdim \@tempskipa >\z@
2433 \if@slide\else
2434 \null
2435 \vspace*{-\baselineskip}%
2436 \fi
2437 \vskip\@tempskipa
2438 \fi
2439 \fi
2440 \noindent
2441 % 追加終わり
2442 \@ifstar
2443 {\@ssect{#3}{#4}{#5}{#6}}%
2444 {\@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、多少変えています。 \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。

```

2445 \ifnum\bxjs@everyparhook=\bxjs@everyparhook@@compat
2446 \let\bxjs@if@ceph\@firstofone
2447 \else \let\bxjs@if@ceph\@gobble
2448 \fi

```

```

2449 \def\@sect#1#2#3#4#5#6[#7]#8{%
2450 \ifnum #2>\c@secnumdepth
2451 \let\@svsec\@empty
2452 \else
2453 \refstepcounter{#1}%
2454 \protected@edef\@svsec{\@secntformat{#1}\relax}%

```



```

2455 \fi
2456 % 見出し後の空きを \@tempskipa にセット
2457 \@tempskipa #5\relax
2458 % 条件判断の順序を入れ換えました
2459 \ifdim \@tempskipa<\z@
2460 \def\@svsechd{%
2461 #6{\hskip #3\relax
2462 \@svsec #8}%
2463 \csname #1mark\endcsname{#7}%
2464 \addcontentsline{toc}{#1}{%
2465 \ifnum #2>\c@secnumdepth \else
2466 \protect\numberline{\bxjs@label@sect{#1}}%
2467 \fi
2468 #7}}% 目次にフルネームを載せるなら #8
2469 \else
2470 \begingroup
2471 \interlinepenalty \@M % 下から移動
2472 #6{%
2473 \@hangfrom{\hskip #3\relax\@svsec}%
2474 % \interlinepenalty \@M % 上に移動
2475 #8\@@par}%
2476 \endgroup
2477 \csname #1mark\endcsname{#7}%
2478 \addcontentsline{toc}{#1}{%
2479 \ifnum #2>\c@secnumdepth \else
2480 \protect\numberline{\bxjs@label@sect{#1}}%
2481 \fi
2482 #7}% 目次にフルネームを載せるならここは #8
2483 \fi
2484 \@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。

```

2485 \def\@xsect#1{%
2486 % 見出しの後ろの空きを \@tempskipa にセット
2487 \@tempskipa #1\relax
2488 % 条件判断の順序を変えました
2489 \ifdim \@tempskipa<\z@
2490 \@nbreakfalse
2491 \global\@noskipsectrue
2492 \everypar{%
2493 \if@noskipsec
2494 \global\@noskipsecfalse
2495 {\setbox\z@\lastbox}%
2496 \clubpenalty\@M
2497 \begingroup \@svsechd \endgroup

```

```

2498      \unskip
2499      \@tempskipa #1\relax
2500      \hskip -\@tempskipa
2501      \else
2502      \clubpenalty \@clubpenalty
2503      \everypar\expandafter{\bxjs@if@ceph\everyparhook}%
TODO: ↑ナニコレ？
2504      \fi\bxjs@if@ceph\everyparhook}%
2505  \else
2506    \par \nobreak
2507    \vskip \@tempskipa
2508    \@afterheading
2509  \fi
2510  \if@slide
2511    {\vskip\if@twocolumn-5\jsc@mpt\else-6\jsc@mpt\fi
2512     \maybeblue\hrule height0\jsc@mpt depth1\jsc@mpt
2513     \vskip\if@twocolumn 4\jsc@mpt\else 7\jsc@mpt\fi\relax}%
2514  \fi
2515  \par % 2000-12-18
2516  \ignorespaces}
2517 \def\@ssect#1#2#3#4#5{%
2518   \@tempskipa #3\relax
2519   \ifdim \@tempskipa<\z@
2520     \def\@svsechd{#4{\hskip #1\relax #5}}%
2521   \else
2522     \begingroup
2523       #4{%
2524         \@hangfrom{\hskip #1}%
2525         \interlinepenalty \@M #5\@@par}%
2526     \endgroup
2527   \fi
2528   \@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 2529 \newcommand*\chaptermark[1]{ }
\subsubsectionmark 2530 % \newcommand*\sectionmark[1]{ }
2531 % \newcommand*\subsectionmark[1]{ }
\paragraphmark 2532 % \newcommand*\subsubsectionmark[1]{ }
\subparagraphmark 2533 % \newcommand*\paragraphmark[1]{ }
2534 % \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。

```
2535 %<!book&!report>\setcounter{secnumdepth}{3}
2536 %<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 2537 \newcounter{part}
\c@subsubsection 2538 %<book|report>\newcounter{chapter}
2539 %<book|report>\newcounter{section}[chapter]
\c@paragraph 2540 %<!book&!report>\newcounter{section}
\c@subparagraph 2541 \newcounter{subsection}[section]
2542 \newcounter{subsubsection}[subsection]
2543 \newcounter{paragraph}[subsubsection]
2544 \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。

```
2545 \renewcommand{\thepart}{\@Roman\c@part}
2546 %<*&!book&!report>
2547 \ifnum\bxjs@label@section=\bxjs@label@section@@compat
2548 \renewcommand{\thesection}{\presectionname\@arabic\c@section\postsectionname}
2549 \renewcommand{\thesubsection}{\@arabic\c@section.\@arabic\c@subsection}
2550 \else
2551 \renewcommand{\thesection}{\@arabic\c@section}
2552 \renewcommand{\thesubsection}{\thesection.\@arabic\c@subsection}
2553 \fi
2554 %</&!book&!report>
2555 %<*&book|report>
2556 \renewcommand{\thechapter}{\@arabic\c@chapter}
2557 \renewcommand{\thesection}{\thechapter.\@arabic\c@section}
2558 \renewcommand{\thesubsection}{\thesection.\@arabic\c@subsection}
2559 %</&book|report>
2560 \renewcommand{\thesubsubsection}{\%
2561 \thesubsection.\@arabic\c@subsubsection}
2562 \renewcommand{\theparagraph}{\%
2563 \thesubsubsection.\@arabic\c@paragraph}
2564 \renewcommand{\thesubparagraph}{\%
2565 \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。

```
2566 %<book|report>\newcommand{\@chapapp}{\prechaptername}
```

```
2567 %<book|report>\newcommand{\@chappos}{\postchaptername}
```

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

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

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

```
2568 %<*book|report>
2569 \newcommand\frontmatter{%
2570   \pltx@cleartooddpage
2571   \@mainmatterfalse
2572   \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。

```
2573 \newcommand\mainmatter{%
2574   \pltx@cleartooddpage
2575   \@mainmattertrue
2576   \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。

```
2577 \newcommand\backmatter{%
2578   \if@openleft
2579     \cleardoublepage
2580   \else\if@openright
2581     \cleardoublepage
2582   \else
2583     \clearpage
2584   \fi\fi
2585   \@mainmatterfalse}
2586 %</book|report>
```

■部

`\part` 新しい部を始めます。

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

```
\secdef{星なし}{星あり}
```

星なし * のない形の定義です。

星あり * のある形の定義です。

\secdef は次のようにして使います。

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

まず book と report のクラス以外です。

```
2587 %<!*book&!report>
2588 \newcommand\part{%
2589   \if@noskipsec \leavevmode \fi
2590   \par
2591   \addvspace{4ex}%
2592   \if@english \@afterindentfalse \else \@afterindenttrue \fi
2593   \secdef\@part\@spart}
2594 %</!*book&!report>
```

book および report クラスの場合は、少し複雑です。

```
2595 %<*book|report>
2596 \newcommand\part{%
2597   \if@openleft
2598     \cleardoublepage
2599   \else\if@openright
2600     \cleardoublepage
2601   \else
2602     \clearpage
2603   \fi\fi
2604   \thispagestyle{empty}% 欧文用標準スタイルでは plain
2605   \if@twocolumn
2606     \onecolumn
2607     \@restonecoltrue
2608   \else
2609     \@restonecolfalse
2610   \fi
2611   \null\vfil
2612   \secdef\@part\@spart}
2613 %</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。

```
2614 %<!*book&!report>
2615 \def\@part[#1]#2{%
2616   \ifnum \c@secnumdepth >\m@ne
2617     \refstepcounter{part}%
2618     \addcontentsline{toc}{part}{%
2619       \prepartname\thepart\postpartname\hspace{1\jsZw}#1}%
2620   \else
```

```

2621 \addcontentsline{toc}{part}{#1}%
2622 \fi
2623 \markboth{}{}%
2624 {\parindent\z@
2625 \raggedright
2626 \interlinepenalty \@M
2627 \normalfont
2628 \ifnum \c@secnumdepth >\m@ne
2629 \Large\headfont\prepartname\thepart\postpartname
2630 \par\nobreak
2631 \fi
2632 \huge \headfont #2%
2633 \markboth{}{}\par}%
2634 \nobreak
2635 \vskip 3ex
2636 \@afterheading}
2637 %<!/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。

```

2638 %<*book|report>
2639 \def\@part[#1]#2{%
2640 \ifnum \c@secnumdepth >-2\relax
2641 \refstepcounter{part}%
2642 \addcontentsline{toc}{part}{%
2643 \prepartname\thepart\postpartname\hspace{1\jsZw}#1}%
2644 \else
2645 \addcontentsline{toc}{part}{#1}%
2646 \fi
2647 \markboth{}{}%
2648 {\centering
2649 \interlinepenalty \@M
2650 \normalfont
2651 \ifnum \c@secnumdepth >-2\relax
2652 \huge\headfont \prepartname\thepart\postpartname
2653 \par\vskip20\jsc@empt
2654 \fi
2655 \Huge \headfont #2\par}%
2656 \@endpart}
2657 %</book|report>

```

\@spart 番号を付けない部です。

```

2658 %<*!book&!report>
2659 \def\@spart#1{%
2660 \parindent \z@ \raggedright
2661 \interlinepenalty \@M
2662 \normalfont
2663 \huge \headfont #1\par}%
2664 \nobreak
2665 \vskip 3ex

```

```

2666 \afterheading}
2667 %<!/book&!report>
2668 %<*book|report>
2669 \def\@spart#1{%
2670     \centering
2671     \interlinepenalty \@M
2672     \normalfont
2673     \Huge \headfont #1\par}%
2674 \@endpart}
2675 %</book|report>

```

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

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

```

2676 %<*book|report>
2677 \def\@endpart{\vfil\newpage
2678     \if@twoside
2679     \if@openleft %% added (2017/02/24)
2680     \null\thispagestyle{empty}\newpage
2681     \else\if@openright %% added (2016/12/13)
2682     \null\thispagestyle{empty}\newpage
2683     \fi\fi %% added (2016/12/13, 2017/02/24)
2684     \fi
2685     \if@restonecol
2686     \twocolumn
2687     \fi}
2688 %</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。

```

2689 %<*book|report>
2690 \newcommand{\chapter}{%
2691     \if@openleft\cleardoublepage\else
2692     \if@openright\cleardoublepage\else\clearpage\fi\fi
2693     \plainifnotempty % 元: \thispagestyle{plain}
2694     \global\@topnum\z@
2695     \if@english \@afterindentfalse \else \@afterindenttrue \fi
2696     \secdef
2697     {\@omit@numberfalse\@chapter}%
2698     {\@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。

```

2699 \def\@chapter[#1]#2{%
2700   \ifnum \c@secnumdepth >\m@ne
2701     \if@mainmatter
2702       \refstepcounter{chapter}%
2703       \typeout{\@chapapp\thechapter\@chappos}%
2704       \addcontentsline{toc}{chapter}%
2705         {\protect\numberline
2706 %       %{\if@english\thechapter\else\@chapapp\thechapter\@chappos\fi}%
2707       {\@chapapp\thechapter\@chappos}%
2708       #1}%
2709     \else\addcontentsline{toc}{chapter}{#1}\fi
2710   \else
2711     \addcontentsline{toc}{chapter}{#1}%
2712   \fi
2713   \chaptermark{#1}%
2714   \addtocontents{lof}{\protect\addvspace{10\jsc@empt}}%
2715   \addtocontents{lot}{\protect\addvspace{10\jsc@empt}}%
2716   \if@twocolumn
2717     \@topnewpage[\@makechapterhead{#2}]%
2718   \else
2719     \@makechapterhead{#2}%
2720     \@afterheading
2721   \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。

```

2722 \def\@makechapterhead#1{%
2723   \vspace*{2\Cvs}% 欧文は50pt
2724   {\parindent \z@ \raggedright \normalfont
2725     \ifnum \c@secnumdepth >\m@ne
2726       \if@mainmatter
2727         \huge\headfont \@chapapp\thechapter\@chappos
2728         \par\nobreak
2729         \vskip \Cvs % 欧文は20pt
2730       \fi
2731     \fi
2732     \interlinepenalty\@M
2733     \Huge \headfont #1\par\nobreak
2734     \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。

```

2735 \def\@schapter#1{%
2736   \chaptermark{#1}%
2737   \if@twocolumn
2738     \@topnewpage[\@makeschapterhead{#1}]%
2739   \else
2740     \@makeschapterhead{#1}\@afterheading
2741   \fi}

```

\@makeschapterhead 番号なしの章見出しです。


```

2742 \def\@makeschapterhead#1{%
2743   \vspace*{2\Cvs}% 欧文は50pt
2744   {\parindent \z@ \raggedright
2745     \normalfont
2746     \interlinepenalty\@M
2747     \Huge \headfont #1\par\nobreak
2748     \vskip 3\Cvs}} % 欧文は40pt
2749 %</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。

```

2750 \if@twocolumn
2751   \newcommand{\section}{%
2752     %<jspf>\ifx\maketitle\relax\else\maketitle\fi
2753     \@startsection{section}{1}{\z@}%
2754     %<kiyou> {0.6\Cvs}{0.4\Cvs}%
2755     %<kiyou> {\Cvs}{0.5\Cvs}%
2756     % {\normalfont\large\headfont\@secapp}%
2757     {\normalfont\large\headfont\raggedright}}
2758 \else
2759   \newcommand{\section}{%
2760     \if@slide\clearpage\fi
2761     \@startsection{section}{1}{\z@}%
2762     {\Cvs \@plus.5\Cdp \@minus.2\Cdp}% 前アキ
2763     {.5\Cvs \@plus.3\Cdp}% 後アキ
2764     % {\normalfont\Large\headfont\@secapp}%
2765     {\normalfont\Large\headfont\raggedright}}
2766 \fi

```

`\subsection` 同上です。

```

2767 \if@twocolumn
2768   \newcommand{\subsection}{\@startsection{subsection}{2}{\z@}%
2769     {\z@}{\if@slide .4\Cvs \else \z@ \fi}%
2770     {\normalfont\normalsize\headfont}}
2771 \else
2772   \newcommand{\subsection}{\@startsection{subsection}{2}{\z@}%
2773     {\Cvs \@plus.5\Cdp \@minus.2\Cdp}% 前アキ
2774     {.5\Cvs \@plus.3\Cdp}% 後アキ
2775     {\normalfont\large\headfont}}
2776 \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)。

```

2777 \if@twocolumn
2778   \newcommand{\subsubsection}{\@startsection{subsubsection}{3}{\z@}%

```

```

2779     {\z@}{\if@slide .4\Cvs \else \z@ \fi}%
2780     {\normalfont\normalsize\headfont}}
2781 \else
2782   \newcommand{\subsubsection}{\@startsection{subsubsection}{3}{\z@}%
2783     {\Cvs \@plus.5\Cdp \@minus.2\Cdp}%
2784     {\if@slide .5\Cvs \@plus.3\Cdp \else \z@ \fi}%
2785     {\normalfont\normalsize\headfont}}
2786 \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。

```
2787 \let\jsJaChar\@empty
```

```

2788 \newcommand\jsParagraphMark{\relax\jsJaChar{■}}
2789 \let\bxjs@org@paragraph@mark\jsParagraphMark
2790 \ifx\bxjs@paragraph@mark\@empty
2791   \let\jsParagraphMark\@empty
2792 \else\ifx\bxjs@paragraph@mark\undefined\else
2793   \bxjs@nclong\edef\jsParagraphMark{\noexpand\jsJaChar{\bxjs@paragraph@mark}}
2794 \fi\fi
2795 \if@twocolumn
2796   \newcommand{\paragraph}{\@startsection{paragraph}{4}{\z@}%
2797     {\z@}{\if@slide .4\Cvs \else -1\jsZw\fi}% 改行せず 1\jsZw のアキ
2798     %<jspf> {\normalfont\normalsize\headfont}}
2799     %<!jspf> {\normalfont\normalsize\headfont\jsParagraphMark}}
2800 \else
2801   \newcommand{\paragraph}{\@startsection{paragraph}{4}{\z@}%
2802     {0.5\Cvs \@plus.5\Cdp \@minus.2\Cdp}%
2803     {\if@slide .5\Cvs \@plus.3\Cdp \else -1\jsZw\fi}% 改行せず 1\jsZw のアキ
2804     %<jspf> {\normalfont\normalsize\headfont}}
2805     %<!jspf> {\normalfont\normalsize\headfont\jsParagraphMark}}
2806 \fi

```

`\subparagraph` 見出しの後ろで改行されません。

```
2807 \if@twocolumn
```

```

2808 \newcommand{\subparagraph}{\@startsection{subparagraph}{5}{\z@}%
2809   {\z@}{\if@slide .4\Cvs \@plus.3\Cdp \else -1\jsZw\fi}%
2810   {\normalfont\normalsize\headfont}}
2811 \else
2812 \newcommand{\subparagraph}{\@startsection{subparagraph}{5}{\z@}%
2813   {\z@}{\if@slide .5\Cvs \@plus.3\Cdp \else -1\jsZw\fi}%
2814   {\normalfont\normalsize\headfont}}
2815 \fi

```

8.3 リスト環境

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

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

[2002-05-11] 3zw に変更しました。

[2005-03-19] 二段組は 2zw に戻しました。

```

2816 \if@slide
2817   \setlength\leftmargini{1\jsZw}
2818 \else
2819   \if@twocolumn
2820     \setlength\leftmargini{2\jsZw}
2821   \else
2822     \setlength\leftmargini{3\jsZw}
2823   \fi
2824 \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 2825 \if@slide
\leftmarginiv 2826   \setlength\leftmarginii {1\jsZw}
\leftmarginiv 2827   \setlength\leftmarginiii{1\jsZw}
\leftmarginiv 2828   \setlength\leftmarginiv {1\jsZw}
\leftmarginiv 2829   \setlength\leftmarginv  {1\jsZw}
\leftmarginiv 2830   \setlength\leftmarginvi {1\jsZw}
\leftmarginiv 2831 \else
\leftmarginiv 2832   \setlength\leftmarginii {2\jsZw}
\leftmarginiv 2833   \setlength\leftmarginiii{2\jsZw}
\leftmarginiv 2834   \setlength\leftmarginiv {2\jsZw}
\leftmarginiv 2835   \setlength\leftmarginv  {1\jsZw}
\leftmarginiv 2836   \setlength\leftmarginvi {1\jsZw}
\leftmarginiv 2837 \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。

```

2838 \setlength \labelsep {0.5\jsZw} % .5em

```

```
2839 \setlength \labelwidth{\leftmargini}
2840 \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。

```
2841 \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 2842 \@beginparpenalty -\@lowpenalty
\@itempenalty 2843 \@endparpenalty -\@lowpenalty
2844 \@itempenalty -\@lowpenalty
```

`\@listi` `\@listi` は `\leftmargin`, `\parsep`, `\topsep`, `\itemsep` などのトップレベルの定義をします。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 $_{-0.1}^{+0.2}$ `\baselineskip` を思い切って外しました。

```
2845 \def\@listi{\leftmargin\leftmargini
2846 \parsep \z@
2847 \topsep 0.5\baselineskip
2848 \itemsep \z@ \relax}
2849 \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）。

```
2850 \@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 2851 \def\@listii{\leftmargin\leftmarginii
\@listiv 2852 \labelwidth\leftmarginii \advance\labelwidth-\labelsep
2853 \topsep \z@
\@listv 2854 \parsep \z@
\@listvi 2855 \itemsep\parsep}
2856 \def\@listiii{\leftmargin\leftmarginiii
2857 \labelwidth\leftmarginiii \advance\labelwidth-\labelsep
2858 \topsep \z@
2859 \parsep \z@
2860 \itemsep\parsep}
2861 \def\@listiv {\leftmargin\leftmarginiv
2862 \labelwidth\leftmarginiv
2863 \advance\labelwidth-\labelsep}
2864 \def\@listv {\leftmargin\leftmarginv
2865 \labelwidth\leftmarginv
2866 \advance\labelwidth-\labelsep}
2867 \def\@listvi {\leftmargin\leftmarginvi
```

```

2868 \labelwidth\leftmarginvi
2869 \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。

```

2870 \renewcommand{\theenumi}{\@arabic\c@enumi}
2871 \renewcommand{\theenumii}{\@alph\c@enumii}
2872 \renewcommand{\theenumiii}{\@roman\c@enumiii}
2873 \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。

```

2874 \newcommand*{\jsInJaParen}[1]{%
2875   \mbox{\jsInhibitGlue (#1) \jsInhibitGlue}}
2876 \newcommand{\labelenumi}{\theenumi.}
2877 \newcommand{\labelenumii}{\jsInJaParen{\theenumii}}
2878 \newcommand{\labelenumiii}{\theenumiii.}
2879 \newcommand{\labelenumiv}{\theenumiv.}

```

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

```

\p@enumiv 2880 \renewcommand{\p@enumii}{\theenumi}
2881 \renewcommand{\p@enumiii}{\theenumi\jsInhibitGlue (\theenumii )}
2882 \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 2883 \newcommand\labelitemi{\textbullet}
2884 \newcommand\labelitemii{\normalfont\bfseries \textendash}
\labelitemiii 2885 \newcommand\labelitemiii{\textasteriskcentered}
\labelitemiv 2886 \newcommand\labelitemiv{\textperiodcentered}

```

■description 環境

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

```
2887 \newenvironment{description}{%
2888   \list{}{%
2889     \labelwidth=\leftmargin
2890     \labelsep=1\jsZw
2891     \advance \labelwidth by -\labelsep
2892     \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。

```
2893 \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。

```
2894 %<*book|report>
2895 \newenvironment{chapterabstract}{%
2896   \begin{list}{}{%
2897     \listparindent=1\jsZw
2898     \itemindent=\listparindent
2899     \rightmargin=0pt
2900     \leftmargin=5\jsZw}\item[]{}\end{list}\vspace{\baselineskip}}
2901 %</book|report>
```

“普通の” abstract 環境の定義。

```
2902 %<*article|report|slide>
2903 \newbox\@abstractbox
2904 \if@titlepage
2905   \newenvironment{abstract}{%
2906     \titlepage
2907     \null\vfil
```

```

2908 \beginparpenalty\@lowpenalty
2909 \begin{center}%
2910 \headfont \abstractname
2911 \endparpenalty\@M
2912 \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。

```

2913 \par}%
2914 {\par\vfil\null\endtitlepage}
2915 \else
2916 \newenvironment{abstract}{%
2917 \if@twocolumn
2918 \ifx\maketitle\relax
2919 \section*{\abstractname}%
2920 \else
2921 \global\setbox\@abstractbox\hbox\bgroup
2922 \begin{minipage}[b]{\textwidth}
2923 \small\parindent1\jsZw
2924 \begin{center}%
2925 {\headfont \abstractname\vspace{-.5em}\vspace{\z@}}%
2926 \end{center}%
2927 \list{}{%
2928 \listparindent\parindent
2929 \itemindent \listparindent
2930 \rightmargin \leftmargin}%
2931 \item\relax
2932 \fi
2933 \else
2934 \small
2935 \begin{center}%
2936 {\headfont \abstractname\vspace{-.5em}\vspace{\z@}}%
2937 \end{center}%
2938 \list{}{%
2939 \listparindent\parindent
2940 \itemindent \listparindent
2941 \rightmargin \leftmargin}%
2942 \item\relax
2943 \fi}{\if@twocolumn
2944 \ifx\maketitle\relax
2945 \else
2946 \endlist\end{minipage}\egroup
2947 \fi
2948 \else
2949 \endlist
2950 \fi}
2951 \fi
2952 %</article|report|slide>
2953 %<*jspf>
2954 \newbox\@abstractbox

```

```

2955 \newenvironment{abstract}{%
2956   \global\setbox\@abstractbox\hbox\bgroup
2957   \begin{minipage}[b]{157\jsc@mmm}{\sffamily Abstract}\par
2958     \small
2959     \if@english \parindent6\jsc@mmm \else \parindent1\jsZw \fi}%
2960   {\end{minipage}\egroup}
2961 %</jspf>

```

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

```

2962 %<*book|report>
2963 \ifbxjs@force@chapterabstract
2964   \let\abstract\chapterabstract
2965   \let\endabstract\endchapterabstract
2966 \fi
2967 %</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。

```

2968 %<*jspf>
2969 %\newbox\@keywordsbox
2970 %\newenvironment{keywords}{%
2971 %   \global\setbox\@keywordsbox\hbox\bgroup
2972 %   \begin{minipage}[b]{1570\jsc@mmm}{\sffamily Keywords:}\par
2973 %     \small\parindent0\jsZw}%
2974 %   {\end{minipage}\egroup}
2975 %</jspf>

```

■verse 環境

`verse (env.)` 詩のための `verse` 環境です。

```

2976 \newenvironment{verse}{%
2977   \let \=\@centercr
2978   \list{}{%
2979     \itemsep \z@
2980     \itemindent -2\jsZw % 元: -1.5em
2981     \listparindent\itemindent
2982     \rightmargin \z@
2983     \advance\leftmargin 2\jsZw}% 元: 1.5em
2984   \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。


```

2985 \newenvironment{quotation}{%
2986   \list{}{%
2987     \listparindent\parindent
2988     \itemindent\listparindent
2989     \rightmargin \z@}%
2990   \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。

```

2991 \newenvironment{quote}%
2992   {\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。

```

2993 \def\@begintheorem#1#2{\trivlist\labelsep=1\jsZw
2994   \item[\hskip \labelsep{\headfont #1\ #2}]}
2995 \def\@opargbegintheorem#1#2#3{\trivlist\labelsep=1\jsZw
2996   \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。

```

2997 \newenvironment{titlepage}{%
2998   %<book>      \pltx@cleartooddpage %% 2017-02-24
2999   \if@twocolumn
3000     \@restonecoltrue\onecolumn
3001   \else
3002     \@restonecolfalse\newpage
3003   \fi
3004   \thispagestyle{empty}%
3005   \ifodd\c@page\setcounter{page}\@ne\else\setcounter{page}\z@\fi %% 2017-
3006   }%

```

```

3007 {\if@restonecol\twocolumn \else \newpage \fi
3008 \if@twoside\else
3009 \setcounter{page}\@ne
3010 \fi}

```

■付録

`\appendix` 本文と付録を分離するコマンドです。

```

3011 %<!*book&!report>
3012 \newcommand{\appendix}{\par
3013 \setcounter{section}{0}%
3014 \setcounter{subsection}{0}%
3015 \ifnum\bxjs@label@section=\bxjs@label@section@@compat
3016 \gdef\presectionname{\appendixname}%
3017 \gdef\postsectionname{}%
3018 % \gdef\thesection{\@Alph\c@section}% [2003-03-02]
3019 \gdef\thesection{\presectionname\@Alph\c@section\postsectionname}%
3020 \gdef\thesubsection{\@Alph\c@section.\@arabic\c@subsection}%
3021 \else
3022 \gdef\@secapp{\appendixname}%
3023 \gdef\@secpos{}%
3024 \gdef\thesection{\@Alph\c@section}%
3025 \fi}
3026 %</!*book&!report>
3027 %<*book|report>
3028 \newcommand{\appendix}{\par
3029 \setcounter{chapter}{0}%
3030 \setcounter{section}{0}%
3031 \gdef\@chapapp{\appendixname}%
3032 \gdef\@chappos{}%
3033 \gdef\thechapter{\@Alph\c@chapter}}
3034 %</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。

