Marginalia ?.?.?

Write into the margins!

Contents

1 Setup

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

```
#import "@preview/marginalia:?.?.?" as marginalia: note, notefigure, wideblock

#show: marginalia.setup.with(
  // inner: ( far: 5mm, width: 15mm, sep: 5mm ),
  // outer: ( far: 5mm, width: 15mm, sep: 5mm ),
  // top: 2.5cm,
  // bottom: 2.5cm,
  // book: false,
  // clearance: 12pt,
)
```

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

If `book` is `false`, `inner` and `outer` correspond to the left and right margins respectively. If book is true, the margins swap sides on even and odd pages. Notes are placed in the outside margin by default.

See the appendix for a more detailed explanation of the `setup()` function and its options.

Additionally, I recommend using Typst's partial function application feature to customize other aspects of the notes consistently:

```
#let note = note.with(/* options here */)
#let notefigure = notefigure.with(/* same options here */)
```

● Do not `#import "...": *`, this will shadow built-in functions.

2 Margin-Notes

By default, the `#note[]` command places a note to the right/outer margin, like so:○. By giving the argument `side: "inner"`, we obtain a note on the inner (left) margin.◆ If `setup.book` is `true`, the side will of course be adjusted automatically. It is also possible to pass `side: "left"` or `side: "right"` if you want a fixed side even in books.

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[]`.

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

○ This is a note.

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

◇ Note 1

■ Note 2

□ Note 3

▲ Note 4

△ Note 5

◆ Reversed.

♡ This note was given `15pt` dy.

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

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

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 `#block[]`s or `#column[]`s.○ By default, they are placed aligned with their anchor.◆ To force the notes to appear in the margin in the same order as they appear in the text, set `keep-order: true`◇ on *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 `flush-numbering: true` on the note.■

■ This note has flush numbering.

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

You can also change the `counter` used. This can be useful if you want some of your notes to have independent numbering.A□B

A alphabetized note

□ regular one

B another alphabetized note

```
#let a-note-counter = counter("a-note")
#let a-note = note.with(
  counter: a-note-counter,
  numbering: (..i) => text(weight: 900, font: "Inter", size: 5pt, style: "normal", fill: rgb(54%, 72%, 95%), numbering("A", ..i)),
)
```

Setting the `counter` to `none`,▲ we obtain notes without number:

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

Like this.

References

There are two ways to reference another note:

1. You can add a `<label>` to the note and then `@label` reference it. Note that any supplement is ignored.
2. You can use `marginalia.ref()` and tell it how many notes away the target is. This is mostly useful to reference the most recent note again.

   Be aware that notes without anchor/number still count towards the offset, and you can also reference them, but doing so results in an invisible link and is a bit pointless.

```
Original: #note[This is a note]<label>
Label Reference: @label @label2
Count Reference: #marginalia.ref(-1) #marginalia.ref(1)
Original: #note[This is another note]<label2>
```

△ This is a note

♥ This is another note

Original:△ Label Reference:△♥ Offset Reference:△♥ Original:♥

Advanced Markers

If a different style is desired for the marker in the text and in the margins, you can use the `anchor-numbering` parameter to control the in-text marker:[20]

20 This note has a custom numbering, but the same counter.

```
#note(
  numbering: (.., i) => text(font: "Inria Sans")[#i#h(0.5em)],
  anchor-numbering: (.., i) => super[#i],
)[...]
```

Note that doing this implies `flush-numbering: true`. This is based on the assumption that if you have set two different numbering functions, you want to handle the placement yourself. Non-flush numbers, which are placed, complicate this.

This can also be used to create notes that have an anchor,→ but no numbering in the note itself.[21]

Like this one. The `counter` has to be not none, as it will not display the anchor otherwise.

21 (the `notecounter` is unaffected by the previous note, as it has `numbering: none`)

2.2 Styling

Both `note()` and `notefigure()` accept `text-style`, `par-style`, and `block-style` parameters:

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

○ Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do.

◆ Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do.

The default options here are meant to be as close as possible to the stock footnote style given 11pt text. For other text sizes, set the `text-style` size to 0.85 times your body text size if you want to match the stock footnotes.

Lorem ipsum dolor sit.

`block-style`

To style the block containing the note body, use the `block-style` parameter.

