Package pxgentombow v0.7

Hironobu Yamashita

2017/07/23

日本の出版では、たとえば版面が A4 サイズの場合、一回り大きな B4 サイズの用紙の中央にトンボ付きで印刷するということがなされることがあるようです。このドキュメントは、実際に A4 用紙を B4 用紙の中央にトンボ付きで配置している事例です。

pxgentombow パッケージは、クラスファイルなどが想定している用紙サイズ情報（\paperwidth, \paperheight）を検知し、それより一回り大きなサイズの用紙の中央にトンボ付きで出力するために必要な機能を提供します。ただし、このパッケージは実際の出力サイズ指定を発行しませんので、bxpapersize パッケージまたは bounddvi パッケージと併用するとよいでしょう[*1]。

使い方の例は

```
\documentclass[a4j]{jarticle}
\usepackage{pxgentombow}
\usepackage{bounddvi}
\begin{document}
…本文…
\end{document}
```

です。pLaTeX における横組と縦組の両方で使え、また jsclasses のような版面拡大 (\mag) が使われた場合にも対応しています。なお、現時点では pLaTeX と upLaTeX および LuaTeX-ja のみで動作します。

本パッケージは、platex-tools バンドルの一部として配布されています：

https://github.com/aminophen/platex-tools

用紙サイズの自動検知による出力サイズ決定

パッケージにはあらかじめ A 系列 (a0–a10)、B 系列 (b0–b10), C 系列 (c0–c10) と letter, legal, executive の用紙サイズが定義されています。ここで、B 系列は ISO ではなく JIS です。また、変形版として a4var（a4 の変形）と b5var（b5 の変形）も定義されています。

これらのうちいずれか（ただし a0, b0, c0 を除く）の用紙サイズを検知すると、出力サイズが次の規則で自動的に決定されます[*2]。

[*1] bounddvi パッケージは、pxgentombow と同じく platex-tools バンドルに収録されていますが、名前のとおり DVI を経由する場合にしか利用できません。一方、bxpapersize パッケージは汎用で、LuaTeX のような PDF 直接出力の場合にも利用できます。

[*2] なお、C 系列と letter, legal, executive については日本での慣習が不明のため、現時点では A 系列のサイズで出力することにしています。

- 用紙サイズが A 系列のとき：出力サイズは一回り大きな B 系列
- 用紙サイズが B, C 系列または letter, legal, executive のとき：出力サイズは一回り大きな A 系列

この場合、パッケージを読みこんだだけでトンボが付きます。なお、用紙サイズが横長の場合は自動的に出力も横長になり、縦長の場合は自動的に縦長になります。

よく使われる用紙サイズの例を挙げます。

用紙サイズ	出力サイズ
a6	b6
b6	a5
a5	b5
b5	a4
a4	b4
b4	a3
a3	b3
b3	a2
c6	a5
c5	a4
c4	a3
c3	a2
letter	a3
legal	a3
executive	a4

用紙サイズの自動検知に失敗した場合の出力サイズ決定

仮に用紙サイズが定義済みのいずれとも異なる場合は、デフォルトでは用紙の天地左右に 1 インチずつのノビを付けたサイズで出力します。たとえば、幅 100 mm、高さ 200 mm の用紙の場合、出力サイズは幅 100 mm + 2 in、高さ 200 mm + 2 in になります。

オプションによる出力サイズの明示指定

自動決定されるサイズと異なるサイズに出力したい場合、パッケージオプションで明示的に指定することができます。たとえば

```
\documentclass[a4j]{jarticle}
\usepackage[tombow-a3]{pxgentombow}
\begin{document}
…本文…
\end{document}
```

とすると、出力サイズは（自動決定の b4 は無視されて）a3 に変わります。指定可能なサイズは、定義済みの

用紙サイズと同じものです。すなわち、A 系列 (a0–a10)、B 系列 (b0–b10), C 系列 (c0–c10) と a4var, b5var, letter, legal, executive です。なお、ここでも用紙サイズが横長の場合は自動的に出力も横長になり、縦長の場合は自動的に縦長になります。

オプションの書式は、トンボ形式とサイズをハイフン (-) で結びます。トンボ形式は、pLaTeX の標準クラスと同じで `tombow`, `tombo`, `mentuke` のいずれかを選びます（`tombow` はジョブ情報を表示し、`tombo` は表示しません。また、`mentuke` はトンボの線を表示しません)。

トンボに表示するジョブ情報の有無

用紙サイズの自動検知によって出力サイズが決まる場合、デフォルトでは `pxgentombow (2017-07-29 17:20)` のようにトンボにジョブ情報が出力されます。これを無効化するには

```
\documentclass[a4j]{jarticle}
\usepackage[notombowdate]{pxgentombow}
\begin{document}
…本文…
\end{document}
```

とします。

jsclasses で使用する場合の注意

奥村晴彦氏による jsclasses のクラス（2016 年以降は日本語 TeX 開発コミュニティが管理）を使用していて、10pt 以外のサイズオプションを指定する場合は、以下のいずれかの方法をとってください。