```
3035 \setlength\arraycolsep{5\jsc@mp}
```

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

```
3036 \setlength\tabcolsep{6\jsc@mp}
```

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

```
3037 \setlength\arrayrulewidth{.4\p@}
```

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

```
3038 \setlength\doublerulesep{2\p@}
```

■tabbing 環境

`\tabbingsep` `\'` コマンドで入るアキです。

```
3039 \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。

```
3040 \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。

```
3041 \setlength\fboxsep{3\jsc@mp}
```

```
3042 \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。

```
3043 %<!book&!report>\renewcommand \theequation {\@arabic\c@equation}
```

```
3044 %<*book|report>
```

```
3045 \@addtoreset{equation}{chapter}
```

```
3046 \renewcommand\theequation
```

```
3047 {\ifnum \c@chapter>\z@ \thechapter.\fi \@arabic\c@equation}
```

```
3048 %</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。

```
3049 % \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。

```
3050 % \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。

```
3051 % \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。

```
3052 %<!*book&!report>
3053 \newcounter{figure}
3054 \renewcommand \thefigure {\@arabic\c@figure}
3055 %</!*book&!report>
3056 %<*book|report>
3057 \newcounter{figure}[chapter]
3058 \renewcommand \thefigure
3059     {\ifnum \c@chapter>\z@ \thechapter.\fi \@arabic\c@figure}
3060 %</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 3061 \def\fps@figure{tbp}
3062 \def\ftype@figure{1}
\fnum@figure 3063 \def\ext@figure{lof}
3064 \def\fnum@figure{\figurename\nobreak\thefigure}
```

`figure (env.)` * 形式は段抜きのフロートです。

```
figure* (env.) 3065 \newenvironment{figure}%
3066                 {\@float{figure}}%
3067                 {\end@float}
3068 \newenvironment{figure*}%
3069                 {\@dblfloat{figure}}%
3070                 {\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。

```
3071 %<!*book&!report>
3072 \newcounter{table}
3073 \renewcommand\thetable{\@arabic\c@table}
3074 %</!*book&!report>
3075 %<*book|report>
3076 \newcounter{table}[chapter]
```

```

3077 \renewcommand \thetable
3078     {\ifnum \c@chapter>\z@ \thechapter.\fi \@arabic\c@table}
3079 %</book|report>

```

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

```

\ext@table 3080 \def\fps@table{tbp}
3081 \def\ftype@table{2}
\fnun@table 3082 \def\ext@table{lot}
3083 \def\fnun@table{\tablename\nobreak\thetable}

```

`table (env.)` * は段抜きのフロートです。

```

table* (env.) 3084 \newenvironment{table}%
3085             {\@float{table}}%
3086             {\end@float}
3087 \newenvironment{table*}%
3088             {\@dblfloat{table}}%
3089             {\end@dblfloat}

```

8.6 キャプション

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

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

```

3090 \newlength\abovecaptionskip
3091 \newlength\belowcaptionskip
3092 \setlength\abovecaptionskip{5\jsc@mp} % 元: 10\p@
3093 \setlength\belowcaptionskip{5\jsc@mp} % 元: 0\p@

```

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

[2003-11-05] ロジックを少し変えてみました。

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

```

3094 %<*\jpf>
3095 % \long\def\@makecaption#1#2{\small
3096 %     \advance\leftskip10\jsc@mm
3097 %     \advance\rightskip10\jsc@mm
3098 %     \vskip\abovecaptionskip
3099 %     \sbox\@tempboxa{#1\hskip1\jsZw\relax #2}%
3100 %     \ifdim \wd\@tempboxa >\hsize
3101 %         #1\hskip1\jsZw\relax #2\par
3102 %     \else

```

```

3103 %      \global \@minipagefalse
3104 %      \hb@xt@\hsize{\hfil\box\@tempboxa\hfil}%
3105 %      \fi
3106 %      \vskip\belowcaptionskip}}
3107 \long\def\@makecaption#1#2{{\small
3108   \advance\leftskip .0628\linewidth
3109   \advance\rightskip .0628\linewidth
3110   \vskip\abovcaptionskip
3111   \sbox\@tempboxa{\small\sffamily #1\quad #2}%
3112   \ifdim \wd\@tempboxa <\hsize \centering \fi
3113   #1\zwspace#2\par
3114   \vskip\belowcaptionskip}}
3115 %</!jspf>
3116 %<*jspf>
3117 \long\def\@makecaption#1#2{%
3118   \vskip\abovcaptionskip
3119   \sbox\@tempboxa{\small\sffamily #1\quad #2}%
3120   \ifdim \wd\@tempboxa >\hsize
3121     {\small\sffamily
3122       \list{#1}{%
3123         \renewcommand{\makelabel}[1]{##1\hfil}
3124         \itemsep      \z@
3125         \itemindent    \z@
3126         \labelsep      \z@
3127         \labelwidth    11\jsc@mmm
3128         \listparindent \z@
3129         \leftmargin    11\jsc@mmm}\item\relax #2\endlist}
3130   \else
3131     \global \@minipagefalse
3132     \hb@xt@\hsize{\hfil\box\@tempboxa\hfil}%
3133     \fi
3134     \vskip\belowcaptionskip}
3135 %</jspf>

```

9 フォントコマンド

ここでは L^AT_EX 2.09 で使われていたコマンドを定義します。これらはテキストモードと数式モードのどちらでも動作します。これらは互換性のためのもので、できるだけ `\text...` と `\math...` を使ってください。

[2016-07-15] KOMA-Script 中の `\scr@DeclareOldFontCommand` に倣い、これらの命令を使うときには警告を発することにしました。

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

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

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

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

```
3136 \newif\if@jsc@warnoldfontcmd
3137 \@jsc@warnoldfontcmdtrue
3138 \newif\if@jsc@warnoldfontcmdexception
3139 \@jsc@warnoldfontcmdexceptionfalse
```

`\jsc@DeclareOldFontCommand`

```
3140 \newcommand*{\jsc@DeclareOldFontCommand}[3]{%
3141   \g@addto@macro\bxjs@oldfontcmd@list{\do#1}%
3142   \DeclareOldFontCommand{#1}{%
3143     \bxjs@oldfontcmd{#1}#2%
3144   }{%
3145     \bxjs@oldfontcmd{#1}#3%
3146   }%
3147 }
3148 \DeclareRobustCommand*\jsc@warnoldfontcmd[1]{%
3149   \ClassInfo\bxjs@clsname
3150   {Old font command '\string#1' is used!!\MessageBreak
3151     The first occurrence is}%
3152 }
```

`\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。

```
3153 \DeclareRobustCommand*\allowoldfontcommands{%
3154   \@jsc@warnoldfontcmdexceptiontrue}
3155 \DeclareRobustCommand*\disallowoldfontcommands{%
3156   \@jsc@warnoldfontcmdexceptionfalse}

3157 \let\bxjs@oldfontcmd@list\@empty
3158 \def\bxjs@oldfontcmd#1{%
3159   \expandafter\bxjs@oldfontcmd@a\csname bxjs@ofc/\string#1\endcsname#1}
3160 \def\bxjs@oldfontcmd@a#1#2{%
3161   \if@jsc@warnoldfontcmdexception\else
3162     \global\@jsc@warnoldfontcmdfalse
3163     \ifx#1\relax
3164       \global\let#1=t%
3165       \jsc@warnoldfontcmd{#2}%
3166     \fi
3167   \fi}
3168 \def\bxjs@warnoldfontcmd@final{%
3169 % \par
3170   \global\let\bxjs@warnoldfontcmd@final\@empty
3171   \let\@tempa\@empty
3172   \def\do##1{%
```

```

3173 \ifundefined{bxjs@ofc\string##1}{\%else
3174 \edef\@tempa{\@tempa \space\string##1}}}%
3175 \bxjs@oldfontcmd@list
3176 \ifx\@tempa\@empty\else
3177 \ClassWarningNoLine{bxjs@clsname
3178 {Some old font commands were used in text:\MessageBreak
3179 \space\@tempa\MessageBreak
3180 You should note, that since 1994 LaTeX2e provides a\MessageBreak
3181 new font selection scheme called NFSS2 with several\MessageBreak
3182 new, combinable font commands. The
3183 class provides\MessageBreak
3184 the old font commands only for compatibility}
3185 \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。

```

3186 \ifbxjs@old@hook@system
3187 \AtEndDocument{%
3188 \g@addto@macro\clearpage{\bxjs@warnoldfontcmd@final}}
3189 \else
3190 \AddToHook{enddocument/afterlastpage}{\bxjs@warnoldfontcmd@final}
3191 \fi

```

`\mc` フォントファミリーを変更します。

```

\gt 3192 \jsc@DeclareOldFontCommand{\mc}{\normalfont\mcfamily}{\mathmc}
\rm 3193 \jsc@DeclareOldFontCommand{\gt}{\normalfont\gtfamily}{\mathgt}
\sf 3194 \jsc@DeclareOldFontCommand{\rm}{\normalfont\rmfamily}{\mathrm}
\tt 3195 \jsc@DeclareOldFontCommand{\sf}{\normalfont\sffamily}{\mathsf}
3196 \jsc@DeclareOldFontCommand{\tt}{\normalfont\ttfamily}{\mathtt}

```

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

```

3197 \jsc@DeclareOldFontCommand{\bf}{\normalfont\bfseries}{\mathbf}

```

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

```

3198 \jsc@DeclareOldFontCommand{\it}{\normalfont\itshape}{\mathit}
3199 \jsc@DeclareOldFontCommand{\sl}{\normalfont\slshape}{\@nomath\sl}
3200 \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 3201 \DeclareRobustCommand*\cal{\@fontswitch\relax\mathcal}
3202 \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。

```
3203 \newcommand\@pnumwidth{1.55em}
```

```
3204 \newcommand\@tocrmarg{2.55em}
```

```
3205 \newcommand\@dotsep{4.5}
```

```
3206 %<!book&!report>\setcounter{tocdepth}{2}
```

```
3207 %<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)

```
3208 \newdimen\jsc@tocl@width
3209 \newcommand{\tableofcontents}{%
3210 %<*book|report>
3211 \settowidth\jsc@tocl@width{\headfont\prechaptername\postchaptername}%
3212 \settowidth\@tempdima{\headfont\appendixname}%
3213 \ifdim\jsc@tocl@width<\@tempdima \setlength\jsc@tocl@width{\@tempdima}\fi
3214 \ifdim\jsc@tocl@width<2\jsZw \divide\jsc@tocl@width by 2 \advance\jsc@tocl@width 1\jsZw\fi
3215 \if@twocolumn
3216 \@restonecoltrue\onecolumn
3217 \else
3218 \@restonecolfalse
3219 \fi
3220 \chapter*{\contentsname}%
3221 \@mkboth{\contentsname}{}%
3222 %</book|report>
3223 %<!*book&!report>
3224 \settowidth\jsc@tocl@width{\headfont\presectionname\postsectionname}%
3225 \settowidth\@tempdima{\headfont\appendixname}%
3226 \ifdim\jsc@tocl@width<\@tempdima\relax\setlength\jsc@tocl@width{\@tempdima}\fi
3227 \ifdim\jsc@tocl@width<2\jsZw \divide\jsc@tocl@width by 2 \advance\jsc@tocl@width 1\jsZw\fi
3228 \section*{\contentsname}%
3229 \@mkboth{\contentsname}{\contentsname}%
3230 %</!book&!report>
3231 \@starttoc{toc}%
3232 %<book|report> \if@restonecol\twocolumn\fi
3233 }
```

`\l@part` 部の目次です。

```
3234 \newcommand*{\l@part}[2]{%
3235 \ifnum \c@tocdepth >-2\relax
3236 %<!book&!report> \addpenalty\@secpenalty
3237 %<book|report> \addpenalty{-\@highpenalty}%
3238 \addvspace{2.25em \@plus\jsc@empt}%
3239 \begingroup
3240 \parindent \z@
3241 % \@pnumwidth should be \@tocrmarg
3242 % \rightskip \@pnumwidth
3243 \rightskip \@tocrmarg
3244 \parfillskip -\rightskip
3245 {\leavevmode
3246 \large \headfont
3247 \setlength\@lnumwidth{4\jsZw}%
3248 #1\hfil \hb@xt@\@pnumwidth{\hss #2}}\par
3249 \nobreak
3250 %<book|report> \global\@nobreaktrue
3251 %<book|report> \everypar{\global\@nobreakfalse\everypar{}}%
```

```

3252 \endgroup
3253 \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)

```

3254 %<*book|report>
3255 \newcommand*{\l@chapter}[2]{%
3256 \ifnum \c@tocdepth >\m@ne
3257 \addpenalty{-\@highpenalty}%
3258 \addvspace{1.0em \@plus\jsc@mp}
3259 % \vskip 1.0em \@plus\p@ % book.clsでは↑がこうなっている
3260 \begingroup
3261 \parindent\z@
3262 % \rightskip\@pnumwidth
3263 \rightskip\@tocrmarg
3264 \parfillskip-\rightskip
3265 \leavevmode\headfont
3266 % % \if@english\setlength\@lnumwidth{5.5em}\else\setlength\@lnumwidth{4.683\jsZw}\fi
3267 \setlength\@lnumwidth{\jsc@tocl@width}\advance\@lnumwidth 2.683\jsZw
3268 \advance\leftskip\@lnumwidth \hskip-\leftskip
3269 #1\nobreak\hfil\nobreak\hbox to\@pnumwidth{\hss#2}\par
3270 \penalty\@highpenalty
3271 \endgroup
3272 \fi}
3273 %</book|report>

```

\l@section 節の目次です。

```

3274 %<!*book&!report>
3275 \newcommand*{\l@section}[2]{%
3276 \ifnum \c@tocdepth >\z@
3277 \addpenalty{\@secpenalty}%
3278 \addvspace{1.0em \@plus\jsc@mp}%
3279 \begingroup
3280 \parindent\z@
3281 % \rightskip\@pnumwidth
3282 \rightskip\@tocrmarg
3283 \parfillskip-\rightskip
3284 \leavevmode\headfont
3285 % % \setlength\@lnumwidth{4\jsZw}% 元1.5em [2003-03-02]
3286 \setlength\@lnumwidth{\jsc@tocl@width}\advance\@lnumwidth 2\jsZw
3287 \advance\leftskip\@lnumwidth \hskip-\leftskip
3288 #1\nobreak\hfil\nobreak\hbox to\@pnumwidth{\hss#2}\par
3289 \endgroup
3290 \fi}
3291 %</*!book&!report>

```

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

```

3292 %<book|report> % \newcommand*{\l@section}{\@dottedtocline{1}{1\jsZw}{3.683\jsZw}}

```

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

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

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

```
\l@subparagraph 3293 %<!*book&!report>
3294 % \newcommand*{\l@section} {\@dottedtocline{2}{1.5em}{2.3em}}
3295 % \newcommand*{\l@subsubsection}{\@dottedtocline{3}{3.8em}{3.2em}}
3296 % \newcommand*{\l@paragraph} {\@dottedtocline{4}{7.0em}{4.1em}}
3297 % \newcommand*{\l@subparagraph} {\@dottedtocline{5}{10em}{5em}}
3298 %
3299 % \newcommand*{\l@section} {\@dottedtocline{2}{1zw}{3zw}}
3300 % \newcommand*{\l@subsubsection}{\@dottedtocline{3}{2\jsZw}{3\jsZw}}
3301 % \newcommand*{\l@paragraph} {\@dottedtocline{4}{3\jsZw}{3\jsZw}}
3302 % \newcommand*{\l@subparagraph} {\@dottedtocline{5}{4\jsZw}{3\jsZw}}
3303 %
3304 \newcommand*{\l@section}{%
3305     \@tempdima\jsc@tocl@width \advance\@tempdima -1\jsZw
3306     \@dottedtocline{2}{\@tempdima}{3\jsZw}}
3307 \newcommand*{\l@subsubsection}{%
3308     \@tempdima\jsc@tocl@width \advance\@tempdima 0\jsZw
3309     \@dottedtocline{3}{\@tempdima}{4\jsZw}}
3310 \newcommand*{\l@paragraph}{%
3311     \@tempdima\jsc@tocl@width \advance\@tempdima 1\jsZw
3312     \@dottedtocline{4}{\@tempdima}{5\jsZw}}
3313 \newcommand*{\l@subparagraph}{%
3314     \@tempdima\jsc@tocl@width \advance\@tempdima 2\jsZw
3315     \@dottedtocline{5}{\@tempdima}{6\jsZw}}
3316 %</!*book&!report>
3317 %<*book|report>
3318 % \newcommand*{\l@section} {\@dottedtocline{2}{3.8em}{3.2em}}
3319 % \newcommand*{\l@subsubsection}{\@dottedtocline{3}{7.0em}{4.1em}}
3320 % \newcommand*{\l@paragraph} {\@dottedtocline{4}{10em}{5em}}
3321 % \newcommand*{\l@subparagraph} {\@dottedtocline{5}{12em}{6em}}
3322 \newcommand*{\l@section}{%
3323     \@tempdima\jsc@tocl@width \advance\@tempdima -1\jsZw
3324     \@dottedtocline{1}{\@tempdima}{3.683\jsZw}}
3325 \newcommand*{\l@subsubsection}{%
3326     \@tempdima\jsc@tocl@width \advance\@tempdima 2.683\jsZw
3327     \@dottedtocline{2}{\@tempdima}{3.5\jsZw}}
3328 \newcommand*{\l@subsubsection}{%
3329     \@tempdima\jsc@tocl@width \advance\@tempdima 6.183\jsZw
3330     \@dottedtocline{3}{\@tempdima}{4.5\jsZw}}
3331 \newcommand*{\l@paragraph}{%
3332     \@tempdima\jsc@tocl@width \advance\@tempdima 10.683\jsZw
3333     \@dottedtocline{4}{\@tempdima}{5.5\jsZw}}
3334 \newcommand*{\l@subparagraph}{%
3335     \@tempdima\jsc@tocl@width \advance\@tempdima 16.183\jsZw
```

```

3336 \dottedtocline{5}{\@tempdima}{6.5\jsZw}}
3337 %</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` ですが、アスキー版では `\@tempdima` の代わりに `\@lnumwidth` という変数で幅を決めるよう
に再定義しています。後続文字が全角か半角かでスペースが変わらないように `\hspace` を
入れておきました。

```

3338 \newdimen\@lnumwidth
3339 \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。

```

3340 \def\jsTocLine{\leaders\hbox{%
3341   $\m@th \mkern \@dotsep mu\hbox{.}\mkern \@dotsep mu$\hfill}
3342 \def\@dottedtocline#1#2#3#4#5{\ifnum #1>\c@tocdepth \else
3343   \vskip \z@ \@plus.2\jsc@mp
3344   {\leftskip #2\relax \rightskip \@tocrmarg \parfillskip -\rightskip
3345    \parindent #2\relax\@afterindenttrue
3346    \interlinepenalty\@M
3347    \leavevmode
3348    \@lnumwidth #3\relax
3349    \advance\leftskip \@lnumwidth \null\nobreak\hskip -\leftskip
3350    {#4}\nobreak
3351    \jsTocLine \nobreak\hb@xt@\@pnumwidth{%
3352      \hfil\normalfont \normalcolor #5\par}\fi}

```

■ 図目次と表目次

`\listoffigures` 図目次を出力します。

```

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

```

```

3365 %<book|report> \if@restonecol\twocolumn\fi
3366 }

```

`\l@figure` 図目次の項目を出力します。

```

3367 \newcommand{\l@figure}{\@dottedtocline{1}{1\jsZw}{3.683\jsZw}}

```

`\listoftables` 表目次を出力します。

```

3368 \newcommand{\listoftables}{%
3369 %<*book|report>
3370 \if@twocolumn\@restonecoltrue\onecolumn
3371 \else\@restonecolfalse\fi
3372 \chapter*{\listtablename}%
3373 \@mkboth{\listtablename}{}%
3374 %</book|report>
3375 %<!*book&!report>
3376 \section*{\listtablename}%
3377 \@mkboth{\listtablename}{\listtablename}%
3378 %</!*book&!report>
3379 \@starttoc{lot}%
3380 %<book|report> \if@restonecol\twocolumn\fi
3381 }

```

`\l@table` 表目次は図目次と同じです。

```

3382 \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。

```

3383 \newdimen\bibindent
3384 \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。

```

3385 \newenvironment{thebibliography}[1]{%
3386 \@jsc@warnoldfontcmdexceptiontrue
3387 \global\let\presectionname\relax
3388 \global\let\postsectionname\relax
3389 %<article|slide> \section*{\refname}\@mkboth{\refname}{\refname}%
3390 %<*kiyou>
3391 \vspace{1.5\baselineskip}
3392 \subsubsection*{\refname}\@mkboth{\refname}{\refname}%
3393 \vspace{0.5\baselineskip}
3394 %</kiyou>
3395 %<book|report> \chapter*{\bibname}\@mkboth{\bibname}{}%
3396 %<book|report> \addcontentsline{toc}{chapter}{\bibname}%

```

```

3397 \list{\@biblabel{\@arabic\c@enumiv}}%
3398     {\settowidth\labelwidth{\@biblabel{#1}}%
3399     \leftmargin\labelwidth
3400     \advance\leftmargin\labelsep
3401     \@openbib@code
3402     \usecounter{enumiv}%
3403     \let\p@enumiv\@empty
3404     \renewcommand\theenumiv{\@arabic\c@enumiv}}%
3405 %<kiyou> \small
3406 \sloppy
3407 \clubpenalty4000
3408 \@clubpenalty\clubpenalty
3409 \widowpenalty4000%
3410 \sfcode`\.\@m}
3411 {\def\@noitemerr
3412   {\@latex@warning{Empty `thebibliography' environment}}%
3413 \endlist}

```

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

```

3414 \newcommand{\newblock}{\hskip .11em\@plus.33em\@minus.07em}

```

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

```

3415 \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。

```

3416 % \def\@biblabel#1{\jsInhibitGlue [#1] \jsInhibitGlue}

```

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

```

3417 % \def\@citex[#1]#2{\leavevmode
3418 %   \let\@citea\@empty
3419 %   \@cite{\@for\@citeb:=#2\do
3420 %     {\@citea\def\@citea{, \inhibitglue\penalty\@m\ }%
3421 %     \edef\@citeb{\expandafter\@firstofone\@citeb\@empty}}%
3422 %     \if@filesw\immediate\write\@auxout{\string\citation{\@citeb}}\fi
3423 %     \ifundefined{b@\@citeb}{\mbox{\normalfont\bfseries ?}}%
3424 %     \G@refundefinedtrue
3425 %     \@latex@warning
3426 %       {Citation `'\@citeb' on page \thepage \space undefined}}%
3427 %     {\@cite@ofmt{\csname b@\@citeb\endcsname}}}{#1}}
3428 % \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。

```
3429 % \DeclareRobustCommand\cite{\unskip
3430 %   \@ifnextchar [{\@tempswatrue\@citex}{\@tempswafalse\@citex[]}]
3431 % \def\@cite#1#2{${\hbox{\scriptsize{#1}\if@tempswa
3432 %   , \jsInhibitGlue\ #2\fi}) }}$}
```

10.3 索引

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

```
3433 \newenvironment{theindex}{% 索引を3段組で出力する環境
3434   \if@twocolumn
3435     \onecolumn\@restonecolfalse
3436   \else
3437     \clearpage\@restonecoltrue
3438   \fi
3439   \columnseprule.4pt \columnsep 2\jsZw
3440   \ifx\multicols\undefined
```

`hyperref` 使用時に索引へのリンクが正常に作られるように、`hyperref` の説明書の解説に従って `\phantomsection` を配置した。

```
3441 %<book|report>      \twocolumn[\bxjs@phantomsection
3442 %<book|report>      \addcontentsline{toc}{chapter}{\indexname}%
3443 %<book|report>      \@makeschapterhead{\indexname}]]%
3444 %<!book&!report>    \def\presectionname{}\def\postsectionname{}%
3445 %<!book&!report>    \twocolumn[\section*{\indexname}]]%
3446   \else
3447     \ifdim\textwidth<\fullwidth
3448       \setlength{\evensidemargin}{\oddsidemargin}
3449       \setlength{\textwidth}{\fullwidth}
3450       \setlength{\linewidth}{\fullwidth}
3451 %<book|report>      \begin{multicols}{3}[\chapter*{\indexname}%
3452 %<book|report>      \addcontentsline{toc}{chapter}{\indexname}]]%
3453 %<!book&!report>    \def\presectionname{}\def\postsectionname{}%
3454 %<!book&!report>    \begin{multicols}{3}[\section*{\indexname}]]%
3455   \else
3456 %<book|report>      \begin{multicols}{2}[\chapter*{\indexname}%
3457 %<book|report>      \addcontentsline{toc}{chapter}{\indexname}]]%
3458 %<!book&!report>    \def\presectionname{}\def\postsectionname{}%
3459 %<!book&!report>    \begin{multicols}{2}[\section*{\indexname}]]%
3460   \fi
3461   \fi
3462 %<book|report>      \@mkboth{\indexname}{}%
3463 %<!book&!report>    \@mkboth{\indexname}{\indexname}%
3464   \plainifnotempty % \thispagestyle{plain}
```



```

3465 \parindent\z@
3466 \parskip\z@ \@plus .3\jsc@empt\relax
3467 \let\item\@idxitem
3468 \raggedright
3469 \footnotesize\narrowbaselines
3470 }{
3471 \ifx\multicols\@undefined
3472 \if@restonecol\onecolumn\fi
3473 \else
3474 \end{multicols}
3475 \fi
3476 \clearpage
3477 }

```

`\@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 3478 \newcommand{\@idxitem}{\par\hangindent 4\jsZw} % 元 40pt
\subsubitem 3479 \newcommand{\subitem}{\@idxitem \hspace*{2\jsZw}} % 元 20pt
3480 \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。

```

3481 \newcommand{\indexspace}{\par \vskip 10\jsc@empt \@plus5\jsc@empt \@minus3\jsc@empt\relax}

```

`\seenname` 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) などでいいでしょう。

```

3482 \newcommand\seenname{\if@english see\else →\fi}
3483 \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。

```

3484 \ifx\pltx@foot@penalty\@undefined
3485 \let\footnotes@ve=\footnote
3486 \def\footnote{\jsInhibitGlue\footnotes@ve}
3487 \let\footnotemarks@ve=\footnotemark
3488 \def\footnotemark{\jsInhibitGlue\footnotemarks@ve}
3489 \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 `newtexttext` や `newpertext` 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 `defaultsup` オプションを付けてください (qa:57284, qa:57287)。

```
3490 \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。

```
3491 % \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。

```
3492 \renewcommand{\footnoterule}{%
3493   \kern-2.6\jsc@mp \kern-.4\p@
3494   \hrule width .4\columnwidth
3495   \kern 2.6\jsc@mp}
```

`\c@footnote` 脚注番号は章ごとにリセットされます。

```
3496 %<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 pLaTeX 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 pLaTeX のバグ修正に従いました。

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

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

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

```
3497 \long\def\@footnotetext{%
3498   \insert\footins\bgroup
3499     \normalfont\footnotesize
3500     \interlinepenalty\interfootnotelinepenalty
3501     \splittopskip\footnotesep
3502     \splitmaxdepth \dp\strutbox \floatingpenalty \@MM
3503     \hsize\columnwidth \@parboxrestore
3504     \def\@currentcounter{footnote}%
3505     \protected@edef\@currentlabel{%
3506       \csname p@footnote\endcsname\@thefnmark
3507     }%
3508     \color@begingroup
3509       \@makefntext{%
3510         \rule{z@}\footnotesep\ignorespaces}%
3511       \futurelet\jsc@next\jsc@fo@t}
3512 \def\jsc@fo@t{\ifcat\bgroup\noexpand\jsc@next \let\jsc@next\jsc@fo@t
3513               \else \let\jsc@next\jsc@fo@t\fi \jsc@next}
3514 \def\jsc@fo@t@prefix{\jsInhibitGlue\ignorespaces}
3515 \def\jsc@fo@t@bgroup\aftergroup\jsc@foot\afterassignment\jsc@fo@t@prefix\let\jsc@next}
3516 \def\jsc@fo@t#1{\jsc@fo@t@prefix#1\jsc@foot}
3517 \def\jsc@foot{\@finalstrut\strutbox\color@endgroup\egroup
3518   \ifx\pltx@foot@penalty\@undefined\else
3519     \ifhmode\null\fi
3520     \ifnum\pltx@foot@penalty=z@\else
3521       \penalty\pltx@foot@penalty
3522       \pltx@foot@penaltyz@
3523     \fi
3524   \fi}
```