- `block-style: (stroke: (top: (thickness: 0.5pt, dash: "dotted")), outset: (top: 6pt /* clearance is 12pt */), width: 100%)` gives:◇
- `block-style: (fill: oklch(90%, 0.06, 140deg), outset: (left: 10pt, rest: 4pt), width: 100%, radius: 4pt)` gives:■
- `block-style: (fill: oklch(90%, 0.06, 140deg), inset: (x: 4pt), outset: (y: 4pt), width: 100%, radius: 4pt)` gives:□

◇ This is a note with a dotted stroke above.

So is this.

■ This is a note with a green background and `flush-numbering: true`.

So is this.

□ This is a note with an outset green background.

So is this.

For more advanced use-cases, you can also pass a function as the `block-style`. It will be called with one argument, either `"left"` of `"right"`, depending on the side the note will be placed on. Inside the function, context is available.▲△

△ Purple 2

▲ Purple

```
#let block-style = (side) => {
  if side == "left" {
    (stroke: (left: none, rest: 0.5pt + purple), outset: (left: marginalia.get-left().far,
rest: 4pt))
  } else {
    (stroke: (right: none, rest: 0.5pt + purple), outset: (right: marginalia.get-
right().far, left: 9pt, rest: 4pt))
  }
}
#note(block-style: block-style)[Purple]
#note(side: "inner", block-style: block-style)[Purple 2]
```

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 asymmetric page layout with `setup.book: true`, wideblocks which span across pagebreaks are messy, because there is no way for the wideblock to detect the pagebreak and adjust its position after it.

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

♡ Wow!

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

♥ Voila.

`#wideblock(side: "inner")[...]`: The `side` option allows extending the block into the inside margin instead. This is analogous to the `side` option on notes and notefigures and allows placing notes in their usual column.

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

`#wideblock(side: "both")[...]`: Additionally, wideblocks can extend on both sides, for extra wide content. This is especially useful for figures, more on that below.

○ 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 `#notefigure()`. It accepts all arguments `#figure()` takes (except `placement` and `scope`), plus all arguments `#note[]` takes.

Figure 1: A notefigure.

However, by default it has no marker, and to get a marker like other notes, you must pass `counter: marginalia.notecounter`, and it will get a marker like other notes:◆

◆ Figure 2: A marked notefigure.

If you want, you can override the `counter` and `anchor-numbering` to get an anchor using the figure-numbering.[fig. 3]

Figure 3: reusing figure counter

```
#notefigure(
  /**/,
  counter: counter(figure.where(kind: image)),
  anchor-numbering: (.., i) => super[fig. #(i + 1) ],
  numbering: none,
)
```

Additionally, the `alignment` parameter can now also be `"caption-top"`, which results in alignment with the top of the caption.◇

◇ Figure 4: Aligned to top of caption.

By default, like normal `#note[]`s, it uses `alignment: "baseline"` which leads to the caption's being aligned with the main text.■

■ Figure 5: Aligned to top of figure with `alignment: "top"`.

A label can be attached to the figure normally as was done for Figure 2. You can also add a label to the *note* by using the `note-label` argument.◆

Notefigures can also be given `side`, `text-style`, `par-style`, and `block-style` parameters, – like `#note[]` – as is demonstrated in Figure 6. Furthermore, the `numbering`, `anchor-numbering`, and `flush-numbering` parameters work as expected.[36]

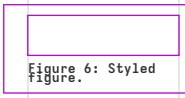


Figure 6: Styled figure.

36 Figure 7: Figure with custom numbering

Note that `#show figure.caption: /**/` rules are ignored for `#notefigure[]`s, use the `show-caption` parameter instead.▲

▲ NB: `show-caption` expects a function with two arguments, check the docs.

4.2 Large Figures

For larger figures, use the following set and show rules if you want top-aligned captions:

```
#set figure(gap: 0pt) // neccessary in both cases
#set figure.caption(position: top)
#show figure.caption.where(position: top): note.with(
  alignment: "top", counter: none, shift: "avoid", keep-order: true,
  dy: -0.01pt, // this is so that the caption is placed above wide figures.
)
```

Figure 8: A figure.

