photobook — Document class for building photo-books *

Alex A. Naanou†

Released 2024-07-18

Abstract

The `photobook` LaTeX document class extends the `book` class defining a set of parameters, meta-macros, macros and environments with reasonable defaults to help typeset, build and print books mainly based on visual/image content.

Contents

Introduction 2

Usage 2

Options 2
- Page geometry . . . 2
- layout . . . 3
- Image clearance . . . 4
- Image block layout . . . 5
- PDF Viewer layout . . . 5
- Other options . . . 5

Packages 5

Globals 5

Initialization 7

Generic commands 7

Environments and Cells 7
- Generic . . . 7
- Cells . . . 8
- Save cells . . . 9
- Cell macros . . . 9
- Paper cells . . . 14
- Page cells . . . 14
- Foldout page cells . . . 16
- Multi-page cells . . . 18
- Endpaper cells . . . 19
- Cover and dust jacket cells . . . 19

Caption Templates 19

Page Templates 20
- Basic book information . . . 20
- Tweaking . . . 20
- General interface . . . 21
- No bleed . . . 22
- Partial bleed . . . 23
- Full bleed . . . 24

Spread Templates 25
- No bleed . . . 25
- Partial bleed . . . 26
- Full bleed . . . 27

Meta Macros / Environments 27

Miscellaneous 28

*This file describes version v0.1.30, last revised 2024-07-18.

†E-mail: alex.nanou@gmail.com

Introduction

LaTeX is great with textual and text-primary content with figures peppered in, as long as pictures/figures flow within, with or as part of text, vanilla LaTeX is fantastic.

One can relatively easily stretch the flow concept to more complex layouts introducing *even* and *odd* pages (the `book` class is one example) and flow rules based on them, but the next step, *bleeds*, combining page pairs into *spreads* as is often needed when designing image-oriented books is lacking. Full-bleed images/pages can be implemented manually, some more effort is needed to split something horizontally into a page spread, but doing so for whole book's worth of content is not practical – automating and experimenting with this process is how `photobook` began.

`photobook` extends the `book` class adding page layout types, bleeds and other global geometry configuration, introduces the *page* and *spread* as first-class elements into the document flow. These concepts are generalized as `cell`s. A `cell` is similar to a figure, it can be placed within the document flow, but unlike a figure a `cell` can be aligned relative to a page, it can fill a page, a cell can even be horizontally split to fill several pages (how spreads are implemented).

On top of the `cell`, *page*, and *spread* concepts, `photobook` also builds a set of configurable high level macros and templates for common use cases like full bleed image spreads, foldouts, ... etc.

Usage

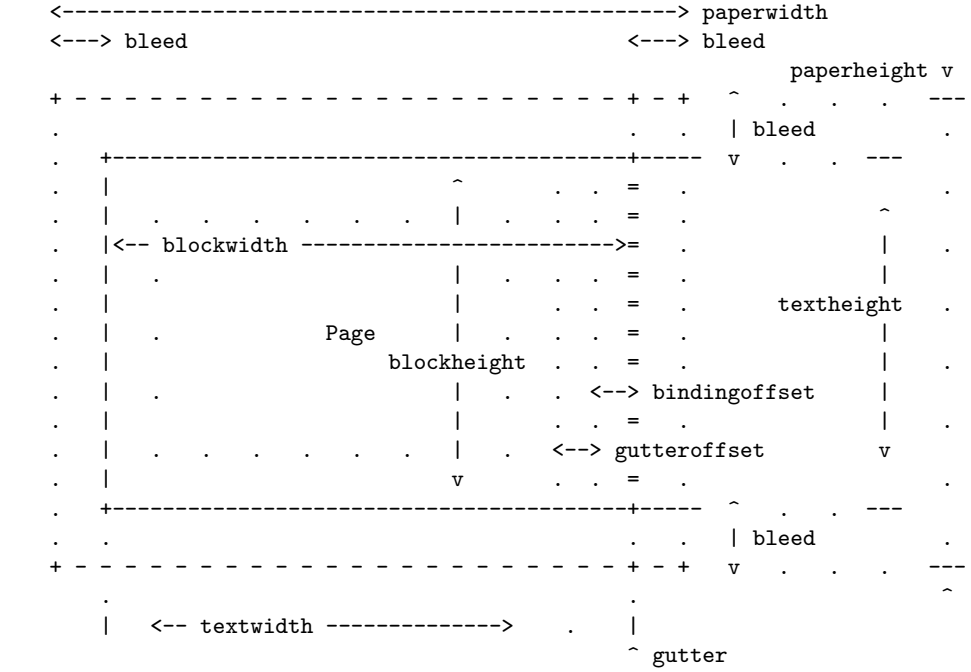
```
\documentclass[<options>]{photobook}
```

Options

Page geometry

blockwidth=<len>
blockheight=<len>
bindingoffset=<len>
gutteroffset=<len>
bleed=<len>

This is similar to what `geometry` does, but adds bleed support.



Note that all macros respect `\bindingoffset` but only some macros account for `\gutteroffset`, namely macros that do not display content with bleeds.

Also note that bleeds are included in geometry's `\paperwidth` and `\paperheight`.

`flatfold=<len>` Sets the clearance set aside for a flat fold, used for foldouts (see: Foldout page cells section).

`pagefold=<fold>` Sets the default fold type.

Can be `in` or `out`.

`foldout=<fold-spec>` Sets the default fold specification (i.e. sets `\defaultfoldout`).

For more information see: Foldout page cells section.

layout

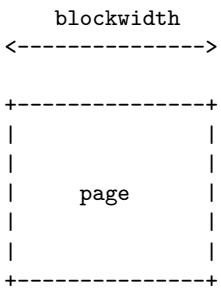
`layoutmode=<layout>` Sat page layout mode.

`block`
`spread`
`endpaper`
`cover`
`jacket`

```
layoutmode=<option>
```

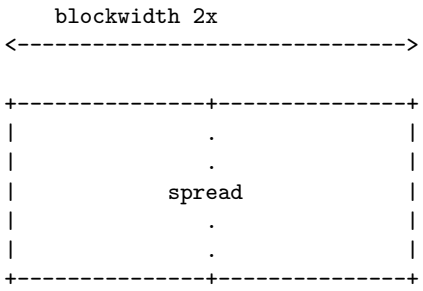
`block` (default)

Basic page layout.



`spread/endpaper`:

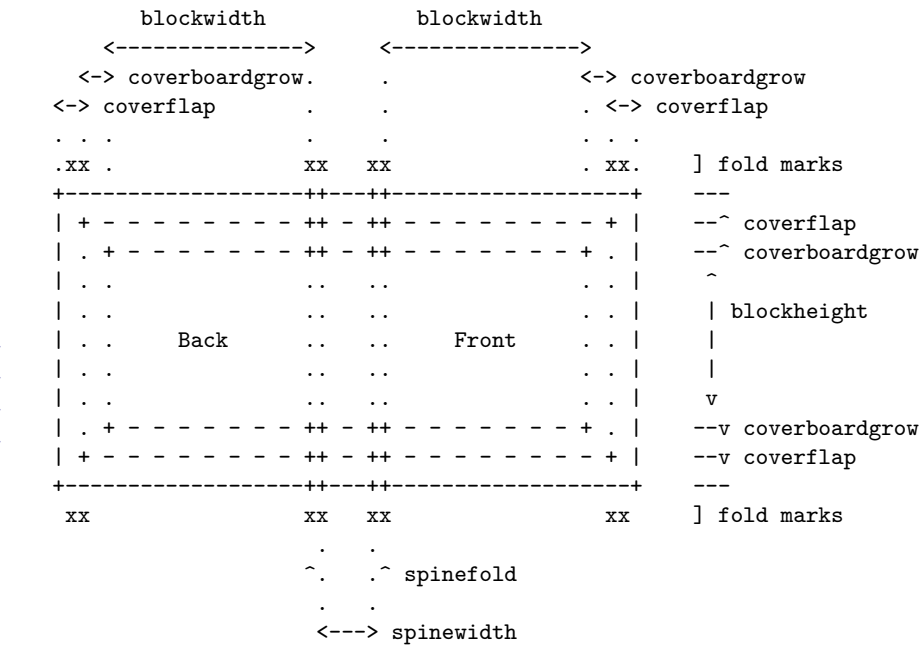
Spread/endpaper layout.



Note that `endpaper` is simply an alias to `spread`, this helps make the book source to be more semantic and readable.

