The pagelayout class

Manual for Version 1.0.5

https://github.com/friedemannbartels/latex-pagelayout

Friedemann Bartels

January 13, 2024

Contents

1 Introduction

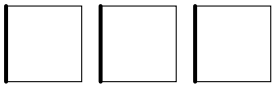
When Apple discontinued its photo book feature in Apple Photos, I was missing an easy-to-use photo book application. In 2020, I started to develop my own solution. What started with some SVG templates and shell scripts ended up in the pagelayout class, a declarative desktop publishing approach.

With the pagelayout class you can create single- and double-sided documents, create pages with margins, safety margins, and bleed, use templates, align text and graphics in a grid, wrap text across multiple pages and use before pages. Automatic grid layout and a simple and consistent user interface make it easier than ever to create graphics-rich documents with LaTeX. Under the hood the TikZ and tcolorbox packages, ImageMagick and Inkscape are used.

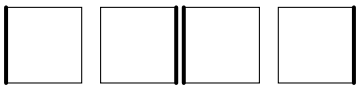
2 Document

The document arguments described in the following can be either set as a document argument (`\documentclass[twoside]{pagelayout}`) or with a command (`\twoside`) in the preamble.

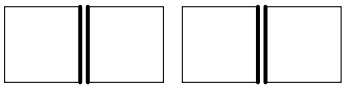
`\documentclass` Use the class with the `\documentclass[`⟨*arguments*⟩`]{pagelayout}` command. A single sided document is the default.



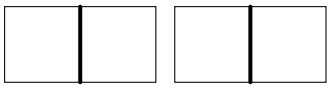
`\twoside` Using the document argument `twoside` results in a document with left and right pages. When you create a two-sided document, your document always has an even number of pages. If you create a document with an odd number of pages, an additional page is automatically inserted. Two-sided documents start with a right page by default.



`\beginleft` Use the argument `beginleft` to begin with a left page.

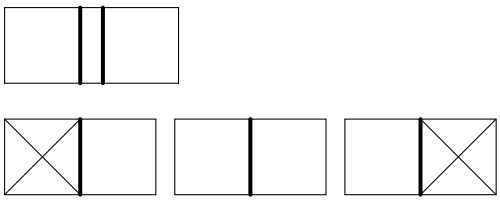


`\fanfold` By default, each page is placed on one paper. With the `fanfold` argument you can change this behavior so that a left and a right page are placed on one paper. In this case, the first page is a left one by default.



`\cover` The `cover` argument is similar to the `fanfold` argument, but only applied to cover pages. Read more about cover pages in section 3.

`\beginright` If you want your fanfold document to start with a right hand page, you can use the argument `beginright`.



2.1 Lengths

The document lengths described below can be set either as a document argument (`\documentclass[page width=21cm]{pagelayout}`) or by assigning the dimension directly (`\pagewidth=21cm`) in the preamble or in the document.

`\pagewidth` `\pageheight` You can set the lengths `page width` and `page height`. The default page size is `210mm` by `297mm`.

`\bleed` You can set the `bleed` length. It applies to all paper edges. The default bleed is `0mm`. You can override the bleed for each edge separately by defining the lengths `top bleed`, `inner bleed`, `bottom bleed`, `outer bleed`. The paper size results from the page size and the bleed.

`\safetymargin` The `safety margin` is the only length that has no influence on the final result. Its only purpose is to control the safe zone when editing. It is `0mm` by default and can be configured for each edge by defining `top safety margin`, `inner safety margin`, `bottom safety margin` and `outer safety margin`.

`\margin` `\gutter` You can set the lengths `margin` and `gutter` for the document, for a page (see section 3) or for a grid (see section 4). The default margin is `20mm`, the default gutter is `0mm`. Similar to bleed and safety margin, the margin can be defined for each edge individually by setting `top margin`, `outer margin`, `bottom margin` and `inner margin`.

`\coverwidth` `\coverheight` You can create documents with cover pages. Read more about creating cover pages in section 3. You can set the lengths `coverwidth` and `coverheight`. By default, the cover pages are the same size as the inside pages. If you want to create a cover with a spine, use the `fanfold` or `cover` argument described in section 2. In this case the length `spinewidth` takes effect. The default spine width is `3mm`.

You can overwrite the bleed and the safety margin for cover pages. To do this, prefix the respective lengths described above. For example, define `cover inner bleed` or `cover safety margin`.

2.2 Page Graphics and Colors

You can define a page graphic and a page color with the arguments `page graphic` and `page color`. Set the argument `color` to define the text color. These arguments can be defined for the document or for each page (see section 3).