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

```
3525 \newcommand\@makefntext[1]{%
3526   \advance\leftskip 3\jsZw
3527   \parindent 1\jsZw
3528   \noindent
3529   \llap{\@makefnmark\hskip0.3\jsZw}#1}
```

`\@xfootnotenext` 最初の `\footnotetext{...}` は番号が付きません。著者の所属などを脚注の欄に書くときに便利です。

すでに `\footnote` を使った後なら `\footnotetext[0]{...}` とすれば番号を付けない脚注になります。ただし、この場合は脚注番号がリセットされてしまうので、工夫が必要です。

[2002-04-09] インプリメントの仕方を変えたため消しました。

```
3530 % \def\@xfootnotenext[#1]{%
3531 %   \begingroup
```

```

3532 %      \ifnum#1>\z@
3533 %      \csname c@\mpfn\endcsname #1\relax
3534 %      \unrestored@protected@xdef\@thefnmark{\thempfn}%
3535 %      \else
3536 %      \unrestored@protected@xdef\@thefnmark{}%
3537 %      \fi
3538 %  \endgroup
3539 %  \@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。

```
3540 \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。

```

3541 \def\everyparhook{\jsInhibitGlueAtParTop}
3542 \ifnum\bxjs@everyparhook=\bxjs@everyparhook@@compat
3543 \g@addto@macro\bxjs@begin@document@hook{\everypar{\everyparhook}}
3544 \fi

```

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

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

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

```

3545 \ifx j\jsEngine
3546 \def\@inhibitglue{%
3547   \futurelet\@let@token\@@inhibitglue}
3548 \begingroup

```

```

3549 \let\GDEF=\gdef
3550 \let\CATCODE=\catcode
3551 \let\ENDGROUP=\endgroup
3552 \CATCODE`k=12
3553 \CATCODE`a=12
3554 \CATCODE`n=12
3555 \CATCODE`j=12
3556 \CATCODE`i=12
3557 \CATCODE`c=12
3558 \CATCODE`h=12
3559 \CATCODE`r=12
3560 \CATCODE`t=12
3561 \CATCODE`e=12
3562 \GDEF\KANJI@CHARACTER{kanji character }
3563 \ENDGROUP
3564 \def\@@inhibitglue{%
3565   \expandafter\expandafter\expandafter\jsc@inhibitglue\expandafter\meaning\expandafter\@let@
3566   \expandafter\def\expandafter\jsc@inhibitglue\expandafter#\expandafter1\KANJI@CHARACTER#2#3\j
3567   \def\jsc@ig@temp{#1}%
3568   \ifx\jsc@ig@temp\@empty
3569     \ifnum\the\inhibitxspcode`#2=2\relax
3570       \inhibitglue
3571     \fi
3572   \fi}
3573 \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。

```

3574 \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。

```

3575 \def\@doendpe{%
3576   \@endpetrue
3577   \def\par{%
3578     \@restorepar\clubpenalty\@clubpenalty\everypar{\everyparhook}\par\@endpefalse}%
3579   \everypar{\setbox\z@\lastbox}\everypar{\everyparhook}\@endpefalse\everyparhook}}

```

[2017-08-31] minipage 環境にも対策します。

```

3580 \def\@setminipage{%
3581   \@minipagetrue
3582   \everypar{\@minipagefalse\everypar{\everyparhook}}%
3583 }

```

\item 命令の直後です。

```
3584 \def\@item[#1]{%
3585   \if@nparitem
3586     \donoparitem
3587   \else
3588     \if@inlabel
3589       \indent \par
3590     \fi
3591     \ifhmode
3592       \unskip\unskip \par
3593     \fi
3594     \if@newlist
3595       \if@nobreak
3596         \@nbitem
3597       \else
3598         \addpenalty\@beginparpenalty
3599         \addvspace\@topsep
3600         \addvspace{-\parskip}%
3601       \fi
3602     \else
3603       \addpenalty\@itempenalty
3604       \addvspace\itemsep
3605     \fi
3606     \global\@inlabeltrue
3607   \fi
3608   \everypar{%
3609     \@minipagefalse
3610     \global\@newlistfalse
3611     \if@inlabel
3612       \global\@inlabelfalse
3613       {\setbox\z@\lastbox
3614        \ifvoid\z@
3615          \kern-\itemindent
3616        \fi}%
3617     \box\@labels
3618     \penalty\z@
3619   \fi
3620   \if@nobreak
3621     \@nobreakfalse
3622     \clubpenalty \@M
3623   \else
3624     \clubpenalty \@clubpenalty
3625     \everypar{\everyparhook}%
3626   \fi\everyparhook}%
3627 \if@noitemarg
3628   \@noitemargfalse
3629   \if@nmbrrlist
3630     \refstepcounter\@listctr
```

```

3631 \fi
3632 \fi
3633 \sbox\@tempboxa{\makelabel{#1}}%
3634 \global\setbox\@labels\hbox{%
3635   \unhbox\@labels
3636   \hskip \itemindent
3637   \hskip -\labelwidth
3638   \hskip -\labelsep
3639   \ifdim \wd\@tempboxa >\labelwidth
3640     \box\@tempboxa
3641   \else
3642     \hbox to\labelwidth {\unhbox\@tempboxa}%
3643   \fi
3644   \hskip \labelsep}%
3645 \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。

```

3646 \def\@afterheading{%
3647   \@nobraektrue
3648   \everypar{%
3649     \if@nobraek
3650       \@nobraekfalse
3651       \clubpenalty \@M
3652       \if@afterindent \else
3653         {\setbox\z@\lastbox}%
3654       \fi
3655     \else
3656       \clubpenalty \@clubpenalty
3657       \everypar{\everyparhook}%
3658     \fi\everyparhook}}

```

「`\everyparhook` 用のパッチ処理」はここまで。

```

3659 \fi

```

`\@gnewline` についてはちょっと複雑な心境です。もともとの $\text{p}\text{A}\text{T}\text{E}\text{X} 2_{\epsilon}$ は段落の頭にグルーが入る方で統一されていました。しかし `\` の直後にはグルーが入らず、不統一でした。そこで `\` の直後にもグルーを入れるように直していただいた経緯があります。しかし、ここでは逆にグルーを入れない方で統一したいので、また元に戻してしまいました。

しかし単に戻すだけでも駄目みたいなので、ここでも最後にグルーを消しておきます。

※ `luatexja` を読み込んだ場合に `lltjcore.sty` によって上書きされるのを防ぐため遅延させる。

```

3660 \ifnum\bxjs@everyparhook=\bxjs@everyparhook@@none\else

```

```

3661 \AtEndOfClass{%
3662 \def\@gnewline #1{%
3663   \ifvmode
3664     \@nolnerr
3665   \else
3666     \unskip \reserved@e {\reserved@f#1}\nobreak \hfil \break \null
3667     \jsInhibitGlue \ignorespaces
3668   \fi}
3669 }
3670 \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。

```

3671 \if@jslogo
3672   \IfFileExists{jslogo.sty}{%
3673     \RequirePackage{jslogo}%
3674   }{%
3675     \ClassWarningNoLine{bxjs@clsname}
3676       {The package 'jslogo' is not installed.\MessageBreak
3677        It is included in the recent release of\MessageBreak
3678        the 'jsclasses' bundle}
3679   }
3680 \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。


```

3681 %\let\ltx@ifnextchar\@ifnextchar
3682 %\def\ProvidesFile#1{%
3683 % \begingroup
3684 % \catcode\ 10 %
3685 % \ifnum \endlinechar<256 %
3686 % \ifnum \endlinechar>\m@ne
3687 % \catcode\endlinechar 10 %
3688 % \fi
3689 % \fi
3690 % \@makeother\/%
3691 % \@makeother\&%
3692 % \ltx@ifnextchar[\@providesfile{#1}]{\@providesfile{#1}[]}}

```

14 初期設定

■いろいろな語

```

\prepartname
\postpartname 3693 \newcommand{\prepartname}{\if@english Part~\else 第\fi}
\prechaptername 3694 \newcommand{\postpartname}{\if@english\else 部\fi}
\postchaptername 3695 %<book|report>\newcommand{\prechaptername}{\if@english Chapter~\else 第\fi}
\postchaptername 3696 %<book|report>\newcommand{\postchaptername}{\if@english\else 章\fi}
\presectionname 3697 \newcommand{\presectionname}{}% 第
\postsectionname 3698 \newcommand{\postsectionname}{}% 節

\contentsname
\listfigurename 3699 \newcommand{\contentsname}{\if@english Contents\else 目次\fi}
\listtablename 3700 \newcommand{\listfigurename}{\if@english List of Figures\else 図目次\fi}
3701 \newcommand{\listtablename}{\if@english List of Tables\else 表目次\fi}

\refname
\bibname 3702 \newcommand{\refname}{\if@english References\else 参考文献\fi}
\indexname 3703 \newcommand{\bibname}{\if@english Bibliography\else 参考文献\fi}
3704 \newcommand{\indexname}{\if@english Index\else 索引\fi}

\figurename
\tablename 3705 %<!\jspf>\newcommand{\figurename}{\if@english Fig.~\else 図\fi}
3706 %<jspf>\newcommand{\figurename}{Fig.~}
3707 %<!\jspf>\newcommand{\tablename}{\if@english Table~\else 表\fi}
3708 %<jspf>\newcommand{\tablename}{Table~}

\appendixname
\abstractname 3709 % \newcommand{\appendixname}{\if@english Appendix~\else 付録\fi}
3710 \newcommand{\appendixname}{\if@english \else 付録\fi}
3711 %<!\book>\newcommand{\abstractname}{\if@english Abstract\else 概要\fi}

```

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

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

※ 2.0 版より、完全に bxwareki に任せる。

\西暦 8 ビット欧文 T_EX ではそもそも非 ASCII の制御綴は使えないのであるが、JS クラスのユーザーザ命令である \西暦/\和暦 だけは擬似的に使えるようにする。欧文 T_EX では

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

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

T_EX エンジンの種類により処理を分ける。

```
3712 \onlypreamble\bxjs@decl@Seireki@cmds
3713 \@tempswafalse
3714 \if p\jsEngine \@tempswattrue \fi
3715 \if n\jsEngine \@tempswattrue \fi
3716 \bxjs@cond\if@tempswa\fi{%
```

8 ビット欧文 T_EX の場合。

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

```
3717 \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。

```
3718 \def\bxjs@decl@Seireki@cmds{%
3719   \def\西暦{\jsSeirekitrue}%
3720   \def\和暦{\jsSeirekifalse\bxjs@wareki@used}}
```

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

```
\Wareki 3721 \def\Seireki{\jsSeirekitrue}
3722 \def\Wareki{\jsSeirekifalse\bxjs@wareki@used}

3723 \def\bxjs@if@use@seireki{\bxjs@cond\ifjsSeireki\fi}
3724 \def\bxjs@iaif{\noexpand~}
3725 }{%
```

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

```
3726 \newif\if西暦 \西暦true
3727 \def\bxjs@decl@Seireki@cmds{%
3728   \def\西暦{\西暦true}%
3729   \def\和暦{\西暦false\bxjs@wareki@used}}
3730 \def\Seireki{\西暦true}
3731 \def\Wareki{\西暦false\bxjs@wareki@used}
```

```

3732 \def\bxjs@if@use@seireki{\bxjs@cond\if西暦\fi}
3733 \let\bxjs@iaai\@empty
3734 }
3735 \bxjs@decl@Seireki@cmds

```

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

※ 8 ビット欧文でかつ非 e-TeX なエンジン（現状ではサポート外だが）では `bxwareki` を読むだけでエラーが発生してしまうので、この場合は読込を回避する。

```

3736 \newif\ifbxjs@bxwareki@avail
3737 \IfFileExists{bxwareki.sty}{%
3738   \if \if n\jsEngine \ifjsWitheTeX T\else F\fi\else T\fi T%
3739     \RequirePackage{bxwareki}[2018/04/08]%v0.2
3740     \bxjs@bxwareki@availtrue
3741   \fi}{%

```

`\bxjs@wareki@used` `bxwareki` が利用できないのに和暦出力をしようとした場合に警告を出す。

```

3742 \ifbxjs@bxwareki@avail \let\bxjs@wareki@used\@empty
3743 \else
3744   \bxjs@robust@def\bxjs@wareki@used{%
3745     \global\let\bxjs@wareki@used\@empty
3746     \ClassWarning\bxjs@clsname
3747       {Wareki mode is not supported, since\MessageBreak
3748         'bxwareki' is unavailable, reported}}
3749     \g@addto@macro\bxjs@begin@document@hook{%
3750       \let\bxjs@wareki@used\@empty}
3751 \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。

```

3752 \ifbxjs@bxwareki@avail
3753   \let\jayear\warekiyear
3754   \def\bxjs@tmpa{H}\ifx\bxjs@tmpa\warekigengoinitial
3755     \newcount\heisei \heisei=\value{warekiyear}
3756   \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。

```

3757 \else
3758   \edef\jayear{\the\year \bxjs@iaai}
3759 \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。

```

3760 %<*jsclasses>
3761 \newif\if西暦 \西暦true

```

```

3762 \def\西暦{\西暦true}
3763 \def\和暦{\西暦false}
3764 \newcount\heisei \heisei\year \advance\heisei-1988\relax
3765 \def\pltx@today@year@#1{%
3766   \ifnum\numexpr\year-#1=1 元\else
3767     \ifnum1=\iftdir\ifmdir0\else1\fi\else0\fi
3768     \kansuji\numexpr\year-#1\relax
3769   \else
3770     \number\numexpr\year-#1\relax\nobreak
3771   \fi
3772   \fi 年
3773 }
3774 \def\pltx@today@year{%
3775   \ifnum\numexpr\year*10000+\month*100+\day<19890108
3776     昭和\pltx@today@year@{1925}%
3777   \else\ifnum\numexpr\year*10000+\month*100+\day<20190501
3778     平成\pltx@today@year@{1988}%
3779   \else
3780     令和\pltx@today@year@{2018}%
3781   \fi\fi}
3782 %</jsclasses>
3783 \begingroup
3784 \let\bxjs@next\relax
3785 \ifbxjs@bxwareki@avail \ifx\warekigengo\@empty\else
3786   \def\bxjs@next{\warekitoday}
3787   \bxjs@test@engine\unexpanded{%
3788     \def\bxjs@next{\unexpanded\expandafter{\warekitoday}}}}
3789 \fi\fi
3790 \def\!#1#2#3{\noexpand#1\noexpand#2\noexpand#3}
3791 \ifx\bxjs@iai\@empty \let\!\@empty \fi
3792 \xdef\bxjs@today{%
3793   \if@english
3794     \ifcase\month\or
3795       January\or February\or March\or April\or May\or June\or
3796       July\or August\or September\or October\or November\or December\fi
3797     \space\number\day, \number\year
3798   \else
3799     \ifx\bxjs@next\relax \expandafter\@firstoftwo
3800     \else \noexpand\bxjs@if@use@seireki
3801     \fi {%
3802       \number\year\bxjs@iai\!年%
3803       \bxjs@iai\number\month\bxjs@iai\!月%
3804       \bxjs@iai\number\day\bxjs@iai\!日%
3805     }\bxjs@next}%
3806   \fi}
3807 \endgroup
3808 \let\today\bxjs@today

```

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

```
3809 \g@addto@macro\bxjs@begin@document@hook{%
3810   \ifx\bb1@jpn@maybekansuji\@undefined\else
3811     \bxjs@decl@Seireki@cmds
3812     \g@addto@macro\datejapanese{%
3813       \let\today\bxjs@today}%
3814   \fi}
```

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

```
3815 \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。

```
3816 %<slide>\pagestyle{empty}%
3817 %<article|report>\pagestyle{plain}%
3818 %<book>\pagestyle{headings}%
3819 \pagenumbering{arabic}
3820 \if@twocolumn
3821   \twocolumn
3822   \sloppy
3823   \flushbottom
3824 \else
3825   \onecolumn
3826   \raggedbottom
3827 \fi
3828 %<*slide>
3829 \renewcommand\familydefault{\sfdefault}
3830 \raggedright
3831 %</slide>
```

15 実験的コード

この節は JS クラスの話で、BXJS クラスには当てはまらない。

[2016-11-29] コミュニティ版 pLaTeX で新設されたテスト用パッケージ (expp12e パッケージ) が文書クラスより先に読み込まれていた場合は、jsclasses もテスト版として動作します。この処置は jsarticle, jsbook, jsreport にのみ行い、jspf と kiyou は除外しておきます。expp12e パッケージが読みこまれていない場合は通常版として動作しますので、ここで終了します。

以上です。

16 BXJS 独自の追加処理

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

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

\jsStrongText 強調用の宣言型命令。

```
3832 \bxjs@robust@def\jsStrongText{\bxjs@strong@text}%
```

fontspec と互換の \strongfontdeclare 命令も提供する。既定の設定は \bfseries (太字) である。

※\strongfontdeclare は試験的機能とする。

```
3833 \chardef\bxjs@strong@level=0
3834 \DeclareRobustCommand*\jsStrongDeclare[1]{%
3835   \bxjs@set@array@from@clist{\bxjs@strong}{#1}%
3836   \chardef\bxjs@strong@level\z@}
3837 \jsStrongDeclare{\bfseries}
3838 \def\bxjs@strong@text{%
3839   \bxjs@csletcs{\bxjs@tmpa}{\bxjs@strong/\the\bxjs@strong@level}%
3840   \ifx\bxjs@tmpa\relax
3841     \bxjs@advance@qc\bxjs@strong@level\m@ne \bxjs@strong@text
3842   \else \bxjs@advance@qc\bxjs@strong@level\@ne \bxjs@tmpa
3843   \fi}
3844 % \end{macro}
3845 %
3846 % \paragraph{共通命令の実装}
3847 %
3848 % |\jQ| 等の「単位」系の共通命令を実装する。
3849 %
3850 % \begin{macro}{\bxjs@const@unit}
3851 % 固定値の単位として使える制御綴を定義する。
3852 %
3853 % {\eTeX}拡張が使える場合は、
3854 % 「|\dimexpr|外部寸法表記|\relax|」の形式
3855 % (これは内部値なので単位として使える)
3856 % に展開されるマクロとして定義する。
3857 %   \begin{macrocode}
3858 \@onlypreamble\bxjs@const@unit
3859 \@onlypreamble\bxjs@const@unit@a
3860 \ifjsWitheTeX
3861   \def\bxjs@const@unit#1#2#3{%
3862     \protected\edef#1{\dimexpr\the\dimexpr#3\relax\relax}}
```

ϵ -TeX 拡張が使えない場合は、何らかの寸法パラメタに値を保持する必要があるが、レジスタは貴重なので代わりに「ダミーの TFM を定義してその \fontdimen を使う」というテク

ニックを用いる (アレ)。

```
3863 \else
3864   \let\bxjs@Ct\fontdimen \font\bxjs@Ut=cmtex9 at 0.98245pt
3865   \bxjs@Ct8\bxjs@Ut=8sp \bxjs@Ct16\bxjs@Ut=\z@
3866   \def\bxjs@const@unit#1#2{%
3867     \expandafter\bxjs@const@unit@a\csname bxjs@#2@\endcsname#1}
3868   \def\bxjs@const@unit@a#1#2#3{%
3869     \chardef#1\bxjs@Ct8\bxjs@Ut \bxjs@advance@qc#1\@ne \bxjs@Ct8\bxjs@Ut#1sp
3870     \bxjs@Ct#1\bxjs@Ut=#3\relax \def#2{\bxjs@Ct#1\bxjs@Ut}}
3871 \fi
```

\jQ \jQ と \jH はともに 0.25 mm に等しい。

```
\jH 3872 \bxjs@const@unit\jQ{jQ}{0.25mm}
3873 \let\jH\jQ
```

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

```
\trueH 3874 \ifjsc@mag
3875   \@tempdimb=\jsBaseFontSize\relax
3876   \edef\bxjs@tmpa{\strip@pt\@tempdimb}%
3877   \@tempdima=2.5mm
3878   \bxjs@invscale\@tempdima\bxjs@tmpa
3879   \bxjs@const@unit\trueQ{trueQ}{\@tempdima}
3880   \@tempdima=10pt
3881   \bxjs@invscale\@tempdima\bxjs@tmpa
3882   \bxjs@const@unit\bxjs@truept{truept}{\@tempdima}
3883 \else \let\trueQ\jQ \let\bxjs@truept\p@
3884 \fi
3885 \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。

```
3886 \@tempdima\trueQ \bxjs@invscale\@tempdima\jsScale
3887 \bxjs@const@unit\ascQ{ascQ}{\@tempdima}
3888 \@tempdima\bxjs@truept \bxjs@invscale\@tempdima\jsScale
3889 \bxjs@const@unit\ascpt{ascpt}{\@tempdima}
```

\jafontsize \jafontsize{〈フォントサイズ〉}{〈行送り〉}： 和文フォント規準で、すなわち、1zw が〈
フォントサイズ〉に等しくなるようにフォントサイズを指定する。この命令の引数では、Q/H
の単位が使用できる。

```
3890 \DeclareRobustCommand*\jsJaFontSize[2]{%
3891   \begingroup
3892     \bxjs@jafontsize@a{#1}%
3893     \@tempdimb\jsInverseScale\@tempdima
3894     \bxjs@jafontsize@a{#2}%
3895     \xdef\bxjs@g@tmpa{%
3896       \noexpand\fontsize{\the\@tempdimb}{\the\@tempdima}}%
3897   \endgroup\bxjs@g@tmpa}
```

```

3898 \def\bxjs@jafontsize@a#1{%
3899   \bxjs@parse@qh{#1}%
3900   \ifx\bxjs@tmpb\relax \def\bxjs@tmpb{#1}\fi
3901   \@defaultunits\@tempdima\bxjs@tmpb pt\relax\@nnil}

```

続いて、和文間空白・和欧文間空白関連の命令を実装する。この実装はエンジンや和文処理パッケージに依存するが、ここでは共通の基盤となる部分を実装する。

```

3902 \def\bxjs@let@lenexpr{\edef}

```

`\bxjs@kanjiskip` 和文間空白の量を表すテキスト。

```

3903 \def\bxjs@kanjiskip{0pt}

```

`\jsSetKanjiSkip` 和文間空白の量を設定する。

※`\setkanjiskip` の実体。

```

3904 \DeclareRobustCommand*\jsSetKanjiSkip[1]{%
3905   \bxjs@let@lenexpr\bxjs@kanjiskip{#1}%
3906   \bxjs@reset@kanjiskip}

```

`\jsGetKanjiSkip` 和文間空白の量を表すテキストに展開する。

※`\getkanjiskip` の実体。

```

3907 \newcommand*\jsGetKanjiSkip{%
3908   \bxjs@kanjiskip}

```

`\ifbxjs@kanjiskip@enabled` 和文間空白の挿入が有効か。

※エンジン側の機能で制御する場合は、このスイッチは常に真にしておく。

```

3909 \newif\ifbxjs@kanjiskip@enabled \bxjs@kanjiskip@enabledtrue

```

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

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

```

3910 \bxjs@robust@def\jsEnableKanjiSkip{%
3911   \bxjs@kanjiskip@enabledtrue
3912   \bxjs@reset@kanjiskip}
3913 \bxjs@robust@def\jsDisableKanjiSkip{%
3914   \bxjs@kanjiskip@enabledfalse
3915   \bxjs@reset@kanjiskip}

```

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

```

3916 \bxjs@robust@def\bxjs@reset@kanjiskip{%
3917   \ifbxjs@kanjiskip@enabled
3918     \setlength{\@tempskipa}{\bxjs@kanjiskip}%
3919   \else \@tempskipa\z@
3920   \fi
3921   \jsApplyKanjiSkip\@tempskipa}

```

`\jsApplyKanjiSkip` `\jsApplyKanjiSkip{〈グループ値〉}`： 和文間空白を実際に設定するためのエンジン依存のコード。

```

3922 \let\jsApplyKanjiSkip\@gobble

```


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

```

\jsSetXKanjiSkip 3923 \def\bxjs@xkanjiskip{0pt}
\jsGetXKanjiSkip 3924 \DeclareRobustCommand*\jsSetXKanjiSkip[1]{%
\ifbxjs@xkanjiskip@enabled 3925 \bxjs@let@lenexpr\bxjs@xkanjiskip{#1}%
3926 \bxjs@reset@xkanjiskip}
\jsEnableXKanjiSkip 3927 \newcommand*\jsGetXKanjiSkip{%
3928 \bxjs@xkanjiskip}
\jsDisableXKanjiSkip 3929 \newif\ifbxjs@xkanjiskip@enabled \bxjs@xkanjiskip@enabledtrue
\bxjs@reset@xkanjiskip 3930 \bxjs@robust@def\jsEnableXKanjiSkip{%
\jsApplyXKanjiSkip 3931 \bxjs@xkanjiskip@enabledtrue
3932 \bxjs@reset@xkanjiskip}
3933 \bxjs@robust@def\jsDisableXKanjiSkip{%
3934 \bxjs@xkanjiskip@enabledfalse
3935 \bxjs@reset@xkanjiskip}
3936 \bxjs@robust@def\bxjs@reset@xkanjiskip{%
3937 \ifbxjs@xkanjiskip@enabled
3938 \setlength{\@tempskipa}{\bxjs@xkanjiskip}%
3939 \else \@tempskipa\z@
3940 \fi
3941 \jsApplyXKanjiSkip\@tempskipa}
3942 \let\jsApplyXKanjiSkip@gobble

```

`\jsResetDimen` を用いて、フォントサイズが変更された時に空白の量が追従するようにする。

```

3943 \g@addto@macro\jsResetDimen{%
3944 \bxjs@reset@kanjiskip
3945 \bxjs@reset@xkanjiskip}

```

和文・和欧文間空白の初期値。

```

3946 \AtEndOfPackage{%
3947 \jsSetKanjiSkip{0pt plus.1\jsZw minus.01\jsZw}%
3948 \ifx\jsDocClass\jsSlide \jsSetXKanjiSkip{0.1em}%
3949 \else \jsSetXKanjiSkip{0.25em plus 0.15em minus 0.06em}%
3950 \fi
3951 }

```

■和文空白命令

```

3952 \ifbxjs@jaspace@cmd

```

`\jaenspace` 半角幅の水平空き。

```

3953 \def\jaenspace{\hskip.5\jsZw\relax}

```

`\jathinspace` 和欧文間空白を入れるユーザ命令。

```

3954 \def\jathinspace{\hskip\bxjs@xkanjiskip\relax}

```

`_` 全角空白文字 1 つからなる名前の制御綴。 `\zwspace` と等価になる。

```

3955 \def\_ {\zwspace}

```

`\>` 非数式中では `\jathinspace` と等価になるように再定義する。

※数式中では従来通り（\: と等価）。

```
3956 \bxjs@protected\def\bxjs@choice@jathinspace{%
3957   \relax\ifmmode \mskip\medmuskip
3958   \else \jathinspace\ignorespaces
3959   \fi}
3960 \jsAtEndOfClass{%
3961   \ifjsWithTeX \let\>\bxjs@choice@jathinspace
3962   \else \def\>\protect\bxjs@choice@jathinspace}%
3963   \fi}
```

\jaspace jlreq クラスと互換の命令。

```
3964 \DeclareRobustCommand*\jaspace}[1]{%
3965   \expandafter\ifx\csname bxjs@jaspace@@#1\endcsname\relax
3966   \ClassError\bxjs@clsname
3967   {Unknown jaspac: #1}{\@eha}%
3968   \else
3969   \csname bxjs@jaspace@@#1\endcsname
3970   \fi}
3971 \def\bxjs@jaspace@@zenkaku{\hskip 1\jsZw\relax}
3972 \def\bxjs@jaspace@@nibu{\hskip .5\jsZw\relax}
3973 \def\bxjs@jaspace@@shibu{\hskip .25\jsZw\relax}
```

終わり。