`cover`

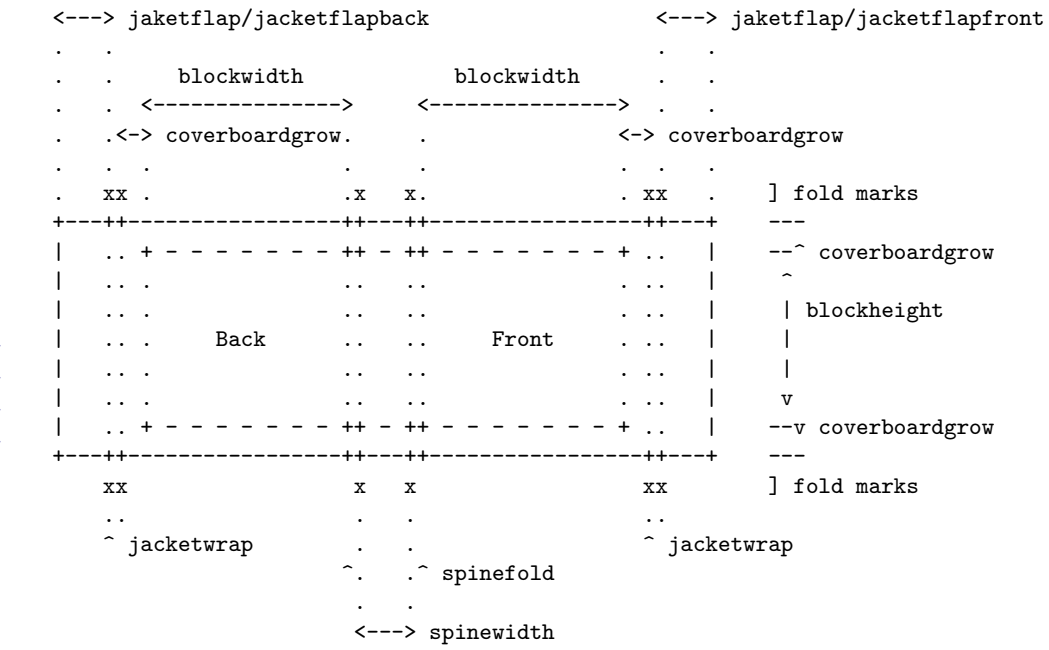
Book cover layout



coverboardgrow=<len>
coverflap=<len>
spinewidth=<len>
spinewidth=<len>

`jacket`

Dust jacket layout



jacketwrap=<len>
jacketflap=<len>
jacketflapfront=<len>
jacketflapback=<len>

Note that for `cover`, `spread/endpaper`, and `jacket`, fancyhdr's `\pagestyle{..}` is set to `empty` by default.

Image clearance

clearimage=<len> Distance from image to paper border (clearance) for full-page images.

this can be: - negative value set image bleed, - positive value set distance from paper edge to image.

Image block layout

imageblockwidth=<len> Image block size relative to text block.
imageblockheight=<len>
imageblockoffsettop=<len>

PDF Viewer layout

pdfpagelayout=<mode> Sets PDF page display mode.
SinglePage
OneColumn
TwoColumnLeft
TwoColumnRight
TwoPageLeft
TwoPageRight

Defaults to `TwoColumnLeft` for `layoutmode=block` and `SinglePage` for other modes.

See: hyperref's `pdfpagelayout` for more options.

Other options

nofoldmarks If given disable fold marks.

This sets `\iffoldmarks` to false, otherwise it is set to true.

foldmarksize=<len> Sets the fold mark size (default: 6mm).

geometrynodefaults let the user set geometry defaults.

If this is not set `photobook` will override some user settings when initializing geometry.

If set `photobook` will only set override:

```
paperwidth=\bleedblockwidth
paperheight=\bleedblockheight
bindingoffset=\bindingoffset
```

roundprintedlengthsto=<num> Number of digits to round printed lengths to (default: 1).

This is a shorthand to `numprint`'s `\nprounddigits{..}`, us it to change values mid-document if needed.

This is mostly used for `\GenerateTemplate`.

Packages

`photobook` adds and uses internally the following packages: geometry, kvoptions, calc, xargs, ifthen, iftex, pgffor, xint, xinttools, listofitems, xkeyval, etoolbox, atbegshi, hyperref, eso-pic, environ, numprint, trimclip, xcolor, pagecolor, colorspace, graphicx, adjustbox, adjustbox, fancyvrb, tikz, rotating, fancyhdr, and pdfpages.
Most of these packages can be used by the book author without explicitly importing them.

Globals

\layoutmode=<mode> Layout mode

\pdfpagelayout=<mode> Controls the default layout in the pdf viewer.

\spinewidth=<len> Spine width

\spinefold=<len> Spine fold width

\coverboardgrow=<len> Controls how much biger the cover board is than the page block

\coverflap=<len> Cover flap

\jacketwrap=<len> Jacker configuration

\jacketflap=<len>
\jacketflapfront=<len>
\jacketflapback=<len>
\blockwidth=<len>
\blockheight=<len>

Page block size

`\bleedblockwidth=<len>` Page block size with bleeds

`\bleedblockheight=<len>` These are equivalent to `\paperwidth` and `\paperheight` but are independent of them...

`\pageblockwidth=<len>` Original page block size

`\pageblockheight=<len>` for the block layout these are the same as `\blockwidth` and `\blockheight` for other layouts these are the original page layout size while the `\blockwidth` are set tho the current layout visible size.

`\bleed=<len>` Page bleed size

`\bindingoffset=<len>` Binding offset

Used to offset content to account glue-up/roughing of pages for certain types of binds like perfect binds.

`\gutteroffset=<len>` Gutter offsets

Mainly used to offset content without bleeds away from or into the gutter.

`\defaultfoldout=<fold-spec>` Set the default fold specification.

This can be set via the `foldout` class option.

For more information see: Foldout page cells section.

`\flatfold=<len>` Sets the amount of paper to account for a flat fold.

`\pagefold=<fold>` Default fold type.

This defaults to `out` for `jacket` and `cover` and to `in` for other layouts.

`\foldmarkoffset=<len>` Sets the offset of fold marks from outside of page edge (default: `0.5\bleed`).

Note that this can either be a command or a dimension.

`\foldmarksize=<len>` Sets fold marker size (vertical).

`\iffoldmarks` Disable/enable fold marks.

The default is true, to toggle on/off use `\foldmarkstrue` / `\foldmarksfalse`.

This can be set globally via the `nofoldmarks` class option.

`\clearimage=<len>` Image clearance

`\imageblockwidth=<ratio>` Image block width relative to `\textwidth`

`\imageblockheight=<ratio>` Root page text width/height.

`\pagetextwidth=<len>` The ration by which the image is raised in `\ImagePage{..}` and derived templates.

`\pagetextheight=<len>` Block caption clearance from edge of parent cell.

`\imageblockoffsettop=<ratio>` Used as default by `\captionblockcell{..}`

`\clearcaption=<len>` Block caption spacing from parent cell.

`\captioncellspacing=<len>` Used as default by `\captionblockcell{..}`

`\captionclearpage=<len>` The distance between a caption block on a template page and page borders (defaults to: `\clearimage`).

This is relevant only in page/spread templates.

`\cellparentwidth=<len>` Cell geometry.

`\cellparentheight=<len>` These are set automatically by cells, thus it is not recommended to edit them manually.

`\cellwidth=<len>`

`\cellheight=<len>`

`\celloffsetleft=<len>` For more details see the Cells section.

`\celloffsettop=<len>` Foldout geometry.

`\clearfoldoutgutter=<len>` For more info see Foldout page cells section.

`\clearfoldoutfold=<len>`

`\clearfoldoutedge=<len>`

Initialization

`\InitPages` Initialize page dimensions.

This is not intended for direct use.

`\ReInitPages` Reset and re-initialize page dimentions.

`\ChangeLayout` Change document layout.

```
\ChangeLayout{<layoutmode>}
```

This is a shorthand for `\def\layoutmode{<layoutmode>}` and then `\ReInitPages`, and as this is changing any other parameters that may affect the layout, this should be done last.

Generic commands

`\keywords{..}` Set pdf metadata keywords

```
\keywords{<keywords>}
```

`\subject{..}` Set pdf metadata subject

```
\subject{<subject>}
```

`\mindim{..}` Get min/max dimension.
`\maxdim{..}`

```
\mindim{A}{B}
```

```
\maxdim{A}{B}
```

`\emptypage{..}` Create an empty page.

```
\emptypage
```

```
\emptypage[<style>]
```

`<style>` is the style name as defined via `fancyhdr` (default: `empty`).

`\cleartoleftpage` Forces content to left page.

This is a companion to `\cleardoublepage`.

Environments and Cells

Generic

`page` Page environment.

This is mainly designed to wrap other cell environment described later.

Note that this may span more than one page if there is enough stuff packed in. Also note that items within the page environment are placed in the same way as in any normal page, for absolute placement use either specific cells/environments like `papercell` or more generic `call`.

`leftpage` EXPERIMENTAL

Cells

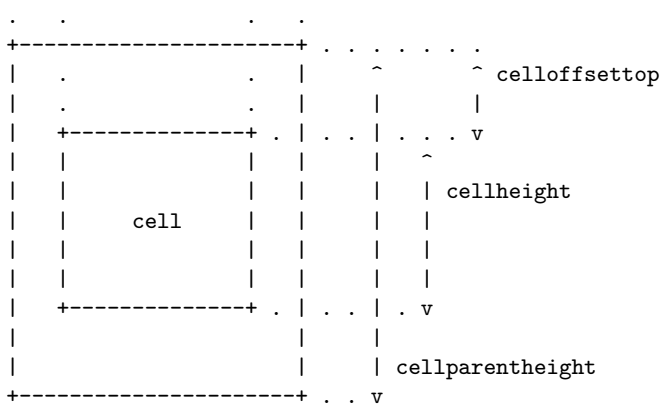
A cell environment is a box of specified size.