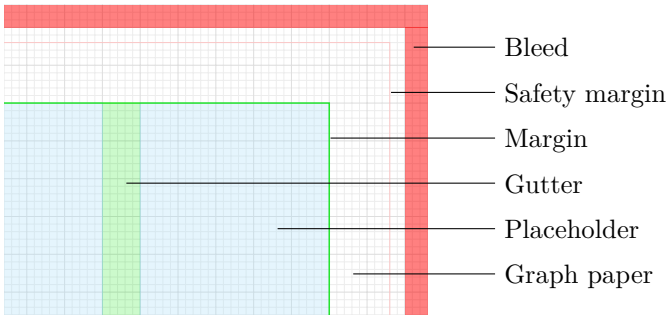
Read how to set the scale and crop of a page graphic with the `\newgraphic` command in section 5.

2.3 Layout Guides

The layout guides described in the following can be either set as a document argument (`\documentclass[grid]{pagelayout}`) or using a command (`\grid`) in the preamble or the document. Layout guides can be switched on (`\safezone`) or off (`\nosafezone`).

\grid \safezone \cuttingmarks \graphpaper

The `grid` argument visualizes the margin and the grid gutters with green lines. Read more about the grid in section 4. The `safezone` layout guide shows a red border for the bleed and a thin pink line to indicate the safety margin. As an alternative to `safezone`, you can also use `cutting marks` to display cutting marks. Show a graph paper with the `graph paper` argument.



\placeholders \fillpages

The `placeholders` argument shows template placeholders. Read more about templates in section 6. If you create a document with a cover and a first right hand page, use the `fill pages` argument during editing to add blank pages to get a better impression of the final result.

2.4 Draft Mode

The `draft` argument is a shortcut to enable the arguments `placeholders`, `cutting marks` and `fill pages`. It also speeds up rebuilding the PDF when changing the document.

3 Pages

\page

Create a page with the command `\page[`⟨*arguments*⟩`]{`⟨*content*⟩`}`. You can overwrite the document arguments `margin`, `top margin`, `bottom margin`, `inner margin`, `outer margin`, `gutter`, `page color`, `page graphic` and `color` for each page.

Use the argument `double` to create a double page. When creating a double page after a left page, a right page is automatically inserted before the double page. The argument `double` is only recognized for double sided documents.

Use the arguments `front cover` or `back cover` to create a front or back cover. If you have a back cover and a front cover, the back cover page must be defined first. If no argument is active, the front cover is placed as the first page and the back cover as the last page of the document. In this case, the only difference from normal pages is that the cover pages are not counted in the page numbering.

If the argument `fanfold` or `cover` is selected, the back and front covers are laid out on one paper so that the cover is suitable for a book with a hard or soft cover (see section 2). Space is reserved between the back and front covers for the spine. Set the `spine width` document argument as described in section 2.1.

With the arguments `hpos` and `vpos` you can set the default grid alignment for a page. Read more about the grid in the next section.

\setpagecolor

Alternativly to the argument you can use the command `\setpagecolor{`⟨*color*⟩`}` to set the page color for all following pages.

\setpagegraphic

With the command `\setpagegraphic[`⟨*arguments*⟩`]{`⟨*name*⟩`}` you can define

a page graphic for all following pages. The name references a graphic version created with the `\newgraphic` command (see section 5). If no graphic version with the given name exists, the command uses the name as the file name. You can change the clipping by defining relative decimal values for the arguments `scale` (>1), `hpos` and `vpos` (0-1).

`\newbeforepage` `\setbeforepage` With the command `\newbeforepage{`⟨*name*⟩`}{`⟨*content*⟩`}` you can create a reusable page layer, that can be set before pages. Use the command `\setbeforepage{`⟨*name*⟩`}` to define a before page for all following pages or use the page argument `before page` to specify a before page for a individual page.

`\ifleftpage` `\ifrightpage` Use the condition `\ifleftpage{`⟨*content*⟩`}` and `\ifrightpage{`⟨*content*⟩`}` to check for the current page alignment.

`\leftpage` Within a page scope you can use the `\leftpage{`⟨*content*⟩`}` command to place content only on a left page. In comparison to the `\ifleftpage{`⟨*content*⟩`}` condition, the command sets a grid for a single page. That way you can combine single `\rightpage` with double page layouts. Accordingly you can use the `\rightpage{`⟨*content*⟩`}` command.

4 Grid

The grid allows you to create rows with cells. You can define width and height relations between rows and cells. All definitions are relative. The size and position results from these relations and the configured dimensions page width, page height, margin and gutter.