```
3974 \fi
```

■和文ドライバ読込 フックを実行する。

```
3975 \bxjs@pre@jadriver@hook
```

和文ドライバのファイルを読み込む。

```
3976 \input{bxjsja-\bxjs@jadriver.def}
```

おしまい。

```
3977 %</class>
```

付録 A 和文ドライバの仕様

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

- `\jsDocClass` [文字トークンの `let`] 文書クラスの種類を示し、次のいずれかと一致する (`\if` で判定可能)。
 - `\jsArticle` `bxjsarticle` クラス
 - `\jsBook` `bxjsbook` クラス
 - `\jsReport` `bxjsreport` クラス
 - `\jsSlide` `bxjsslide` クラス
- `\jsEngine` [文字トークンの `let`] 使用されているエンジンの種別。 (`\if` で判定可能)。
 - `p` `pdfTeX` (DVI モードも含む)
 - `l` `LuaTeX` (〃)
 - `x` `XYTeX`
 - `j` `pTeX` または `upTeX`
 - `n` 以上の何れでもない
- `\ifjsWithupTeX` [スイッチ] 使用されているエンジンが `upTeX` であるか。
- `\ifjsWitheTeX` [スイッチ] 使用されているエンジンが 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。

```
3978 %<*drv>
```

付録 B 和文ドライバ：minimal

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

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

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

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

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

B.1 準備

```
3979 %<*minimal>
```

```
3980 %% このファイルは日本語文字を含みます
```

■環境検査 minimal 和文ドライバの処理系バージョン要件はクラス本体と同じとする。

ただし「公式にはサポート外」のエンジンが使われている場合は強制終了させる。

※ NTT jIEX と Omega 系。

```
3981 \let\bxjs@tmpa\relax
3982 \ifx J\jsEngine \def\bxjs@tmpa{NTT-jTeX}\fi
3983 \ifx O\jsEngine \def\bxjs@tmpa{Omega}\fi
3984 \ifx\bxjs@tmpa\relax \expandafter\@gobble
3985 \else
3986   \ClassError\bxjs@clsname
3987   {The engine in use (\bxjs@tmpa) is not supported}
```

```

3988 {It's a fatal error. I'll quit right now.}
3989 \expandafter\@firstofone
3990 \fi{\endinput\@@end}

```

■補助マクロ

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

```

3991 \def\DeclareJaTextFontCommand#1#2{%
3992   \DeclareRobustCommand#1[1]{%
3993     \relax
3994     \ifmmode \expandafter\nfss@text \fi
3995     {#2##1}}%
3996 }

```

`\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。

```

3997 \def\DeclareJaMathFontCommand#1#2{%
3998   \DeclareRobustCommand#1[1]{%
3999     \relax
4000     \ifmmode\else \non@alpherr{#1\space}\fi
4001     \nfss@text{\fontfamily\familydefault
4002               \fontseries{m}\fontshape{n}\selectfont\relax
4003               #2##1}%
4004   }%
4005 }

```

`\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。

```

4006 \bxjs@nclong\def\bxjs@@CSsfdefault{\sfdefault}%
4007 \@onlypreamble\bxjs@if@sf@default
4008 \def\bxjs@if@sf@default#1{%
4009   \ifx\familydefault\bxjs@@CSsfdefault#1\fi
4010   \g@addto@macro\bxjs@begin@document@hook{%
4011     \ifx\familydefault\bxjs@@CSsfdefault#1\fi}%
4012 }

```

`\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。

```

4013 \@tempdima\p@ \bxjs@invscale\@tempdima\jsScale
4014 \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。

```

4015 \def\jsLetHeadChar#1#2{%
4016   \begingroup
4017   \escapechar=`\ %
4018   \let\bxjs@tmpa={% brace-match-hack
4019   \bxjs@let@hchar@exp#2}%
4020   \endgroup
4021   \let#1\bxjs@g@tmpa}
4022 \def\bxjs@let@hchar@exp{%
4023   \futurelet\@let@token\bxjs@let@hchar@exp@a}
4024 \def\bxjs@let@hchar@exp@a{%
4025   \bxjs@cond@ifcat\noexpand\@let@token\bgroup\fi{% 波括弧
4026     \bxjs@let@hchar@out\let\relax
4027   }\bxjs@cond@ifcat\noexpand\@let@token\@sptoken\fi{% 空白
4028     \bxjs@let@hchar@out\let\space%
4029   }\bxjs@cond@if\noexpand\@let@token\@backslashchar\fi{% バックスラッシュ
4030     \bxjs@let@hchar@out\let\@backslashchar
4031   }\bxjs@let@hchar@exp@b}}}}
4032 \def\bxjs@let@hchar@exp@b#1{%
4033   \expandafter\bxjs@let@hchar@exp@c\string#1?\@nil#1}
4034 \def\bxjs@let@hchar@exp@c#1#2\@nil{%
4035   %\message{<#1#2>}%
4036   \bxjs@cond@if#1\@backslashchar\fi{% 制御綴
4037     \bxjs@cond\expandafter\ifx\noexpand\@let@token\@let@token\fi{%
4038       \bxjs@let@hchar@out\let\relax
4039     }{%else
4040       \expandafter\bxjs@let@hchar@exp
4041     }%
4042   }\fi%else
4043     \bxjs@let@hchar@chr#1%
4044   }}
4045 \def\bxjs@let@hchar@chr#1{%
4046   \bxjs@let@hchar@out\def{{#1}}}
4047 \def\bxjs@let@hchar@out#1#2{%
4048   \global#1\bxjs@g@tmpa#2\relax
4049   \toks@\bgroup}% skip to right brace

```

UTF-8 のバイト列を扱うコード。

```

4050 \chardef\bxjs@let@hchar@csta=128
4051 \chardef\bxjs@let@hchar@cstb=192
4052 \chardef\bxjs@let@hchar@cstc=224
4053 \chardef\bxjs@let@hchar@cstd=240
4054 \chardef\bxjs@let@hchar@cste=248
4055 \let\bxjs@let@hchar@chr@ue@a\bxjs@let@hchar@chr
4056 \def\bxjs@let@hchar@chr@ue#1{%
4057   \@tempcnta=`#1\relax
4058   %\message{\the\@tempcnta}%
4059   \bxjs@cond@ifnum\@tempcnta<\bxjs@let@hchar@csta\fi{%
4060     \bxjs@let@hchar@chr@ue@a#1%
4061   }\bxjs@cond@ifnum\@tempcnta<\bxjs@let@hchar@cstb\fi{%

```

```

4062 \bxjs@let@hchar@out\let\relax
4063 }{\bxjs@cond@ifnum\@tempcnta<\bxjs@let@hchar@cstc\fi{%
4064 \bxjs@let@hchar@chr@ue@b
4065 }{\bxjs@cond@ifnum\@tempcnta<\bxjs@let@hchar@cstd\fi{%
4066 \bxjs@let@hchar@chr@ue@c
4067 }{\bxjs@cond@ifnum\@tempcnta<\bxjs@let@hchar@cste\fi{%
4068 \bxjs@let@hchar@chr@ue@d
4069 }{%else
4070 \bxjs@let@hchar@out\let\relax
4071 }}}}
4072 \def\bxjs@let@hchar@chr@ue@a#1{%
4073 \bxjs@let@hchar@out\def{{#1}}}
4074 \def\bxjs@let@hchar@chr@ue@b#1#2{%
4075 \bxjs@let@hchar@out\def{{#1#2}}}
4076 \def\bxjs@let@hchar@chr@ue@c#1#2#3{%
4077 \bxjs@let@hchar@out\def{{#1#2#3}}}
4078 \def\bxjs@let@hchar@chr@ue@d#1#2#3#4{%
4079 \bxjs@let@hchar@out\def{{#1#2#3#4}}}

```

B.2 (u)pTeX 用の設定

```
4080 \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。

```

4081 \def\bxjs@let@hchar@chr@pp#1#2{%
4082 \expandafter\bxjs@let@hchar@chr@pp@a\meaning#2\relax#1#2}
4083 \def\bxjs@let@hchar@chr@pp@a#1#2\relax#3#4{%
4084 %\message{(\meaning#3:\meaning#4)}%
4085 \bxjs@cond@if#1k\fi{%
4086 \bxjs@let@hchar@out\def{{#4}}}%
4087 }{%else
4088 \bxjs@let@hchar@chr@ue#3#4%
4089 }}
4090 \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。

```

4091 \edef\jsc@JYn{\ifjsWithupTeX JY2\else JY1\fi}
4092 \edef\jsc@JTn{\ifjsWithupTeX JT2\else JT1\fi}
4093 \edef\jsc@pfx0{\ifjsWithupTeX u\fi}

```

`\bxjs@declarefontshape` は標準の和文フォント宣言である。後で `\bxjs@scale` を求

めるため一旦マクロにしておく。`\bxjs@sizereference` は全角幅を測定する時に参照するフォント。

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

```
4094 \@onlypreamble\bxjs@declarefontshape
4095 \ifjsWithupTeX
4096 \def\bxjs@declarefontshape{%
4097 \DeclareFontShape{JY2}{mc}{m}{n}{<->s*[\bxjs@scale]upjpnrm-h}{}%
4098 \DeclareFontShape{JY2}{gt}{m}{n}{<->s*[\bxjs@scale]upjpngt-h}{}%
4099 \DeclareFontShape{JT2}{mc}{m}{n}{<->s*[\bxjs@scale]upjpnrm-v}{}%
4100 \DeclareFontShape{JT2}{gt}{m}{n}{<->s*[\bxjs@scale]upjpngt-v}{}%
4101 }
4102 \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。

```
4103 \else
4104 \def\bxjs@declarefontshape{%
4105 \DeclareFontShape{JY1}{mc}{m}{n}{<->s*[\bxjs@scale]jis}{}%
4106 \DeclareFontShape{JY1}{gt}{m}{n}{<->s*[\bxjs@scale]jisg}{}%
4107 \DeclareFontShape{JT1}{mc}{m}{n}{<->s*[\bxjs@scale]tmin10}{}%
4108 \DeclareFontShape{JT1}{gt}{m}{n}{<->s*[\bxjs@scale]tgoth10}{}%
4109 }
4110 \def\bxjs@sizereference{jis}
4111 \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。

```
4112 \def\bxjs@next#1/#2/#3/#4/#5\relax{%
4113   \def\bxjs@tmpb{#5}}
4114 \ifjsWithpTeXng \def\bxjs@tmpb{10}%
4115 \else
4116 \expandafter\expandafter\expandafter\bxjs@next
4117 \expandafter\string\the\jfont\relax
4118 \fi
4119 \@for\bxjs@tmpa:={\jsc@JYn/mc/m/n,\jsc@JYn/gt/m/n,%
4120   \jsc@JTn/mc/m/n,\jsc@JTn/gt/m/n}\do
4121 {\expandafter\let\csname\bxjs@tmpa/10\endcsname=\@undefined
4122   \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。

```
4123 \begingroup
4124 % 参照用フォント(\bxjs@sizereference)の全角空白の幅を取得
4125 \font\bxjs@tmpa=\bxjs@sizereference\space at 10pt
```



```

4126 \setbox\z@\hbox{\bxjs@tmpa\char\jis"2121\relax}
4127 % 幅が丁度10ptなら補正は不要
4128 \ifdim\wd\z@=10pt
4129 \global\let\bxjs@scale\jsScale
4130 \else
4131 % (10*s)/(10*f)として計算、\bxjs@invscaleはBXJSで定義
4132 \edef\bxjs@tmpa{\strip@pt\wd\z@}
4133 \@tempdima=10pt \@tempdima=\jsScale\@tempdima
4134 \bxjs@invscale\@tempdima\bxjs@tmpa
4135 \xdef\bxjs@scale{\strip@pt\@tempdima}
4136 \fi
4137 \endgroup
4138 %\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。

```
4139 \bxjs@declarefontshape
```

フォント代替の明示的定義。

```

4140 \DeclareFontShape{\jsc@JYn}{mc}{m}{it}{<->ssub*mc/m/n}{-}
4141 \DeclareFontShape{\jsc@JYn}{mc}{m}{sl}{<->ssub*mc/m/n}{-}
4142 \DeclareFontShape{\jsc@JYn}{mc}{m}{sc}{<->ssub*mc/m/n}{-}
4143 \DeclareFontShape{\jsc@JYn}{gt}{m}{it}{<->ssub*gt/m/n}{-}
4144 \DeclareFontShape{\jsc@JYn}{gt}{m}{sl}{<->ssub*gt/m/n}{-}
4145 \DeclareFontShape{\jsc@JYn}{mc}{bx}{it}{<->ssub*gt/m/n}{-}
4146 \DeclareFontShape{\jsc@JYn}{mc}{bx}{sl}{<->ssub*gt/m/n}{-}
4147 \DeclareFontShape{\jsc@JYn}{gt}{bx}{it}{<->ssub*gt/m/n}{-}
4148 \DeclareFontShape{\jsc@JYn}{gt}{bx}{sl}{<->ssub*gt/m/n}{-}
4149 \DeclareFontShape{\jsc@JYn}{mc}{b}{n}{<->ssub*mc/bx/n}{-}
4150 \DeclareFontShape{\jsc@JYn}{mc}{b}{it}{<->ssub*mc/bx/n}{-}
4151 \DeclareFontShape{\jsc@JYn}{mc}{b}{sl}{<->ssub*mc/bx/n}{-}
4152 \DeclareFontShape{\jsc@JYn}{gt}{b}{n}{<->ssub*gt/bx/n}{-}
4153 \DeclareFontShape{\jsc@JYn}{gt}{b}{it}{<->ssub*gt/bx/n}{-}
4154 \DeclareFontShape{\jsc@JYn}{gt}{b}{sl}{<->ssub*gt/bx/n}{-}
4155 \DeclareFontShape{\jsc@JTn}{mc}{m}{it}{<->ssub*mc/m/n}{-}
4156 \DeclareFontShape{\jsc@JTn}{mc}{m}{sl}{<->ssub*mc/m/n}{-}
4157 \DeclareFontShape{\jsc@JTn}{mc}{m}{sc}{<->ssub*mc/m/n}{-}
4158 \DeclareFontShape{\jsc@JTn}{gt}{m}{it}{<->ssub*gt/m/n}{-}
4159 \DeclareFontShape{\jsc@JTn}{gt}{m}{sl}{<->ssub*gt/m/n}{-}
4160 \DeclareFontShape{\jsc@JTn}{mc}{bx}{it}{<->ssub*gt/m/n}{-}
4161 \DeclareFontShape{\jsc@JTn}{mc}{bx}{sl}{<->ssub*gt/m/n}{-}
4162 \DeclareFontShape{\jsc@JTn}{gt}{bx}{it}{<->ssub*gt/m/n}{-}
4163 \DeclareFontShape{\jsc@JTn}{gt}{bx}{sl}{<->ssub*gt/m/n}{-}
4164 \DeclareFontShape{\jsc@JTn}{mc}{b}{n}{<->ssub*mc/bx/n}{-}
4165 \DeclareFontShape{\jsc@JTn}{mc}{b}{it}{<->ssub*mc/bx/n}{-}
4166 \DeclareFontShape{\jsc@JTn}{mc}{b}{sl}{<->ssub*mc/bx/n}{-}
4167 \DeclareFontShape{\jsc@JTn}{gt}{b}{n}{<->ssub*gt/bx/n}{-}
4168 \DeclareFontShape{\jsc@JTn}{gt}{b}{it}{<->ssub*gt/bx/n}{-}
4169 \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。

```

4170 \@ifl@t@r\fmtversion{2020/10/01}
4171   {\jsc@needspace@tchfalse}{\jsc@needspace@tchtrue}
4172 \ifjsc@needspace@tch      % --- for 2020-02-02 or older BEGIN
4173 \ifx\@rmfamilyhook\@undefined % old
4174 \DeclareRobustCommand\rmfamily
4175   {\not@math@alphabet\rmfamily\mathrm
4176    \romanfamily\rmdefault\kanjifamily\mcdefault\selectfont}
4177 \DeclareRobustCommand\sffamily
4178   {\not@math@alphabet\sffamily\mathsf
4179    \romanfamily\sfdefault\kanjifamily\gtdefault\selectfont}
4180 \DeclareRobustCommand\ttfamily
4181   {\not@math@alphabet\ttfamily\mathtt
4182    \romanfamily\ttdefault\kanjifamily\gtdefault\selectfont}
4183 \g@addto@macro\bxjs@begin@document@hook{%
4184   \ifx\mweights@init\@undefined\else % mweights.sty is loaded
4185     % my definitions above should have been overwritten, recover it!
4186     % \selectfont is executed twice but I don't care about speed...
4187     \expandafter\g@addto@macro\csname rmfamily \endcsname
4188       {\kanjifamily\mcdefault\selectfont}%
4189     \expandafter\g@addto@macro\csname sffamily \endcsname
4190       {\kanjifamily\gtdefault\selectfont}%
4191     \expandafter\g@addto@macro\csname ttfamily \endcsname
4192       {\kanjifamily\gtdefault\selectfont}%
4193   \fi}
4194 \else % 2020-02-02
4195 \g@addto@macro\@rmfamilyhook
4196   {\prepare@family@series@update@kanji{mc}\mcdefault}
4197 \g@addto@macro\@sffamilyhook
4198   {\prepare@family@series@update@kanji{gt}\gtdefault}
4199 \g@addto@macro\@ttfamilyhook
4200   {\prepare@family@series@update@kanji{gt}\gtdefault}
4201 \fi
4202 \else % --- for 2020-02-02 or older END & for 2020-10-01 BEGIN
4203 \AddToHook{rmfamily}%
4204   {\prepare@family@series@update@kanji{mc}\mcdefault}
4205 \AddToHook{sffamily}%
4206   {\prepare@family@series@update@kanji{gt}\gtdefault}
4207 \AddToHook{ttfamily}%
4208   {\prepare@family@series@update@kanji{gt}\gtdefault}
4209 \fi % --- for 2020-10-01 END
4210 \ifx\DeclareFixJFMCJKTextFontCommand\@undefined
4211 \DeclareJaTextFontCommand{\textmc}{\mcfamily}
4212 \DeclareJaTextFontCommand{\textgt}{\gtfamily}
4213 \fi
4214 \bxjs@if@sf@default{%

```

```
4215 \renewcommand\kanjifamilydefault{\gtdefault}}
```

念のため。

```
4216 \selectfont
```

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

```
4217 \def\bxjs@parse@qh#1{\let\bxjs@tmpb\relax}
```

```
4218 \let\bxjs@parse@qh@a\@undefined
```

```
4219 \let\bxjs@parse@qh@b\@undefined
```

■パラメタの設定

```
4220 \prebreakpenalty\jis"2147=10000
```

```
4221 \postbreakpenalty\jis"2148=10000
```

```
4222 \prebreakpenalty\jis"2149=10000
```

```
4223 \inhibitxspcode`!=1
```

```
4224 \inhibitxspcode`-=2
```

```
4225 \xspcode`+=3
```

```
4226 \xspcode`\%=3
```

"80～"FF の範囲の `\xspcode` を 3 に変更。

```
4227 \@tempcnta="80 \@whilenum\@tempcnta<"100 \do{%
```

```
4228 \xspcode\@tempcnta=3\advance\@tempcnta\@ne}
```

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

```
4229 \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。

```
4230 \begingroup
```

```
4231 \catcode`\!=0
```

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

```
4232 \gdef\bxjs@ptex@dir{%
```

```
4233 !ifttdir t%
```

```
4234 !else!ifydir y%
```

```
4235 !else ?%
```

```
4236 !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。

```
4237 % 古い \@makefnmark の定義
```

```
4238 \bxjs@nclong\def\bxjs@tmpa{\hbox{%
```

```
4239 !ifydir \@textsuperscript{\normalfont\@thefnmark}}%
```

```
4240 !else\hbox{\yoko\@textsuperscript{\normalfont\@thefnmark}}!fi}}
```

```

4241 \ifx\@makefnmark\bxjs@tmpa
4242 \bxjs@nclong\gdef\@makefnmark{%
4243   !ifydir \hbox{}\hbox{\@textsuperscript{\normalfont\@thefnmark}}\hbox{}}%
4244   !else\hbox{\yoko\@textsuperscript{\normalfont\@thefnmark}}!fi}
4245 \fi

```

エスケープ文字の変更はここまで。

```

4246 \endgroup

```

■minijs パッケージのブロック やっておく。

```

4247 \@namedef{ver@minijs.sty}{ }

```

B.3 pdfTeX 用の処理

```

4248 \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。

```

4249 \let\bxjs@let@hchar@chr\bxjs@let@hchar@chr@ue

```

ムニャムニャ。

```

4250 \@onlypreamble\bxjs@cjk@loaded
4251 \def\bxjs@cjk@loaded{%
4252   \def\@footnotemark{%
4253     \leavevmode
4254     \ifhmode
4255       \edef\@x@sf{\the\spacefactor}%
4256       \ifdim\lastkern>\z@\ifdim\lastkern<5sp\relax
4257         \unkern\unkern
4258         \ifdim\lastskip>\z@ \unskip \fi
4259       \fi\fi
4260       \nobreak
4261     \fi
4262     \@makefnmark
4263     \ifhmode \spacefactor\@x@sf \fi
4264     \relax}%
4265   \let\bxjs@cjk@loaded\relax
4266 }
4267 \g@addto@macro\bxjs@begin@document@hook{%
4268   \@ifpackageloaded{CJK}{%
4269     \bxjs@cjk@loaded
4270   }{}%
4271 }

```

B.4 XeTeX 用の処理

```

4272 \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。

```

4273 \def\bxjs@let@hchar@chr#1{%
4274   \@tempcnta`#1\relax \divide\@tempcnta"800\relax

```

```

4275 \bxjs@cond\ifnum\@tempcnta=27 \fi{%
4276 \bxjs@let@hchar@chr@xe
4277 }{\bxjs@let@hchar@out\def{{#1}}}}
4278 \def\bxjs@let@hchar@chr@xe#1{%
4279 \lccode`0=`#1\relax
4280 \lowercase{\bxjs@let@hchar@out\def{{0}}}}

```

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

```

4281 \@onlypreamble\bxjs@do@precisetext
4282 \ifx\XeTeXgenerateactualtext\@undefined\else
4283 \def\bxjs@do@precisetext{%
4284 \XeTeXgenerateactualtext=\@ne}
4285 \fi

```

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

TODO:3.0 バージョン要件を見直して暫定措置を解除する。

```

4286 \@onlypreamble\bxjs@do@simplejasetup
4287 \def\bxjs@do@simplejasetup{%
4288 \@namedef{bxjs@zeroglue/0.0pt}{T}%
4289 \ifnum\XeTeXinterchartokenstate>\z@
4290 \else\expandafter\ifx\csname bxjs@zeroglue/\the\XeTeXlinebreakskip\endcsname\relax\else
4291 \jsSimpleJaSetup
4292 \ClassInfo\bxjs@clsname
4293 {'\string\jsSimpleJaSetup' is applied\@gobble}%
4294 \fi\fi}

```

`\jsSimpleJaSetup` 日本語出力用の超簡易的な設定。

```

4295 \newcommand*{\jsSimpleJaSetup}{%
4296 \XeTeXlinebreaklocale "ja"\relax
4297 \XeTeXlinebreakskip=0pt plus 1pt minus 0.1pt\relax
4298 \XeTeXlinebreakpenalty=0\relax}

```

B.5 後処理（エンジン共通）

```

4299 \fi\fi\fi

```

`simplejasetup` オプションの処理。

```

4300 \ifx\bxjs@do@simplejasetup\@undefined\else
4301 \g@addto@macro\bxjs@begin@document@hook{%
4302 \ifbxjs@simplejasetup
4303 \bxjs@do@simplejasetup
4304 \fi}
4305 \fi

```

`precisetext` オプションの処理。

```

4306 \ifbxjs@precisetext
4307 \ifx\bxjs@do@precisetext\@undefined
4308 \ClassWarning\bxjs@clsname
4309 {The current engine does not support the\MessageBreak

```

```

4310      'precise-text' option\@gobble}
4311  \else
4312    \bxjs@do@precisetext
4313  \fi
4314 \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。

```

4315 \g@addto@macro\bxjs@begin@document@hook{%
4316   \ifx\jsInhibitGlueAtParTop\@empty
4317     \def\bxjs@tmpa{\jsInhibitGlueAtParTop}%
4318     \ifx\everyparhook\bxjs@tmpa
4319       \let\everyparhook\@empty
4320     \fi
4321 \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。

```

4322 \ifnum\bxjs@everyparhook=\bxjs@everyparhook@@modern

```

まず `\everypar` を“乗っ取る” 処理を行う。

```

4323 \let\bxjs@everypar\everypar
4324 \newtoks\everypar
4325 \everypar\bxjs@everypar

```

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

```

4326 \bxjs@everypar{\the\expandafter\everypar\everyparhook}%
4327 \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。

```

4328 \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。

```

4329 \@onlypreamble\bxjs@adjust@fancyhdr
4330 \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。

```

4331 \def\bxjs@tmpa{\fancyplain{}{\sl\rightmark}\strut}%
4332 \def\bxjs@tmpb{\fancyplain{}{\rightmark}\strut}%
4333 \ifx\f@ncyselh\bxjs@tmpa \global\let\f@ncyselh\bxjs@tmpb \fi

```

```

4334 \ifx\@ncyerh\bxjs@tmpa \global\let\@ncyerh\bxjs@tmpb \fi
4335 \ifx\@ncyolh\bxjs@tmpa \global\let\@ncyolh\bxjs@tmpb \fi
4336 \ifx\@ncyorh\bxjs@tmpa \global\let\@ncyorh\bxjs@tmpb \fi
4337 \def\bxjs@tmpa{\fancyplain{}{\sl\leftmark}\strut}%
4338 \def\bxjs@tmpb{\fancyplain{}{\leftmark}\strut}%
4339 \ifx\@ncyelh\bxjs@tmpa \global\let\@ncyelh\bxjs@tmpb \fi
4340 \ifx\@ncyerh\bxjs@tmpa \global\let\@ncyerh\bxjs@tmpb \fi
4341 \ifx\@ncyolh\bxjs@tmpa \global\let\@ncyolh\bxjs@tmpb \fi
4342 \ifx\@ncyorh\bxjs@tmpa \global\let\@ncyorh\bxjs@tmpb \fi
4343 \def\bxjs@tmpa{\rm\thepage\strut}%
4344 \def\bxjs@tmpb{\thepage\strut}%
4345 \ifx\@ncyecf\bxjs@tmpa \global\let\@ncyecf\bxjs@tmpb \fi
4346 \ifx\@ncyocf\bxjs@tmpa \global\let\@ncyocf\bxjs@tmpb \fi

```

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

```

4347 \ifx\fullwidth\@undefined\else \ifdim\textwidth<\fullwidth
4348   \setlength{\@tempdima}{\fullwidth-\textwidth}%
4349   \edef\bxjs@tmpa{\noexpand\fancyhfoffset[EL,OR]{\the\@tempdima}%
4350   }\bxjs@tmpa
4351 \fi\fi
4352 \PackageInfo\bxjs{clsname
4353   {Patch to fancyhdr is applied\@gobble}}

```

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

```

4354 \def\bxjs@pagestyle@hook{%
4355   \@ifpackageloaded{fancyhdr}{%
4356     \bxjs@adjust@fancyhdr
4357     \global\let\bxjs@adjust@fancyhdr\relax
4358   }{}}

```

\pagestyle にフックを入れ込む。

```

4359 \let\bxjs@org@pagestyle\pagestyle
4360 \def\pagestyle{%
4361   \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。

```

4362 \g@addto@macro\bxjs@begin@document@hook{%
4363   \bxjs@pagestyle@hook
4364   \global\let\bxjs@pagestyle@hook\relax}

```

終わり。