Cells can be both placed inline relative to other content or in an absolute location.

Note that absolute cells are placed relative to the page and currently can not be placed relative to other absolute cells (this might change in the future).

\cellwidth=<len>
\cellheight=<len>
\cellparentwidth=<len>
\cellparentheight=<len>
\celloffsettop=<len>
\celloffsetleft=<len>

A cell defines a set of contextual lengths:



For absolutely positioned cells these define the cell offset from parent.

The bare page can be reasonably treated as a cell.

Initally, outside of any cells `\cellwidth`, `\cellparentwidth` and `\cellheight`, `\cellparentheight` are equal to `\paperwidth` and `\paperheight` respectively, and `\celloffsettop` and `\celloffsetleft` are set to `0pt`.

Changing these is not recommended, it likely will not affect the current cell but can mess up nested cells.

inlinecell
inlinecell*

Create a basic inline cell.

```
\begin{inlinecell}{<width>}{<height>} ... \end{inlinecell}

\begin{inlinecell}[<valign>]{<width>}{<height>} ... \end{inlinecell}
```

This will clip oversized content.

`inlinecell*` is like `inlinecell` but will not clip.

```
\begin{inlinecell*}{<width>}{<height>} ... \end{inlinecell*}

\begin{inlinecell*}[<valign>]{<width>}{<height>} ... \end{inlinecell*}
```

`<valign>` can be one of `t` (default) for top, `c` for center or `b` for bottom.

These are just like `minipage` but provide cell mechanics.

minipagecell

EXPERIMENTAL

minipagecell*
zinlinecell

EXPERIMENTAL

zinlinecell*

Like `inlinecell` / `inlinecell*` but will take up zero space and sized to `\cellwidth` x `\cellheight`.

```
\begin{zinlinecell} ... \end{zinlinecell}
```

```
\begin{zinlinecell}[<valign>] ... \end{zinlinecell}
```

cell
cell*

Create a basic absolutely positioned cell.

```
\begin{cell}{<left>, <top>}{<width>}{<height>} ... \end{cell}
```

Oversized content will be clipped.

`cell*` is just like `cell` but will not clip its content.

```
\begin{cell*}{<left>, <top>}{<width>}{<height>} ... \end{cell*}
```

`cell` and `cell*` are absolutely positioned either relative to the current page or to the closest savecell.

adjustcell
adjustcell*

EXPERIMENTAL

This a cell wrapper for `adjustbox`.

Save cells

\savecell{..}
\gsavecell{..}

Create a saved cell.

```
\savecell{<name>}{<width>}{<height>}{ .. }
```

This is similar to `\newsavebox{..}` and `\sbox{..}` but adds cell functionality.

`\gsavecell{..}` is the same as `\savecell{..}` but creates a global cell.

Note that both `\gsavecell{..}` and `\savecell{..}` make the nested `cell` and `cell*` position relative to the cell and not the page. This is done by setting `\TPoptions{absolute=false}` for the cell content which will also affect `textpos`'s macros.

\usecell{..}
\usecell*{..}

Use part of a saved cell.

```
\usecell{<name>}(<top>, <left>){<width>}{<height>}
```

This will clip the content to cell.

`\usecell*{..}` is similar to `\usecell{..}` but will not clip the cell content.

```
\usecell*{<name>}(<top>, <left>){<width>}{<height>}
```

These are similar to `\usebox{..}`.

Cell macros

Cell macros require a cell environment to function correctly.

topdown
bottomup

Rotate cell content vertically, orienting it top-down or bottom-up.

```
\begin{topdown} ... \end{topdown}
```

```
\begin{bottomup} ... \end{bottomup}
```

cliptocell
setcliptocellbleeds

Clip content to parent cell.

```
\begin{cliptocell} ... \end{cliptocell}
```

Clip content to cell plus offsets (bleeds) from each side

```
\begin{cliptocell}[<size>] ... \end{cliptocell}

\begin{cliptocell}[<horizontal> <vertical>] ... \end{cliptocell}

\begin{cliptocell}[<left> <bottom> <right> <top>] ... \end{cliptocell}
```

Offset order, i.e. left-bottom-right-top is made consistent with graphicx and trimclip modules and represents the bottom-left and top-right viewport point offsets from the parent cell.

`cliptocell` does not affect the cell content positioning in any way.

This is designed to simplify filling template cells and adding bleeds to certain sides and clipping flush to others, for example for cells of a dust jacket or a foldout, but `cliptocell` is not limited to this one use-case.

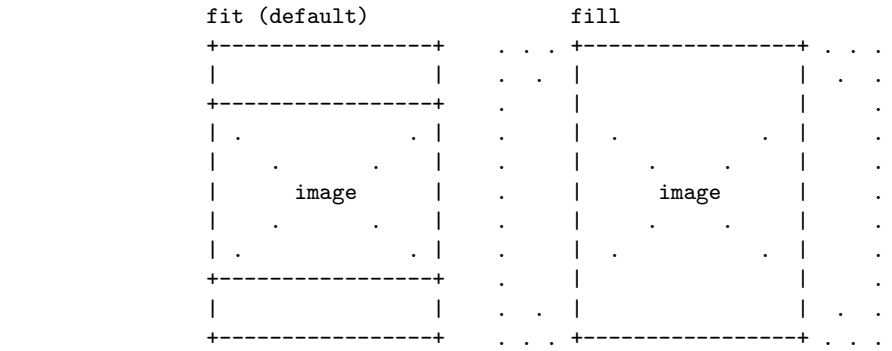
Set the default bleeds for `cliptocell` environments

```
\setcliptocellbleeds{<size>}

\setcliptocellbleeds{<horizontal> <vertical>}

\setcliptocellbleeds{<left> <bottom> <right> <top>}
```

This will only affect `cliptocell` environments on the same level, without affecting the nested `cliptocell`s.

`\imagecell{..}` Place image in cell.

```
\imagecell{<caption-cell>}{<image>}

\imagecell[<key>=<value>, ..]{<caption-cell>}{<image>}

\imagecell[fit]{}{<image>}

\imagecell[fill]{}{<image>}
```

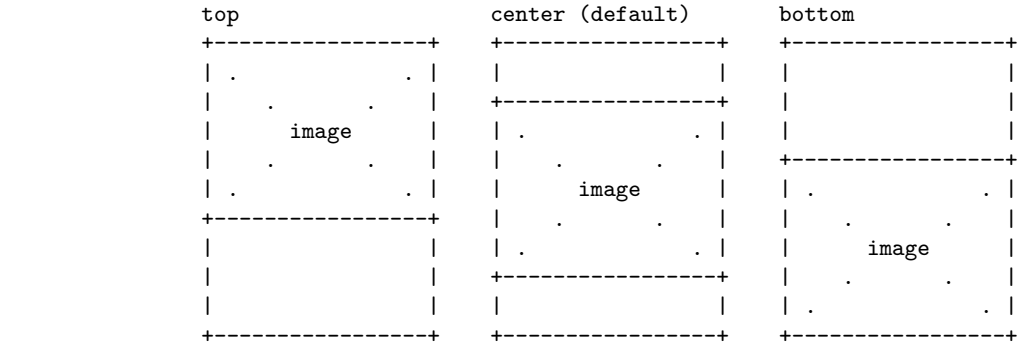


Centering. The image will be centered by default.

```
\imagecell{}{<image>}

\imagecell[center]{}{<image>}
```

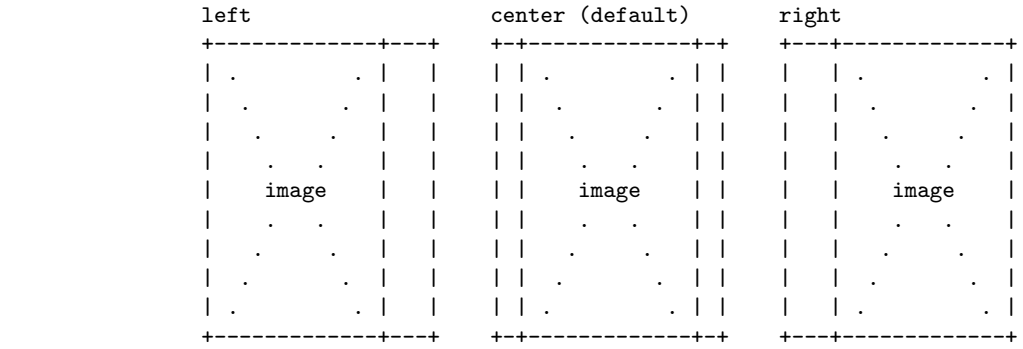
Vertical alignment

```
\imagecell[top]{}{<image>}

\imagecell[bottom]{}{<image>}
```