`\setgrid` Create a grid with the command `\setgrid[`⟨*arguments*⟩`]{`⟨*config*⟩`}`. The grid configuration is a nested list of rows and cells. A cell is defined by a number that describes the width ratio to the other cells in the row. The default grid describes a row with one cell:

```
\setgrid{
  {{}}}
```

You can define a width relation between cells by adding a integer value for each cell:

```
\setgrid{
  {{1}{2}}}
```

With an optional integer value for each row you can set a height relation between rows:

```
\setgrid{
  {[2]{1}{2}}
  {[1]{1}{2}}}
```

You can set an aspect ratio for a cell by adding a `!` to the width. This way the aspect ratio is defined by the width of the cell and the height of the row. If all cells in a row have a fixed aspect ratio, the aspect ratio of the row is fixed too. In the following, a row with only fixed cells is named a fixed row. Because the height of a fixed row is determined by the aspect ratios of the cells, the height relation to the flexible rows is broken. In this case, you can force a height relation

between flexible and fixed rows by adding a `!` to the height of the flexible row. In the following, a row with a forced height is named a forced row.

The example defines a first fixed row with one cell and an aspect ratio of 3:2. The height of the second row is forced half of the height of the first row. The cells in the second row share the available width in the ratio one to two:

```
\setgrid{
  {[2]{3!}}
  {[1!]{1}{2}}}
```

If the height of fixed and forced rows exceeds the available height, the grid shrinks to fit the available space. The defined aspect ratios of the cells are preserved and the grid is centered horizontally. In this case you can define the `hpos` argument to align the grid horizontally. A value of `0` aligns the grid to the left, a value of `1` aligns the grid to the right. The default is a value of `0.5`.

If the height of fixed and forced rows is less than the available height, all flexible rows share the excess height. If there are no flexible rows, the grid is centered vertically. In this case you can define the `vpos` argument to align the grid vertically. A value of `0` aligns the grid to the top, a value of `1` aligns the grid to the bottom. The default is a value of `0.5`.

As for a document or a page you can set `margin` and `gutter` arguments for a specific grid. With the arguments `width`, `height`, `x` and `y` you can override the intrinsic grid size and position. When overriding the grid size and position, you can calculate with the intrinsic values, for example `x=\x-1cm`.

4.1 Placing Content

The grid aligns content (see section 5) automatically in the given grid cells. To set the position of a content box manually use the command `\place{`⟨*fromrow fromcell torow tocell*⟩`}`. To place content in the first row and first cell use the command `\place{0 0 1 1}` before the content. The definded place applies only for the following content box. The next boxes are then again placed with auto layout.

`\place`

The command accepts decimals. That allows you to place content everywhere within the grid. The `gutter` is taken into account when calculating the positions.

5 Content

5.1 Text

With the command `\text[`⟨*arguments*⟩`]{`⟨*text*⟩`}` you can place text. Align the text vertically with the `center` and `bottom` arguments.

`\text`

The `\text` command uses the package tcolorbox and accepts tcolorbox arguments, for example to set text, frame or background color and transparency. The tcolorbox behaves slightly diffenrent then the standalone version: spacings, rules and background are removed by default. You can change the behaviour with the tcolorbox command `\tcbset{`⟨*arguments*⟩`}`. Also the `arc` arguments behaves different: if you set the `arc` to `0pt` the `outer arc` is also set to `0pt`.

Add the argument `breakable` to break text over multiple boxes and pages. Use the command `\usetext[`⟨*name*⟩`]` to place the following text boxes in the grid. If

`\usetext`

you want to use multiple breakable text boxes you can initialize the text with the `name` argument and define a name that you then can pass to the `\usetext` argument. Breakable text boxes may have different heights and positions. To align the baselines use the `baselinesnap` argument and set the baseline dimension, for example `baselinesnap=\baselineskip`.

The `\text` command accepts the arguments `shadow`, `shadow size`, `shadow color`, `shadow opacity`, `shadow xshift`, `shadow yshift`, `border`, `border width`, `border color` and `border radius`. Read more about shadows and borders in section 5.4.

5.2 Graphics

`\newgraphic` Create a reusable graphic version with the `\newgraphic{`⟨*name*⟩`}{`⟨*arguments*⟩`}` command. Define the arguments `scale` (>1), `hpos` and `vpos` (0-1) to set the clipping of the graphic. With the argument `file` you can link a file. If no file is defined, the name is used as the file name.