- クラスオプションに「トンボオプション」（`tombow` または `tombo`）を追加する。
- クラスオプションに「`\mag` を使わないオプション」（`nomag` または `nomag*`）を追加する。

これは、jsclasses クラス内で行われる `\oddsidemargin` と `\topmargin` の計算の都合からくる制約です。たとえば

```
\documentclass[a4j,14pt]{jarticle}
\usepackage{pxgentombow}
```

という使い方は**誤り**です（このままでは誤った余白設定が適用されますので、安全のため pxgentombow パッケージがエラーを出すようにしてあります)。代わりに

```
\documentclass[a4j,14pt,tombow]{jarticle}
\usepackage{pxgentombow}
```

と書くようにしてください。

レイアウト設定の注意

余白などのレイアウト設定でありがちですが、`\hoffset` や `\voffset` の値が 0 以外になっている場合、pxgentombow パッケージはエラーを出します。レイアウト設定のために変更すべきなのはこれらの寸法では

なく、\oddsidemargin や \topmargin であることがほとんどです。したがって、それらを適切な値に設定するか、レイアウトの設定すべてを geometry パッケージに任せてしまうのも一つの方法です。

たとえば左右の余白を 25 mm に、上下の余白を 30 mm にしたいとき[*3]、まず「TEX の 1 インチ」を削除してから \oddsidemargin や \topmargin を変更すると、「見かけ上は」期待どおりの結果になることがあります。

```
\setlength{\hoffset}{-1in}% <== NG
\setlength{\voffset}{-1in}% <== NG
\setlength{\oddsidemargin}{25mm}
\setlength{\topmargin}{30mm}
\setlength{\textwidth}{\paperwidth}
\addtolength{\textwidth}{-2\oddsidemargin}
\setlength{\textheight}{\paperheight}
\addtolength{\textheight}{-2\topmargin}
\addtolength{\topmargin}{-\headheight}
\addtolength{\topmargin}{-\headsep}
```

しかし、この設定では pxgentombow パッケージがトンボを追加するとき、正しい余白を維持することができません。

一方、\hoffset や \voffset は 0 のままで、以下のように \oddsidemargin や \topmargin を設定していれば問題ありません。

```
\setlength{\oddsidemargin}{-0.4mm}% 25mm = 1inch - 0.4mm
\setlength{\topmargin}{4.6mm}%      30mm = 1inch + 4.6mm
\setlength{\textwidth}{\paperwidth}
\addtolength{\textwidth}{-50mm}
\addtolength{\topmargin}{-\headheight}
\addtolength{\topmargin}{-\headsep}
\setlength{\textheight}{\paperheight}
\addtolength{\textheight}{-60mm}
```

これと同等のレイアウト設定は、以下のように geometry パッケージで行うのが簡単です。

```
\usepackage[lmargin=25mm,rmargin=25mm,
            tmargin=30mm,bmargin=30mm]{geometry}
```

雑記

発端はこの話です。

- 齋藤修三郎 (@psi_tau) on Twitter, 2017 年 2 月 9 日
  https://twitter.com/psi_tau/status/829873082911248386

また、現在検討中の事項を挙げておきます。

- 現状では、用紙の横長・縦長がそのまま出力に反映されるので、これを逆転させるオプションの実装。また、任意の出力サイズを指定できるインタインタフェースの実現。
- 自動で決定できる出力サイズの拡張。現状では用紙サイズが定義値に完全一致する場合のみ自動決定さ

[*3] ここでは「本文の領域以外」を余白と定義します。すなわち、ヘッダとフッタは余白の一部です。

れるが、中間のサイズでもその一回り大きなサイズに出力することは可能と思われる。
- 出版用途ではカラー印刷の場合に、CMYK の版ごとにトンボを作る必要がある。color パッケージが読み込まれている場合に、オプション次第で CK など必要に応じた色を選べるようにするとよいのではないか[*4]。
- \mag ≠ 1000 の場合について、jsclasses 以外での動作は未確認。特に、geometry とは共存しない可能性が高い。
- \stockwidth/\stockheight が \paperwidth/\paperheight より小さい場合の動作。現在は警告を出すだけとしているが、さらに天地左右 1in にフォールバックしたほうが無難だろうか。
- pLaTeX/upLaTeX/LuaTeX-ja 以外のサポート。

変更履歴

- 2017/02/10 v0.1 最初の公開版
- 2017/03/01 v0.4 トンボ形式の修正など
- 2017/05/05 v0.5 jsclasses の \mag ≠ 1000 に対応、最初の CTAN リリース版
- 2017/05/06 v0.6 jsclasses との共存時のチェック強化、LuaTeX-ja での動作確認
- 2017/07/23 v0.7 ドキュメント更新、CTAN リリース版

[*4] ただし作者の本業は出版ではないので、商用を含む実用には程遠いかもしれない。