Horizontal alignment

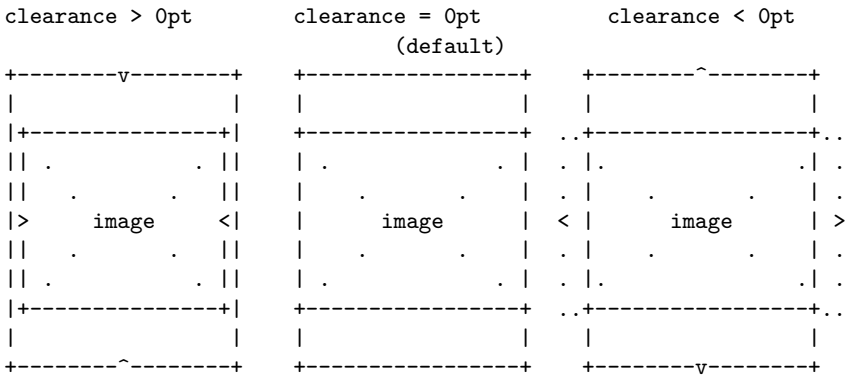
```
\imagecell[left]{}{<image>}

\imagecell[right]{}{<image>}
```



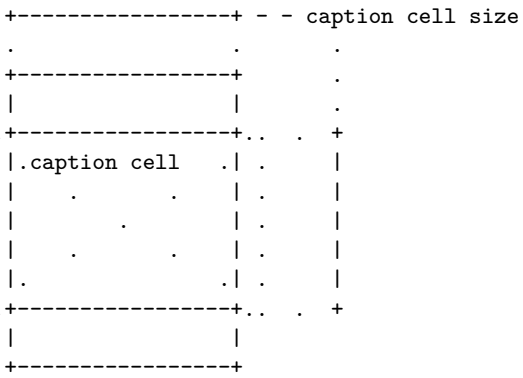
Horizontal and vertical alignment can be combined to control alignment of both vertical and horizontal images at the same time.

Image `clearance`. This sets the amount of clearance around an image (default: `0pt`).

```
\imagecell[clearance=<size>]{}{<image>}

\imagecell[clearance=<horizontal> <vertical>]{}{<image>}

\imagecell[clearance=<left> <bottom> <top> <right>]{}{<image>}
```



Note that if `clearance` is less than 0, the image will take up more space than the containing cell, `\imagecell{..}` will not clip its content and the whole image surface will be shown. If clipping is needed then use `clipcell` environment as a container.



`<caption-cell>` occupies the same space as the image clipped by the containing cell and provides all the cell functionality.

If `captionclearparent` is set, the `<caption-cell>` will fit into an intersection between the image area and the parent cell padded by `captionclearparent`.

This will write image path, page and size to `\jobname`.image-list, this is useful for final image pre-press.

To disable image list set:

```
\writeimagelistfalse
```

`\captioncell{..}` Caption cell

Placement:

```
\captioncell[<position>]{<caption>}
```

```
  over
+-----------------+
| top             |
|                 |
|                 |
|                 |
| center          |
|                 |
|                 |
|                 |
| bottom          |
+-----------------+
  under
```

Default:

```
\captioncell{<caption>}

\captioncell[top]{<caption>}

\captioncell[center]{<caption>}

\captioncell[bottom]{<caption>}

\captioncell[over]{<caption>}

\captioncell[under]{<caption>}
```

Horizontal alignment

```
\captioncell[align=<mode>]{<caption>}
```

```
+-----------------+    +-----------------+    +-----------------+
|flushleft        |    |     center      |    |       flushright|
|                 |    |                 |    |                 |
|                 |    |                 |    |                 |
```

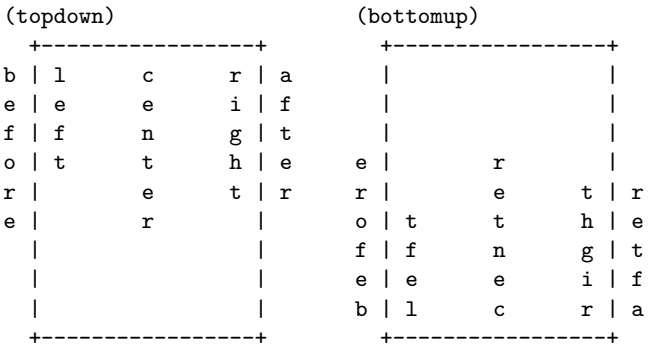
Default:

```
\captioncell{<caption>}

\captioncell[align=flushleft]{<caption>}

\captioncell[align=center]{<caption>}

\captioncell[align=flushright]{<caption>}
```

Note that a caption cell does not take up any space in the parent cell so multiple captions can be used in combination with other elements.

Note that caption cells currently do not play well with other content in the same cell that takes up space, e.g. text, pictures, ...etc. Captions are mainly suited to play well with image cells.

`\vcaptioncell{..}` Vertical caption cell

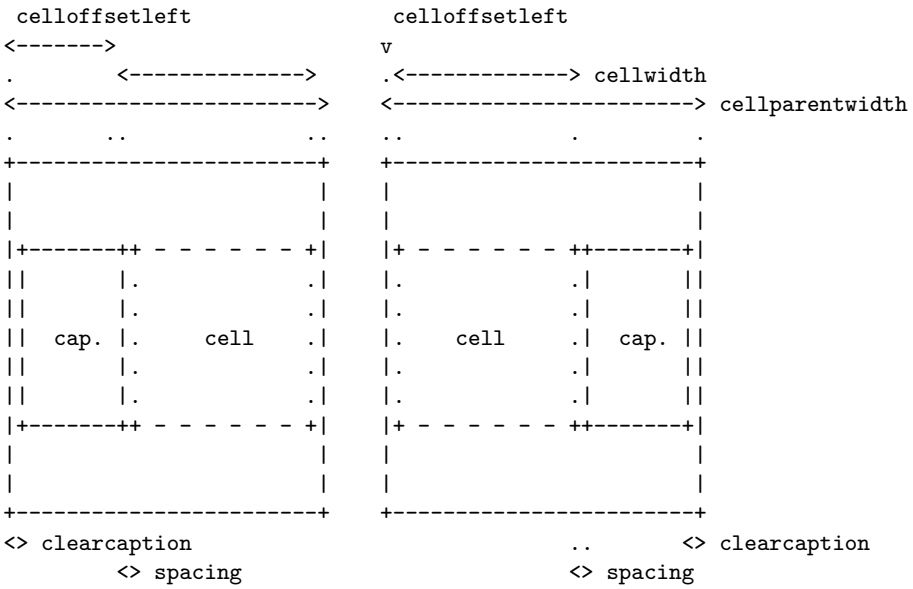


See samples for better illustration.

`\rcaptioncell{..}` EXPERIMENTAL

Upside-down caption cell.

`\captionblockcell{..}` Add caption into a box left/right of current cell.



Paper cells

Paper and page cells are very similar but differ in one aspect – paper cells represent the paper as indicated by `layoutmode=<layout>` while page cells are always the size of the page block, i.e. `layoutmode=block`.

`papercell` Paper cell.

This does not include bleeds.

`paperbleedcell` Paper bleed cell.

Like `papercell` but includes bleeds.

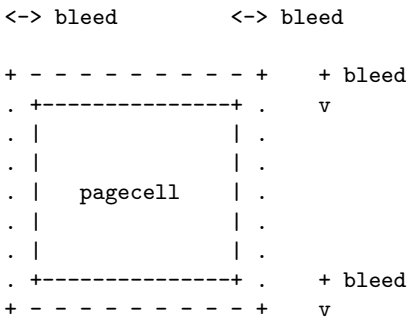
Page cells

Page cells always correspond to the page block size, i.e. `layoutmode=block` regardless of the actual `layoutmode` set currently.

`pagecell` Page cell.

`pagecell*`

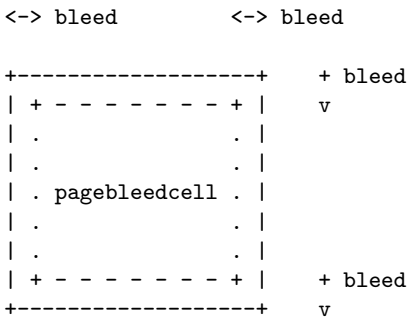
This corresponds to the visible page in the `layoutmode=block` and does not include bleeds.



The star version accounts for `\gutteroffset`.

Note that `layoutmode`'s other than block will change the paper size but will not affect this either in size (block size) or in position (top-left).

`pagebleedcell` Like `page` but includes bleeds.

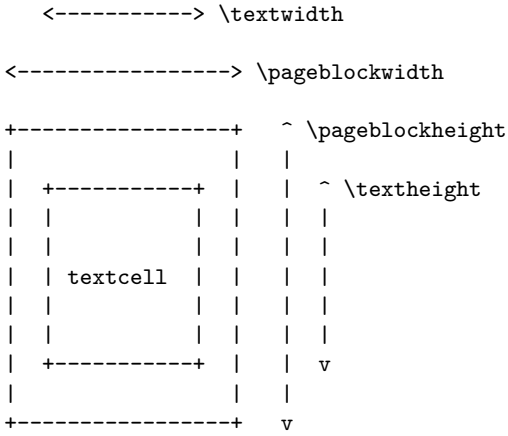


Note that as with `pagecell` this is not affected by `layoutmode`.

