Calculating covers with ConTEXt

Henning Hraban Ramm

Abstract

Every TEX user can typeset a book, but the cover might be a different story. We will learn a bit about dimensions and calculations as we calculate a cover.

Take cover!

I used to create the covers for my ConTEXt books in a graphics application. I still think that's the best way to plan a cover, because I can try new ideas or make changes, and see the results straight away.

But there are disadvantages. It can take a lot of steps to make a small change. You need to hand calculate the spine width, and remember to change it if the number of pages changes. If you change the title or author on the front cover, you need to remember to change it on the spine too. It can get messy if you want several graphics, each slightly different.

I want to show you how I use ConTEXt to calculate covers, without those disadvantages.

Basic setup

Let's start simple: we define a paper size and a whole-page layer, on which we will place our elements:

```
\definepapersize[Cover][width=350mm,height=240mm]
\setuppapersize[Cover]

\definelayer[cover][
  x=0mm,y=0mm,
  width=\paperwidth,
  height=\paperheight,
]
\setupbackgrounds[page]
    [background=cover,state=start]

\starttext
\setlayer[cover][
  x=200mm,
  y=20mm,
]{\ss\bfd My Title}
\strut
\stoptext
```

The `\strut` at the end is necessary, otherwise the page has no content and ConTEXt won't even display the content of the layer.

Dimensions

Where did we get those dimensions? I used the size of the DANTE books. Their page is 165 by 240 mm, i. e. their cover width is two times 165 mm plus a 20 mm spine. Maybe our printshop told us these dimensions, but we would need to ask again if something changes. Let's see if TEX can do the calculations.

```
\definepapersize[Cover][
  width={2 * 165mm + 20mm},
  height=240mm,]
```

That would have been nice, but ConTEXt complained about an "Illegal unit of measure". It's not as easy as it might be.

For ConTEXt to calculate with dimensions, we must use 'dimension expressions'. As a rule, they must start with a dimension:

```
\definepapersize[Cover][
  width=\dimexpr 165mm * 2 + 20mm\relax,
  height=240mm,]
```

Do we need to use such cumbersome code for every element on the cover? No! We can predefine the most important values, most simply as macros:

```
\def\PageWidth{165mm}
\def\SpineWidth{20mm}

\definepapersize[Cover][
  width=\dimexpr \PageWidth*2 + \SpineWidth\relax,
  height=240mm,]
```

That works, but it's cleaner and more reliable if we define our own dimensions. In ε-TEX it looked like this:

```
\newdimen\PageWidth
\PageWidth=165mm
```

But because we use ConTEXt, it should look like ConTEXt:

```
\definemeasure[PageWidth][165mm]
```

You can retrieve such a value in two ways: with `\measure` as a string for assignments in `\setup` commands, or with `\measured` as a dimension for calculations.

In the second argument of `\definemeasure`, we can execute calculations without writing an explicit `\dimexpr`, but internally, since `\definemeasure` uses `\dimexpr`, it has the same limitations:

```
\definemeasure[CoverWidth]
    [2\measured{PageWidth} + 20mm]
```

Oh, this expression doesn't start with a dimension!? Well, simple factors like this are possible, sometimes even with decimal numbers. For example, these are valid: '`2\lineheight`', '`1.5\lineheight`', and '`\lineheight * 2`', but '`\lineheight * 1.5`' is not. In such cases you can cheat with fractions: '`*1.5`' throws an error, while '`*3/2`' works.

Now the complete code is:

```
\definemeasure[PageWidth][165mm]
\definemeasure[PageHeight][240mm]
```

First published in German in DANTE's *Die TEXnische Komödie* 1/2023, pp. 16–25; translation by the author. English version originally edited by Peter Hopcroft.

doi.org/10.47397/tb/44-2/tb137ramm-covers

```
\definemeasure[SpineWidth][20mm]
\definemeasure[CoverWidth]
  [2\measured{PageWidth} + \measured{SpineWidth}]

\definepapersize[Cover][
  width=\measure{CoverWidth},
  height=\measure{PageHeight}]
\setuppapersize[Cover]

\definelayer[Cover][
  x=0mm,y=0mm,
  width=\paperwidth,
  height=\paperheight,]
\setupbackgrounds[page]
    [background=cover,state=start]

\starttext
\setlayer[Cover][
  x=\dimexpr\measured{PageWidth}
            + \measured{SpineWidth}
            + 15mm\relax,
  y=20mm,
]{\ss\bfd My Title}
\strut
\stoptext
```

Page count

If the number of pages changes, we only need to change one number. But it is better to calculate it automatically:

```
\useexternalfigure[content][book.pdf]
\getfiguredimensions[content]
\expanded{\definemeasure[SpineWidth]
    [2mm + (0.09mm * 3/2 * \noffigurepages/2)]}
```

`\useexternalfigure` gives our content PDF the symbolic name `content`. `\getfiguredimensions` detects the properties of the current image, including the number of pages in the PDF, which is stored as `\noffigurepages`. We have to use `\expanded` to execute `\definemeasure` immediately. Otherwise the current image would have changed, and `\noffigurepages` would be wrong.

