Marginalia

Write into the margins!

Contents

1 Setup 1
2 Margin-Notes 1
2.1 Markers 2
2.2 Styling 2
3 Wide Blocks 3
4 Figures 3
4.1 Notefigures 3
4.2 Large Figures 3
5 Other Tidbits 4
5.1 Absolute Placement 4
5.2 Headers and Background 4
6 Troubleshooting / Known Bugs 6
7 Thanks 6
A Detailed Documentation of all Exported Symbols 7

1 Setup

Put something akin to the following at the start of your `.typ` file:

```
#import "@preview/marginalia:0.1.3" as marginalia: note, wideblock
#let config = (
  // inner: ( far: 5mm, width: 15mm, sep: 5mm ),
  // outer: ( far: 5mm, width: 15mm, sep: 5mm ),
  // top: 2.5cm,
  // bottom: 2.5cm,
  // book: false,
  // clearance: 12pt,
  // flush-numbers: false,
  // numbering: /* numbering-function */,
)
#marginalia.configure(..config)
#set page(
  // setup margins:
  ..marginalia.page-setup(..config),
  /* other page setup */
)
```

Where you can then customize `config` to your preferences. Shown here (as comments) are the default values taken if the corresponding keys are unset.

See the appendix for a more detailed explanation of the `configure()` and `page-setup()` functions.

2 Margin-Notes

By default, the `#note[...]` command places a note to the right/outer margin, like so:●. By giving the argument `reverse: true`, we obtain a note on the left/inner margin.○ If `config.book = true`, the side will of course be adjusted automatically.

● This is a note.

They can contain any content, and will wrap within the note column.

○ Reversed.

If ◆ we ◇ place ■ multiple □ notes ▲ in △ one ♥ line,♡ they automatically adjust their positions. Additionally, a `dy` argument can be passed to shift their initial position by that amount vertically. They may still get shifted around, unless configured otherwise via the shift parameter of `#note()`.

◆ Note 1

◇ Note 2

■ Note 3

□ Note 4

▲ Note 5

△ This note was given `15pt` dy, but it was shifted more than that to avoid Notes 1–5.

♥ This note was given `15pt` dy.

♡ This note was given `10cm` dy and was shifted less than that to stay on the page.

Notes will shift vertically to avoid other notes, wideblocks, and the top page margin. It will attempt to move one note below a wide-block if there is not enough space above, but if there are multiple notes that would need to be rearranged you must assist by manually setting `dy` such that their initial position is below the wideblock.

Margin notes also work from within most containers such as blocks or `#column()`s.●

Blah blah.○ To force the notes to appear in the margin in the same order as they appear in the text, use `#note(keep-order: true)[]`◆ for *all* notes whose relative order is important.

○ Note from second column.

● Lorem ipsum dolor sit.

◆ Like so. The lorem-ipsum note was also placed with `keep-order`.

2.1 Markers

The margin notes are decorated with little symbols, which by default hang into the gap. If this is not desired, set the configuration option `flush-numbers: true`. Setting the argument `numbered: false`,◇ we obtain notes without icon/number:

◇ Unnumbered notes `"avoid"` being shifted if possible, preferring to shift other notes up.

Like this.

To change the markers, you can override `config.numbering-function` which is used to generate the markers.

2.2 Styling

Both `note()` and `notefigure()` accept a `text-style` and `par-style` parameter:

- `text-style: (size: 5pt, font: ("Iosevka Extended"))` gives ■
- `par-style: (spacing: 20pt, leading: -2pt)` gives □

■ `Lorem ipsum dolor sit amet.`

□ Lorem ipsum dolor sit.

Lorem ipsum dolor sit.

The default options here are meant to be as close as possible to the stock footnote style.

To add a block around the notes, simply create wrapper-functions as desired, as notes can contain block-level content without messing up the placement of the markers.

Example for dotted separator lines:▲

▲ This is a note with a dotted stroke above.

