Your personal LaTeX bookshelf: Improving your background in a time of lockdown

Peter Flynn

Abstract

This paper describes the development of a LaTeX package to create a bookshelf image from a BibTeX file, suitable for use as a background for a video call in Zoom, Skype, or similar. Each entry is typeset as the spine of a book with title and author, using a randomly-selected font, color, and size. The paper describes the problems of random choice with both fixed-length and [potentially] endless lists, and the algorithm used to fit the author and title onto the spine. The package is available as bookshelf on CTAN for inspection and testing.

Background (literally)

It started on Twitter, when several people were commenting on the way people appeared when suddenly faced with having to do a Skype or Zoom video call during the COVID-19 lockdown. Apart from the lack of a camera crew, makeup team, sound crew, and production control, there were a lot of hastily-cleared walls, bookcases, window-ledges, and even whole rooms on view behind the talking head.

In particular, people who read and write, particularly academics, have lots of bookcases with lots of books, often in a state of considerable disarray. This doesn't look good — people may laugh about notoriously untidy professors, but when you need to sit up and be interviewed about epidemiology, or seroprevalence, or the 1918 influenza pandemic, you need to look calm and professional, and that jumble of books doesn't cut it.

It suddenly dawned on me that in the BibTeX users' environment, we have title and author for practically everything we have ever cited — somewhere. What was needed was a *virtual* bookcase, an image generated from life's collection of reading.

Publishers do keep images of their books, but usually the front cover, not the spine; and even so, they would not be available to the public, nor would they ever be in a sensible, uniform location on their web sites. No, it would need to be random: a random color for background and font; a random font from the huge range available to TeX users; and a random height and width of spine. In fact the only non-random data available would be the BibTeX entries, and rather than sort them, the order could be left to the user.

'Actually it turns out to be rather easy, but it would need an algorithm for colour-pairing, and a few assorted layouts for title an author. But basically, it works.' (May 1st)

Figure 1: First pass

Start-up

In the traditional Internet ethos of 'rough consensus and running code' it didn't take too long to come up with a proof-of-concept, which I ran past `@latex_ninja`, `@damienmulley`, and a few of the usual suspects (Figure 1).

By this time the requirements were becoming more apparent:

Randomness There needed to be a way to generate random values to select at least five aspects: *a*) colors (font and background); *b*) height and width; and *c*) font (well, typeface).

Data The need for selection meant that LaTeX somehow had to be provided with a list of available typefaces and available colors, and that minima and maxima for the book spine height and width needed to be set; and that those would need to be floating-point (lengths) whereas the font and color selection would need to be integer.

Color-pairing It was clear from early on that just picking two random colors was a recipe for conflict. What was needed was a way to say if one color was sufficiently in contrast with the other one to be legible.

Format It would be nice if there was some variation in spine layout, rather than having all the books look the same.

The randomness was easily fixed with Donald Arseneau's wonderful random package, which can generate both random integers and random dimensions.

However, if this was to deal with anyone's BibTeX files, some way to deal with character encodings would be needed, some way to overcome the assorted weirdnesses of old *bibtex* `.bst` files, and some way to choose from the user's installed fonts. That most useful of devices, Occam's Razor, was employed: UTF-8 only, XeLaTeX only, using biblatex and *biber*. I've been using this method for a couple of years now, and while I'm aware of the need for more development, it works for me, and the time has probably come to put the old `.bst` system out to grass.

Implementation

A shell script was created that extracted all entry keys from the user's BibTeX file and formatted each as a command to call the `\makebook` command, which the class defines to handle one entry. This can be `\input` by the user's document.

```
cat "$BIBFILE" |\
    grep '^@' |\
    grep -viE '(@Preamble|@String)' |\
    awk -F\{ '{print $2}' |\
    awk -F, '{print "\\makebook{" $1 "}%"}' \
      >entries.tex
```

That left basically three main actions: pick a font, pick the colors, and size the spine.

Font selection LaTeX has no way to create a list of installed fonts. Operating systems provide this information, so an external preprocessor was going to be needed. A TeX `\ifcase` structure was considered, but the number of installed fonts on many systems would be too large. The method chosen was to create a set of files numbered `1.tex`, `2.tex`, etc., in a subdirectory, each one containing a font selection command. The numbering is easily scripted on Unix-like systems (including GNU/Linux and Apple macOS) by using *fc-list* and the standard text utilities to create the files, simply numbered in order of occurrence.

The final action is to place a command setting the maximum bound for the random choice into another file that gets `\input`. Selection can then be done with `\setrannum` between 1 and the maximum, and then using `\input` to execute the font selection.

Colors In the case of colors, there is again a theoretical infinity of choice. However, practicality suggested one of the named palettes in the xcolor package, and *svgnames* was chosen as a representative sample. It also had the advantage of being small enough to be instantiated as an `\ifcase` structure. Extending the script was straightforward to extract the color names from `svgnam.def` and write the `\ifcase` into a file that can be `\input`. As with font selection, `\setrannum` is used to pick a number to apply to the `\ifcase` for the background, and again for the foreground.

Height and width Random dimensions sounded fine, but needed taming: for any given length of title and author, a certain amount of space is needed. In LaTeX, this tends to be like the choice of column type in a `tabular` environment: left, right, and center only handle single lines of data: for longer data you need a paragraphic cell. So long titles need to be allowed to wrap naturally in a `\vbox`, whereas shorter ones don't, so this is going to affect how much width and height is needed. The starting-point was a height and width set with `\setrandimen` between 5–20mm wide and 70–110mm high.

