

# lltjp-geometry パッケージ

LuaTeX-ja プロジェクト\*

2020 年 8 月 8 日

ページレイアウトの設定として、[geometry パッケージ](#)が有名であるが、これは pLaTeX・LuaTeX-ja の縦組クラスでは利用が不可能という問題があった。本文書で解説する lltjp-geometry パッケージは、geometry パッケージを縦組クラスに対応させるパッチである。

## 1 利用方法

lltjp-geometry パッケージは、LuaTeX-ja に標準で含まれている。本パッケージの動作には [lualatex](#), [etoolbox](#) パッケージが必要である。また、LaTeX 2<sub>ε</sub> 2020-02-02 以前では [filehook](#) パッケージも必要である。

LuaTeX-ja では、自動的に lltjp-geometry パッケージが読み込まれる。縦組クラスか否かの自動判定 (1.1 節) を上書きしたい場合は、

```
% \PassOptionsToPackage{force}{lltjp-geometry} % 強制的に有効
\PassOptionsToPackage{disable}{lltjp-geometry} % 強制的に無効
\documentclass{...}
\usepackage[...]{geometry}
```

のように **luatexja の読み込み前に** `\PassOptionsToPackage` で本パッケージに渡すオプションを指定する (`\usepackage{lltjp-geometry}` を行っても意味がない)。

pTeX 系列では、`tarticle`, `tbook`, `treport` といった縦組クラスを使う場合に、

```
\usepackage[...]{lltjp-geometry}
\usepackage[...]{geometry}
```

と、`geometry` パッケージの**前**に読み込む。

### 1.1 縦組クラスか否かの判定

本パッケージは、以下のいずれかが該当する場合に「現在のクラスは縦組クラス」と自動判定し、`geometry` パッケージ読み込み直後にパッチを当てる：

1. `geometry` パッケージを読み込む際に、現在の組方向が縦組になっている。
2. `\AtBeginDocument` により\*1指定される、`\begin{document}` 時に実行される内容に `\tate` (というトークン) が含まれている。

---

\* <http://osdn.jp/projects/luatex-ja/wiki/FrontPage>

\*1 LaTeX 2<sub>ε</sub> 2020-10-01 以降ではそれと同義な `\AddToHook{begindocument}` も含む。

3. 本パッケージを読み込む際に `force` オプションが指定されている。

LuaTeX-ja で縦組クラスを利用する場合は主に 1. の、pTeX 系列で縦組クラスを利用する場合は主に 2. の状況となる\*2。

上記の自動判定がうまく行かなかったときに備え、本パッケージには `force` オプションと `disable` オプションを用意した。

- `force` オプションが指定されている場合は、自動判定の結果に関わらず `geometry` パッケージ読み込み直後にパッチを当てる。
- `disable` オプションが指定されている場合は、自動判定の結果に関わらず何もしない。

## 2 lltjp-geometry 使用時の注意事項

### 2.1 twoside 指定時

縦組の本は通常右綴じである。これを反映し、`twoside` オプション指定時には

- `left`, `lmargin` は小口側の余白、`right`, `rmargin` はノド側の余白を指す。
- 左右余白比 `hmarginratio` の標準値は 3 : 2 に変更。
- `bindingoffset` は右側に余白を確保する。

と変更している。

### 2.2 width と height

`\textwidth` が字送り方向の長さ (縦) を表すのと同様に、`width`, `totalwidth`, `textwidth` キーの値も字送り方向を、また `height`, `totalheight`, `textheight` キーの値も行送り方向 (横) を表すようになっている。

しかし、用紙サイズについては例外であり、物理的な意味での幅・高さを表す。`paperwidth`, `layoutwidth` はそれぞれ紙の横幅、レイアウトの横幅を、`paperheight`, `layoutheight` はそれぞれ紙の高さ、レイアウトの高さを表している。

### 2.3 傍注

縦組の場合、傍注は本文の上下に配置される\*3。これにより、`includemp` (や `includeall`) が未指定の場合、傍注はヘッダやフッタに重なる。`includemp` 指定時は、`\footskip`, `\headsep` のいずれか (二段組の場合は両方) を `\marginparwidth` + `\marginparsep` だけ増加させる。

## 3 lines オプションに関する注意事項

本節の内容は、`lltjp-geometry` パッケージを読み込まない場合、つまり、横組クラスで `geometry` パッケージを普通に使用した場合にも当てはまる注意事項である。

---

\*2 標準縦組クラスでは、`\begin{document}` の内部で組方向を縦組に変更する。

\*3 二段組の場合は上下共に、一段組の場合は標準では下側だが、`reversemp` が指定されたときには上側に配置される。

### 3.1 fontspec パッケージとの干渉

fontspec パッケージの、読み込み直後に geometry パッケージを用いてレイアウトを設定すると、lines による指定が正しく働かないという症状が生じる：

```
\documentclass{article}
\usepackage{geometry}
\usepackage{fontspec}
\geometry{lines=20}
\begin{document}
\hoge\typeout{\the\topskip, \the\baselineskip, \the\textheight}
\end{document}
```

\typeout で \topskip, \baselineskip, \textheight の値を調べると

$$\frac{\text{\textheight} - \text{\topskip}}{\text{\baselineskip}} = 15.8\dot{3}$$