So is this.

```
#let note-with-separator(..args, body) = marginalia.note(
  ..args,
  block(
    stroke: (top: (thickness: 0.5pt, dash: "dotted")),
    outset: (top: 6pt /* clearance is 12pt */),
    width: 100%,
    body)
)
```

Examples for backgrounds:△♥

△ This is a note with a green background.

So is this.

♥ This is a note with a wide green background.

So is this.

```
#let note-with-wide-background(..args, body) = marginalia.note(
  ..args,
  block(
    fill: oklch(90.26%, 0.058, 140.43deg),
    outset: (left: 10pt, rest: 4pt),
    width: 100%, radius: 4pt,
    body)
)
#let note-with-background(..args, body) = marginalia.note(
  ..args,
  block(fill: oklch(90.26%, 0.058, 140.43deg),
  inset: (x: 4pt), outset: (y: 4pt),
  width: 100%, radius: 4pt,
  body)
)
```

3 Wide Blocks

The command `#wideblock[...]` can be used to wrap content in a wide block which spans into the margin-note-column.

Note: when using an asyymetric page layout with `book: true`, wideblocks which span across pagebreaks are messy, because there is no way for the wideblock to detect the pagebreak and adjust ist position after it.

● Wow!

It is possible to use notes in a wide block:♡●. They will automatically shift downwards to avoid colliding with the wideblock.○

`#wideblock(reverse: true)[...]`: The `reverse` option makes the block extend to the inside margin instead. This is analogous to the `reverse` option on notes and allows placing notes in their usual column.

○ Unless they are given a `dy` argument moving them above the block.

♡ Voila.

In this manual, a reverse wideblock is used to set the appendix to make it take up fewer pages. This is also why the appendix is no longer using `book: true`.◆

`#wideblock(double: true)[...]`: The `double` option makes it extend both ways. Note that setting both `reverse: true` and `double: true` is disallowed and will panic.

◆ Notes above a `wideblock` will shift upwards if necessary.

4 Figures

4.1 Notefigures

For small figures, you can place them in the margin with `marginalia.notefigure`. It accepts all arguments `figure` takes (except `placement` and `scope`), plus all arguments `note` takes (except `align-baseline`). However, by default it has no marker, and to get a marker like other notes, you must pass `numbered: true`, it will get a marker like other notes: ◇

Figure 1: A notefigure.

◇ Figure 2: A marked notefigure.

Additionally, the `dy` argument now takes a relative length, where `100%` is the distance between the top of the figure content and the first baseline of the caption. By default, figures have a `dy` of `0pt - 100%`, which results in the caption being aligned horizontally to the text. ■

■ Figure 3: Aligned to top of figure with `dy: 0pt`.

A label can be attached to the figure using the `label` argument.

4.2 Large Figures

For larger figures, use the following set and show rules:

```
#set figure(gap: 0pt)
#set figure.caption(position: top)
#show figure.caption.where(position: top): note.with(numbered:false, dy:1em)
```

Figure 4: A figure.

For wide figures, simply place a figure in a wideblock. The caption gets placed beneath the figure automatically, courtesy of regular wide-block-avoidance.

```
#wideblock(figure(image(..), caption: [A figure in a wide block.]))
```

Figure 5: A figure in a wide block.

Top (no shift)
Top (no shift, no baseline align)

Top

Figure 6: A figur
wide block.

Figure 7: A figure in a double-wide block.

5 Other Tidbits

5.1 Absolute Placement

You can place notes in absolute positions relative to the page using `place`:

```
#place(top, note(numbered: false, reverse: true)[Top])
#place(bottom, note(numbered: false, reverse: true)[Bottom])
```

To avoid these notes moving about, use `shift: false` (or `shift: "ignore"` if you don't mind overlaps.)

```
#place(top, note(numbered: false, shift: false)[Top (no shift)])
#place(bottom, note(numbered: false, shift: false)[Bottom (no shift)])
```

By default, notes are aligned to their first baseline. To align the top of the note instead, set `align-baseline` to `false`.

5.2 Headers and Background

This is not (yet) a polished feature and requires to access `marginalia._config.get().book` to read the respective config option. In your documents, consider removing this check and simplifying the `if` a bit.