Paper thickness

How did we get that formula? We need the number of sheets. And my vocational teachers used to remind us: "Paper has two sides!" Our book will be printed on 90 gsm[1] paper. Standard 90 gsm paper is 0.09 mm thick. Our paper has some light filler material, and is thicker: 1.5 times 0.09 mm thick. The 1.5 (written above as `3/2`) is called bulk. Of course, we could define paper thickness as a ConTEXt dimension, but we don't need it elsewhere.

The 2 mm above is a fold allowance, about 1 mm for each fold between a cover and the spine. Strictly speaking, the fold allowance calculation should take into account the thickness of the cover cardboard.

Now our cover automatically adapts to the number of pages in the book. Not too bad.

Layers

We want to place the title on the front cover, as well as the subtitle and author. We want these on the spine as well. On the back cover, we want some blurb and an ISBN barcode. Must we repeat the same laborious calculations time and again? No!

We will define separate layers for front cover, back cover, and spine. Then we can give the offsets of elements relative to their parent layer:

```
% ...
\definelayer[BC][ % back cover
  hoffset=0mm,
  y=0mm,
  width=\measure{PageWidth},
  height=\measure{PageHeight},
]
\definemeasure[FrontStart]
  [\measured{PageWidth} + \measured{SpineWidth}]
\definelayer[FC][ % front cover
  hoffset=\measure{FrontStart},
  y=0mm,
  width=\measure{PageWidth},
  height=\measure{PageHeight},
]
\definelayer[Spine][
  hoffset=\measure{PageWidth},
  y=0mm,
  width=\measure{SpineWidth},
  height=\measure{PageHeight},
]
\setupbackgrounds[page]
    [background={Cover,BC,FC,Spine},
     state=start]
% ...
\setlayer[FC][
  x=15mm,
  y=20mm,
]{\ss\bfd My Title}
\setlayerframed[Spine][
  y=12mm,
  offset=overlay,
  frame=off,
  align=flushleft,
  width=\measure{SpineWidth},
  height=0.66\measured{PageHeight},
]{%
  \rotate[
    rotation=90,
    height=\measure{SpineWidth},
    width=0.66\measured{PageHeight},
    align={lohi,flushright},
  ]{Author: Title}%
}
```

[1] grams per square meter, also known as 'grammage'

For the spine text I used \setlayerframed so we have all the options of \framed to hand. While planning a cover, I like to turn on the frames to check the positions of the elements. We can make this setting a command-line argument:

```
\setupframed[offset=overlay] % no border distance
\startnotmode[debug]
  \setupframed[frame=off]
\stopnotmode
```

To turn the frame on, call ConTEXt with the option --mode=debug. The above code also sets offset=overlay for every frame, so we don't have to do this for every frame individually.

Buffers

Next, let's take care of the blurb. We can place it with \setlayerframed[BC]. But I find it confusing to have long text strings loitering within ConTEXt calculations. Therefore, we define the text as a buffer in advance:

```
\startbuffer[Blurb]
\quotation{I never read a better book!}
\wordright{(M. Reich-Radecki)}
\blank[2*line]
Something about the brilliant content...
\stopbuffer

\startsetups[blurb]
% font/alignment/indent settings
\stopsetups

% ...
\setlayerframed[BC][
  x=.15\measured{PageWidth},
  y=20mm,
  width=.7\measured{PageWidth},
  height=.8\measured{PageHeight},
  setups=blurb,
]{\getbuffer[Blurb]}
```

Variables

Let's define some book data as variables, all in one place:

```
\setvariables[book][
  contentPdf={vol01},% name of the content file
  author={Donald E. Knuth},
  title={The \TeX book},
  subtitle={about command-based typesetting},
  series={Computers & Typesetting},
  volume={A},
  isbn={978-3-12345-007-Z},
  coverimage={lion},
]
% ...
\useexternalfigure[content]
    [\getvariable{book}{contentPdf}]
\getfiguredimensions[content]
% ...
\setlayer[FC][
  x=15mm,y=20mm,
  setups=maintitle
]{\getvariable{book}{title}}
```

When we change any of the above book data, it automatically changes on all the layers where it appears. Wonderful. I have seen books with a different title or author on the front cover and on the spine.