```
4365 \fi
```

以上で終わり。

```
4366 %</minimal>
```

付録 C 和文ドライバ：standard 🐼

標準のドライバ。

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

C.1 準備

```
4367 %<*standard>
4368 %% このファイルは日本語文字を含みます
```

まず minimal ドライバを読み込む。

```
4369 \input{bxjsja-minimal.def}
```

`simplejasetup` は `standard` では無効になる。

```
4370 \bxjs@simplejasetupfalse
```

■環境検査

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

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

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

加えて、以下の要件を定める。

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

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

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

```
4371 \ifjsWithTeX
4372 \IfFileExists{etoolbox.sty}{%
```



```

4373 \RequirePackage{etoolbox}[2007/10/08]% v1.3
4374 }{}
4375 \fi

```

C.2 和文ドライバパラメタ

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

\ifbxjs@jp@jismmiv 2004JIS 字形を優先させるか。

```

4376 \newif\ifbxjs@jp@jismmiv

    jis2004 オプションの処理。
4377 \bxjs@cslet{bxjs@kv@jis2004@true}\bxjs@jp@jismmivtrue
4378 \bxjs@cslet{bxjs@kv@jis2004@false}\bxjs@jp@jismmivfalse
4379 \define@key{bxjsStd}{jis2004}[true]{%
4380 \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。

```

4381 \newif\ifbxjs@jp@units

    units オプションの処理。
4382 \let\bxjs@kv@units@true\bxjs@jp@unitstrue
4383 \let\bxjs@kv@units@false\bxjs@jp@unitsfalse
4384 \define@key{bxjsStd}{units}[true]{%
4385 \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。

```

4386 \let\bxjs@jp@font\@empty

    font オプションの処理。
※ 2.9 版より、複数回指定した場合には累積させる。
4387 \define@key{bxjsStd}{font}{%
4388 \edef\bxjs@jp@font{\bxjs@catopt\bxjs@jp@font{#1}}}

```

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

```

4389 \newif\ifbxjs@jp@strong@cmd \bxjs@jp@strong@cmdtrue

    strong-cmd オプションの処理。
4390 \let\bxjs@kv@strongcmd@true\bxjs@jp@strong@cmdtrue
4391 \let\bxjs@kv@strongcmd@false\bxjs@jp@strong@cmdfalse
4392 \define@key{bxjs}{strong-cmd}[true]{\bxjs@set@keyval{strongcmd}{#1}{}}

```

実際の japaram の値を適用する。

```

4393 \def\bxjs@next#1{\bxjs@safe@setkeys{bxjsStd}{#1}}
4394 \expandafter\bxjs@next\expandafter{\jsJaParam}

```

C.3 共通処理 (1)

```
4395 \let\jafontsize\jsJaFontSize
```

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

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

```
4396 \@onlypreamble\bxjs@apply@mmiv
4397 \def\bxjs@apply@mmiv{%
4398   \bxjs@add@class@option{jis2004}
4399 % \@ifpackagewith 判定への対策
4400   \PassOptionsToPackage{jis2004}{otf}
4401   \global\let\bxjs@apply@mmiv\relax}
4402 \ifbxjs@jp@jismmiv \bxjs@apply@mmiv \fi
```

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

```
4403 \if j\jsEngine
4404   \bxjs@jp@unitsfalse
4405 \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。

```
4406 \ifbxjs@jp@units
4407   \IfFileExists{bxcalc.sty}{%
4408     \RequirePackage{bxcalc}[2018/01/28]%v1.0a
4409     \ifx\usepTeXunits\@undefined
4410       \PackageWarningNoLine\bxjs@clsname
4411         {Cannot support pTeX units (zw etc.), since\MessageBreak
4412           the package 'bxcalc' is too old}%
4413       \bxjs@jp@unitsfalse
4414     \else \usepTeXunits
4415     \fi
4416   }{%else
4417     \PackageWarningNoLine\bxjs@clsname
4418       {Cannot support pTeX units (zw etc.), since\MessageBreak
4419         the package 'bxcalc' is unavailable}%
4420     \bxjs@jp@unitsfalse
4421   }
4422 \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。

```
4423 \ifbxjs@jp@units
4424 \def\bxjs@parse@qh#1{\let\bxjs@tmpb\relax}
4425 \let\bxjs@parse@qh@a\@undefined
4426 \let\bxjs@parse@qh@b\@undefined
4427 \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。

```

4428 \ifbxjs@jp@units
4429   \def\bxjs@let@lenexpr#1#2{%
4430     \edef#1{#2}%
4431     \expandafter\CUXParseExpr\expandafter#1\expandafter{#1}}
4432 \else
4433   \def\bxjs@let@lenexpr{\edef}
4434 \fi

```

■\strong 命令の補填

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

```

4435 \ifbxjs@jp@strong@cmd\jsAtEndOfClass{%
4436   \ifx\strong\@undefined\ifx\strongenv\@undefined
4437     \newcommand*{\strongenv}{\jsStrongText}%
4438     \DeclareTextFontCommand{\strong}{\jsStrongText}%
4439     \newcommand*{\strongfontdeclare}{\jsStrongDeclare}%
4440   \fi\fi
4441 }

```

■和文フォント指定の扱い `standard` 和文ドライバでは `\jsJaFont` の値を和文フォントの“プリセット”の指定として用いる。プリセットの値は、`TeX Live` の `kanji-config-updmap` コマンドで使う“ファミリ”と同じにすることを想定する。特別な値として、`auto` は `kanji-config-updmap` で現在指定されているファミリを表す。

`\bxjs@adjust@jafont` `\jsJaFont` に入っている和文フォント設定の値を“調整”して、その結果を `\bxjs@tmpa` に返す。`#1` が `f` の場合は“非埋込 (`noEmbed`)”の設定が禁止される。この禁止の場合も含め、何か異常がある場合は `\bxjs@tmpa` は空になる。

```

4442 \@onlypreamble\bxjs@adjust@jafont
4443 \def\bxjs@adjust@jafont#1{%
4444   \ifx\jsJaFont\bxjs@@auto
4445     \bxjs@get@kanjiEmbed
4446     \ifx\bxjs@jaEmbed\relax
4447       \let\bxjs@tmpa\@empty
4448     \else
4449       \let\bxjs@tmpa\bxjs@jaEmbed
4450       \ifx\bxjs@jaVariant\bxjs@@hziv
4451         \bxjs@apply@mmiv
4452       \fi
4453     \fi
4454   \else
4455     \let\bxjs@tmpa\jsJaFont
4456   \fi
4457   \if f#1\ifx\bxjs@tmpa\bxjs@@noEmbed
4458     \ClassWarningNoLine\bxjs@clsname
4459     {Option 'jafont=noEmbed' is ignored, because it is\MessageBreak
4460      not available on the current situation}%

```

```

4461 \let\bxjs@tmpa\@empty
4462 \fi\fi
4463 }
4464 \def\bxjs@@auto{auto}
4465 \def\bxjs@@noEmbed{noEmbed}
4466 \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。

```

4467 \let\bxjs@jaEmbed\relax
4468 \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。

```

4469 \@onlypreamble\bxjs@get@kanjiEmbed
4470 \def\bxjs@get@kanjiEmbed{%
4471 \begingroup\setbox\z@=\hbox{%
4472 \global\let\bxjs@tmpdo\@empty
4473 \def\bxjs@next##1##2##3{%
4474 \def##1####1##3 ####2\@nil####3\@nnil{%
4475 \ifx$####1$\gdef##2{####2}\fi}%
4476 \g@addto@macro\bxjs@tmpdo{%
4477 \expandafter##1\bxjs@tmpa\@nil##3 \@nil\@nnil}}}%
4478 \bxjs@next\bxjs@tmpdo@a\bxjs@g@tmpa{kanjiEmbed}%
4479 \bxjs@next\bxjs@tmpdo@b\bxjs@g@tmpa{jaEmbed}%
4480 \bxjs@next\bxjs@tmpdo@c\bxjs@g@tmpb{kanjiVariant}%
4481 \bxjs@next\bxjs@tmpdo@d\bxjs@g@tmpb{jaVariant}%
4482 %
4483 \global\let\bxjs@g@tmpa\relax
4484 \global\let\bxjs@g@tmpb\relax
4485 \endlinechar\m@ne
4486 \let\do\@makeother\dospecials
4487 \catcode32=10 \catcode12=10 %form-feed
4488 \let\bxjs@tmpa\@empty
4489 \openin\@inputcheck="|kpsewhich updmap.cfg"\relax
4490 \ifeof\@inputcheck\else
4491 \read\@inputcheck to\bxjs@tmpa
4492 \closein\@inputcheck
4493 \fi
4494 \ifx\bxjs@tmpa\@empty\else
4495 \openin\@inputcheck="\bxjs@tmpa"\relax
4496 \@tempwattrue
4497 \loop\if@tempswa
4498 \read\@inputcheck to\bxjs@tmpa
4499 \bxjs@tmpdo
4500 \ifeof\@inputcheck \@tempwafalse \fi
4501 \repeat
4502 \fi
4503 }\endgroup

```

```

4504 \let\bxjs@jaEmbed\bxjs@g@tmpa
4505 \let\bxjs@jaVariant\bxjs@g@tmpb
4506 }

```

`\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。

```

4507 \@onlypreamble\bxjs@resolve@jafont@paren
4508 \def\bxjs@resolve@jafont@paren#1{%
4509   \def\bxjs@tmpb{\let#1}%
4510   \expandafter\bxjs@resolve@jafont@paren@a#1\@nil()\@nil\@nnil#1}
4511 \@onlypreamble\bxjs@resolve@jafont@paren@a
4512 \def\bxjs@resolve@jafont@paren@a#1(#2)#3\@nil#4\@nnil#5{%
4513   \ifx\relax#4\relax \bxjs@tmpb#5%
4514   \else
4515     \edef\bxjs@tmpa{#1\bxjs@jafont@paren{#2}#3}%
4516     \bxjs@tmpb\bxjs@tmpa
4517   \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。

```

4518 \newcommand*\jachar[1]{%
4519   \begingroup

```

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

```

4520   \jsLetHeadChar\bxjs@tmpa{#1}%
4521   \ifx\bxjs@tmpa\relax
4522     \ClassWarningNoLine\bxjs@clsname
4523       {Illegal argument given to \string\jachar}%
4524   \else
4525     \expandafter\bxjs@jachar\expandafter{\bxjs@tmpa}%
4526   \fi
4527 \endgroup}

```

`\jsJaChar` を `\jachar` と等価にする。

```

4528 \let\jsJaChar\jachar

```

下請けの `\bxjs@jachar` の実装はエンジンにより異なる。

```

4529 \let\bxjs@jachar\@firstofone

```

■hyperref 対策 出力ページサイズに館する処理は `geometry` パッケージが行うので、`hyperref` 側の処理は無効にしておく。

```

4530 \PassOptionsToPackage{setpagesize=false}{hyperref}

```

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

```

4531 \@onlypreamble\bxjs@fix@hyperref@unicode
4532 \def\bxjs@fix@hyperref@unicode#1{%
4533   \PassOptionsToPackage{bxjs/hook=#1}{hyperref}%

```

```

4534 \@namedef{KV@Hyp@bxjs/hook}##1{%
4535   \KV@Hyp@unicode{##1}%
4536   \def\KV@Hyp@unicode####1{%
4537     \expandafter\ifx\csname if##1\expandafter\endcsname
4538     \csname if####1\endcsname\else
4539     \ClassWarningNoLine{bxjs@clsname
4540       {Blcoked hyperref option 'unicode=####1'}}%
4541     \fi
4542   }%
4543 }%
4544 }

```

`\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。

```

4545 \@onlypreamble\jsCheckHyperrefUnicode
4546 \let\jsCheckHyperrefUnicode\empty
4547 \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。

```

4548 \@onlypreamble\bxjs@check@hyperref@unicode
4549 \def\bxjs@check@hyperref@unicode#1{%
4550   \g@addto@macro\jsCheckHyperrefUnicode{%
4551     \@tempswafalse
4552     \begingroup
4553     \expandafter\ifx\csname ifHy@unicode\endcsname\relax
4554       \aftergroup\@tempwattrue \fi
4555     \expandafter\ifx\csname ifHy@unicode\expandafter\endcsname
4556       \csname if#1\endcsname
4557     \aftergroup\@tempwattrue \fi
4558   \endgroup
4559   \if@tempswa\else
4560     \ClassError{bxjs@clsname
4561       {The value of hyperref 'unicode' key is not suitable\MessageBreak
4562         for the present engine (must be #1)}}%
4563     {\@ehc}%
4564   \fi}}

```

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

```

4565 \@onlypreamble\bxjs@urgent@special
4566 \@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。

```

4567 \ifbxjs@old@hook@system
4568 \def\bxjs@urgent@special#1{%
4569   \AtBeginDvi{\special{#1}}%
4570   \g@addto@macro\bxjs@begin@document@hook{%
4571     \@ifpackageloaded{atbegshi}{%
4572       \begingroup

```

```

4573      \toks\z@{\special{#1}}%
4574      \toks\tw@\expandafter{\AtBegShi@HookFirst}%
4575      \xdef\AtBegShi@HookFirst{\the\toks@\the\toks\tw@}%
4576      \endgroup
4577  }{}%
4578  }%
4579 }

```

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。

```

4580 \else
4581   \def\bxjs@urgent@special#1{%
4582     \bxjs@urgent@special@a
4583     \AddToHook{shipout/firstpage}[pxjahyper/enc]{\special{#1}}
4584   \def\bxjs@urgent@special@a{%
4585     \DeclareHookRule{shipout/firstpage}{pxjahyper/enc}{<}{hyperref}%
4586     \global\let\bxjs@urgent@special@a\relax}
4587 \fi

```

■bm パッケージ対策 `\reDeclareMathAlphabet` を適用した数式英字フォント命令は通常の場合とは定義文の形が異なる。このため bm パッケージを利用して `\bm{\mathrm{A}}` を実行するとエラーが発生する。これを回避するため、「`\bm` の引数中では数式英字フォントの和文連動を無効にする」ことにする。

`\reDeclareMathAlphabet` の適用により例えば `\mathrm` は以下のように変更される。

- 適用前は `\mathrm` の一回展開は `\protect\[mathrm_]` である。
- 適用後は `\mathrm` の一回展開が `\protect\[mathrm_]` になる。
- `\[mathrm_]` の一回展開は以下になる。
`\DualLang@mathalph@bet{\RDMAorg@mathrm}{\RDMAorg@mathmc}`
- `\RDMAorg@mathrm` の一回展開は `\[mathrm_]` になる。

`\bxjs@patch@RDMA@for@bm` `\reDeclareMathAlphabet` の機能に対して bm パッケージ対策のパッチを当てる。

```

4588 \@onlypreamble\bxjs@patch@RDMA@for@bm
4589 \def\bxjs@patch@RDMA@for@bm{%
  実際に改変するのは \DualLang@mathalph@bet である。
4590   \let\bxjs@org@DualLang@mathalph@bet\DualLang@mathalph@bet
4591   \def\DualLang@mathalph@bet{%

```

`\bm` の引数の中（そこでは `\bm` が `\@firstofone` に等置されているのでこれを判定に利用する）では、`\DualLang@mathalph@bet` を `\@firstoftwo` の動作に変える。これにより、`\mathrm` の（`\protect` を無視する場合の）先頭完全展開形が、「適用」前のものと一致する。このため `\bm` は「適用」の影響を受けずに正常動作できる。

```

4592   \ifx\bm@\@firstofone \expandafter\@firstoftwo
4593   \else \expandafter\bxjs@org@DualLang@mathalph@bet
4594   \fi}%

```

4595 }

C.4 pTeX 用設定

4596 \if j\jsEngine

■共通命令の実装

```
4597 \newcommand*\setkanjiskip{\jsSetKanjiSkip}
4598 \newcommand*\getkanjiskip{\jsGetKanjiSkip}
4599 \def\jsApplyKanjiSkip#1{%
4600   \kanjiskip=#1\relax}
4601 \newcommand*\setxkanjiskip{\jsSetXKanjiSkip}
4602 \newcommand*\getxkanjiskip{\jsGetXKanjiSkip}
4603 \def\jsApplyXKanjiSkip#1{%
4604   \xkanjiskip=#1\relax}
```

\jaJaChar のサブマクロ。

```
4605 \def\bxjs@jachar#1{%
4606   \bxjs@jachar@a#1...\@nil}
4607 \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。

```
4608   \ifx.#2#1%
```

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

```
4609   \else\ifx.#3%
4610     \@tempcnta`#1 \multiply\@tempcnta64
4611     \advance\@tempcnta`#2 \advance\@tempcnta-"3080
4612     \bxjs@jachar@b
4613   \else\ifx.#4%
4614     \@tempcnta`#1 \multiply\@tempcnta64
4615     \advance\@tempcnta`#2 \multiply\@tempcnta64
4616     \advance\@tempcnta`#3 \advance\@tempcnta-"E2080
4617     \bxjs@jachar@b
4618   \else
4619     \@tempcnta`#1 \multiply\@tempcnta64
4620     \advance\@tempcnta`#2 \multiply\@tempcnta64
4621     \advance\@tempcnta`#3 \multiply\@tempcnta64
4622     \advance\@tempcnta`#4 \advance\@tempcnta-"3C82080
4623     \bxjs@jachar@b
4624   \fi\fi\fi}
```

符号値が \@tempcnta の和文文字を出力する処理。

```
4625 \ifjsWithupTeX
4626   \def\bxjs@jachar@b{\kchar\@tempcnta}
4627 \else
4628   \def\bxjs@jachar@b{%
4629     \ifx\bxUInt\@undefined\else
4630       \bxUInt{\@tempcnta}%
```



```
4631 \fi}
4632 \fi
```

和欧文間空白の命令 `\jathinspace` の実装。

```
4633 \ifbxjs@jaspace@cmd
4634 \def\jathinspace{\hskip\xkanjiskip}
4635 \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。

```
4636 \ifbxjs@jp@jismmiv
4637 \PassOptionsToPackage{prefer2004jis}{pxchfon}
4638 \PassOptionsToPackage{prefer2004jis}{pxbabel}
4639 \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。

```
4640 \let\bxjs@jafont@paren\@firstofone
4641 \let\bxjs@tmpa\jsJaFont
4642 \ifx\bxjs@tmpa\bxjs@@auto
4643 \let\bxjs@tmpa\@empty
4644 \else\ifx\bxjs@tmpa\bxjs@@noEmbed
4645 \def\bxjs@tmpa{noembed}
4646 \fi\fi
4647 \bxjs@resolve@jafont@paren\bxjs@tmpa
4648 \edef\bxjs@tmpa{\bxjs@catopt\bxjs@tmpa\bxjs@jp@font}
4649 \ifx\bxjs@tmpa\@empty\else
4650 \edef\bxjs@next{%
4651 \noexpand\RequirePackage[\bxjs@tmpa]{pxchfon}[2010/05/12]% v0.5
4652 }\bxjs@next
4653 \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……。)

```
4654 \begingroup
4655 \global\let\bxjs@g@tmpa\relax
4656 \catcode`\|=0 \catcode`\|=12
4657 |def|bxjs@tmpdo#1|@nil{%
4658 |bxjs@tmpdo@a#1|@nil\RequirePackage|@nnil}%
4659 |def|bxjs@tmpdo@a#1\RequirePackage#2|@nnil{%
4660 |ifx$#1$|bxjs@tmpdo@b#2|@nil keyval|@nnil |fi}%
```

```

4661 |catcode`\|=0 \catcode`\|=12
4662 \def\bxjs@tmpdo@b#1keyval#2\@nnil{%
4663   \ifx$#2$\else
4664     \xdef\bxjs@g@tmpa{%
4665       \noexpand\PassOptionsToPackage{scale=\jsScale}{otf}}%
4666   \fi}
4667 \@firstofone{%
4668   \catcode10=12 \endlinechar\m@ne
4669   \let\do\@makeoother \dospecials \catcode32=10
4670   \openin\@inputcheck=otf.sty\relax
4671   \@tempswatrue
4672   \loop\if@tempswa
4673     \ifeof\@inputcheck \@tempswafalse \fi
4674     \if@tempswa
4675       \read\@inputcheck to\bxjs@next
4676       \expandafter\bxjs@tmpdo\bxjs@next\@nil
4677     \fi
4678   \repeat
4679   \closein\@inputcheck
4680 \endgroup}
4681 \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。

```
4682 \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。

```
4683 \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。

```
4684 \bxjs@if@package@atleast{plautopatch}{2020/05/25}{% v0.9g
```

```
4685   \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。

```
4686 \let\bxjs@avail@hy@unicode=f
```

```

4687 \if \ifx t\bxjs@plautopatch@new T%
4688     \else\ifbxjs@old@hook@system F\else T\fi\fi T%
4689     \IfFileExists{pxjahyper-uni.def}{\let\bxjs@avail@hy@unicode=t}{%}
4690 \fi
4691 \if t\bxjs@avail@hy@unicode
4692     \ifjsWithupTeX

```

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

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

```

4693     \ifbxjs@old@hook@system\else
4694         \AddToHook{\bxjs@CGHN{package/hyperref/after}}{%
4695             \input{pxjahyper-uni.def}}
4696 \fi
4697 \else

```

必要条件が満たされていて、かつ p \LaTeX である場合の処理。pxjahyper が「p \LaTeX 上の unicode 対応をもつほど新しい版（1.3 版以降）」であるかを判定する方法はない。しかし、新しい \LaTeX システムで unicode を無効にするのは避けたいので、 \LaTeX カーネルが 2023/06/01 版以降である場合に pxjahyper も十分に新しいと推定することにする。すなわち「pxjahyper が読み込まれるはず」かつ「 \LaTeX がカーネルが新しい」かを判定する。

```

4698     \let\bxjs@avail@hy@unicode=f
4699     \ifx t\bxjs@plautopatch@new
4700         \bxjs@if@format@at@least{2023/06/01}{\let\bxjs@avail@hy@unicode=t}{%}
4701 \fi
4702 \fi
4703 \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。

```

4704 \if f\bxjs@avail@hy@unicode
4705     \PassOptionsToPackage{unicode=false}{hyperref}
4706 \fi
4707 \fi

```

tounicode special 命令を出力する。

```

4708 \if \ifx\bxjs@driver@given\bxjs@driver@@dvipdfmx T%
4709     \else\ifjsWithpTeXng T\else F\fi\fi T%
4710 \IfFileExists{pxjahyper-enc.sty}{\@tempwattrue}{\@tempwafalse}
4711 \if@tempwa
4712     \RequirePackage{pxjahyper-enc}[2020/10/05]%v0.6
4713     \ifbxjs@bigcode\else \suppressbigcode \fi
4714 \else
4715     \ifnum\jis"2121="A1A1 %euc
4716         \bxjs@urgent@special{pdf:tounicode EUC-UCS2}
4717     \else\ifnum\jis"2121="8140 %sjis
4718         \bxjs@urgent@special{pdf:tounicode 90ms-RKSJ-UCS2}

```

```

4719 \else\ifnum\jis"2121="3000 %uptex
4720 \ifbxjs@bigcode
4721 \bxjs@urgent@special{pdf:tounicode UTF8-UTF16}
4722 \PassOptionsToPackage{bigcode}{pxjahyper}
4723 \else
4724 \bxjs@urgent@special{pdf:tounicode UTF8-UCS2}
4725 \PassOptionsToPackage{nobigcode}{pxjahyper}
4726 \fi
4727 \fi\fi\fi
4728 \let\bxToUnicodeSpecialDone=t
4729 \fi
4730 \fi

```

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

```

4731 \ifx f\bxjs@enablejfam\else
4732 \@enablejfamtrue
4733 \fi

```

実際に和文用の数式ファミリーの設定を行う。

```

4734 \if@enablejfam
4735 \DeclareSymbolFont{mincho}{\jsc@JYn}{mc}{m}{n}
4736 \DeclareSymbolFontAlphabet{\mathmc}{mincho}
4737 \SetSymbolFont{mincho}{bold}{\jsc@JYn}{gt}{m}{n}
4738 \jfam\symmincho
4739 \DeclareMathAlphabet{\mathgt}{\jsc@JYn}{gt}{m}{n}
4740 \g@addto@macro\bxjs@begin@document@hook{%
4741 \ifx\reDeclareMathAlphabet\undefined\else

```

`bm` パッケージが読込済であればパッチを適用する。

```

4742 \@ifpackageloaded{bm}{\bxjs@patch@RDMA@for@bm}{}%
4743 \reDeclareMathAlphabet{\mathrm}{\@mathrm}{\@mathmc}%
4744 \reDeclareMathAlphabet{\mathbf}{\@mathbf}{\@mathgt}%
4745 \reDeclareMathAlphabet{\mathsf}{\@mathsf}{\@mathgt}%
4746 \fi}
4747 \fi

```

C.5 pdfTeX 用設定：CJK + bxcjkatype

```

4748 \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。

```

4749 \bxjs@adjust@jafont{f}
4750 \let\bxjs@jafont@paren\@firstofone

```

```

4751 \bxjs@resolve@jafont@paren\bxjs@tmpa
4752 \edef\bxjs@tmpa{\bxjs@catopt\bxjs@tmpa\bxjs@jp@font}
4753 \edef\bxjs@tmpa{\bxjs@catopt\bxjs@tmpa{whole}}
4754 \ifx\bxjs@jadriver\bxjs@@pandoc\else
4755   \edef\bxjs@tmpa{\bxjs@catopt\bxjs@tmpa{autotilde}}
4756 \fi
4757 \edef\bxjs@next{%
4758   \noexpand\RequirePackage[\bxjs@tmpa]{bxcjkatype}[2013/10/15]% v0.2c
4759 }\bxjs@next
4760 \bxjs@cjk@loaded

```

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

```

4761 \ifbxjs@hyperref@enc
4762   \PassOptionsToPackage{unicode}{hyperref}
4763 \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。

```

4764 \ifx\bxcjkatypeHyperrefPatchDone\@undefined
4765 \begingroup
4766   \CJK@input{UTF8.bdg}
4767 \endgroup
4768 \g@addto@macro\pdfstringdefPreHook{%
4769   \@nameuse{CJK@UTF8Binding}%
4770 }
4771 \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。

```

4772 \ifx\bxcjkatypeHyperrefPatchDone\@undefined
4773 \g@addto@macro\pdfstringdefPreHook{%
4774   \ifx~\bxjs@@CJKtilde
4775     \let\bxjs@org@LetUnexpandableSpace\HyPsd@LetUnexpandableSpace
4776     \let\HyPsd@LetUnexpandableSpace\bxjs@LetUnexpandableSpace
4777     \let~\@empty
4778   \fi
4779 }
4780 \def\bxjs@@CJKtilde{\CJKecglue\ignorespaces}
4781 \def\bxjs@@tildecmd{~}
4782 \def\bxjs@LetUnexpandableSpace#1{%
4783   \def\bxjs@tmpa{#1}\ifx\bxjs@tmpa\bxjs@@tildecmd\else
4784     \bxjs@org@LetUnexpandableSpace#1%
4785   \fi}
4786 \fi

```

■共通命令の実装

```

4787 \newskip\jsKanjiSkip
4788 \newskip\jsXKanjiSkip
4789 \ifx\CJKecglue\@undefined
4790   \def\CJKtilde{\CJK@global\def~{\CJKecglue\ignorespaces}}
4791 \fi
4792 \newcommand*{\setkanjiskip}{\jsSetKanjiSkip}
4793 \newcommand*{\getkanjiskip}{\jsGetKanjiSkip}
4794 \newcommand*{\autospaceing}{\jsEnableKanjiSkip}
4795 \newcommand*{\noautospaceing}{\jsDisableKanjiSkip}
4796 \protected\def\bxjs@CJKglue{\hskip\jsKanjiSkip}
4797 \def\jsApplyKanjiSkip#1{%
4798   \jsKanjiSkip=#1\relax
4799   \let\CJKglue\bxjs@CJKglue}
4800 \newcommand*{\setxkanjiskip}{\jsSetXKanjiSkip}
4801 \newcommand*{\getxkanjiskip}{\jsGetXKanjiSkip}
4802 \newcommand*{\autoxspaceing}{\jsEnableXKanjiSkip}
4803 \newcommand*{\noautoxspaceing}{\jsDisableXKanjiSkip}
4804 \protected\def\bxjs@CJKecglue{\hskip\jsXKanjiSkip}
4805 \def\jsApplyXKanjiSkip#1{%
4806   \jsXKanjiSkip=#1\relax
4807   \let\CJKecglue\bxjs@CJKecglue}