And if you want bottom-aligned captions, use the following:

```
#set figure(gap: 0pt) // neccessary in both cases
#set figure.caption(position: bottom) // (this is the default)
#show figure.caption.where(position: bottom): note.with(
  alignment: "bottom", counter: none, shift: "avoid", keep-order: true,)
```

Figure 9: A figure.

Wide Figures

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

(this is assuming you have one of the above `show` rules)

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

Figure 10: A figure in a wide block.

Figure 11: A figure in a reversed wide block.

Figure 12: A figure in an extra-wide wideblock.

5 Other Tidbits

5.1 Background Lines

They're mostly here to showcase the columns and help me verify that everything gets placed in the right spot, but if you want, you can enable the lines in the background simply by using

```
#show: marginalia.show-frame
```

You can also hide the lines for the header and footer with

Top (no shift)
Top (no shift, top-aligned)

```
#show: marginalia.show-frame.with(header: false, footer: false)
```

Top

5.2 Absolute Placement

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

```
#place(top, note(counter: none, side: "inner")[Top])
#place(bottom, note(counter: none, side: "inner")[Bottom])
```

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

```
#place(top, note(counter: none, shift: false)[Top (no shift)])
#place(bottom, note(counter: none, shift: false)[Bottom (no shift)])
```

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

△(Or set `alignment` to `"bottom"` to align the bottom of the note.)

5.3 Headers

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

```
#set page(
  header: context if here().page() > 1 {
    marginalia.header(
      text-style: (size: 8.5pt, number-type: "old-style"),
      [Page #counter(page).display("1 of 1", both: true)],
      [#smallcaps[Marginalia] #text(fill: luma(60%))[#VERSION]],
      [],
    )
  },
)
```

Despite the name, this function can be used anywhere, and not solely for headers. It simply creates a wideblock and fills it with properly sized boxes.

The `#marginalia.header()` function is pretty flexible in the arguments it expects. Any of the following will work:

```
#marginalia.header(                  [outer]) = #marginalia.header[outer]
#marginalia.header(         [center], [outer]) = #marginalia.header[center][outer]
#marginalia.header([inner], [center], [outer]) = #marginalia.header[inner][center][outer]
#marginalia.header(
  even: ([left outer], [center], [right inner]),
  odd: ([left inner], [center], [right outer]),
)
```

For convenience, you may pass a `text-style` parameter also.

left outer *this is on an even page* *right inner*

5.4 Pages with automatic sizing.

Pages with `width: auto` are not supported at all.

Pages with `height: auto` work – with the limitation that notes may run over the bottom of the page as they are not considered by Typst when determining the page height.

There are two workarounds for this: (these can be combined)

1. If there is not enough space on the page to fit the notes, you can add some vertical space (`#v(__pt)`) to make the page taller.

Bottom (no shift)
Bottom (no shift, top-aligned)

Bottom

2. If there is left-over space above the notes,♥ you can try the following:

♥ I.e. moving notes up would make them fit inside the page

```
// at the TOP of your content (before all notes)
//   -- this ensures that notes aren't moved below the "barrier"
#let note = note.with(keep-order: true)

/* Your content */

// at the END of your content
//   -- this serves as a "barrier" that moves the previous notes up
#context marginalia.note(shift: false, alignment: "top", dy:
marginalia._config.get().clearance, keep-order: true, counter: none)[]
```

6 Troubleshooting / Known Limitations

- 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 necessary, 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.

(This project is not affiliated with https://marginalia-search.com/, but that is *also* a cool project.)

A Detailed Documentation of all Exported Symbols

Breaking Changes

0.2.0

- The functions `configure()` and `page-setup()` have been combined into one `setup()` function.

A.1 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.)

Requires context.

Parameters

`get-left() → dictionary`

A.2 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.)

Requires context.

Parameters

`get-right() → dictionary`

A.3 header

This generates a `wideblock()` and divides its arguments into three boxes sized to match the margin setup.

Parameters

`header(text-style: dictionary, ⟨..args⟩, even: array, odd: array) → content`

`text-style: (:)` `dictionary` Will be used to `set` the text style.

`⟨..args⟩` Positional

Up to three positional arguments. They are interpreted as `⟨outer⟩`, `⟨center⟩⟨outer⟩`, or `⟨inner⟩⟨center⟩⟨outer⟩`, depending on how many there are.