`textcell` EXPERIMENTAL

A cell taking up the page text block.

```
\begin{textcell}{<width>}{<height>} ... \end{textcell}
```



Note that this is an inline cell and if something is on the page it may not be centered properly.

Foldout page cells

EXPERIMENTAL

A foldout is a special case cell that changes the page format.

In addition to cell lengths a foldout defines a set of additional lengths and lists within its context.

`\pagefoldpanels=<list>` Lists page folded panel widths and fold types respectively.

`\pagefoldpanelfolds=<list>`

`pagefoldpanels` is set automatically for layouts other than `block` and for foldout pages, changing this can mess up fold markers and page cell placement.

`pagefoldpanelfolds` is generated when creating a foldout.

See: listofitems for list API.

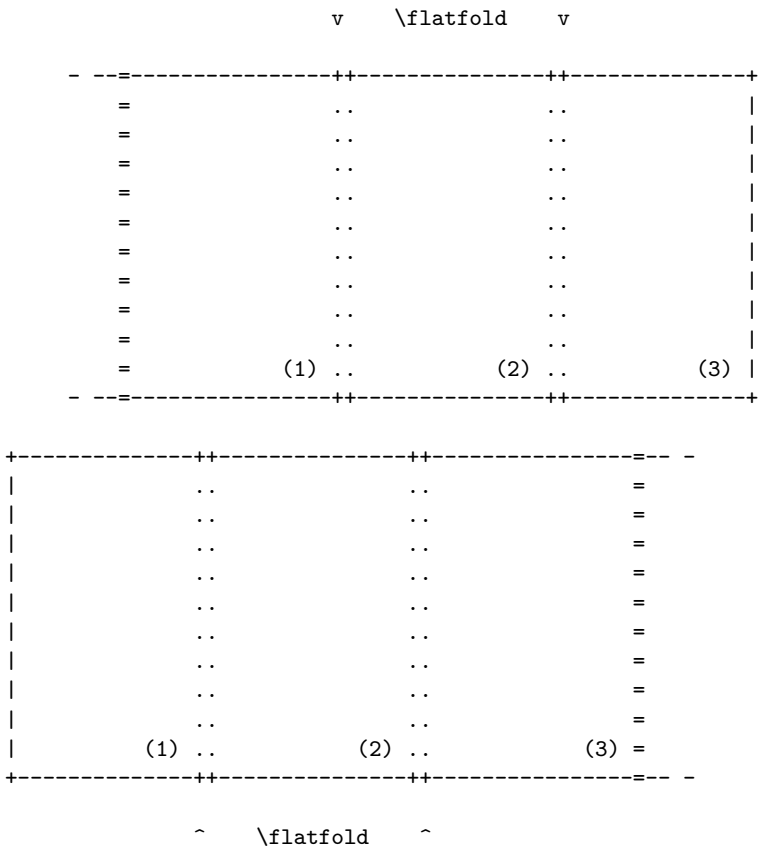
`\foldoutwidth=<len>` Total foldout page width, calculated when a foldout page is created.

`foldout` Create 2 or more pages in foldout mode.

```
\begin{foldout}[<fold-spec>] ... \end{foldout}
```

`<fold-spec>` can be a number of folds, or contain one or more fold types (`in` or `out`), two or more panel widths (dimensions) or a sequence of both (i.e. dim fold dim ...) where missing widths are automatically calculated and missing folds are the same as the previous fold or the default if no folds are specified.

If no `<fold-spec>` is given, then the value of `\defaultfoldout` is used (default: `2`).

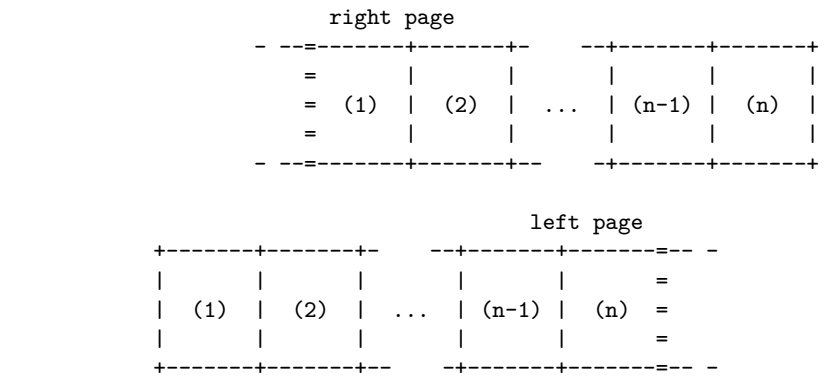


A foldout always starts on the right/odd page and will always end on the left/even page.

If `\foldoutpages{..}` is called on an even page, an empty normal page will be created pushing the foldout to the right page of the spread. If an odd number of pages is

created the set will be padded with an empty page before `\foldoutpages{..}` exits.

Foldout panel numbering

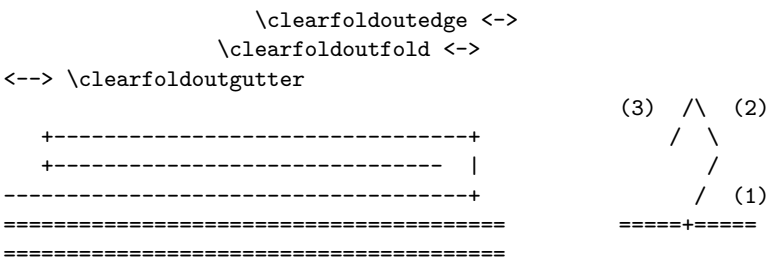


Formal `<fold-spec>` grammar:

```
<fold-spec> ::
    <count>
    | <fold>
    | <panels>
<panels> ::=
    <panel>
    | <panel> <panels>
<panel> ::=
    <width> <fold>
    | <width>
    | <fold>
<fold> ::=
   in | out
```

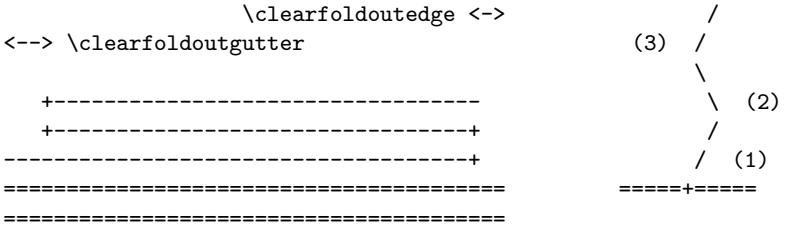
`\clearfoldoutgutter=<len>`
`\clearfoldoutedge=<len>`
`\clearfoldoutfold=<len>`

Fold panel sizing for similar fold sequence:



Note that `in in` fold sequence is drawn, `out out` is identical in sizing but reflectd.

Fold panel sizing for dissimilar fold sequence:



`in out` is drawn, `out it` is the same but reflected.

The size of the paper fold is set by `\flatfold`.

foldoutcell
foldoutcell*

Create a cell spanning one or more foldout panels.

```
\begin{foldoutcell}[<num>] ... \end{foldoutcell}

\begin{foldoutcell}[<from>-<to>] ... \end{foldoutcell}
```

If no arguments are given this will create a panel at `foldoutpanel` counter and increment it by one.

```
\begin{foldoutcell} ... \end{foldoutcell}
```

This will also auto-advance the page when all panels are filled.

Note that manual panel placement has no effect on the panel counter thus care must be taken when mixing manual and auto-advanced panels. Also note that `foldoutpanel`'s value is not maintained within manually placed panels and should be treated as undefined.

`foldoutcell*` is the same as the non-star version but creates cells including bleeds.

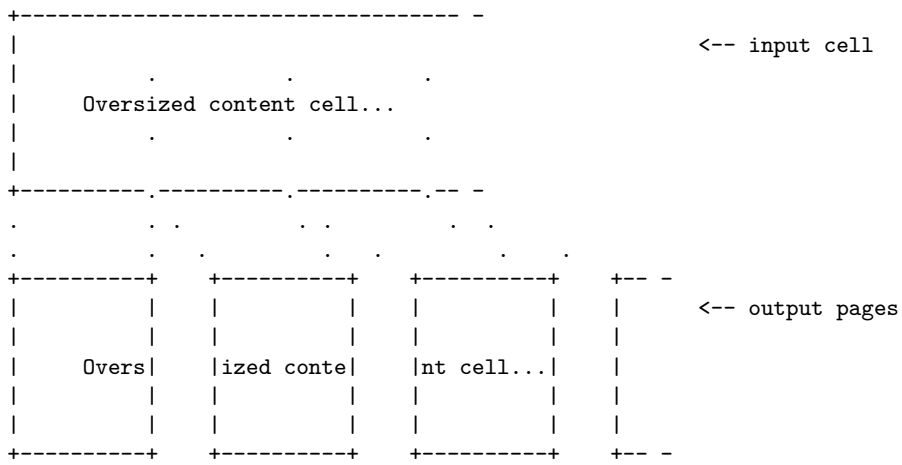
Neither version clips its content, to explicitly clip use the `cliptocell` environment.