\jachar のサブマクロの実装。
4808 \def\bxjs@jachar#1{%
4809   \CJKforced{#1}}

和欧文間空白の命令 \jathinspace の実装。
4810 \ifbxjs@jaspace@cmd
4811   \protected\def\jathinspace{\CJKecglue}
4812 \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。

```

4813 \ifx t\bxjs@enablejfam
4814   \ClassWarningNoLine\bxjs@clsname
4815     {You cannot use 'enablejfam=true', since the\MessageBreak
4816       CJK package does not support Japanese math}
4817 \fi

```

C.6 Xe_{La}TeX 用設定：xeCJK + zxjatype

```

4818 \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。

```

4819 \RequirePackage{zxjatype}
4820 \PassOptionsToPackage{no-math}{fontspec}%!
4821 \PassOptionsToPackage{xetex}{graphicx}%!
4822 \PassOptionsToPackage{xetex}{graphics}%!
4823 \ifx\zxJaFamilyName\@undefined

```

```

4824 \ClassError\bxjs@clsname
4825 {xeCJK or zxjatype is too old}\@ehc
4826 \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。

```

4827 \bxjs@adjust@jafont{f}
4828 \let\bxjs@jafont@paren\@gobble
4829 \bxjs@resolve@jafont@paren\bxjs@tmpa
4830 \edef\bxjs@tmpa{\bxjs@catopt\bxjs@tmpa\bxjs@jp@font}
4831 \ifx\bxjs@tmpa\@empty
4832 \setCJKmainfont[BoldFont=HaranoAjiGothic-Medium.otf]{HaranoAjiMincho-
Regular.otf}
4833 \setCJKsansfont[BoldFont=HaranoAjiGothic-Medium.otf]{HaranoAjiGothic-
Medium.otf}
4834 \else
4835 \edef\bxjs@next{%
4836 \noexpand\RequirePackage[\bxjs@tmpa]{zxjafont}[2013/01/28]% v0.2a
4837 }\bxjs@next
4838 \fi

```

■hyperref 対策 unicode オプションの指定に関する話。

X_ƎTeX 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_ƎTeX の場合にもパッケージ側で文字コード変換を行う方が望ましいと考えている。実際、unicode を無効にしていると警告が出て強制的に有効化される。一方で、過去 (r35125 まで) の xdvipdfmx では、文字列を UTF-16 に変換した状態で与えるのは不正と見なしていて警告が発生する。

これを踏まえて、ここでは、「X_ƎTeX のバージョンが 0.99992 以上の場合に unicode を既定で有効にする」ことにする。

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

TODO:3.0 バージョン要件を見直して暫定措置を解除する。

```

4839 \ifx\strcmp\@undefined\else %未定義なら条件を満たさない
4840 \ifnum\strcmp{\the\XeTeXversion\XeTeXrevision}{0.99992}>\m@ne
4841 \ifbxjs@hyperref@enc
4842 \PassOptionsToPackage{unicode}{hyperref}
4843 \fi
4844 \fi
4845 \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。

```
4846 \AtEndOfClass{%
4847 \def\@gnewline #1{%
4848   \ifvmode \@nolnerr
4849   \else
4850     \unskip \reserved@e {\reserved@f#1}\nobreak \hfil \break \null
4851     \nobreak \hskip-1sp\hskip1sp\relax
4852     \ignorespaces
4853   \fi}
4854 }
```

■共通命令の実装

```
4855 \newskip\jsKanjiSkip
4856 \newskip\jsXKanjiSkip
4857 \ifx\CJKecglue\@undefined
4858   \def\CJKtilde{\CJK@global\def~{\CJKecglue\ignorespaces}}
4859 \fi
4860 \newcommand*{\setkanjiskip}{\jsSetKanjiSkip}
4861 \newcommand*{\getkanjiskip}{\jsGetKanjiSkip}
4862 \newcommand*{\autospacing}{\jsEnableKanjiSkip}
4863 \newcommand*{\noautospacing}{\jsDisableKanjiSkip}
4864 \protected\def\bxjs@CJKglue{\hskip\jsKanjiSkip}
4865 \def\jsApplyKanjiSkip#1{%
4866   \jsKanjiSkip=#1\relax
4867   \xeCJKsetup{CJKglue={\bxjs@CJKglue}}}
4868 \newcommand*{\setxkanjiskip}{\jsSetXKanjiSkip}
4869 \newcommand*{\getxkanjiskip}{\jsGetXKanjiSkip}
4870 \newcommand*{\autoxspacing}{\jsEnableXKanjiSkip}
4871 \newcommand*{\noautoxspacing}{\jsDisableXKanjiSkip}
4872 \protected\def\bxjs@CJKecglue{\hskip\jsXKanjiSkip}
4873 \def\jsApplyXKanjiSkip#1{%
4874   \jsXKanjiSkip=#1\relax
4875   \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。

```
4876 \ifx\mcfamily\@undefined
4877   \protected\def\mcfamily{\CJKfamily{\CJKrmdefault}}
4878   \protected\def\gtfamily{\CJKfamily{\CJKsfdefault}}
4879 \fi
```

\jachar のサブマクロの実装。

```
4880 \def\bxjs@jachar#1{%
4881   \xeCJKDeclareCharClass{CJK}{`#1}\relax
4882   #1}
```


`\jathinspace` の実装。

```

4883 \ifbxjs@jaspacespace@cmd
4884 \protected\def\jathinspace{\CJKecglue}
4885 \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。

```

4886 \ifx t\bxjs@enablejfam
4887 \@enablejfamtrue
4888 \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。

```

4889 \if@enablejfam
4890 \xeCJKsetup{CJKmath=true}
4891 \fi

```

C.7 LuaTeX 用設定：LuaTeX-ja

```

4892 \else\if 1\jsEngine

```

■LuaTeX-ja パッケージの読み込み `luatexja` とともに `luatexja-fontspec` パッケージを読み込む。

`luatexja` は自前の `\zw` (これは実際の現在和文フォントに基づく値を返す) を定義するので、`\zw` の定義を消しておく。なお、レイアウト定義の「全角幅」は「規定」に基づく `\jsZw` であることに注意が必要。

※ 1.0b 版から「graphics パッケージに `pdftex` オプションを渡す」処理を行っていたが、1.4 版で廃止された。

```

4893 \let\zw\@undefined
4894 \RequirePackage{luatexja}
4895 \edef\bxjs@next{%
4896 \noexpand\RequirePackage[scale=\jsScale]{luatexja-fontspec}[2015/08/26]%
4897 }\bxjs@next

```

`\set@fontsize` へのパッチ適用を再度行う。

```

4898 \bxjs@patch@set@fontsize

```

フォント代替の明示的定義。

```

4899 \DeclareFontShape{JY3}{mc}{m}{it}{<->ssub*mc/m/n}{f}
4900 \DeclareFontShape{JY3}{mc}{m}{sl}{<->ssub*mc/m/n}{f}
4901 \DeclareFontShape{JY3}{mc}{m}{sc}{<->ssub*mc/m/n}{f}
4902 \DeclareFontShape{JY3}{gt}{m}{it}{<->ssub*gt/m/n}{f}
4903 \DeclareFontShape{JY3}{gt}{m}{sl}{<->ssub*gt/m/n}{f}
4904 \DeclareFontShape{JY3}{mc}{bx}{it}{<->ssub*gt/m/n}{f}
4905 \DeclareFontShape{JY3}{mc}{bx}{sl}{<->ssub*gt/m/n}{f}
4906 \DeclareFontShape{JY3}{gt}{bx}{it}{<->ssub*gt/m/n}{f}
4907 \DeclareFontShape{JY3}{gt}{bx}{sl}{<->ssub*gt/m/n}{f}

```

```

4908 \DeclareFontShape{JY3}{mc}{b}{n}{<->ssub*mc/bx/n}{}
4909 \DeclareFontShape{JY3}{mc}{b}{it}{<->ssub*mc/bx/n}{}
4910 \DeclareFontShape{JY3}{mc}{b}{sl}{<->ssub*mc/bx/n}{}
4911 \DeclareFontShape{JY3}{gt}{b}{n}{<->ssub*gt/bx/n}{}
4912 \DeclareFontShape{JY3}{gt}{b}{it}{<->ssub*gt/bx/n}{}
4913 \DeclareFontShape{JY3}{gt}{b}{sl}{<->ssub*gt/bx/n}{}
4914 \DeclareFontShape{JT3}{mc}{m}{it}{<->ssub*mc/m/n}{}
4915 \DeclareFontShape{JT3}{mc}{m}{sl}{<->ssub*mc/m/n}{}
4916 \DeclareFontShape{JT3}{mc}{m}{sc}{<->ssub*mc/m/n}{}
4917 \DeclareFontShape{JT3}{gt}{m}{it}{<->ssub*gt/m/n}{}
4918 \DeclareFontShape{JT3}{gt}{m}{sl}{<->ssub*gt/m/n}{}
4919 \DeclareFontShape{JT3}{mc}{bx}{it}{<->ssub*gt/m/n}{}
4920 \DeclareFontShape{JT3}{mc}{bx}{sl}{<->ssub*gt/m/n}{}
4921 \DeclareFontShape{JT3}{gt}{bx}{it}{<->ssub*gt/m/n}{}
4922 \DeclareFontShape{JT3}{gt}{bx}{sl}{<->ssub*gt/m/n}{}
4923 \DeclareFontShape{JT3}{mc}{b}{n}{<->ssub*mc/bx/n}{}
4924 \DeclareFontShape{JT3}{mc}{b}{it}{<->ssub*mc/bx/n}{}
4925 \DeclareFontShape{JT3}{mc}{b}{sl}{<->ssub*mc/bx/n}{}
4926 \DeclareFontShape{JT3}{gt}{b}{n}{<->ssub*gt/bx/n}{}
4927 \DeclareFontShape{JT3}{gt}{b}{it}{<->ssub*gt/bx/n}{}
4928 \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。

```

4929 \bxjs@adjust@jafont{t}
4930 \ifx\bxjs@tmpa\bxjs@noEmbed
4931   \def\bxjs@tmpa{noembed}
4932 \fi
4933 \let\bxjs@jafont@paren@gobble
4934 \bxjs@resolve@jafont@paren\bxjs@tmpa
4935 \edef\bxjs@tmpa{\bxjs@catopt\bxjs@tmpa\bxjs@jp@font}
4936 \ifx\bxjs@tmpa@empty
4937   \defaultjfontfeatures{ Kerning=Off }
4938   \setmainfont[BoldFont=HaranoAjiGothic-Medium.otf,JFM=ujis]{HaranoAjiMincho-
Regular.otf}
4939   \setsansfont[BoldFont=HaranoAjiGothic-Medium.otf,JFM=ujis]{HaranoAjiGothic-
Medium.otf}
4940 \else
4941   \edef\bxjs@next{%
4942     \noexpand\RequirePackage[\bxjs@tmpa]{luatexja-preset}%
4943   }\bxjs@next
4944 \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。

```

4945 \@ifpackagelater{luatexja}{2016/03/31}{}{\%else

```

```

4946 \DeclareRobustCommand\rmfamily
4947 {\not@math@alphabet\rmfamily\mathrm
4948 \romanfamily\rmdefault\kanjifamily\mcdefault\selectfont}
4949 \DeclareRobustCommand\sffamily
4950 {\not@math@alphabet\sffamily\mathsf
4951 \romanfamily\sfdefault\kanjifamily\gtdefault\selectfont}
4952 \DeclareRobustCommand\ttfamily
4953 {\not@math@alphabet\ttfamily\mathtt
4954 \romanfamily\ttdefault\kanjifamily\gtdefault\selectfont}
4955 }
4956 \bxjs@nclong\def\jttdefault{\gtdefault}
4957 \unless\ifx\@ltj@match@familytrue\@undefined
4958 \@ltj@match@familytrue
4959 \fi
4960 \g@addto@macro\bxjs@begin@document@hook{%
4961 \@ifpackageloaded{bm}{\bxjs@patch@RDMA@for@bm}{}%
4962 \reDeclareMathAlphabet{\mathrm}{\mathrm}{\mathmc}{%
4963 \reDeclareMathAlphabet{\mathbf}{\mathbf}{\mathgt}{%
4964 \reDeclareMathAlphabet{\mathsf}{\mathsf}{\mathgt}}}%
4965 \bxjs@if@sf@default{%
4966 \renewcommand\kanjifamilydefault{\gtdefault}}

```

■和文パラメタの設定

```

4967 % 次の3つは既定値の通り
4968 %\ltjsetparameter{prebreakpenalty={`',10000}}
4969 %\ltjsetparameter{postbreakpenalty={`“,10000}}
4970 %\ltjsetparameter{prebreakpenalty={`” ,10000}}
4971 \ltjsetparameter{jaxspmode={`!,1}}
4972 \ltjsetparameter{jaxspmode={`〒,2}}
4973 \ltjsetparameter{alxspmode={`+,3}}
4974 \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-jā で定義されているマクロで、段落中で段落冒頭用の処理を発動する。未定義である場合にに備えて同等のものを用意する。

```

4975 \ifx\ltjfakeparbegin\@undefined
4976 \protected\def\ltjfakeparbegin{%
4977 \ifhmode
4978 \relax\directlua{%
4979 luatexja.jfmglue.create_beginpar_node()}}
4980 \fi}
4981 \fi

```

`ltjs*` クラスの定義と同等になるようにパッチを当てる。

```

4982 \unless\ifnum\bxjs@everyparhook=\bxjs@everyparhook@@none
4983 \begingroup

```

```

4984 \let\% \@percentchar \def\@#1{[[\detokenize{#1}]]}
4985 \@gobble\if\def\bxjs@tmpa{\@{\everypar{}\fi}}
4986 \ifnum\bxjs@everyparhook=\bxjs@everyparhook@@compat
4987   \@gobble\if\def\bxjs@tmpa{\@{\everypar{\everyparhook}\fi}}\fi
4988 \directlua{
4989   local function patchcmd(cs, code, from, to)
4990     tex.sprint(code:gsub(from:gsub("%W", "%\\%\\%0"), "%0"..to)
4991       :gsub("macro:", \@gdef..cs, 1):gsub("->", "{", 1)..")
4992   end
4993   patchcmd(\@xsect, [[\meaning\xsect]],
4994     \@{\hskip-\@tempskipa}, \@{\ltjfakeparbegin})
4995   patchcmd(\@item, [[\meaning\@item]],
4996     \bxjs@tmpa, \@{\ltjfakeparbegin})
4997 \endgroup
4998 \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。

```

4999 \ifbxjs@hyperref@enc
5000   \PassOptionsToPackage{unicode}{hyperref}
5001   \bxjs@check@hyperref@unicode{true}
5002 \fi

```

■共通命令の実装

```

5003 \newcommand*{\setkanjiskip}{\jsSetKanjiSkip}
5004 \newcommand*{\getkanjiskip}{\jsGetKanjiSkip}
5005 \newcommand*{\setxkanjiskip}{\jsSetXKanjiSkip}
5006 \newcommand*{\getxkanjiskip}{\jsGetXKanjiSkip}
5007 \protected\def\autospacing{%
5008   \ltjsetparameter{autospacing=true}}
5009 \protected\def\noautospacing{%
5010   \ltjsetparameter{autospacing=false}}
5011 \protected\def\autoxspacing{%
5012   \ltjsetparameter{autoxspacing=true}}
5013 \protected\def\noautoxspacing{%
5014   \ltjsetparameter{autoxspacing=false}}
5015 \def\jsApplyKanjiSkip#1{%
5016   \ltjsetparameter{kanjiskip={#1}}}
5017 \def\jsApplyXKanjiSkip#1{%
5018   \ltjsetparameter{xkanjiskip={#1}}}

```

\jachar のサブマクロの実装。

```

5019 \def\bxjs@jachar#1{%
5020   \ltjjachar`#1\relax}

```

\jathinspace の実装。

```

5021 \ifbxjs@jaspace@cmd
5022   \protected\def\jathinspace{%
5023     \hskip\ltjgetparameter{xkanjiskip}\relax}

```

```
5024 \fi
```

■和文数式ファミリー LuaTeX-já では和文数式ファミリーは常に有効で、既にこの時点で必要な設定は済んでいる。従って @enablejfam は常に真になる。

```
5025 \ifx f\bxjs@enablejfam
5026   \ClassWarningNoLine\bxjs@clsname
5027   {You cannot use 'enablejfam=false', since the\MessageBreak
5028     LuaTeX-já always provides Japanese math families}
5029 \fi
```

C.8 共通処理 (2)

```
5030 \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 5031 \ifx\DeclareFixJFMCJKTextFontCommand\@undefined
5032   \DeclareJaTextFontCommand{\textmc}{\mcfamily}
5033   \DeclareJaTextFontCommand{\textgt}{\gtfamily}
5034 \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。

```
5035 \ifx\mathmc\@undefined
5036   \DeclareJaMathFontCommand{\mathmc}{\mcfamily}
5037   \DeclareJaMathFontCommand{\mathgt}{\gtfamily}
5038 \fi
```

以上で終わり。

```
5039 %</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。

```
5040 %<*modern>
5041 \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。

```
5042 \ifnum0\if x\jsEngine1\fi\if 1\jsEngine1\fi=\z@
5043 \def\encodingdefault{T1}%
5044 \input{t1enc.def}%
```

```
5045 \fontencoding\encodingdefault\selectfont
5046 \fi
```

基本フォントを Latin Modern フォントファミリに変更する。

※以下は `\usepackage[noamth]{lmodern}` と同じ。ユーザは後で `lmodern` を好きなオプションを付けて読み込むことができる。

```
5047 \ifnum0\if x\jsEngine1\fi\if 1\jsEngine1\fi=\z@
5048 \renewcommand{\rmdefault}{lrm}
5049 \renewcommand{\sfdefault}{lss}
5050 \renewcommand{\ttdefault}{lmtt}
5051 \fi
```

大型演算子用の数式フォントの設定。

※ `amsmath` パッケージと同等にする。

```
5052 \DeclareFontShape{OMX}{cmex}{m}{n}{%
5053   <-7.5>cmex7<7.5-8.5>cmex8%
5054   <8.5-9.5>cmex9<9.5->cmex10}{}%
5055 \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。
5056 \def\cmex@opt{10}
```

D.2 fixltx2e 読込

※ `fixltx2e` 廃止前の L^AT_EX カーネルの場合。

```
5057 \ifx\@IncludeInRelease\@undefined
5058 \RequirePackage{fixltx2e}
5059 \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。

```
5060 \RequirePackage{bxjscjkat}
```

D.4 完了

おしまい。

```
5061 %</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

5062 %<*pandoc>

xeCJK で space が有効になるのを阻止する。

※ bxjsja-standard.def の中で xeCJK が読み込まれるためこの位置に置いている。

5063 \if x\jsEngine

5064 \PassOptionsToPackage{nospace}{xeCJK}

5065 \fi

standard ドライバの設定を引き継ぐ。

5066 \input{bxjsja-standard.def}

■環境検査

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

pandoc 和文ドライバの処理系バージョン要件は standard と同じとする。加えて、以下の要件を定める。

- pTeX 系も含めて全てのエンジン種別で ϵ -TeX 拡張を要求する。
- 特に etoolbox の 2.0 版以上を要求する。
※もちろん他にも追加の依存パッケージがある。

■パッケージ読込 bxjspandoc パッケージを読み込む。

5067 \RequirePackage{bxjspandoc}

ϵ -TeX ではない場合に警告を出す。

5068 \ifjsWitheTeX\else

5069 \ClassWarningNoLine{bxjs@clsname

5070 {!!!!!!! WARNING !!!!!!!\MessageBreak

5071 This engine does not support ϵ -TeX extension!\MessageBreak

5072 Some feature might not work properly}

5073 \fi

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

5074 \newif\ifbxjs@bxghost@available

5075 \ifjsWitheTeX

5076 \RequirePackage{pdftexcmds}[2009/09/22]% v0.5

5077 \IfFileExists{bxghost.sty}{%

5078 \bxjs@bxghost@availabletrue

5079 \@namedef{bxjs@bgbv/79E70A0991967E27981832C84DB5DF99}{1}%v0.2.0

5080 \ifx\pdf@filemdfivesum\undefined\else

5081 \expandafter\ifx\cename{bxjs@bgbv}\pdf@filemdfivesum{bxghost.sty}%

5082 \endcsname\relax\else \bxjs@bxghost@availablefalse \fi

5083 \fi

5084 }{}

5085 \fi

その他の依存パッケージを読み込む。

```

5086 \RequirePackage{iftex}[2013/04/04]% v0.2
5087 \ifjsWitheTeX
5088 \RequirePackage{etoolbox}[2010/08/21]% v2.0
5089 \RequirePackage{filehook}[2011/10/12]% v0.5d
5090 \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。

```

5091 \newif\ifbxjs@jp@fix@strong \bxjs@jp@fix@strongtrue

fix-strong オプションの処理。

5092 \let\bxjs@kv@fixstrong@true\bxjs@jp@fix@strongtrue
5093 \let\bxjs@kv@fixstrong@false\bxjs@jp@fix@strongfalse
5094 \define@key{bxjsPan}{fix-strong}[true]{%
5095 \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。

```

5096 \newif\ifbxjs@jp@fix@code \bxjs@jp@fix@codetrue

fix-code オプションの処理。

5097 \let\bxjs@kv@fixcode@true\bxjs@jp@fix@codetrue
5098 \let\bxjs@kv@fixcode@false\bxjs@jp@fix@codefalse
5099 \define@key{bxjsPan}{fix-code}[true]{%
5100 \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。

```

5101 \chardef\bxjs@jp@strong=0

strong オプションの処理。

5102 \def\bxjs@kv@strong@bold{\chardef\bxjs@jp@strong=0 }
5103 \def\bxjs@kv@strong@sans{\chardef\bxjs@jp@strong=1 }
5104 \def\bxjs@kv@strong@boldsans{\chardef\bxjs@jp@strong=2 }
5105 \define@key{bxjsPan}{strong}{%
5106 \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 5107 \newif\ifbxjs@jp@or@indent \bxjs@jp@or@indenttrue
\ifbxjs@jp@or@block@heading 5108 \newif\ifbxjs@jp@or@secnumdepth \bxjs@jp@or@secnumdepthtrue
5109 \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。

```

5110 \define@key{bxjsPan}{_plus}[]{}
5111 \bxjs@jp@or@indentfalse

```



```

5112 \bxjs@jp@or@secnumdepthfalse
5113 \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。

```

5114 \let\bxjs@kv@orindent@true\bxjs@jp@or@indenttrue
5115 \let\bxjs@kv@orindent@false\bxjs@jp@or@indentfalse
5116 \define@key{bxjsPan}{or-indent}[true]{%
5117 \bxjs@set@keyval{orindent}{#1}{}}
5118 \let\bxjs@kv@orsecnumdepth@true\bxjs@jp@or@secnumdepthtrue
5119 \let\bxjs@kv@orsecnumdepth@false\bxjs@jp@or@secnumdepthfalse
5120 \define@key{bxjsPan}{or-secnumdepth}[true]{%
5121 \bxjs@set@keyval{orsecnumdepth}{#1}{}}
5122 \let\bxjs@kv@orblockheading@true\bxjs@jp@or@block@headingtrue
5123 \let\bxjs@kv@orblockheading@false\bxjs@jp@or@block@headingfalse
5124 \define@key{bxjsPan}{or-block-heading}[true]{%
5125 \bxjs@set@keyval{orblockheading}{#1}{}}

```

実際の japaram の値を適用する。

```

5126 \def\bxjs@next#1{\bxjs@safe@setkeys{bxjsPan}{#1}}
5127 \expandafter\bxjs@next\expandafter{\jsJaParam}

```

E.3 dupload システム

TODO: 新しいカーネルで利用可能な機構での代替を検討する。カーネルへのパッチは排除したいので。

パッケージが重複して読み込まれたときに “option clash” の検査をスキップする。この時に何らかのコードを実行させることができる。

`\bxjs@set@dupload@proc \bxjs@set@dupload@proc{〈ファイル名〉}{〈定義本体〉}`： 指定の名前の特定のファイルの読込が `\@filewithoptions` で指示されて、しかもそのファイルが読込済である場合に、オプション重複検査をスキップして、代わりに〈定義本体〉のコードを実行する。このコード中で #1 は渡されたオプション列のテキストに置換される。

```

5128 \@onlypreamble\bxjs@set@dupload@proc
5129 \def\bxjs@set@dupload@proc#1{%
5130 \expandafter\bxjs@set@dupload@proc@a\csname bxjs@dlp/#1\endcsname}
5131 \@onlypreamble\bxjs@set@dupload@proc@a
5132 \def\bxjs@set@dupload@proc@a#1{%
5133 \@onlypreamble#1\def#1#1}
5134 \def\bxjs@unset@dupload@proc#1{%
5135 \bxjs@cslet{bxjs@dlp/#1}\@undefined}

```

`\@ifoptions \@ifoptions` の再定義。

```

5136 \@onlypreamble\bxjs@org@ifoptions
5137 \let\bxjs@org@ifoptions\@ifoptions
5138 \@onlypreamble\bxjs@org@resetoptions
5139 \let\bxjs@org@resetoptions\relax

```

```

5140 \def\@if@options#1#2#3{%
5141   \let\bxjs@next\@secondoftwo
5142   \def\bxjs@tmpa{#1}\def\bxjs@tmpb{\@currentx}%
5143   \ifx\bxjs@tmpa\bxjs@tmpb
5144     \expandafter\ifx\csname bxjs@dlp/#2.#1\endcsname\relax\else
5145     \let\bxjs@next\@firstoftwo \fi
5146   \fi
5147   \bxjs@next\bxjs@do@dupload@proc\bxjs@org@if@options{#1}{#2}{#3}}
5148 \g@addto@macro\bxjs@begin@document@hook{%
5149   \let\@if@options\bxjs@org@if@options}
5150 \@onlypreamble\bxjs@do@dupload@proc
5151 \def\bxjs@do@dupload@proc#1#2#3{%
5152   \ifx\bxjs@org@reset@options\relax
5153     \let\bxjs@org@reset@options\@reset@options
5154   \fi
5155   \bxjs@csletcs{bxjs@next}{bxjs@dlp/#2.#1}%
5156   \def\@reset@options{%
5157     \let\@reset@options\bxjs@org@reset@options
5158     \@reset@options
5159     \bxjs@next{#3}}%
5160   \@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 \LaTeX 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 \XeTeX か \LuaTeX の場合に利用されていた。

`\bxjs@polyglossia@options` Polyglossia のオプション列のテキスト。“実際には読み込まれていない” 場合は `\relax` になる。

```
5161 \let\bxjs@polyglossia@options\relax
```

エンジンが \XeTeX か \LuaTeX の場合が対象になる。

※この場合 etoolbox が使用可能になっている。

```
5162 \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。

```
5163 \pandocSkipLoadPackage{polyglossia}
5164 \bxjs@set@dupload@proc{polyglossia.sty}{%
5165   \bxjs@unset@dupload@proc{polyglossia.sty}%
5166   \ClassWarning\bxjs@clsname
5167     {Package polyglossia is requested}%
5168   \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。

```
5169   \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。

```
5170     \ifboolexpr{test{\ifblank{##2}}or test{\ifstrequal{##2}{japanese}}}{%
5171       \ClassWarning\bxjs@clsname
5172         {Main language is 'japanese', thus fallback\MessageBreak
5173           definitions will be employed}%
5174       \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。