`even: ()` `array`

This is ignored if there are positional parameters or if `setup.book` is `false`.

Otherwise, it is interpreted as `(⟨outer⟩, ⟨center⟩, ⟨inner⟩)` on even pages.

odd: () array

This is ignored if there are positional parameters.

Otherwise, it is interpreted as (⟨inner⟩, ⟨center⟩, ⟨outer⟩) on odd pages or, if setup.book is false, on all pages.

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.

Notes can be attached a label and are referenceable (if setup() was run).

Breaking Changes

0.1.5

- reverse has been replaced with note.side.
  ⟶ use `side: "inner"` instead of `reverse: true`
- numbered has been replaced with note.numbering.
  ⟶ use `numbering: "none"` instead of `numbered: false`

0.2.2

- align-baseline has been replaced with note.alignment.
  ⟶ use `alignment: "top"` instead of `align-baseline: false`

Parameters

```
note(
  counter: counter none,
  numbering: none function string,
  anchor-numbering: none auto function string,
  link-anchor: bool,
  flush-numbering: auto bool,
  side: auto "outer" "inner" "left" "right" "near",
  alignment: "baseline" "top" "bottom",
  dy: length,
  keep-order: bool,
  shift: bool auto "avoid" "ignore",
  text-style: dictionary,
  par-style: dictionary,
  block-style: dictionary function,
  ⟨body⟩: content
)
```

counter: notecounter counter or none

Counter to use for this note. Can be set to none do disable numbering this note.

Will only be stepped if numbering is not none.

`numbering: note-numbering` `none` or `function` or `string`

Function or `numbering`-string to generate the note markers from the `notecounter`.
- If `none`, will not step the `counter`.
- Will be ignored if `counter` is `none`.

Examples:
- `(..i) => super(numbering("1", ..i))` for superscript numbers[44]
- `(..i) => super(numbering("a", ..i))` for superscript letters[as]
- `marginalia.note-numbering.with(repeat: false, markers: ())` for small blue numbers[46]

[44] E.g.

[as] E.g.

[46] E.g.

`anchor-numbering: auto` `none` or `auto` or `function` or `string`

Used to generate the marker for the anchor (i.e. the one in the surrounding text)

- If `auto`, will use the given `note.numbering`.
- Will be ignored if `counter` is `none`.

`link-anchor: true` `bool`

Whether to have the anchor link to the note, and vice-versa.

`flush-numbering: auto` `auto` or `bool`

Disallow note markers hanging into the whitespace.
- If `auto`, acts like `false` if `note.anchor-numbering` is `auto`.

`side: auto` `auto` or `"outer"` or `"inner"` or `"left"` or `"right"` or `"near"`

Which side to place the note. `auto` defaults to `"outer"`. In non-book documents, `"outer"`/`"inner"` are equivalent to `"right"`/`"left"` respectively. `"near"` will place the note in the left or right margin, depending which is nearer.

`alignment: "baseline"` `"baseline"` or `"top"` or `"bottom"`

Vertical alignment of the note.
- `"bottom"` aligns the bottom edge of the note with the main text baseline.▲
- `"baseline"` aligns the first baseline of the note with the main text baseline.▵
- `"top"` aligns the top edge of the note with the main text baseline.♥

▲ Bottom
...

▵ Baseline
...

♥ Top
...

`dy: 0pt` `length`

Inital vertical offset of the note, relative to the alignment point. The note may get shifted still to avoid other notes depending on `note.shift`.

`keep-order: false` `bool`

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

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

Whether the note may get shifted vertically 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. E.g. if it would collide with a wideblock or a note with `shift: false`.
- `"ignore"`: Like `false`, but other notes do not try to avoid it.
- `auto`: `true` if numbered, `"avoid"` otherwise.

text-style: `(size: 9.35pt, 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.

block-style: `(width: 100%)` `dictionary` or `function`

Will be passed to the `block` containing the note body. If this is a function, it will be called with `"left"` or `"right"` as its argument, and the result is passed to the `block`.

A.5 note-numbering

Format note marker.