These can only be used from within a `foldout` cell.

Multi-page cells

spreadtopages
spreadtopages*
\usespreadpage{..}
\usespreadpage*{..}

Spread cell into pages.



`spreadtopages` creates a cell and spreads it into pages right away.

```
\begin{spreadtopages} .. \end{spreadtopages}

\begin{spreadtopages}[<page-count>] .. \end{spreadtopages}
```

`spreadtopages*` creates a named save cell only allowing the pages to be placed manually via `\usespreadpage{..}`.

```
\begin{spreadtopages*}{<name>} .. \end{spreadtopages*}

\begin{spreadtopages*}[<page-count>]{<name>} .. \end{spreadtopages*}
```

`\usespreadpage{..}` places a page from a saved cell.

```
\usespreadpage{<name>}
```

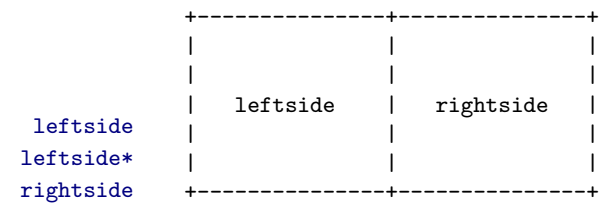
```
\usespreadpage[<page-num>]{<name>}
```

`\usespreadpage*{..}` is the same as the non-star version but will not use a `page` environment, enabling the user to populate the page with other elements.

Page numbers are 1-based.

Note that the cell created by `spreadtopages*` env is a normal save cell and can be manipulated via `\usecell{..}` and `\usecell*{..}`.

Endpaper cells

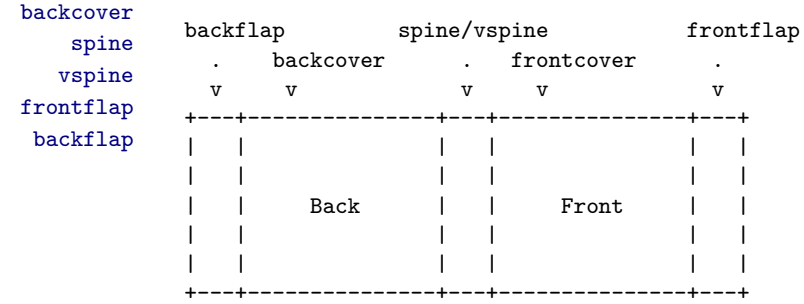


The non-star versions will clip to bleeds except for the gutter side that has no bleeds.

The star versions will not clip.

Cover and dust jacket cells

`frontcover` `backcover` `spine` `vspine` `frontflap` `backflap` Covers and dust jackets differ only in that covers do not usually have flaps.



Note that when typesetting a spine with both top-down and bottom-up text it is recommended to use `topdown`/`bottomup` in `zinlinecell`'s in a normal `spine` rather than using a `vspine` and trying to rotate part of the content.

Caption Templates

`\captionsize{..}` Defines the caption font.

Can be redefined to control caption font/style.

`\captionformat{..}` General caption format.

```
\captionformat{<code>}
```

This can be redefined to control the image caption.

Page Templates

This section contains a set of predefined configurable single-page templates.

Note that most page templates do not `\clearpage` before expanding, this enables one to prepend elements (like pdf comments), if clearing a page is required add `\clearpage` manually before the template.

Basic book information

`BookType=<text>` Used in page macro text to indicate the type of document built, by default this is set to `book` for the most common case but can be set to anything (e.g. `brochure`, `magazine`, `presentation`, ...etc.)

`BookTitle=<text>` These provide the default information used by the `\BookInfoPage` and `\BookSoftwareInfoPage`.

`BookVersion=<text>`
`BookAuthors=<text>`
`BookYear=<text>`
`ByNotice=<text>`
`ThanksTo=<text>`
`ISBN=<text>`
`BookEdition=<text>`
`License=<text>`
`CopyrightNotice=<text>`
`OtherSoftware=<text>`
`BookFonts=<text>`
`SoftwareNotice=<text>`
`\BookInfoPage` Generate book information page.

```
+---------------+
|               |
|               |
|               |
| copyright     |
| ISBN          |
| info          |
+---------------+
```

This page is *usually* included near the start of the book, before any of the logical sections of the book start, *usually* just after the title pages but before any of the epigraphs, forewords, TOCs and prefaces. This can also in some cases be pushed to the rear of the book.

`\BookSoftwareInfoPage` Generate software info page.

```
+---------------+
|               |
|               |
|               |
|               |
|               |
| notice        |
+---------------+
```

This page if present is usually placed at the very rear of the book.

`\BookFullInfoPage` This page combines the `\BookInfoPage` and `\BookSoftwareInfoPage` information into a single page.

Tweaking

Most page/spread templates provide an ability to externally ”tweak” some image proportions.

This is an alternative means to setting template properties, e.g:

```
\tweakimagescale{0.8}
\ImagePage{image}
```

Is equivalent to:

```
\ImagePage[scale=0.8]{image}
```

Tweaks get reset after each template.

`\imagescale=<num>` Tweak next image templates scale/offsettop/offsetleft

`\imageoffsettop=<len>` Note that `\imagescale` is not a length.

`\imageoffsetleft=<len>`

```
\edef\imagescale{<scale>}

\setlength\imageoffsettop{<len>}

\setlength\imageoffsetleft{<len>}
```

`\tweakimagescale{..}` Convenience commands

`\tweakimageoffsettop{..}` These provide a uniform interface for tweaking.

`\tweakimageoffsetleft{..}`

```
\tweakimagescale{<scale>}

\tweakimageoffsettop{<len>}

\tweakimageoffsetleft{<len>}
```

`\resetimagetweaks{..}` Reset tweaked values.

General interface

A templates provide a uniform interface consisting of several commands:

Base template command:

```
\<template-name>{<caption>}{<image>}

\<template-name>[<options>]{<caption>}{<image>}
```

`<options>` is the same as for `\imagecell{..}`.

Template-specific caption command:

```
\<template-name>Caption{<caption>}
```

This can be redefined to control typesetting the caption for all consecutive templates.

Reset template caption to default:

```
\reset<template-name>Caption
```

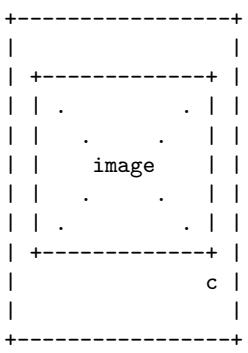
An equivalent to the non-star version but use the caption as-is:

```
\<template-name>*{<caption>}{<image>}

\<template-name>*[<options>]{<caption>}{<image>}
```

Note that thogh some template versions differ only in `<options>` defaults passed to `\imagecell{..}`, they are split to provide ability to configure the template defaults separately.

No bleed

\ImagePage{..} Basic image page
\ImagePageCaption{..}
\resetImagePageCaption
\ImagePage*{..}

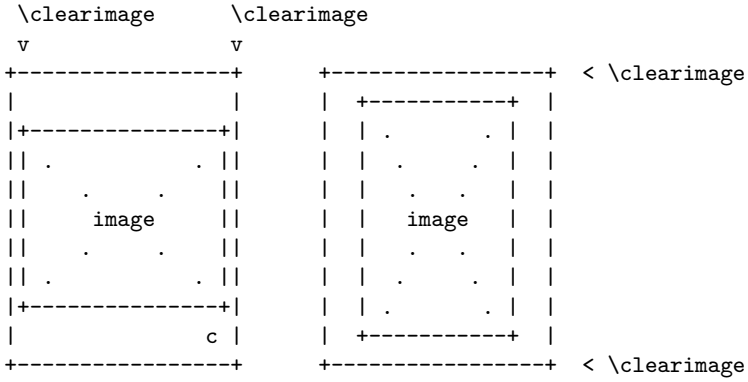


Create an image page:

Note that `\ImagePage{..}` only supports tweaking `scale`, this is intentional ad tweaking offset will mess up page alignment relative to other pages in a book.

\ImagePageClear{..}
\ImagePageClearCaption{..}
\resetImagePageClearCaption
\ImagePageClear*{..}

```
\ImagePageClear[<options>]{<caption>}{<image>}
```

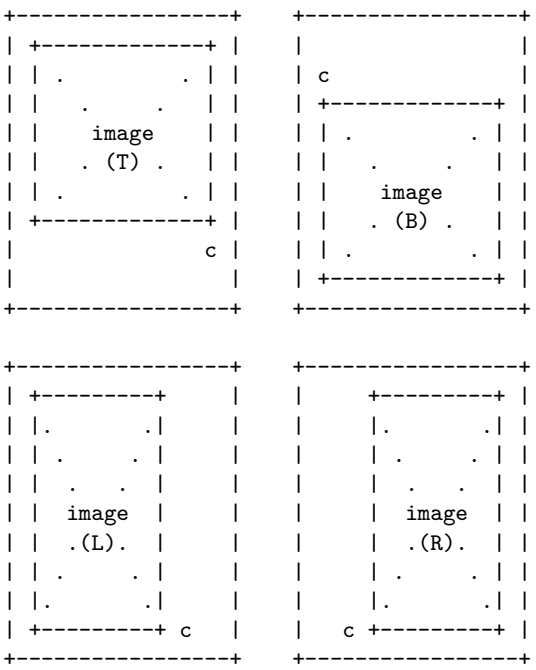


Default image clearance is set by `\clearimage` global length.