```
5175     }{%else
5176       \ClassWarning\bxjs@clsname
5177         {Main language is '##2',\MessageBreak
5178           thus polyglossia will be loaded}%
5179       \csundef{ver@polyglossia.sty}%
5180       \edef\bxjs@next{%
5181         \noexpand\RequirePackage[\bxjs@polyglossia@options]{polyglossia}[]%
5182       }\bxjs@next
5183       \setmainlanguage[##1]{##2}%
5184     }}
```

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。

```
5185 \g@addto@macro\bxjs@endpreamble@hook{%
5186   \ifx\bxjs@polyglossia@options\relax
5187     \csundef{ver@polyglossia.sty}%
5188   \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。

```
5189 \@onlypreamble\bxjs@pandoc@polyglossia@ja
5190 \def\bxjs@pandoc@polyglossia@ja{%
```

```

5191 \renewcommand*\setmainlanguage[2] [] {%
5192 \newcommand*\setotherlanguage[2] [] {%
5193   \ifblank{##2}{-}{%else
5194     \cslet{##2}\@empty \cslet{end##2}\@empty
5195     \cslet{text##2}\@firstofone}}%
5196 \newcommand*\setotherlanguages[2] [] {%
5197   \@for\bxjs@tmpa:={##2}\do{%
5198     \setotherlangauge{\bxjs@tmpa}}}%

```

Polyglossia の読込済マークは外れるようにしておく。

```

5199 \let\bxjs@polyglossia@options\relax}%

5200 \fi

```

■**Babel** について 現在の Pandoc では、テンプレートで用いられる多言語パッケージとしてエンジンの種別によらずに Babel が使われる。

※ X_YTEX では 2.15 版で、LuaTEX は 2.6 版で Polyglossia から Babel に変更されている。

`\bxjs@babel@options` Babel のオプション列のテキスト。“実際には読み込まれていない” 場合は `\relax` になる。

```

5201 \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。

```

5202 \pandocSkipLoadPackage{babel}
5203 \bxjs@set@dupload@proc{babel.sty}{%
5204   \bxjs@unset@dupload@proc{babel.sty}%
5205   \ClassWarning\bxjs@clsname
5206     {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。

5207 \@tempswafalse \let\bxjs@babel@options\@empty
5208 \def\bxjs@tmpb{main=}
5209 \@for\bxjs@tmpa:=#1\do{%
5210   \ifx\bxjs@tmpa\bxjs@tmpb \def\bxjs@tmpa{main=japanese}\fi
5211   \edef\bxjs@babel@options{\bxjs@babel@options,\bxjs@tmpa}}%
5212 \bxjs@cslet{ver@babel.sty}\@undefined
5213 \edef\bxjs@next{%
5214   \noexpand\RequirePackage[\bxjs@babel@options]{babel}\relax
5215 }\bxjs@next
5216 \RequirePackage{bxorigcapt}\relax}

```

プレアンプルで babel の読込が指示されなかった場合、読込済マークを外す。

```

5217 \g@addto@macro\bxjs@endpreamble@hook{%
5218   \ifx\bxjs@babel@options\relax
5219     \bxjs@cslet{ver@babel.sty}\@undefined
5220   \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。

```
5221 \ifjsWitheTeX
```

filehook の機能を用いて japanese.ldf の読込にフックを仕込む。

```
5222 \AtBeginOfFile{japanese.ldf}{\bxjs@begin@japanese@ldf@hook}
5223 \def\bxjs@begin@japanese@ldf@hook{%
5224   \let\bxjs@begin@japanese@ldf@hook\relax
5225   \let\bxjs@save@ProvidesLanguage\ProvidesLanguage
5226   \let\bxjs@save@LdfInit\LdfInit
5227   \def\ProvidesLanguage##1[##2]{\bxjs@do@japanese@ldf{##2}}%
5228   \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。

```
5229 \def\bxjs@do@japanese@ldf#1{\bxjs@do@japanese@ldf@a#1\@nil}
5230 \def\bxjs@do@japanese@ldf@a#1/#2/#3#4#5\@nil{%
5231   \let\LdfInit\bxjs@save@LdfInit
5232   \ClassInfo\bxjs@clsname
5233   {Release date of japanese.ldf is:\MessageBreak
5234     \@spaces #1/#2/#3#4\@gobble}%
5235   \ifnum#1#2#3#4<20201206 % v3.0
5236     \let\bxjs@japanese@ldf@skipped=t\csuse{endinginput}%
5237   \fi}
5238 \AtEndOfFile{japanese.ldf}{\bxjs@end@japanese@ldf@hook}
5239 \def\bxjs@end@japanese@ldf@hook{%
5240   \let\bxjs@end@japanese@ldf@hook\relax
5241   \let\ProvidesLanguage\bxjs@save@ProvidesLanguage
5242   \let\LdfInit\bxjs@save@LdfInit
5243   \ifx t\bxjs@japanese@ldf@skipped
5244     \ClassWarningNoLine\bxjs@clsname
5245     {Loading japanese.ldf is skipped}%
5246   \fi}
```

ダミーの言語定義。

```
5246   \ifundef\l@japanese{\chardef\l@japanese\z0}{}%
5247   \let\datejapanese\@empty\let\captionjapanese\@empty
5248   \let\extrajapanese\@empty\let\noextrajapanese\@empty
5249   \main@language{japanese}%
5250 \fi}
5251 \g@addto@macro\bxjs@begin@document@hook{%
5252   \let\bxjs@begin@japanese@ldf@hook\relax
5253   \let\bxjs@end@japanese@ldf@hook\relax}
5254 \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。

```
5255 \bxjs@set@dupload@proc{geometry.sty}{%
5256   \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。

```
5257 \if l\jsEngine
5258   \pandocSkipLoadPackage{xeCJK}
5259   \providecommand*\setCJKmainfont{\setmainfont}
5260 \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。

```
5261 \if x\jsEngine
5262   \expandafter\g@addto@macro\csname opt@xeCJK.sty\endcsname{%
5263     ,space}
5264 \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。

```
5265 \ifjsWithTeX
5266 \@onlypreamble\bxjs@info@or@ban
5267 \def\bxjs@info@or@ban#1{%
5268   \PackageInfo\bxjs@clsname
5269     {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。

```
5270 \unless\ifbxjs@jp@or@indent
5271   \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。

```
5272 \IfFileExists{parskip.sty}{%
5273   \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。

```
5274 }{%else
5275     \eappto\bxjs@endpreamble@hook{%
5276         \parindent=\the\parindent\relax
5277         \parskip=\the\parskip\relax}}
5278 \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。

```
5279 \ifbxjs@jp@or@secnumdepth\else
5280     \bxjs@info@or@ban{secnumdepth}
5281     \eappto\bxjs@endpreamble@hook{%
5282         \c@secnumdepth=\the\c@secnumdepth\relax}
5283 \fi
```

■**block-heading** について `\paragraph`、`\subparagraph` を別行見出しに変える処理を抑制する。

※ 2.7.1 版以前では別行見出し変更が既定で有効であった。

```
5284 \ifbxjs@jp@or@block@heading\else
5285     \let\bxjs@frozen@paragraph\paragraph
5286     \let\bxjs@frozen@subparagraph\subparagraph
5287     \bxjs@info@or@ban{block-heading}
5288     \appto\bxjs@endpreamble@hook{%
5289         \let\oldparagraph\@undefined
5290         \let\paragraph\bxjs@frozen@paragraph
5291         \let\subparagraph\bxjs@frozen@subparagraph}
5292 \fi
```

以上。

```
5293 \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。

```
5294 \g@addto@macro\bxjs@begin@document@hook{%
5295     \@tempwafalse
```

まず、マーク変更が必要かを調べる。`\oldparagraph` という制御綴が定義済の場合、Pandoc が `\paragraph` の様式を変更したということなので、マーク変更が必要である。

```
5296 \ifx\oldparagraph\undefined\else
5297   \@tempswatrue
5298 \fi
```

`\paragraph` が番号付きの場合は、マーク変更が必要である。

```
5299 \ifnum\c@secnumdepth>3
5300   \@tempswatrue
5301 \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。

```
5302 \if@tempswa\ifx\jsParagraphMark\bxjs@org@paragraph@mark
5303   \let\jsParagraphMark\@empty
5304 \fi\fi}
```

E.10 全角空白文字

L^AT_EX でない入力では、全角空きを入れるために全角空白文字 (U+3000) が使われる可能性があるので、全角空白文字を和文文字でなく空きとして扱うようにしておく。

※ (u)pL^AT_EX では対応できないので対象外。

`\pandocZWSpace` 全角空白文字の入力で実行されるコード。

```
5305 \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。

```
5306 \ifnum0\if x\jsEngine1\fi\if 1\jsEngine1\fi>\z@
5307   \catcode"3000=\active
5308   \begingroup \catcode`\!=7
5309   \protected\gdef!!!!3000{\pandocZWSpace}
5310   \endgroup
5311 \else\ifx\DeclareUnicodeCharacter\undefined\else
5312   \DeclareUnicodeCharacter{3000}{\bxjs@zsp@char}
5313   \bxjs@protected\def\bxjs@zsp@char{\pandocZWSpace}
5314 \fi\fi
```

E.11 hyperref 対策

`hyperref` の `unicode` オプションの固定を行う。

TODO: `unicode` オプションの固定処理は可能なら廃止したい。`hyperref` の開発状況を鑑みる限り、固定処理は危険なので。

```
5315 \if j\jsEngine
5316   \bxjs@fix@hyperref@unicode{false}
5317 \else
5318   \bxjs@fix@hyperref@unicode{true}
5319 \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 (うわぁ)。

```
5320 \ifbxjs@jp@fix@strong\ifbxjs@jp@strong@cmd
5321   \let\orgtextbf\textbf
5322   \DeclareRobustCommand\pandocTextbf[1]{%
5323     \begingroup
5324       \let\textbf\orgtextbf
5325       \strong{#1}%
5326     \endgroup}%
5327   \g@addto@macro\bxjs@begin@document@hook{%
5328     \let\textbf\pandocTextbf}
5329 \fi\fi
```

`\strong` の書体を設定する。

```
5330 \jsAtEndOfClass{%
5331   \ifx\strongfontdeclare\@undefined\else
5332     \ifcase\bxjs@jp@strong
5333     \or \strongfontdeclare{\sffamily}%
5334     \or \strongfontdeclare{\sffamily\bfseries}%
5335     \fi
5336   \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。

```
5337 \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。

```
5338 \ifbxjs@bxghost@available
5339   \RequirePackage[verb]{bxghost}[2020/01/31]% v0.3.0
5340   \let\bxjs@eghostguarded\eghostguarded
5341 \else
5342   \chardef\bxjs@eghost@c=23
5343   \ifx j\jsEngine \xspcode\bxjs@eghost@c=3
5344   \else\ifx l\jsEngine \ltjsetParameter{alxspmode={\bxjs@eghost@c,3}}
5345   \else\ifx x\jsEngine %no-op
5346   \else \let\bxjs@eghost@c\@undefined
5347   \fi\fi\fi
5348   \ifx\bxjs@eghost@c\@undefined\else
5349     \font\bxjs@eghost@f=ec-lmr10 at 1.23456pt
5350     \def\bxjs@pan@eghost{\bgroup\bxjs@eghost@f\bxjs@eghost@c\egroup}
5351     \def\bxjs@eghostguarded#1{%
5352       \bxjs@pan@eghost\null#1\null\bxjs@pan@eghost}
5353   \fi
```

```

5354 \fi
5355 \ifx\bxjs@eghostguarded\@undefined\else
5356 \let\orgtexttt\texttt
5357 \DeclareRobustCommand\pandocTexttt[1]{%
5358 \ifmmode \nfss@text{\ttfamily #1}%
5359 \else
5360 \ifvmode \leavevmode \fi
5361 \bxjs@eghostguarded{\begingroup\ttfamily#1\endgroup}%
5362 \fi}
5363 \g@addto@macro\bxjs@begin@document@hook{%
5364 \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。

```

5365 \ifbxjs@bxghost@available\else
5366 \expandafter\def\expandafter\verb\expandafter{%
5367 \expandafter\bxjs@pan@eghost\verb}
5368 \g@addto@macro\verb@egroup{\bxjs@pan@eghost}
5369 \fi
5370 \fi
5371 \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。

```

5372 \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。

```

```

5373 \def\bxjs@check@frontier{%
5374 \expandafter\bxjs@check@frontier@a\@currnamestack\noindent...\@nil}
5375 \def\bxjs@check@frontier@a#1#2#3#4#5\@nil#6{%
5376 \ifx\noindent#4#6\fi}

```

```

\bxjs@unforge@ifPDFTeX \ifPDFTeX を偽（正しい値）にする。

```

```

5377 \onlypreamble\bxjs@unforge@ifPDFTeX
5378 \def\bxjs@unforge@ifPDFTeX{\global\bxjs@csletcs@ifPDFTeX}{iffalse}}

\bxjs@forge@ifPDFTeX \ifPDFTeX を真（偽装した値）にする。

5379 \onlypreamble\bxjs@forge@ifPDFTeX
5380 \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。

5381 \def\bxjs@unload@forge@ifPDFTeX{%
5382   \bxjs@unforge@ifPDFTeX
5383   \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。

5384 \jsAtEndOfClass{\bxjs@forge@ifPDFTeX}
5385 \ifjsWitheTeX
5386   \AtBeginOfEveryFile{\bxjs@check@frontier\bxjs@unforge@ifPDFTeX}%
5387   \AtEndOfEveryFile{\bxjs@check@frontier\bxjs@forge@ifPDFTeX}%
5388   \g@addto@macro\bxjs@endpreamble@hook{\bxjs@unload@forge@ifPDFTeX}
5389 \else
5390   \g@addto@macro\bxjs@begin@document@hook{\bxjs@unload@forge@ifPDFTeX}
5391 \fi
5392 \fi

```

E.14 完了

おしまい。

```
5393 %</pandoc>
```

和文ドライバ実装はここまで。

```
5394 %</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。

```
5395 %<*anc>
```

付録 G 補助パッケージ：bxjscompat 🍡

古いやつをどうにかするためのムニャムニャ。

※すなわち BXJS クラスにおいては「新しいシステムにおいては bxjscompat がなくても正常に動作する」状態を保つべき。

G.1 準備

```
5396 %<*compat>
5397 \def\bxac@pkgname{bxjscompat}

\bxjx@engine エンジンの種別。

5398 \let\bxac@engine=n
5399 \def\bxac@do#1#2{%
5400   \edef\bxac@tmpa{\string#1}%
5401   \edef\bxac@tmpb{\meaning#1}%
5402   \ifx\bxac@tmpa\bxac@tmpb #2\fi}
5403 \bxac@do\kanjiskip{\let\bxac@engine=j}
5404 \bxac@do\XeTeXversion{\let\bxac@engine=x}
5405 \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。

```
5406 \ifx\jsAtEndOfClass\@undefined
5407   \let\bxac@delayed@if@bxjs\@firstofone
5408 \else \let\bxac@delayed@if@bxjs\jsAtEndOfClass
5409 \fi
```

`\ImposeOldLuaTeXBehavior` `\ImposeOldLuaTeXBehavior` は 0.85 版以降の Lua_{TeX} を一時的に pdf_{TeX} 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 Lua_{TeX} 以外の場合は何もしない。

```
5410 \newif\ifbxac@in@old@behavior
5411 \let\ImposeOldLuaTeXBehavior\relax
5412 \let\RevokeOldLuaTeXBehavior\relax
```

G.2 8bit 欧文 T_EX

```
5413 \ifx n\bxac@engine
```

和文を含むマクロ定義を通用させるため、高位バイトをアクティブ化しておく。

```
5414 \@tempcnta="80 \loop \ifnum\@tempcnta<"100
5415   \catcode\@tempcnta\active
5416   \advance\@tempcnta\@ne
5417 \repeat
```

以上。

```
5418 \fi
```

G.3 X₃T_EX

```
5419 \ifx x\bxac@engine
```

■文字クラスの設定 X₃T_EX 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、X_ƎT_EX 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。

```
5420 \ifx\XeTeXcharclass\undefined\else
5421 \bxac@delayed@if@bxjs{%
5422   \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。

```
5423   \ifx\XeTeXalloc@intercharclass\undefined\else
5424     \ifnum\XeTeXalloc@intercharclass=\z@
5425     \PackageInfo\bxac@pkgname
5426       {Setting up interchar class for CJK...\@gobble}%
5427     \InputIfFileExists{load-unicode-xetex-classes.tex}{%
5428       \XeTeXalloc@intercharclass=3
5429     }{\%else
5430       \PackageWarning\bxac@pkgname
5431         {Cannot find file 'load-unicode-xetex-classes.tex'%
5432         \@gobble}%
5433     }%
5434   \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。

```
5435   \ifnum\XeTeXcharclass"3041=\z@
5436     \PackageInfo\bxac@pkgname
5437       {Adjusting interchar class for CJK...\@gobble}%
5438     \@for\bxac@tmpb:={%
5439       3041,3043,3045,3047,3049,3063,3083,3085,3087,308E,%
5440       3095,3096,30A1,30A3,30A5,30A7,30A9,30C3,30E3,30E5,%
5441       30E7,30EE,30F5,30F6,30FC,31F0,31F1,31F2,31F3,31F4,%
5442       31F5,31F6,31F7,31F8,31F9,31FA,31FB,31FC,31FD,31FE,%
5443       31FF%
5444     }\do{\XeTeXcharclass"\bxac@tmpb=\@ne}%
5445   \fi
5446 }%
5447 }
5448 \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。

```
5449 \chardef\bxac@tmpb=11
5450 \def\bxac@do#1#2{%
5451   \@tempcnta=#1\relax
5452   \unless\ifnum\catcode\@tempcnta=\bxac@tmpb
5453     \chardef\bxac@tmpa=#2\relax
5454     \@whilenum{\@tempcnta<\bxac@tmpa}\do{%
5455       \catcode\@tempcnta\bxac@tmpb \advance\@tempcnta\@ne}%
5456   \fi}
```

```
5457 \bxac@do{"4E00"}{"9FCD}
```

以上。

```
5458 \fi
```

G.4 LuaTeX

```
5459 \ifx l\bxac@engine
```

0.82~0.84 版の LuaTeX を (0.81 版以前と同様に) 「pdfTeX の拡張である」 ように見せかける処理。

※恐らく必要な場面はなかったと思われるので、外しておく。

```
5460 %\unless\ifnum\luatexversion<80 \ifnum\luatexversion<85
```

```
5461 % \chardef\pdftexversion=200
```

```
5462 % \def\pdftexrevision{0}
```

```
5463 % \let\pdftexbanner\luatexbanner
```

```
5464 %\fi\fi
```

\ImposeOldLuaTeXBehavior 0.85 版以降であるかを検査する。

```
\RevokeOldLuaTeXBehavior 5465 \begingroup\expandafter\expandafter\expandafter\endgroup
```

```
5466 \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。

```
5467 \def\bxac@ob@list{%
```

```
5468 \do{\let}\pdfoutput{\outputmode}%
```

```
5469 \do{\let}\pdfpagewidth{\pagewidth}%
```

```
5470 \do{\let}\pdfpageheight{\pageheight}%
```

```
5471 \do{\protected\edef}\pdfhorigin{{\pdfvariable horigin}}%
```

```
5472 \do{\protected\edef}\pdfvorigin{{\pdfvariable vorigin}}%
```

```
5473 \def\bxac@ob@do#1#2{\begingroup
```

```
5474 \expandafter\bxac@ob@do@a\csname bxac@\string#2\endcsname{#1}#2}
```

```
5475 \def\bxac@ob@do@a#1#2#3#4{\endgroup
```

```
5476 \ifbxac@in@old@behavior \let#1#3\relax #2#3#4\relax
```

```
5477 \else \let#3#1\relax \let#1@\undefined
```

```
5478 \fi}
```

```
5479 \protected\def\ImposeOldLuaTeXBehavior{%
```

```
5480 \unless\ifbxac@in@old@behavior
```

```
5481 \bxac@in@old@behaviortrue
```

```
5482 \let\do\bxac@ob@do \bxac@ob@list
```

```
5483 \fi}
```

```
5484 \protected\def\RevokeOldLuaTeXBehavior{%
```

```
5485 \ifbxac@in@old@behavior
```

```
5486 \bxac@in@old@behaviorfalse
```

```
5487 \let\do\bxac@ob@do \bxac@ob@list
```

```
5488 \fi}
```

```
5489 \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。

```
5490 \directlua{
```

```
5491 local function range(cs, ce, cc, ff)
```

```

5492         if ff or not tex.getcatcode(cs) == cc then
5493             local setcc = tex.setcatcode
5494             for c = cs, ce do setcc(c, cc) end
5495         end
5496     end
5497     range(0x3400, 0x4DB5, 11, false)
5498     \ifnum\luatexversion>64
5499     range(0x4DB5, 0x4DBF, 11, true)
5500     range(0x4E00, 0x9FCC, 11, false)
5501     range(0x9FCD, 0x9FFF, 11, true)
5502     range(0xAC00, 0xD7A3, 11, false)
5503     range(0x20000, 0x2A6D6, 11, false)
5504     range(0x2A6D7, 0x2A6FF, 11, true)
5505     range(0x2A700, 0x2B734, 11, false)
5506     range(0x2B735, 0x2B73F, 11, true)
5507     range(0x2B740, 0x2B81D, 11, false)
5508     range(0x2B81E, 0x2B81F, 11, true)
5509     range(0x2B820, 0x2CEA1, 11, false)
5510     range(0x2CEA2, 0x2FFFD, 11, true)
5511     \fi
5512 }

```

以上。

```

5513 \fi

```

G.5 完了

おしまい。

```

5514 %</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

```

5515 %<*cjkcat>
5516 \def\bxjx@pkgname{bxjscjkat}
5517 \newcount\bxjx@cna
5518 \@onlypreamble\bxjx@tmpdo
5519 \@onlypreamble\bxjx@tmpdo@a
5520 \@onlypreamble\bxjx@tmpdo@b

```

\bxjx@engine エンジンの種別。

```

5521 \let\bxjx@engine=n
5522 \def\bxjx@tmpdo#1#2{%
5523     \edef\bxjx@tmpa{\string#1}%
5524     \edef\bxjx@tmpb{\meaning#1}%

```

```

5525 \ifx\bxjx@tmpa\bxjx@tmpb #2\fi}
5526 \bxjx@tmpdo\kanjiskip{\let\bxjx@engine=j}
5527 \bxjx@tmpdo\enablecjktoken{%
5528 \ifx\ucs\@undefined\else \ifnum\ucs"3000="3000
5529 \let\bxjx@engine=u\fi\fi}
5530 \bxjx@tmpdo\XeTeXversion{\let\bxjx@engine=x}
5531 \bxjx@tmpdo\pdfTeXversion{\let\bxjx@engine=p}
5532 \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。

```

5533 \def\bxjx@tmpdo#1#2{%
5534 \if#1\bxjx@engine
5535 \@ifpackageloaded{#2}{\fi}{%else
5536 \PackageError\bxjx@pkgname
5537 {Package '#2' must be loaded}%
5538 {Package loading is aborted.\MessageBreak\@ehc}%
5539 \endinput}
5540 \fi}
5541 \bxjx@tmpdo{p}{bxcjkatype}
5542 \bxjx@tmpdo{x}{xeCJK}
5543 \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。

```

5544 \ifx\TextOrMath\@undefined
5545 \RequirePackage{fixltx2e}
5546 \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 LuaL^AT_EX-ja と（ほぼ）等価なものに変更
 する。

※ LuaL^AT_EX-ja との相違点：A830、A960、1B000。

```

5547 \if u\bxjx@engine
5548 \@for\bxjx@tmpa:={%
5549 0080,0100,0180,0250,02B0,0300,0500,0530,0590,0600,%
5550 0700,0750,0780,07C0,0800,0840,0860,08A0,0900,0980,%
5551 0A00,0A80,0B00,0B80,0C00,0C80,0D00,0D80,0E00,0E80,%
5552 0F00,1000,10A0,1200,1380,13A0,1400,1680,16A0,1700,%
5553 1720,1740,1760,1780,1800,18B0,1900,1950,1980,19E0,%
5554 1A00,1A20,1AB0,1B00,1B80,1BC0,1C00,1C50,1C80,1CC0,%
5555 1CD0,1D00,1D80,1DC0,1E00,1F00,2440,27C0,27F0,2800,%
5556 2A00,2C00,2C60,2C80,2D00,2D30,2D80,2DE0,2E00,4DC0,%
5557 A4D0,A500,A640,A6A0,A700,A720,A800,A830,A840,A880,%
5558 A8E0,A900,A930,A980,A9E0,AA00,AA60,AA80,AAE0,AB00,%
5559 AB30,AB70,ABCO,DB80,DC00,E000,FB00,FB50,FE00,%
5560 FE70,FFF0,%
5561 10000,10080,10100,10140,10190,101D0,10280,102A0,%

```



```

5562 102E0,10300,10330,10350,10380,103A0,10400,10450,%
5563 10480,104B0,10500,10530,10600,10800,10840,10860,%
5564 10880,108E0,10900,10920,10980,109A0,10A00,10A60,%
5565 10A80,10AC0,10B00,10B40,10B60,10B80,10C00,10C80,%
5566 10E60,11000,11080,110D0,11100,11150,11180,111E0,%
5567 11200,11280,112B0,11300,11400,11480,11580,11600,%
5568 11660,11680,11700,118A0,11A00,11A50,11AC0,11C00,%
5569 11C70,11D00,12000,12400,12480,13000,14400,16800,%
5570 16A40,16AD0,16B00,16F00,1BC00,1BCA0,1D000,1D100,%
5571 1D200,1D300,1D360,1D400,1D800,1E000,1E800,1E900,%
5572 1EE00,1F000,1F030,1FOA0,1F300,1F600,1F650,1F680,%
5573 1F700,1F780,1F800,1F900,E0000,E0100,F0000,100000,%
5574 00C0%
5575 }\do{%
5576 \@tempcnta="\bxjx@tmpa\relax
5577 \@tempcntb\@tempcnta \advance\@tempcntb\m@ne
5578 \chardef\bxjx@tmpb\kcatcode\@tempcntb
5579 \kcatcode\@tempcnta=15 \kcatcode\@tempcntb\bxjx@tmpb}
5580 \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。