Parameters

```
note-numbering(
  markers: array,
  repeat: bool,
  style: function,
  space: auto bool,
  ⟨..⟩,
  ⟨i⟩: int
) → content
```

markers: note-markers-alternating `array`

```
#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) }

#for i in array.range(1,15) {
  note-numbering(markers: (), i) }
```

◆●■▲❤◇○□△♡◆●■▲

●○◆◇■□▲△❤♡●○◆◇

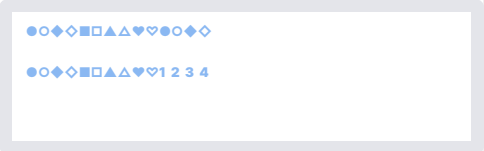
1 2 3 4 5 6 7 8 9 10 11 12 13 14

repeat: true bool

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) }
```



style: text.with(weight: 900, font: "Inter", size: 5pt, style: "normal", fill: rgb(54%, 72%, 95%)) function

Wrap the symbol in a styled text function.

space: auto auto or bool

Whether to add a space of 2pt after the symbol. If auto, a space is only added if it is a number (the symbols have ran out).

A.6 notefigure

Creates a figure in the margin.

Parameters numbering, anchor-numbering, flush-numbering, side, keep-order, shift, text-style, par-style, and block-style work the same as for note().

Notefigures can be attached a label and are referenceable (if setup() was run). Furthermore, the underlying note() can be given a label using the note-label parameter.

Breaking Changes

0.1.5

- reverse has been replaced with notefigure.side.
  ⟶ use `side: "inner"` instead of `reverse: true`
- numbered has been replaced with notefigure.numbering.
  ⟶ use `numbering: marginalia.note-numbering` instead of `numbered: true`

0.2.2

- notefigure.dy no longer takes a relative length, instead notefigure.alignment was added.

0.3.0

- The label argument has been removed.
  ⟶ Instead of #notefigure(label: <l>, ..), use #notefigure(..)<l>.

Parameters

```
notefigure(
  counter: counter none,
  numbering: none function string,
  anchor-numbering: none auto function string,
  link-anchor: bool,
  flush-numbering: auto bool,
  side: auto "outer" "inner" "left" "right" "near",
  alignment: "baseline" "top" "bottom" "caption-top",
  dy: length,
  keep-order: bool,
  shift: bool auto "avoid" "ignore",
  text-style: dictionary,
  par-style: dictionary,
  block-style: dictionary function,
  show-caption: function,
  gap: length,
  note-label: none label,
  ⟨content⟩: content,
  ⟨..figureargs⟩: arguments
) → content
```

counter: none counter or none

Same as note.numbering, but with different default. Set this to marginalia.notecounter (or another counter) to enable numbering this note.

Will only be stepped if numbering is not none.

```
Notefigure with marker:
#notefigure(rect(height: 10pt, width: 100%), caption: [...], counter: marginalia.notecounter)
```

Notefigure with marker:

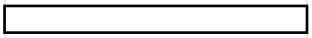


Figure 13: ...

```
Using the figure counter for the numbering:
#notefigure(
  rect(height: 10pt, width: 100%), caption: [...],
  counter: counter(figure.where(kind: image)),
  anchor-numbering: (.., i) => super[fig. #numbering("1", i+1)], numbering: none,
)
```

Using the figure counter for the numbering:[fig. 14]

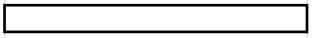


Figure 14: ...

numbering: note-numbering none or function or string Same as note.numbering.

anchor-numbering: auto none or auto or function or string

Same as note.anchor-numbering.

link-anchor: true bool

Whether to have the anchor link to the note, and vice-versa.

flush-numbering: auto auto or bool

Disallow note markers hanging into the whitespace.

- If auto, acts like false if notefigure.anchor-numbering is auto.

side: auto `auto` or `"outer"` or `"inner"` or `"left"` or `"right"` or `"near"`

Which side to place the note. auto defaults to "outer". In non-book documents, "outer"/"inner" are equivalent to "right"/"left" respectively.

alignment: "baseline" `"baseline"` or `"top"` or `"bottom"` or `"caption-top"`

Vertical alignment of the notefigure.

- "top", "bottom" work the same as note.alignment.
- "baseline" aligns the first baseline of the *caption* with the main text baseline.
- "caption-top" aligns the top of the caption with the main text baseline.

Figure 15: Baseline

Figure 16: Caption-top

dy: 0pt `length`

Inital vertical offset of the notefigure, relative to the alignment point.

The notefigure may get shifted still to avoid other notes depending on notefigure.shift.

text-style: (size: 9.35pt, 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.

block-style: (width: 100%) `dictionary` or `function`

Will be passed to the block containing the note body (this contains the entire figure). If this is a function, it will be called with "left" or "right" as its argument, and the result is passed to the block.

show-caption:

```
(number, caption) => {
    number
    caption.supplement
    [ ]
    caption.counter.display(caption.numbering)
    caption.separator
    caption.body
  }
```

`function`

A function with two arguments, the (note-)number and the caption. Will be called as the caption show rule.

If notefigure.numbering is none, number will be none.

gap: 0.55em `length`

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

note-label: none `none` or `label`

A label to attach to the note. Referencing this label will repeat the anchor, so it is only really useful if notefigure.anchor-numbering is not none.

⟨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 ref

Reference a nearby margin note. Will place the same anchor as that note had.

Be aware that notes without an anchor (including notefigures) still count for the offset, but the rendered link is empty.

```
This is a note: #note[Blah Blah]

This is a link to that note:
  #marginalia.ref(-1)

This is an unnumbered note:
  #note(counter: none)[Blah Blah]

This is a useless link to that note:
  #marginalia.ref(-1)
```

This is a note:

This is a link to that note:

This is an unnumbered note:

This is a useless link to that note:

Blah Blah

Blah Blah

Parameters

`ref(⟨offset⟩: integer)`

⟨offset⟩ `integer` Positional How many notes away the target note is.

- `-1`: The previous note.
- `0`: Disallowed
- `1`: The next note.

A.8 setup

This will update the marginalia config and setup the page with the provided config options. (This means this will insert a pagebreak.)

Use as

```
#show: marginalia.setup.with(/* options here */)
```

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.

This function also sets up the neccesary show-rule to allow referencing labelled notes. If you also have a custom `#show ref:` rule, it may be relevant if setup is called before or after that show rule.

Breaking Changes

0.1.5

- `numbering` has been replaced with `note.numbering`/`notefigure.numbering`.
  ⟶ set `numbering: /**/` directly on your notes instead of via `setup()`.
  Use `#let note = note.with(numbering: /**/)` for consistency.
- `flush-numbers` has been replaced by `note.flush-numbering`.
  ⟶ set `flush-numbering: true` directly on your notes instead of via `setup()`.
  Use `#let note = note.with(flush-numbering: /**/)` for consistency.

0.2.0

- This function does no longer apply the configuration partially, but will reset all unspecified options to the default. Additionally, it replaces the `page-setup()` function that was needed previously and is no longer called `configure()`

Parameters

```
setup(
  inner: dictionary,
  outer: dictionary,
  top: length,
  bottom: length,
  book: bool,
  clearance: length,
  ⟨body⟩: content
)
```

`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` `bool`

- 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.

A.9 show-frame

Adds lines to the page background showing the various vertical and horizontal boundaries used by marginalia.

To be used in a show-rule:

```
#show: marginalia.show-frame
```

Parameters

```
show-frame(stroke: color, header: bool, footer: bool, ⟨body⟩: content) → content
```

stroke: 0.5pt + luma(90%) color

Stroke for the lines.

```
#show: marginalia.show-frame.with(stroke: 2pt + red)
```

header: true bool Set to false to hide the header line

footer: true bool Set to false to hide the footer line

A.10 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:

```
#show: marginalia.setup.with(..config, book: false)
#wideblock(side: "inner")[...]
```

Breaking Changes

0.1.5

- `reverse` and `double` have been replaced with `wideblock.side`.
  ⟶ use `side: "inner"` instead of `reverse: true`
  ⟶ use `side: "both"` instead of `double: true`

Parameters

```
wideblock(
  side: auto "outer" "inner" "left" "right" "both",
  ⟨body⟩: content
) → content
```

side: auto auto or "outer" or "inner" or "left" or "right" or "both"

Which side to extend into. `auto` defaults to `"outer"`. In non-book documents, `"outer"`/`"inner"` are equivalent to `"right"`/`"left"` respectively.

A.11 notecounter counter

The default counter used for the note icons.

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

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

A.12 note-markers

Icons to use for note markers.

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

A.13 note-markers-alternating

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

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