This respects `\bindingoffset`.

\ImagePageClear<D>{..}
\ImagePageClear<D>Caption{..}
\resetImagePageClear<D>Caption
\ImagePageClear<D>*{..}

`<D>` can be one of T, B, L or R, for top, bottom, left and right respectively.



Partial bleed

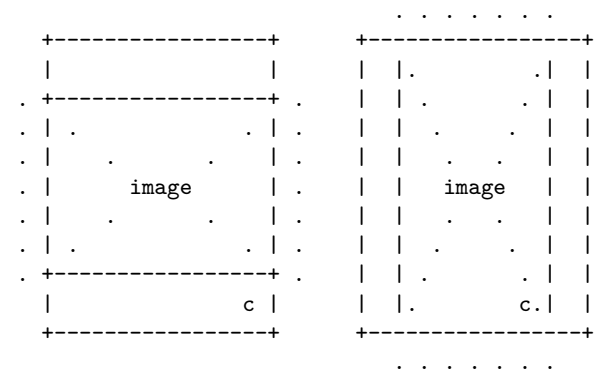
`\ImagePageFit{..}` Fit image into page...

`\ImagePageFitCaption{..}`

`\resetImagePageFitCaption`

`\ImagePageFit*{..}`

```
\ImagePageFit[<options>]{<caption>}{<image>}
```



Default image clearance is set by `\bleed` global length.

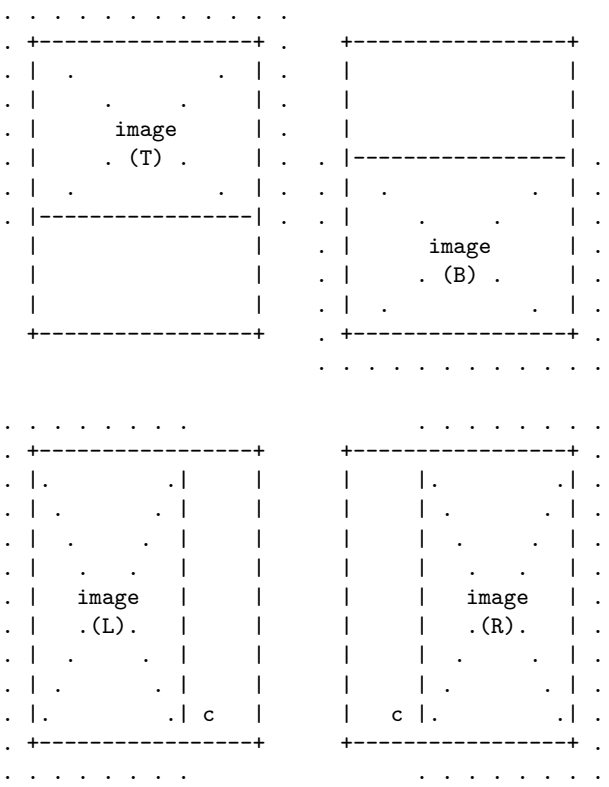
If centered image fits vertically this will account for `\bindingoffset`.

`\ImagePageFit<D>{..}` `<D>` can be one of T, B, L or R, for top, bottom, left and right respectively.

`\ImagePageFit<D>Caption{..}`

`\resetImagePageFit<D>Caption`

`\ImagePageFit<D>*{..}`



Full bleed

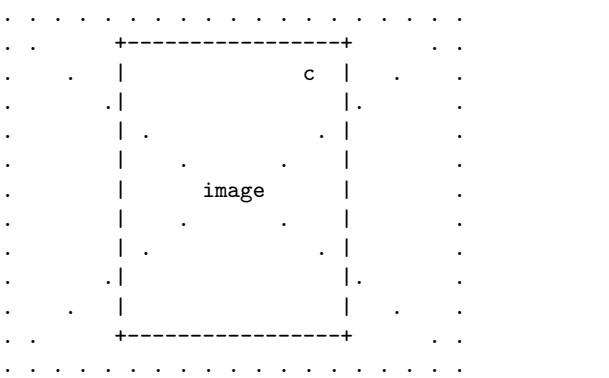
`\ImagePageFill{..}` Like `\ImagePage` but will fill page with image.

`\ImagePageFillCaption{..}`

`\resetImagePageFillCaption`

`\ImagePageFill*{..}`

```
\ImagePageFill[<options>]{<caption>}{<image>}
```



Default image clearance is set by `\bleed` global length.

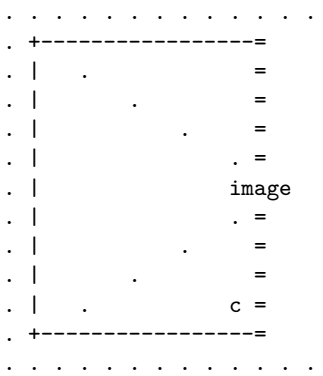
`\ImageHalfPageL{..}` Left half of image filling page with full bleed.

`\ImageHalfPageLCaption{..}`

`\resetImageHalfPageLCaption`

`\ImageHalfPageL*{..}`

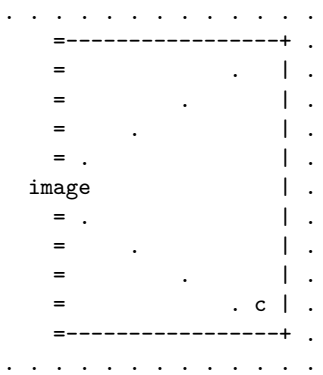
```
\imageleftspreadfullbleed[<vertical-offset>]{<caption>}{<image>}
```



\ImageHalfPageR{..}
\ImageHalfPageRCaption{..}
\resetImageHalfPageRCaption
\ImageHalfPageR*{..}

Right half of image filling page with full bleed.

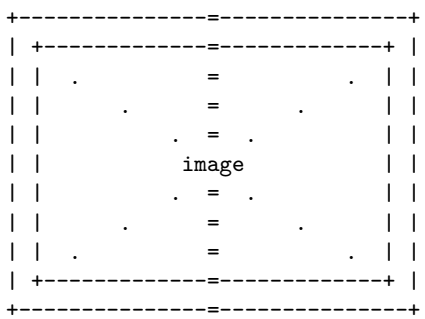
```
\imagerightspreadfullbleed[<vertical-offset>]{<caption>}{<image>}
```



Spread Templates

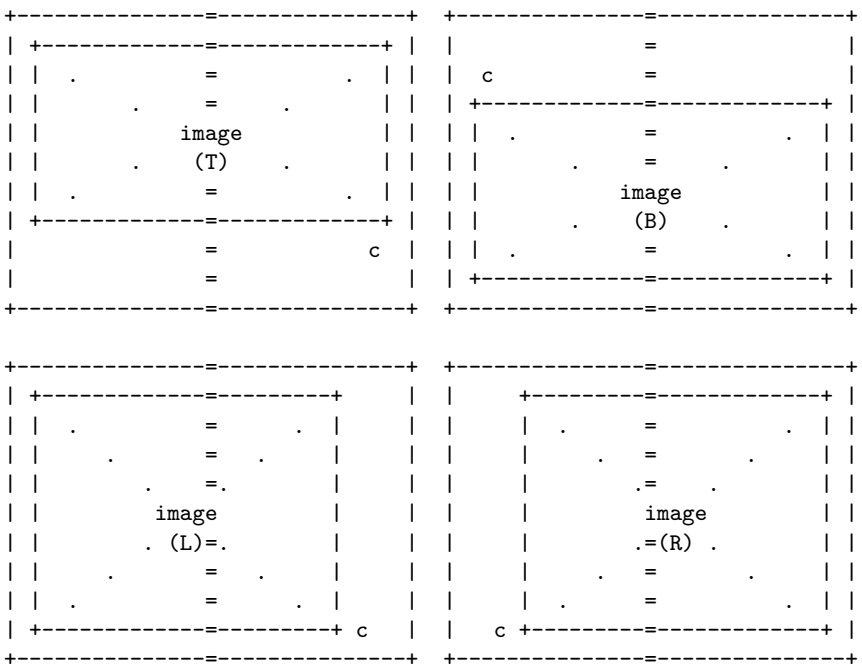
No bleed

\ImageSpread{..}
\ImageSpreadCaption{..}
\resetImageSpreadCaption
\ImageSpread*{..}

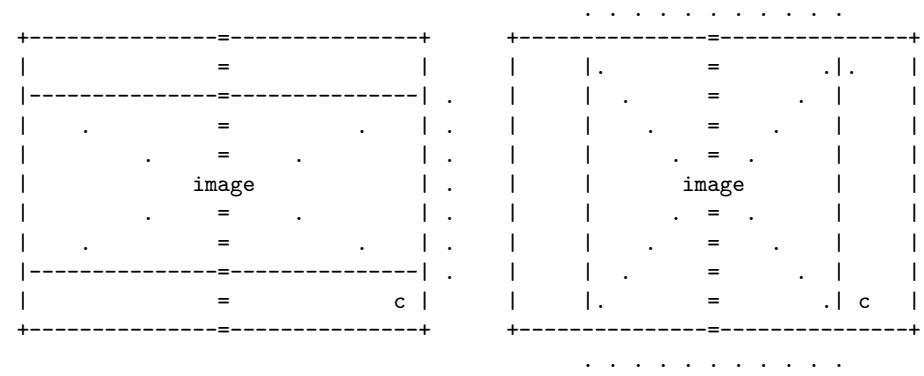


\ImageSpread<D>{..}
\ImageSpread<D>Caption{..}
\resetImageSpread<D>Caption
\ImageSpread<D>*{..}

<D> can be one of T, B, L or R, for top, bottom, left and right respectively.