となるのがわかるから、1 ページには 16 行分入らないことがわかる。

これは、fontspec の読み込みによって \baselineskip がなぜか 10 pt に変えられてしまい、\geometry 命令はその値に従って本文領域の高さを計算するためである。とりあえずの対策は、\normalsize によって \baselineskip を正しい値に再設定し、その後レイアウトを設定すれば良い：

```
\usepackage{geometry}
\usepackage{fontspec}
\normalsize\geometry{lines=20}
```

### 3.2 \maxdepth の調整

L<sup>A</sup>T<sub>E</sub>X では、最後の行の深さ  $d$  と本文領域の上端から最後の行のベースラインまでの距離  $f$  に対し、

$$\text{\textheight} = f + \max(0, d - \text{\maxdepth})$$

が成り立つ。

pT<sub>E</sub>X 系列の標準縦組クラス [u]rticle 等、及びそれを LuaT<sub>E</sub>X-ja 用に移植した ltjarticle 等では、\topskip は横組時における全角空白の高さ 7.77588 pt<sup>\*4</sup>であり、\maxdepth はその半分の値（従って 3.88794 pt）である。

いくつかのフォントについて、その中の文字の深さの最大値を見てみると表 1 のようになっている。欧文フォントのベースラインは、そのままでは和文との組み合わせが悪いので、さらに tbaselineshift = 3.41666 pt だけ下がることを考えると、最後の行に和文文字が来た場合はほぼ確実に深さが \maxdepth を超えてしまうことになる。従って、本文領域を「 $n$  行分」と

---

\*4 標準の 10pt オプション指定時。以下同じ。ところで、この量は公称フォントサイズの 10pt か、もしくは全角空白の高さと深さを合わせた値の 9.16446 pt の間違いではないか、と筆者は考えている。なお、奥村晴彦氏の [pL<sup>A</sup>T<sub>E</sub>X 2<sub>ε</sub> 新ドキュメントクラス](#)では公称ポイントサイズ 10 pt に設定されている。

表1 いくつかのフォント中の、文字の深さの最大値

フォント (10 pt)	深さ (pt 単位)
横組用の標準和文フォント (pTeX)	1.38855
縦組用の標準和文フォント (pTeX)	4.58221
Computer Modern Roman 10 pt	2.5
Computer Modern Sans Serif 10 pt	2.5
Times Roman (ptmr8t)	2.16492
Helvetica Bold Oblique (phvbo8t)	2.22491
Palatino (pplr8t)	2.75989

して指定するときによく使われる

$$\text{\texttheight} = \text{\topskip} + (n - 1)\text{\baselineskip} \quad (1)$$

は `tarticle` クラスのデフォルトでは通用しない。

通常の地の文のみの文章においてほぼ確実に (1) が成り立つようにするため、`lltjp-geometry` では `lines` オプション指定時のみ `\maxdepth` の値が最低でも

公称ポイントサイズの半分に、欧文ベースラインのシフト量を加えた値<sup>\*5</sup>

になるようにしている。`lines` オプション非指定時にはこのような調整は行われない。

### 3.3 見かけ上の基本版面の位置

`LATEX` では、本文の一行目のベースラインは、本文領域の「上端」から `\topskip` だけ「下がった」ところに来ることになっている。あまり `\topskip` が小さいと、ユーザが大きい文字サイズを指定した時に 1 行目のベースライン位置が狂う危険があるため、`geometry` パッケージでは

`lines` オプション指定時、`\topskip` の値を最低でも `\strutbox` の高さ ( $0.7\text{\baselineskip}$ ) まで引き上げる

という仕様になっている。

縦組の場合は、`\strutbox` に対応するボックスは `\tstrutbox` であるため、`lltjp-geometry` では

`lines` オプション指定時、`\topskip` の値を最低でも `\tstrutbox` の高さ ( $\text{\baselineskip}/2$ ) まで引き上げる

という挙動にした。見かけ上は `\topskip` の値制限が緩くなったが、前節で述べたように欧文フォントのベースラインは和文に合うように下にずらされるので、実用上は問題は起きないだろう。

前節の `\maxdepth` の調整も考え合わせると、`LATEX` が認識する本文領域と、実際の見たい目

<sup>\*5</sup> `tarticle` の場合だと、 $5\text{ pt} + 3.41666\text{ pt} = 8.41666\text{ pt}$  である。

の基本版面の位置とは異なることに注意してほしい。

例えば A4 縦を縦組で、公称フォントサイズ 10 pt、行送り 18 pt、30 行左右中央というレイアウトにするため、

```
\documentclass{tarticle}
\usepackage{lltjp-geometry}
\baselineskip=18pt
\usepackage[a4paper,hcentering,lines=30]{geometry}
```

と指定すると、実際には以下のように設定される。

- `\topskip` は `\tstrutbox` の高さ 8.5 pt に設定される。
- 本文領域の「高さ」`\textheight` は

$$\text{\topskip} + (30 - 1)\text{\baselineskip} = 530.5 \text{ pt.}$$

- 従って、左余白と右余白は

$$\frac{210 \text{ mm} - \text{\textheight}}{2} = 33.50394 \text{ pt.}$$

しかし、実際にはページの最初の行のベースラインは、本文領域の右端から `\topskip` だけ左にずれたところにあり、一方ページの最終行のベースラインは本文領域の左端にある。縦組和文フォントのベースラインは文字の左右中央を通ることから、従って、**見た目では、右余白の方が `\topskip` (= 8.5 pt) だけ大きい**ということになってしまう\*6。

---

\*6 同様に、横組で `vcentering` を指定すると、見かけでは `\topskip - \Cht + \Cdp` だけ上余白が大きいように見える。