Also, please don't `.update()` the `marginalia._config` directly, this can easily break the notes.

Here's how the headers in this document were made:

```
#set page(header: context {
  marginalia.notecounter.update(0)
  let book = marginalia._config.get().book
  let leftm = marginalia.get-left()
  let rightm = marginalia.get-right()
  if here().page() > 1 {
    wideblock(double: true, {
      box(width: leftm.width, {
        if not (book) or calc.odd(here().page()) [
          Page
          #counter(page).display("1 of 1", both: true)
        ] else [
          #datetime.today().display(/**/)
        ]
      })
      h(leftm.sep)
      box(width: 1fr, smallcaps[Marginalia])
      h(rightm.sep)
      box(width: rightm.width, {
        if not (book) or calc.odd(here().page()) [
          #datetime.today().display(/**/)
        ] else [
          Page
```

Bottom (no shift)
Bottom (no shift, no baseline al.)

Bottom

```
          #counter(page).display("1 of 1", both: true)
        ]
      })
    })
  }
})
```

And here's the code for the lines in the background:

Not that you should copy them, they're mostly here to showcase the columns and help me verify that everything gets placed in the right spot.

```
#set page(background: context {
  let leftm = marginalia.get-left()
  let rightm = marginalia.get-right()
  place(top, dy: marginalia._config.get().top,
        line(length: 100%, stroke: luma(90%)))
  place(top, dy: marginalia._config.get().top - page.header-ascent,
        line(length: 100%, stroke: luma(90%)))
  place(bottom, dy: -marginalia._config.get().bottom,
        line(length: 100%, stroke: luma(90%)))
  place(dx: leftm.far,
        rect(width: leftm.width, height: 100%, stroke: (x: luma(90%))))
  place(dx: leftm.far + leftm.width + leftm.sep,
        rect(width: 10pt, height: 100%, stroke: (left: luma(90%))))
  place(right, dx: -rightm.far,
        rect(width: rightm.width, height: 100%, stroke: (x: luma(90%))))
  place(right, dx: -rightm.far - rightm.width - rightm.sep,
        rect(width: 10pt, height: 100%, stroke: (right: luma(90%))))
})
```

6 Troubleshooting / Known Bugs

- If the document needs multiple passes to figure out page-breaks,△ it can break the note positioning.
  - This can usually be resolved by placing a `#pagebreak()` or `#pagebreak(weak: true)` in an appropriate location.

△ This can happen for example with outlines which barely fit/don't fit onto the page.

- Nested notes may or may not work.♥ In nearly all cases, they seem to lead to a "layout did not converge within 5 attempts" warning, so it is probably best to avoid them if possible.
  - Just use multiple paragraphs in one note, or place multiple notes in the main text instead.
  - If really neccessary, use `shift: "ignore"` on the nested notes and manually set `dy`.

♥ In this manual, for example, it works fine (with warnings) here, ♡ but not on the first page.●

♡ Probably because there aren't many other notes around.

● Notes on the other side are usually fine though.

- If `book` is `true`, wideblocks that break across pages are broken. Sadly there doesn't seem to be a way to detect and react to page-breaks from within a `block`, so I don't know how to fix this.
- If you encounter anything else which looks like a bug to you, please create an "issue" on GitHub if no-one else has done so already.

7 Thanks

Many thanks go to Nathan Jessurun for their drafting package, which has served as a starting point and was very helpful in figuring out how to position margin-notes.

The `wideblock` functionality was inspired by the one provided in the tufte-memo template.

Also shout-out to tidy, which was used to produce the appendix.

A Detailed Documentation of all Exported Symbols

Functions:
- configure()
- get-left()
- get-right()
- note()
- note-numbering()
- notefigure()
- page-setup()
- wideblock()

Variables:
- notecounter
- note-markers
- note-markers-alternating

A.1 configure

This will update the marginalia config with the provided config options.

The default values for the margins have been chosen such that they match the default typst margins for a4. It is strongly recommended to change at least one of either `inner` or `outer` to be wide enough to actually contain text.