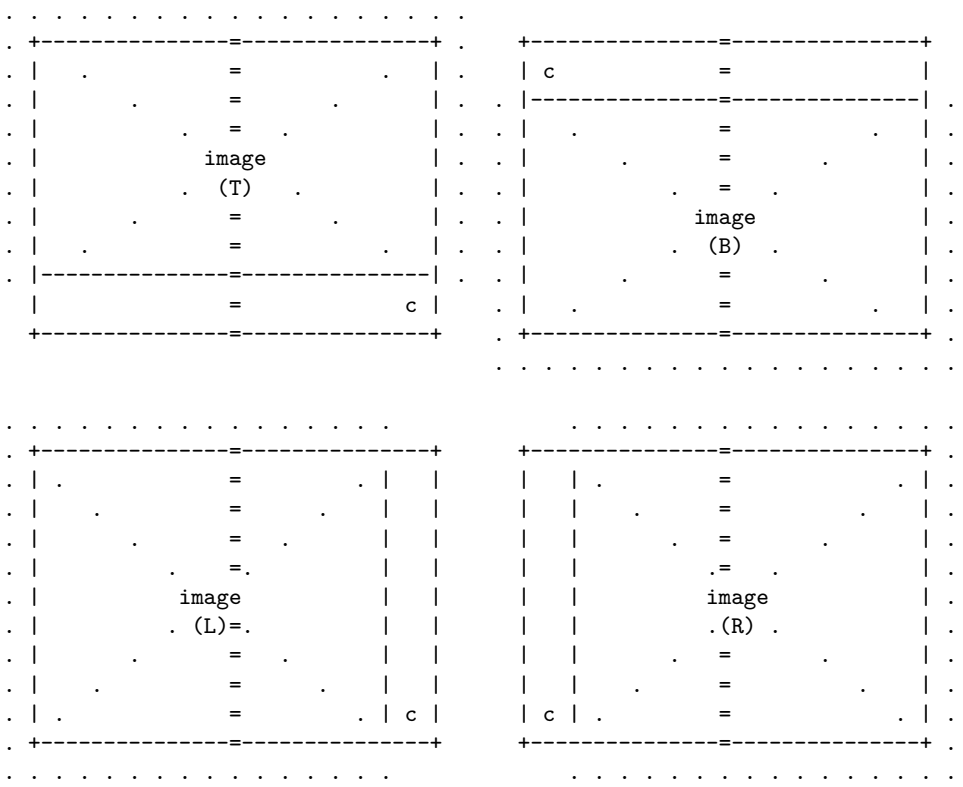


Partial bleed



\ImageSpreadFit{..}
\ImageSpreadFitCaption{..}
\resetImageSpreadFitCaption
\ImageSpreadFit*{..}

\ImageSpread<D>{..} Image spread aligned left (with bleed).
\ImageSpread<D>Caption{..}
\resetImageSpread<D>Caption \ImageSpread<D>[<options>]{<caption>}{<image>}
\ImageSpread<D>*{..}

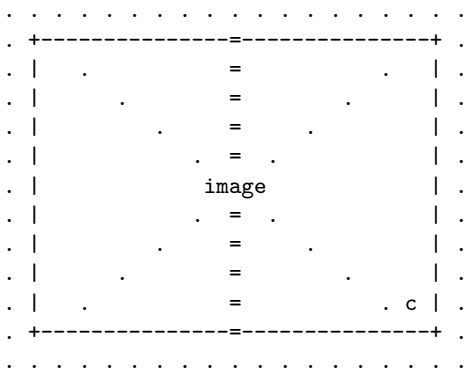


Full bleed

`\ImageSpreadFill{..}`
`\ImageSpreadFillCaption{..}`
`\resetImageSpreadFillCaption`
`\ImageSpreadFill*{..}`

Image spread with full bleed.

`\ImageSpreadFill[<options>]{<caption>}{<image>}`



Meta Macros / Environments

These macros and environments are used internally to help define cells and templates and thus can be useful when extending the functionality of this class both when authoring styles and when creating user macros/templates.[1]

[1] Note that in code the relevant meta-macros are defined in the logical locations for each macro, i.e. before first use and at the start of the relevant context. This section in the reference is placed near the end so as to focus the user's attention on the main class interface, class internal architecture and extension API being secondary.

`\ResettableMacro{..}` Create a resettable command.

```
\ResettableMacro{<name>}{<code>}

\ResettableMacro{<name>}[<arg-count>]{<code>}

\ResettableMacro{<name>}[<arg-count>][<default>]{<code>}
```

This is similar to `\newcommand{..}` but will define two commands:

```
\<name>{..}

\reset<name>
```

`\<name>{..}` can be freely redefined or undefined by user.

`\reset<name>` will reset `\<name>{..}` to its original state.

`\CellContentOptions{..}` EXPERIMENTAL

Define standard cell options.

`CellContent` EXPERIMENTAL

`CellContent*`

Standard cell content wrapper, used by `inlinecell`.

```
\begin{CellContent}[<parent-align>]{<name>} ... \end{CellContent}

\begin{CellContent*}[<parent-align>]{<name>}{<width>}{<height>} ... \end{CellContent}
```

The star version requires width/height to be manually passed while the non-star version will get the values from `\cellwidth` and `\cellheight` respectively.

`MinipageCellContent` EXPERIMENTAL

`MinipageCellContent*`

Like `CellContent` / `CellContent*` but will use `minipage` as the wrapper.

```
\begin{MinipageCellContent}[<parent-align>]{<name>} ... \end{MinipageCellContent}

\begin{MinipageCellContent*}[<parent-align>]{<name>}{<width>}{<height>} ... \end{MinipageCell
```

`\ImagePageTemplate{..}`

```
\ImagePageTemplate{<name>}{<code>}
```

This will define two commands:

```
\<name>[<options>]{<caption>}{<code>}

\<name>*[<options>]{<caption>}{<code>}
```

`\<name>{..}` will use `\<name>Caption{..}` template command to typeset the image caption while `\<name>*{..}` will show the caption as-is.

Miscellaneous

`\PageInfo` Display basic paper / page / cell geometry.

`\GenerateTemplate{..}` Generate template page for current `layoutmode`.

```
\GenerateTemplate
```

Cell size can be printed in `mm` (default) or in any explicit unit supported by LaTeX.

```
\GenerateTemplate{<unit>}
```

This can be useful if one needs to make the cover/jacket/... in either a different software package or by hand.

This is a no-op for `layoutmode=block`.

To change text color set `\textcolor{<color>}` the usual way and for line and fill colors use `\textblockrulecolour{<color>}` and `\textblockcolour{<colour>}` respecitvely.

`\pdfboxesset{..}`

```
\pdfboxesset[<bleed>]{<bleedblockwidth>}{<bleedblockheight>}
```

`\pdfcommentcell{..}` Add pdf comment as margin overlay.

```
\pdfcommentcell[<options>]{<comment>}
```

`\pdfpagecount{..}` Get pdf page count

```
\pdfpagecount{<file.pdf>}
```

`\pdfspinewidth{..}` Calculate spine thickness

```
\pdfspinewidth{<paper-thikness>}{<cover-thikness>}{<block-pdf>}
```

`\pdfspreadstopages{..}` EXPERIMENTAL

Include spreads from a pdf block as pages

```
\pdfspreadstopages[<delta>]{<block-pdf>}
```

`\pdfspreadstopages{..}` inserts an empty page before the first page in the block to push it to the right of the spread.

To display other pages consistently it is recommended to set the block width to `2\blockwidth` and set the `pdfpagelayout` to be set to `SinglePage`, a simple way to do both is to:

```
\ChangeLayout{spread}
```

`\TEX` `\LATEX` Convenience macros to display TeXand LaTeXin the correct font.

Why ASCII diagrams instead of normal graphics, you might ask? Well, for the same reason as photo-books in LaTeX – I liked the idea of it, the simplicity, and thought that it would be fun to see how far can I push things before it all falling apart on me, and... We are here, at the end, and it all is still here too :)