As you may observe, we can use expressions like \getvariable{book}{title} to retrieve values. Of course, we also could have used macros.

Environment

Since we need the data for the book's content (e. g. fly title, imprint) as well, we should save it to an external environment file that we can load in both the cover and the content documents:

```
\startenvironment settings
\project bookbook

\setvariables[book][
%...
]
\stopenvironment
```

Bleed

Next step: we'll add a background image. Since it should cover the whole page, we must set it up to 'bleed'. That means that the image extends a few millimetres past where the book will be trimmed. Otherwise there can be white gaps at the edges if the printed book isn't cut exactly to the trim line. (That's usually because paper changes size as humidity changes during printing, rather than the fault of the printshop or bookbinder.)

```
\definemeasure[Bleed][3mm]
\definemeasure[MaxHeight]
    [\measured{PageHeight} + 2\measured{Bleed}]
\setlayerframed[FC][
  %x=-\measured{Bleed},
  y=-\measured{Bleed},
]{\externalfigure[\getvariable{book}{coverimage}]
                 [height=\measure{MaxHeight}]}
```

Of course, we can also put a background image for the complete cover (back, spine, and front) on the "Cover" layer. If the number of pages changes, the width of the image will change slightly. Usually this doesn't matter.

ConTEXt documents often use MetaPost graphics as background images. For those, you can use the variables OverlayWidth and OverlayHeight.

While the image now has bleed, we can't see it when we look at the PDF on-screen, because we see the trimmed paper size. We can use the oversized

option to expand the paper size by 7.5 mm on all sides:

```
\setuppapersize[Cover][Cover,oversized]
```

But because we also need the dimension for our calculations, we will expand the paper size explicitly:

```
\definemeasure[Trim][7.5mm]
\definemeasure[CoverWidth]
  [2\measured{PageWidth} + \measured{SpineWidth}]
\definemeasure[CoverWidthPlus]
  [2\measured{PageWidth} + 2\measured{Trim}
   + \measured{SpineWidth}]
\definemeasure[CoverHeightPlus]
  [\measured{PageHeight} + 2\measured{Trim}]

\definepapersize[Cover][
  width=\measure{CoverWidth},
  height=\measure{PageHeight}]
\definepapersize[CoverPlus][
  width=\measure{CoverWidthPlus},
  height=\measure{CoverHeightPlus}]
\setuppapersize[Cover][CoverPlus]
```

We don't need to change the layers — their elements don't get trimmed at their borders.

Now we also want to see crop marks, and while we're at it, we should properly set up the invisible 'boxes' in the PDF that outline the trimmed area (TrimBox) and bleed area (BleedBox). You can only *see* them in Acrobat (Preferences > Page Display > Show art, trim & bleed boxes), otherwise you can check the values with `pdfinfo -box`.

```
\setuplayout[
  marking=on,% crop marks
  location=middle,% center page on the sheet
  cropoffset=0mm,
  bleedoffset=\measure{Bleed},
  trimoffset=-\measure{Trim},
]
\setupinteractionscreen[width=max,height=max]
```

- A positive value of `cropoffset` shrinks the visible area and also affects both of the other values.
- A negative value of `trimoffset` defines the offset from TrimBox to CropBox.
- A positive value of `bleedoffset` defines the bleed as the offset from BleedBox to TrimBox.
- Only `\setupinteractionscreen` activates the settings.

Setting TrimBox and BleedBox in this way does not affect the positions of the layers or their contents.

More hints about dimension calculations

Dimension expressions (`\dimexpr`) can be nested. It sometimes makes sense to call `\dimexpr...\relax` within a `\dimexpr`.

Internally, TEX calculates with integer 'scaled points' (`sp`) of 1/65536 pt. The maximum value for dimensions is 16384 pt (about 5.75 m).

If we output dimension values using `\measure`, they get typeset in pt (TEX points). We can convert units like this:

```
\define[2]\Conv{\scratchdimen #1 \the\nodimen #2
                \scratchdimen}
% first parameter: dimension,
% second parameter: unit. For example:
\Conv{1pt}{mm}
```

Final remarks

The code that I use in my publishing house also handles optional flaps.

This article is about softcovers. For hardcovers you need a bigger cover and more bleed (about 15 mm), because the cover paper gets glued around the cover cardboard. The spine also needs more folding allowance (about 4–5 mm) for the hinges. You can change the calculations above.

◇ Henning Hraban Ramm
hraban (at) fiee dot net