The shown default values are for the first usage of this function. On later calls, unspecified options are kept from the previous configuration state:

```
configure(clearance: 5pt)
configure(book: true)
```

is equivalent to `configure(clearance: 5pt, book: true)`.

Parameters

```
configure(
  inner: dictionary,
  outer: dictionary,
  top: length,
  bottom: length,
  book: boolean,
  clearance: length,
  flush-numbers: boolean,
  numbering: function string
)
```

`inner: (far: 5mm, width: 15mm, sep: 5mm)` `dictionary`

Inside/left margins.
- `far`: Distance between edge of page and margin (note) column.
- `width`: Width of the margin column.
- `sep`: Distance between margin column and main text column.

The page inside/left margin should equal `far` + `width` + `sep`.

If partial dictionary is given, it will be filled up with defaults.

`outer: (far: 5mm, width: 15mm, sep: 5mm)` `dictionary`

Outside/right margins. Analogous to `inner`.

`top: 2.5cm` `length` Top margin.

`bottom: 2.5cm` `length` Bottom margin.

`book: false` `boolean`

- If `true`, will use inside/outside margins, alternating on each page.
- If `false`, will use left/right margins with all pages the same.

`clearance: 12pt` `length`

Minimal vertical distance between notes and to wide blocks.

`flush-numbers: false` `boolean`

Disallow note markers hanging into the whitespace.

numbering: `note-numbering` `function` or `string`

Function or `numbering`-string to generate the note markers from the `notecounter`.

Examples:
- `(..i) => super(numbering("1", ..i))` for superscript numbers
- `(..i) => super(numbering("a", ..i))` for superscript letters
- `marginalia.note-numbering.with(repeat: false, markers: ())` for small blue numbers

A.2 get-left

Returns a dictionary with the keys `far`, `width`, `sep` containing the respective widths of the left margin on the current page. (On both even and odd pages.)

Mostly internal.

Requires context.

Parameters

```
get-left() → dictionary
```

A.3 get-right

Returns a dictionary with the keys `far`, `width`, `sep` containing the respective widths of the right margin on the current page. (On both even and odd pages.)

Mostly internal.

Requires context.

Parameters

```
get-right() → dictionary
```

A.4 note

Create a marginnote. Will adjust it's position downwards to avoid previously placed notes, and upwards to avoid extending past the bottom margin.

Parameters

```
note(
  numbered: boolean,
  reverse: boolean,
  dy: length,
  align-baseline: boolean,
  keep-order: boolean,
  shift: boolean auto "avoid" "ignore",
  text-style: dictionary,
  par-style: dictionary,
  ⟨body⟩: content
)
```

numbered: `true` `boolean` Whether to put a mark.

reverse: `false` `boolean` Whether to put it in the opposite (inner/left) margin.

dy: `0pt` `length`

Inital vertical offset of the note. Note may get shifted still to avoid other notes.

align-baseline: `true` `boolean`

Whether to align the baselines or not.
- If `false`, the top of the note is aligned with the main-text baseline.

keep-order: `false` `boolean`

Notes with `keep-order: true` are not re-ordered relative to one another.

shift: `auto` `boolean` or `auto` or `"avoid"` or `"ignore"`

Whether the note may get shifted around to avoid other notes.
- `true`: The note may shift to avoid other notes, wide-blocks and the top/bottom margins.
- `false`: The note is placed exactly where it appears, and other notes may shift to avoid it.
- `"avoid"`: The note is only shifted if shifting other notes is not sufficent to avoid a collision.
- `"ignore"`: Like `false`, but other notes do not try to avoid it.
- `auto`: `true` if numbered, `"avoid"` otherwise.

text-style: `(size: 0.85em, style: "normal", weight: "regular")` `dictionary`

Will be used to `set` the text style.

par-style: `(spacing: 1.2em, leading: 0.5em, hanging-indent: 0pt)` `dictionary`

Will be used to `set` the par style.