```

5581 \@onlypreamble\bxjx@grkcyr@list
5582 \def\bxjx@grkcyr@list{%
5583 \do{0391}{LGR}{\textAlpha}{A}%           % GR. C. L. ALPHA
5584 \do{0392}{LGR}{\textBeta}{B}%           % GR. C. L. BETA
5585 \do{0393}{LGR}{\textGamma}{\Gamma}%     % GR. C. L. GAMMA
5586 \do{0394}{LGR}{\textDelta}{\Delta}%     % GR. C. L. DELTA
5587 \do{0395}{LGR}{\textEpsilon}{E}%       % GR. C. L. EPSILON
5588 \do{0396}{LGR}{\textZeta}{Z}%           % GR. C. L. ZETA
5589 \do{0397}{LGR}{\textEta}{H}%            % GR. C. L. ETA
5590 \do{0398}{LGR}{\textTheta}{\Theta}%     % GR. C. L. THETA
5591 \do{0399}{LGR}{\textIota}{I}%           % GR. C. L. IOTA
5592 \do{039A}{LGR}{\textKappa}{K}%          % GR. C. L. KAPPA
5593 \do{039B}{LGR}{\textLambda}{\Lambda}%   % GR. C. L. LAMDA
5594 \do{039C}{LGR}{\textMu}{M}%            % GR. C. L. MU
5595 \do{039D}{LGR}{\textNu}{N}%            % GR. C. L. NU

```

5596 \do{039E}{LGR}{\textXi}{\Xi}%	% GR. C. L. XI
5597 \do{039F}{LGR}{\textOmicron}{O}%	% GR. C. L. OMICRON
5598 \do{03A0}{LGR}{\textPi}{\Pi}%	% GR. C. L. PI
5599 \do{03A1}{LGR}{\textRho}{\rho}%	% GR. C. L. RHO
5600 \do{03A3}{LGR}{\textSigma}{\Sigma}%	% GR. C. L. SIGMA
5601 \do{03A4}{LGR}{\textTau}{\tau}%	% GR. C. L. TAU
5602 \do{03A5}{LGR}{\textUpsilon}{\Upsilon}%	% GR. C. L. UPSILON
5603 \do{03A6}{LGR}{\textPhi}{\Phi}%	% GR. C. L. PHI
5604 \do{03A7}{LGR}{\textChi}{\chi}%	% GR. C. L. CHI
5605 \do{03A8}{LGR}{\textPsi}{\Psi}%	% GR. C. L. PSI
5606 \do{03A9}{LGR}{\textOmega}{\Omega}%	% GR. C. L. OMEGA
5607 \do{03B1}{LGR}{\textalpha}{\alpha}%	% GR. S. L. ALPHA
5608 \do{03B2}{LGR}{\textbeta}{\beta}%	% GR. S. L. BETA
5609 \do{03B3}{LGR}{\textgamma}{\gamma}%	% GR. S. L. GAMMA
5610 \do{03B4}{LGR}{\textdelta}{\delta}%	% GR. S. L. DELTA
5611 \do{03B5}{LGR}{\textepsilon}{\epsilon}%	% GR. S. L. EPSILON
5612 \do{03B6}{LGR}{\textzeta}{\zeta}%	% GR. S. L. ZETA
5613 \do{03B7}{LGR}{\texteta}{\eta}%	% GR. S. L. ETA
5614 \do{03B8}{LGR}{\texttheta}{\theta}%	% GR. S. L. THETA
5615 \do{03B9}{LGR}{\textiota}{\iota}%	% GR. S. L. IOTA
5616 \do{03BA}{LGR}{\textkappa}{\kappa}%	% GR. S. L. KAPPA
5617 \do{03BB}{LGR}{\textlambda}{\lambda}%	% GR. S. L. LAMDA
5618 \do{03BC}{LGR}{\textmu}{\mu}%	% GR. S. L. MU
5619 \do{03BD}{LGR}{\textnu}{\nu}%	% GR. S. L. NU
5620 \do{03BE}{LGR}{\textxi}{\xi}%	% GR. S. L. XI
5621 \do{03BF}{LGR}{\textomicron}{o}%	% GR. S. L. OMICRON
5622 \do{03C0}{LGR}{\textpi}{\pi}%	% GR. S. L. PI
5623 \do{03C1}{LGR}{\textrho}{\rho}%	% GR. S. L. RHO
5624 \do{03C2}{LGR}{\textvarsigma}{\varsigma}%	% GR. S. L. FINAL SIGMA
5625 \do{03C3}{LGR}{\textsigma}{\sigma}%	% GR. S. L. SIGMA
5626 \do{03C4}{LGR}{\texttau}{\tau}%	% GR. S. L. TAU
5627 \do{03C5}{LGR}{\textupsilon}{\upsilon}%	% GR. S. L. UPSILON
5628 \do{03C6}{LGR}{\textphi}{\phi}%	% GR. S. L. PHI
5629 \do{03C7}{LGR}{\textchi}{\chi}%	% GR. S. L. CHI
5630 \do{03C8}{LGR}{\textpsi}{\psi}%	% GR. S. L. PSI
5631 \do{03C9}{LGR}{\textomega}{\omega}%	% GR. S. L. OMEGA
5632 \do{0401}{T2A}{\CYRYO}{Y}%	% CY. C. L. IO
5633 \do{0410}{T2A}{\CYRA}{A}%	% CY. C. L. A
5634 \do{0411}{T2A}{\CYRB}{B}%	% CY. C. L. BE
5635 \do{0412}{T2A}{\CYRV}{V}%	% CY. C. L. VE
5636 \do{0413}{T2A}{\CYRG}{G}%	% CY. C. L. GHE
5637 \do{0414}{T2A}{\CYRD}{D}%	% CY. C. L. DE
5638 \do{0415}{T2A}{\CYRE}{E}%	% CY. C. L. IE
5639 \do{0416}{T2A}{\CYRZH}{ZH}%	% CY. C. L. ZHE
5640 \do{0417}{T2A}{\CYRZ}{Z}%	% CY. C. L. ZE
5641 \do{0418}{T2A}{\CYRI}{I}%	% CY. C. L. I
5642 \do{0419}{T2A}{\CYRISHRT}{SHRT}%	% CY. C. L. SHORT I
5643 \do{041A}{T2A}{\CYRK}{K}%	% CY. C. L. KA
5644 \do{041B}{T2A}{\CYRL}{L}%	% CY. C. L. EL

5645 \do{041C}{T2A}{\CYRM}{}%	% CY. C. L. EM
5646 \do{041D}{T2A}{\CYRN}{}%	% CY. C. L. EN
5647 \do{041E}{T2A}{\CYRO}{}%	% CY. C. L. O
5648 \do{041F}{T2A}{\CYRP}{}%	% CY. C. L. PE
5649 \do{0420}{T2A}{\CYRR}{}%	% CY. C. L. ER
5650 \do{0421}{T2A}{\CYRS}{}%	% CY. C. L. ES
5651 \do{0422}{T2A}{\CYRT}{}%	% CY. C. L. TE
5652 \do{0423}{T2A}{\CYRU}{}%	% CY. C. L. U
5653 \do{0424}{T2A}{\CYRF}{}%	% CY. C. L. EF
5654 \do{0425}{T2A}{\CYRH}{}%	% CY. C. L. HA
5655 \do{0426}{T2A}{\CYRC}{}%	% CY. C. L. TSE
5656 \do{0427}{T2A}{\CYRCH}{}%	% CY. C. L. CHE
5657 \do{0428}{T2A}{\CYRSH}{}%	% CY. C. L. SHA
5658 \do{0429}{T2A}{\CYRSHCH}{}%	% CY. C. L. SHCHA
5659 \do{042A}{T2A}{\CYRHRSN}{}%	% CY. C. L. HARD SIGN
5660 \do{042B}{T2A}{\CYRERY}{}%	% CY. C. L. YERU
5661 \do{042C}{T2A}{\CYRSFTSN}{}%	% CY. C. L. SOFT SIGN
5662 \do{042D}{T2A}{\CYREREV}{}%	% CY. C. L. E
5663 \do{042E}{T2A}{\CYRYU}{}%	% CY. C. L. YU
5664 \do{042F}{T2A}{\CYRYA}{}%	% CY. C. L. YA
5665 \do{0430}{T2A}{\cyra}{}%	% CY. S. L. A
5666 \do{0431}{T2A}{\cyrb}{}%	% CY. S. L. BE
5667 \do{0432}{T2A}{\cyrv}{}%	% CY. S. L. VE
5668 \do{0433}{T2A}{\cyrg}{}%	% CY. S. L. GHE
5669 \do{0434}{T2A}{\cyrd}{}%	% CY. S. L. DE
5670 \do{0435}{T2A}{\cyre}{}%	% CY. S. L. IE
5671 \do{0436}{T2A}{\cyrrzh}{}%	% CY. S. L. ZHE
5672 \do{0437}{T2A}{\cyrrz}{}%	% CY. S. L. ZE
5673 \do{0438}{T2A}{\cyri}{}%	% CY. S. L. I
5674 \do{0439}{T2A}{\cyrishrt}{}%	% CY. S. L. SHORT I
5675 \do{043A}{T2A}{\cyrk}{}%	% CY. S. L. KA
5676 \do{043B}{T2A}{\cyr1}{}%	% CY. S. L. EL
5677 \do{043C}{T2A}{\cyrm}{}%	% CY. S. L. EM
5678 \do{043D}{T2A}{\cyrn}{}%	% CY. S. L. EN
5679 \do{043E}{T2A}{\cyro}{}%	% CY. S. L. O
5680 \do{043F}{T2A}{\cyrp}{}%	% CY. S. L. PE
5681 \do{0440}{T2A}{\cyrr}{}%	% CY. S. L. ER
5682 \do{0441}{T2A}{\cyrsh}{}%	% CY. S. L. ES
5683 \do{0442}{T2A}{\cyrt}{}%	% CY. S. L. TE
5684 \do{0443}{T2A}{\cyru}{}%	% CY. S. L. U
5685 \do{0444}{T2A}{\cyrf}{}%	% CY. S. L. EF
5686 \do{0445}{T2A}{\cyrh}{}%	% CY. S. L. HA
5687 \do{0446}{T2A}{\cyrc}{}%	% CY. S. L. TSE
5688 \do{0447}{T2A}{\cyrch}{}%	% CY. S. L. CHE
5689 \do{0448}{T2A}{\cyrrsh}{}%	% CY. S. L. SHA
5690 \do{0449}{T2A}{\cyrrshch}{}%	% CY. S. L. SHCHA
5691 \do{044A}{T2A}{\cyrrhdsn}{}%	% CY. S. L. HARD SIGN
5692 \do{044B}{T2A}{\cyrrery}{}%	% CY. S. L. YERU
5693 \do{044C}{T2A}{\cyrrsftsn}{}%	% CY. S. L. SOFT SIGN

```

5694 \do{044D}{T2A}{\cyrrrev}{}%           % CY. S. L. E
5695 \do{044E}{T2A}{\cyryu}{}%             % CY. S. L. YU
5696 \do{044F}{T2A}{\cyrya}{}%             % CY. S. L. YA
5697 \do{0451}{T2A}{\cyryo}{}%             % CY. S. L. IO
5698 \do{00A7}{TS1}{\textsection}{\mathsection}% SECTION SYMBOL
5699 \do{00A8}{TS1}{\textasciidieresis}{}%   % DIAERESIS
5700 \do{00B0}{TS1}{\textdegree}{\mathdegree}% % DEGREE SIGN
5701 \do{00B1}{TS1}{\textpm}{\pm}%           % PLUS-MINUS SIGN
5702 \do{00B4}{TS1}{\textasciicute}{}%       % ACUTE ACCENT
5703 \do{00B6}{TS1}{\textparagraph}{\mathparagraph}% PILCROW SIGN
5704 \do{00D7}{TS1}{\texttimes}{\times}%      % MULTIPLICATION SIGN
5705 \do{00F7}{TS1}{\textdiv}{\div}%          % DIVISION SIGN
5706 }

```

`\mathdegree` 面倒なので補っておく。

```

5707 \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。

```

5708 \newif\ifbxjx@gcc@cjkl

```

`\greekasCJK` [公開命令]「特定 CJK 曖昧文字」を和文扱いにする。

```

5709 \newcommand*\greekasCJK{%
5710   \bxjx@gcc@cjkltrue}

```

`\nogreekasCJK` [公開命令]「特定 CJK 曖昧文字」を欧文扱いにする。

```

5711 \newcommand*\nogreekasCJK{%
5712   \bxjx@gcc@cjklfalse}

```

`\bxjx@fake@grkl` `\bxjx@fake@grkl{〈出力文字〉}{〈基準文字〉}`： ラテン文字で代用される数式ギリシャ文字の出力を行う。〈基準文字〉(`\mathchardef` の制御綴) の数式クラスと数式ファミリを引き継いで、〈出力文字〉(ASCII 文字トークン) の文字コードの数式文字を出力する。例えば、`\Pi` の意味が `\mathchar"7005` である場合、`\bxjx@fake@grkl{B}{\Pi}` は `\mathchar"7042` を実行する。

※フォントパッケージ使用時の再定義を考慮して、〈基準文字〉が `\mathchardef` であるかを検査し、そうでない場合はフォールバックとして単に 〈出力文字〉 を実行する。

```

5713 \def\bxjx@tmpdo#1\relax{%
5714   \def\bxjx@fake@grkl##1##2{%
5715     \expandafter\bxjx@fake@grkl@a\meaning##2#1\@nil{##1}{##2}}%
5716   \def\bxjx@fake@grkl@a##1#1##2\@nil##3##4{%
5717     \ifx\##1\%
5718       \bxjx@canta##4\divide\bxjx@canta\@cclvi
5719       \multiply\bxjx@canta\@cclvi \advance\bxjx@canta`##3\relax
5720       \mathchar\bxjx@canta
5721     \else ##3\fi}
5722 }\expandafter\bxjx@tmpdo\string\mathchar\relax

```

■pdfTeX・upTeX の場合

```

5723 \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。

```
5724 \ifpackageloaded{inputenc}{}{%else
5725   \RequirePackage[utf8]{inputenc}}
5726 \def\bxjx@tmpa{utf8}
5727 \ifx\bxjx@tmpa\inputencdoingname
5728   \PackageWarningNoLine\bxjx@pkgname
5729     {Input encoding changed to utf8}%
5730   \inputencoding{utf8}%
5731 \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。

```
5732 \if u\bxjx@engine
5733   \kcatcode"0370=15
5734   \kcatcode"0400=15
5735   \kcatcode"0500=15
5736 \fi
```

各文字について `\DeclareUnicodeCharacter` を実行する。

```
5737 \def\bxjx@tmpdo#1{%
5738   \@tempcnta="#1\relax
5739   \expandafter\bxjx@tmpdo@a\csname bxjx@KC/\the\@tempcnta\endcsname{#1}}
5740 \def\bxjx@tmpdo@a#1#2#3#4#5{%
```

引数 = `\[bxjx@KC/〈符号値〉]{〈符号値〉}{〈fontenc〉}{〈LICR〉}{〈数式 LICR〉}`

“数式中の動作”を決定する。〈数式 LICR〉が空（数式非対応）なら警告を出す。

```
5741   \ifx\#5\%
5742     \def\bxjx@tmpa{\@inmathwarn#4}%
```

〈数式 LICR〉が英字である場合は `\bxjx@fake@grk` で出力する。大文字なら `\Pi`、小文字なら `\pi` を基準文字にする。

```
5743   \else\ifcat A\noexpand#5%
5744     \edef\bxjx@tmpa{\noexpand\bxjx@fake@grk{#5}}%
5745     {\ifnum\uccode`#5=`#5\noexpand\Pi\else\noexpand\pi\fi}}%
```

それ以外は〈数式 LICR〉をそのまま実行する。

```
5746   \else \def\bxjx@tmpa{#5}%
5747   \fi\fi
5748   \def\bxjx@tmpb{\bxjx@tmpdo@b{#1}{#2}{#3}{#4}}%
5749   \expandafter\bxjx@tmpb\expandafter{\bxjx@tmpa}}
```

以降はエンジン種別で分岐する。 upTeX の場合。

```
5750 \if u\bxjx@engine
5751 \def\bxjx@tmpdo@b#1#2#3#4#5{%
```

引数 = $\backslash\text{bxjx@KC}/\langle\text{符号値}\rangle\{\langle\text{符号値}\rangle\}\{\langle\text{fontenc}\rangle\}\{\langle\text{LICR}\rangle\}\{\langle\text{数式中の動作}\rangle\}$

当該の Unicode 文字の動作は「テキストでは $\langle\text{LICR}\rangle$ 、数式では $\langle\text{数式中の動作}\rangle$ 」となる。
LICR は現在エンコーディングで有効な定義がある場合はそれが実行されるはずである。（つまり、現在が LGR である場合はギリシャ文字は常に欧文扱いになる。）それ以外の場合は LICR を $\backslash\text{bxjx@ja@or@not}$ に帰着させる。この際に、和文用の定義として当該の `kchardef` を使用し、その制御綴として $\backslash\text{bxjx@KC}/\dots$ を流用している。

```
5752 \kchardef#1=\@tempcnta
5753 \DeclareTextCommandDefault{#4}\backslash\text{bxjx@ja@or@not}{#1}{#3}{#4}}%
5754 \DeclareUnicodeCharacter{#2}\text{TextOrMath}{#4}{#5}}}
```

pdf \TeX の場合も処理はほとんど同じ。ただし、和文用の定義として $\backslash\text{UTF}\{\langle\text{符号値}\rangle\}$ を使う（ $\backslash\text{UTF}$ は `bxckjatype` の命令）。 $\backslash\text{bxjx@KC}/\dots$ は使わないが定義済にする必要がある。

```
5755 \else\if p\backslash\text{bxjx@engine
5756 \def\backslash\text{bxjx@tmpdo@b#1#2#3#4#5}{%
5757 \mathchardef#1=\@tempcnta
5758 \DeclareTextCommandDefault{#4}\backslash\text{bxjx@ja@or@not}\backslash\text{UTF}{#2}}{#3}{#4}}%
5759 \DeclareUnicodeCharacter{#2}\text{TextOrMath}{#4}{#5}}}
```

以上の処理を「特定 CJK 曖昧文字」の各々に適用する。

```
5761 \let\do\backslash\text{bxjx@tmpdo} \backslash\text{bxjx@grkcyr@list}
```

$\backslash\text{bxjx@DeclareUnicodeCharacter}$ $\backslash\text{bxjx@DeclareUnicodeCharacter}$ を改変して、「特定 CJK 曖昧文字」の場合に再定義を抑止したもの。

```
5762 \@onlypreamble\backslash\text{bxjx@org@DeclareUnicodeCharacter
5763 \let\backslash\text{bxjx@org@DeclareUnicodeCharacter}\backslash\text{DeclareUnicodeCharacter}
5764 \@onlypreamble\backslash\text{bxjx@DeclareUnicodeCharacter
5765 \def\backslash\text{bxjx@DeclareUnicodeCharacter#1#2}{%
5766 \count@=#1\relax
5767 \expandafter\ifx\csname \backslash\text{bxjx@KC}/\the\count@\endcsname\relax
5768 \backslash\text{bxjx@org@DeclareUnicodeCharacter}{#1}{#2}}%
5769 \else
5770 \wlog{\space\space skipped defining Unicode char U+#1}%
5771 \fi}
```

$\backslash\text{bxjx@ja@or@not}$ $\backslash\text{bxjx@ja@or@not}\{\langle\text{和文用定義}\rangle\}\{\langle\text{対象 fontenc}\rangle\}\{\langle\text{LICR}\rangle\}$: $\backslash\text{[no]greekasCJK}$ の状態に応じて和文または欧文で文字を出力する。

```
5772 \def\backslash\text{bxjx@ja@or@not#1#2#3}{%
```

$\backslash\text{greekasCJK}$ の場合は、無条件に $\langle\text{和文用定義}\rangle$ を実行する。

```
5773 \ifbxjx@gcc@CJK #1%
```

$\backslash\text{nogreekasCJK}$ の場合は、対象のエンコーディングに変更して LICR を実行するが、そのエンコーディングが未定義の場合は（フォールバックとして）和文用定義を使う。

```
5774 \else\expandafter\ifx\csname T@#2\endcsname\relax #1%
5775 \else \UseTextSymbol{#2}{#3}}%
5776 \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。

```
5777 \begingroup
5778 \toks@\expandafter{\DeclareFontEncoding@{#1}{#2}{#3}}
5779 \xdef\next{\def\noexpand\DeclareFontEncoding@##1##2##3{%
5780   \noexpand\bxjx@swap@DUC@cmd
5781   \the\toks@
5782   \noexpand\bxjx@swap@DUC@cmd}}
5783 \endgroup\next
5784 \def\bxjx@swap@DUC@cmd{%
5785   \let\bxjx@tmpa\DeclareUnicodeCharacter
5786   \let\DeclareUnicodeCharacter\bxjx@DeclareUnicodeCharacter
5787   \let\bxjx@DeclareUnicodeCharacter\bxjx@tmpa
5788   \let\bxjx@tmpa\relax}
```

以上。

■X_YTeX・LuaTeX の場合

```
5789 \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。

```
5790 \def\bxjx@tmpdo#1{%
5791   \bxjx@cmta="#1\relax
5792   \begingroup
5793     \lccode`~=\bxjx@cmta
5794     \lowercase{\endgroup
5795       \bxjx@tmpdo@a{~}}{#1}}
5796 \def\bxjx@tmpdo@a#1#2#3#4#5{%
5797   \ifx\\#5\\\let\bxjx@tmpa\relax
5798   \else\ifcat A\noexpand#5%
5799     \edef\bxjx@tmpa{\noexpand\bxjx@fake@grk{#5}%
5800       {\ifnum\uccode`#5=`#5\noexpand\Pi\else\noexpand\pi\fi}}%
5801   \else \def\bxjx@tmpa{#5}%
5802   \fi\fi
5803   \ifx\bxjx@tmpa\relax\else
5804     \mathcode\bxjx@cmta"8000 \let#1\bxjx@tmpa
5805   \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。

```
5806 \mathchardef\bxjx@tmpa="119
5807 \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。

```
5808 \if l\bxjx@engine
5809   \protected\def\greekasCJK{%
5810     \bxjx@gcc@cjctrue
5811     \ltjsetparameter{jacharrange={+2, +8}}
5812   \protected\def\nogreekasCJK{%
5813     \bxjx@gcc@cjcfalse
5814     \ltjsetparameter{jacharrange={-2, -8}}
5815 \fi
```

X_ƎTeX の場合、xeCJK は X_ƎTeX 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。

```
5816 \if x\bxjx@engine
5817   \let\bxjx@gcc@cjkl@list\@empty
5818   \def\do#1#2#3#4{%
5819     \edef\bxjx@gcc@cjkl@list{\bxjx@gcc@cjkl@list
5820       \noexpand\XeTeXcharclass"#1\bxjx@canta}}
5821   \bxjx@grkcyr@list
5822   \protected\def\greekasCJK{%
5823     \bxjx@gcc@cjctrue
5824     \bxjx@canta=\@ne \bxjx@gcc@cjkl@list}
5825   \protected\def\nogreekasCJK{%
5826     \bxjx@gcc@cjcfalse
5827     \bxjx@canta=\z@ \bxjx@gcc@cjkl@list}
5828 \fi
```

以上。

```
5829 \fi\fi
```

H.4 初期設定

「特定 CJK 曖昧文字」を欧文扱いにする。

```
5830 \nogreekasCJK
```

H.5 完了

おしまい。

```
5831 %</cjkcat>
```

付録 I 補助パッケージ：bxjspandoc 🐼

Pandoc の L^AT_EX 用標準テンプレートをより幸せに使うための設定。BXJS クラスの pandoc ドライバのコードの中の、“汎用的”に使える部分を切り出したもの。つまり現在の pandoc ドライバはこのパッケージを読みこむ。

※テンプレートの T_EX コードより前に読み込む必要があるため、専ら文書クラス内での読込に限られる。

1.1 準備

```
5832 %<*ancpandoc>
5833 %% このファイルは日本語文字を含みます.
5834 \def\bxjsp@pkgname{bxjspandoc}
```

\bxjsp@engine エンジンの種別。

```
5835 \let\bxjsp@engine=n
5836 \@onlypreamble\bxjsp@do
5837 \def\bxjsp@do#1#2{%
5838   \edef\bxjsp@tmpa{\string#1}%
5839   \edef\bxjsp@tmpb{\meaning#1}%
5840   \ifx\bxjsp@tmpa\bxjsp@tmpb #2\fi}
5841 \bxjsp@do\kanjiskip{\let\bxjsp@engine=j}
5842 \bxjsp@do\XeTeXversion{\let\bxjsp@engine=x}
5843 \bxjsp@do\pdftexversion{\let\bxjsp@engine=p}
5844 \bxjsp@do\luatexversion{\let\bxjsp@engine=l}
```

\bxjsp@begin@document@hook 文書本体開始時フック。

```
5845 \@onlypreamble\bxjsp@begin@document@hook
5846 \let\bxjsp@begin@document@hook\@empty
5847 \AtBeginDocument{\bxjsp@begin@document@hook}
```

\ifbxjsp@babel@used [スイッチ] Babel が読み込まれたか。

```
5848 \newif\ifbxjsp@babel@used
5849 \g@addto@macro\bxjsp@begin@document@hook{%
5850   \@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。

```
5851 \newif\ifbxjsp@english
5852 \DeclareOption{english}{\bxjsp@englishttrue}
```

オプション定義はおしまい。

```
5853 \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。

```
5854 \@onlypreamble\pandocSkipLoadFile
5855 \newcommand*\pandocSkipLoadFile[1]{%
```

```

5856 \expandafter\bxjsp@skip@load@file@a\csname ver@#1\endcsname{#1}}
5857 \def\bxjsp@skip@load@file@a#1#2{%
5858 \ifx#1\relax
5859 \def#1{2001/01/01}%
5860 \PackageInfo\bxjsp@pkgname
5861 {File '#2' marked as loaded\@gobble}%
5862 \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。

```

5863 \@onlypreamble\pandocSkipLoadPackage
5864 \newcommand*\pandocSkipLoadPackage[1]{%
5865 \pandocSkipLoadFile{#1.sty}}

```

1.4 fixltx2e パッケージ

テンプレートでは `fixltx2e` パッケージを読み込むが、最近（2015 年版以降）の `LaTeX` ではこれで警告が出る。これを抑止する。

`LaTeX` カーネルが新しい場合は `fixltx2e` を読込済にする。

```

5866 \ifx\@IncludeInRelease\@undefined\else
5867 \pandocSkipLoadPackage{fixltx2e}
5868 \fi

```

1.5 cmap パッケージ

エンジンが `(u)pLaTeX` のときに `cmap` パッケージが読み込まれるのを阻止する。（実際は警告が出るだけで無害であるが。）

```

5869 \if j\bxjsp@engine
5870 \pandocSkipLoadPackage{cmap}
5871 \fi

```

1.6 microtype パッケージ

警告が多すぎなので消す。

```

5872 \if j\bxjsp@engine \else
5873 \PassOptionsToPackage{verbose=silent}{microtype}
5874 \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。

```

5875 \if j\bxjsp@engine
5876 \pandocSkipLoadPackage{microtype}
5877 \newcommand*\UseMicrotypeSet[2][]{ }

```

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。

```
5879 \DeclareRobustCommand{\pandocLdots}{%
5880   \let\bxjsp@do\bxjsp@ja@ellipsis
5881   \ifmmode \let\bxjsp@do\bxjsp@org@ldots
5882   \else\ifbxjsp@babel@used
5883     \expandafter\ifx\csname bxjsp@ld/\language\endcsname\relax
5884       \let\bxjsp@do\bxjsp@org@ldots \fi
5885   \fi\fi \bxjsp@do}
5886 \@namedef{bxjsp@ld/japanese}{1}
5887 \def\bxjsp@ja@ellipsis{…}
5888 \let\bxjsp@org@ldots\ldots
```

`\ldots` の実装を `\pandocLdots` に置き換える。

```
5889 \g@addto@macro\bxjsp@begin@document@hook{%
5890   \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。

```
5891 \bxjs@nclong\def\bxjsp@tmpa{\ldots}%
5892 \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。

```
5893   \ifnum0\ifbxjsp@english\ifbxjsp@babel@used\else1\fi\fi=0
5894   \let\ldots\pandocLdots
5895   \fi
5896 \fi}
```

`\ldots` の直後の文字が非英字の場合、Pandoc は「`\ldots。`」のように空白を入れずに並べて出力する。「Pandoc は非英字と見なすが $\text{X}\text{\LaTeX}$ ・ $\text{Lua}\text{\LaTeX}$ 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。

```
5897 \chardef\bxjsp@cc@other=12
5898 \@onlypreamble\bxjsp@makeother@range
5899 \def\bxjsp@makeother@range#1#2{%
5900   \@tempcnta"#1\relax \@tempcntb"#2\relax
5901   \loop\ifnum\@tempcnta<\@tempcntb
5902     \catcode\@tempcnta\bxjsp@cc@other
5903     \advance\@tempcnta\@ne
5904   \repeat}
5905 \ifnum0\if x\bxjsp@engine1\fi\if 1\bxjsp@engine1\fi>0
5906   \catcode"1F23B=\bxjsp@cc@other
5907   \bxjsp@makeother@range{9FCD}{A000}
5908   \bxjsp@makeother@range{1B002}{1B170}
5909   \bxjsp@makeother@range{2B820}{2EBF0}
5910 \fi
```

1.8 PandoLa モジュール

インストール済であれば読み込む。

```
5911 \IfFileExists{bxpandola.sty}{%
5912   \RequirePackage{bxpandola}\relax
5913   \PackageInfo\bxjsp@pkgname
5914   {PandoLa module is loaded\@gobble}
5915 }{}
```

1.9 完了

おしまい。

```
5916 %</ancpandoc>
```

補助パッケージ実装はここまで。

```
5917 %</anc>
```