In addition, an alternative layout was created: author name across the top, rather than run-in with the title. The sizing algorithm was therefore:

1. an author name shorter than the randomly-chosen width of the spine would be typeset horizontally across the width of the spine, at the top, and its height deducted from the randomly-chosen height of the spine;
2. measure the width of the typeset title (or the title and author, joined by an em dash);
3. if the result was longer than the available height, typeset the title (and author, if needed) into a box of width at the available height in ragged-right mode so that it will run naturally to as many lines as needed;
4. measure the height of that box and if necessary increase the chosen width of the spine to accommodate it.

Theoretically you could then cycle round and see those that affected the choice of where the author was typeset, but this was felt to be a step too far for an initial solution.

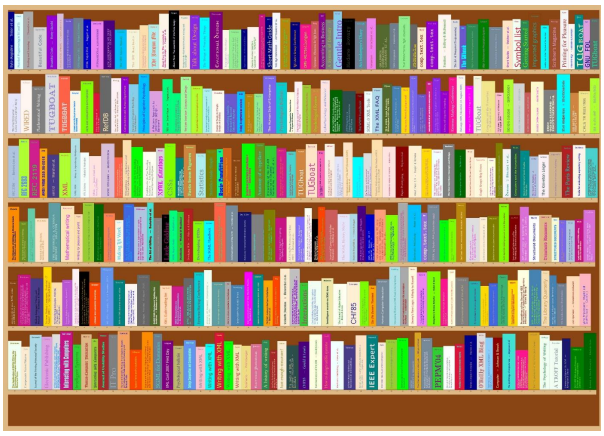
Adjustments

One immediate problem was known — colour clash or brightness and contrast in pairing — but its effect was not apparent until a large bookshelf was created. A workable solution is due to Nir Dobovizki [1], which proposes the formula

$$\text{brightness} = \sqrt{.241r^2 + .691g^2 + .068b^2}$$

where r, g, and b are the red, green, and blue values expressed as integers between 0 and 255.

Code to compute this was added to the script so that color selection and brightness selection could



'Fixed the colour-pairing and random font selection and picked two layouts. Basically working but needs more test data. Thid is my thesis bibliography' [sic] (May 16th)

Figure 2: Working solution

be done in parallel, and a clash rejected, within a loop. By inspection of the gamut, the approximate location of the median brightness value appeared to be 0.7, so the code ensures that each of the two colors chosen falls either side of this value. A notional value of 10 was used for regulating loop exit, after which the current values are used regardless; this appears to be sufficient.

This created a working solution (Figure 2), but left an unresolved issue: the data-preparation script was including all TTF and OTF fonts regardless of their type, whereas it needed limiting to text typefaces with a Latin register (that is, excluding math, symbols, and display fonts). In addition, on the author's system, some directories of older, experimental, and test fonts needed to be excluded.

Some inspection and experimentation showed that a reasonable list could be created by excluding any font name with a match in a regular expression containing suitable strings:

```
(Bitmap|Emoji|Dingbats|Jazz|STIX|dings|
 Symbol|Numeric|DIN|Ornament|OCR|CJK|
 Awesome|Dummy|Math)
```

A cyclical pattern of test-as-you-go had been established, and I am grateful to the numerous people who sent me their thesis BibTeX files. One late addition was to shade the background to a dark color for the inside of the bookshelf, and to color the shelves themselves a pale cream, for which I used a technique suggested by Ulrike Fischer [2].

The final stage, left to the user, is to convert the PDF to image format. The default size is a landscape A0 page, which is huge, but accommodates a few hundred volumes. It shrinks well to a screen size.

Conclusions

The most recent step was to put the package on CTAN and see if there were suggestions (none so far). By this time a number of helpful suggestions had been received, and offers of testing were accepted. By May 24 I was able to report on Twitter:

> May 24 ▪ Replying to `@latex_ninja` `@TeX4Publication` `@erdmaennchen42` It has just been uploaded to CTAN. Thank you.

What could be done better?

- The script works in *bash* (Linux) and *zsh* (Mac). It needs extending to Windows (*cygwin*? *Powershell*?);
- The colors currently are too bright on-screen, although reportedly OK for printing: perhaps the color selection algorithm needs revising;
- Some more spine layouts would be interesting, as would more bookshelf layouts: books at an angle, or stacked horizontally;
- Can something from the biblatex field selection provide for a place to store color, font, layout, and size as one would for bibliometrics or a *catalogue raisonné*;
- 180° rotation for spine titles is needed for some non-English languages, and math in titles needs more testing;
- In essence, this is just an output format from a `.bbl` file. Perhaps it would be more useful rewritten as a biblatex style option.

References

[1] N. Dobovizki. Calculating the Perceived Brightness of a Color. *Making Time-Tracking Software*, Apr 2008. `https://www.nbdtech.com/Blog/archive/2008/04/27/Calculating-the-Perceived-Brightness-of-a-Color.aspx`

[2] U. Fischer. How to set a certain color (other than white) to margin areas? *tex.stackexchange.com*, Dec 2010. `https://tex.stackexchange.com/questions/7725/how-to-set-a-certain-color-other-than-white-to-margin-areas`

◇ Peter Flynn
Textual Therapy Division,
Silmaril Consultants
Cork, Ireland
Phone: +353 86 824 5333
`peter (at) silmaril dot ie`
`blogs.silmaril.ie/peter`