A.5 note-numbering

Format note marker

Parameters

```
note-numbering(
  markers: array(string),
  repeat: boolean,
  (..),
  (number): int
) → content
```

markers: `note-markers-alternating` `array(string)`

```
#for i in array.range(1,15) [
  #note-numbering(markers: note-
markers, i)
]\
#for i in array.range(1,15) [
  #note-numbering(markers: note-
markers-alternating, i)
]
```

```
◆ ● ■ ▲ ♥ ◇ ○ □ △ ♡ ◆ ● ■ ▲
● ○ ◆ ◇ ■ □ ▲ △ ♥ ♡ ● ○ ◆ ◇
```

repeat: true boolean

Whether to (true) loop over the icons, or (false) continue with numbers after icons run out.

```
#for i in array.range(1,15) [
  #note-numbering(repeat: true, i)
]\
#for i in array.range(1,15) [
  #note-numbering(repeat: false, i)
]
```

```
● ○ ◆ ◇ ■ □ ▲ △ ♥ ♡ ● ○ ◆ ◇
● ○ ◆ ◇ ■ □ ▲ △ ♥ ♡ 1 2 3 4
```

A.6 notefigure

Creates a figure in the margin.

Parameters `numbered`, `reverse`, `keep-order`, `shift`, `text-style` and `par-style` work the same as for `note()`.

Parameters

```
notefigure(
  numbered: boolean,
  reverse: boolean,
  dy: relative length,
  keep-order: boolean,
  shift: boolean auto "avoid" "ignore",
  text-style: dictionary,
  par-style: dictionary,
  gap: length,
  label: none label,
  ⟨content⟩: content,
  ⟨..figureargs⟩: arguments
) → content
```

dy: 0pt - 100% relative length

How much to shift the note. 100% corresponds to the height of `content` + `gap` + the first baseline.

Thus `dy: 0pt - 100%` aligns the text and caption baselines.

gap: 0.55em length

Pass-through to `#figure()`, but used to adjust the vertical position.

label: none none or label A label to attach to the figure.

⟨content⟩ content Positional

The figure content, e.g. an image. Pass-through to `#figure()`, but used to adjust the vertical position.

⟨..figureargs⟩ arguments Positional Pass-through to `#figure()`.

(E.g. `caption`)

A.7 page-setup

Page setup helper

This will generate a dictionary (`margin: ..`) compatible with the passed config. This can then be spread into the page setup like so:

```
#set page( ..page-setup(..config) )
```

Takes the same options as `configure()`.

Parameters

```
page-setup(⟨..config⟩: dictionary ) → dictionary
```

⟨..config⟩ `dictionary` Positional

Missing entries are filled with package defaults. Note: missing entries are *not* taken from the current marginalia config, as this would require context.

A.8 wideblock

Creates a block that extends into the outside/right margin.

Note: This does not handle page-breaks sensibly. If `config.book = false`, this is not a problem, as then the margins on all pages are the same. However, when using alternating page margins, a multi-page `wideblock` will not work properly. To be able to set this appendix in a many-page wideblock, this code was used:

```
#configure(..config, book: false)
#set page(..page-setup(..config, book: false))
#wideblock(reverse: true)[...]
```

Parameters

```
wideblock(reverse: boolean , double: boolean , ⟨body⟩: content ) → content
```

reverse: `false` `boolean` Whether to extend into the inside/left margin instead.

double: `false` `boolean`

Whether to extend into both margins. Cannot be combined with `reverse`.

A.9 notecounter `counter`

The counter used for the note icons.

Mostly internal.

If you use `note-numbering()` without `note-numbering.repeat`, it is reccommended you reset this occasionally, e.g. per heading or per page.

```
notecounter.update(1)
```

A.10 note-markers

Icons to use for note markers.

```
("◆", "●", "■", "▲", "♥", "◇", "○", "□", "△", "♡")
```

A.11 note-markers-alternating

Icons to use for note markers, alternating filled/outlined.

```
("●", "○", "◆", "◇", "■", "□", "▲", "△", "♥", "♡")
```