`\graphic` The command `\graphic[`⟨*arguments*⟩`]{`⟨*name*⟩`}` places a graphic. The name references a graphic version created with the command `\newgraphic`. If no graphic version with the given name exists, the command uses the name as the file name.

The graphic is scaled to fit in the content box. If the aspect ratio of the graphic differs from that of the defined content box, the graphic is cropped. You can change the cropping by defining relative decimal values for the arguments `scale` (>1), `hpos` and `vpos` (0-1). With the `orientation` argument you can set the Exif orientation flag (1-8). To flip a graphic horizontally or vertically use the arguments `hflip` or `vflip`. Same as the `\text` command, the `\graphic` command accepts the `border` and `shadow` arguments described in section 5.4.

`\graphicspath` You can use the command `\graphicspath{`⟨*dir-list*⟩`}` of the graphics package to specify a list of directories in which to search for graphic files.

5.3 Other

`\xput` The command `\xput{`⟨*code*⟩`}` is an easy way to place arbitrary content in the grid. Within the code block the dimensions `\width` and `\height` represent the size of the content area and can be used to create graphics, that fit and adapt to the available size.

`\tikzgraphic` The command `\tikzgraphic{`⟨*tikz code*⟩`}` works like the `\xput` command, but wraps your input in a `tikzpicture`.

5.4 Shadows and Borders

Text and graphic content can be placed with a border or a shadow. You can define a border or shadow for a specific content box by using the arguments `border width`, `border color`, `border radius` to add a border or `shadow size`, `shadow color`, `shadow opacity`, `shadow xshift`, `shadow yshift` to add a shadow.

`\newborder` Define a reusable border or shadow with the `\newborder{`⟨*name*⟩`}{`⟨*arguments*⟩`}`
`\newshadow` or `\newshadow{`⟨*name*⟩`}{`⟨*arguments*⟩`}` commands. Apply the border or shadow to `\text` or `\graphic` with the argument `border` or `shadow`.

`\setborder` To set a border or shadow within a page or document scope, use the commands
`\setshadow` `\setborder[`⟨*arguments*⟩`]{`⟨*name*⟩`}` or `\setshadow[`⟨*arguments*⟩`]{`⟨*name*⟩`}`.

6 Templates

\newtemplate With the command \newtemplate{⟨*name*⟩}{⟨*layout*⟩} you can create a reusable layout. Creating a template works the same way as creating a page. In addition to graphics and text you can define placeholders. Use the command \placeholder{⟨*fromrow fromcell torow tocell*⟩} to define a content area that can later be filled with a text or a graphic.

\placeholder

\template Use a template with the command \template[⟨*arguments*⟩]{⟨*name*⟩}{⟨*content*⟩}. The available arguments are the same as for pages.

The command \template comes with an easy way to generate templates by using a name pattern. For example the template name *sp* creates a template with a square and a portrait format placeholder. In this case the letter `s` specifies a square (1:1) and the letter `p` a portrait (2:3) format placeholder. Use the letter `l` to define a landscape ratio (3:2), `g` for golden ratio (5:3), `o` for golden upright ratio (3:5), `w` for wide ratio (2:1) or `f` for a flexible ratio. With the character `-` you can start a new row. Within a row you can mix the fixed placeholders (`s`, `p`, `l`, `g`, `o`, `w`) or use only flexible placeholders (`f`).

7 Image Optimization

\optimize With the command \optimize[⟨*arguments*⟩] you can enable image optimization. When enabled, JPG and PNG images are croped, resized and cached. This results in fast rebuilds when changing the document. You can set the arguments `density`, `quality`, `unsharp` and `downsample threshold`. The `density` argument takes a number of the pixels per inch (default `300`), the `quality` argument takes numbers between 1 and 100, the `unsharp` argument an ImageMagick unsharp configuration (default `2x1`) and the `downsample threshold` a decimal value greater or equal 1 (default `1.2`). The `unsharp` argument can also be applied for individual graphics.

If you use image optimization and change the original graphics files, you must delete the cached versions so that the cache will update. The \import command helps you to streamline this process. When enabled, you need an import directory in which to place the modified files. When the document is created, the files are moved to your image directory and the corresponding cached versions are automatically regenerated. The import directory is either the directory *import* in your working directory. Or you can set a system wide import directory by defining the shell variable `PAGELAYOUT_IMPORT_DIRECTORY`.

\import

\preflight Use the \preflight command to proof the image resolution. If images have a final resolution less then 300 ppi you get a warning, if less then 200 ppi you get an error. With preflight enabled the maximum possible resolution is shown next to each graphic.