# Marginalia

*Write into the margins!*

## 1 Setup

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

```typ
#import "@preview/marginalia:version": 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,
  // 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.

◆ *Reversed.*

◆ *This is a note.*
*They can contain any content, and will wrap within the note column.*

If ■ we ▲ place multiple notes in one line, they automatically adjust their positions (Up to a limit of apparently up to three. I am not sure why exactly this is, as the shifts should not have cyclical dependencies but it should be able to calculate them in-order). However, a `dy` argument can be passed to shift them by that length vertically.

■ *Note 1*
▲ *Note 2*

### 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. To change the

*Like this.*

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

It is recommended to reset the `notecounter` regularly, either per page:

```
#set page(
  header: {
    marginalia.notecounter.update(0)
  }
)
```

or per heading:

```
#show heading.where(level: 1): it => {
  marginalia.notecounter.update(0)
  it
}
```

## 3 Wide Blocks

The command `#wideblock[...]` can be used to wrap content in a wide block which spans into the margin-note-column. It is a bit cluttered, but is possible to use notes in wide blocks:◆, but make sure to set the `reverse` argument appropriately.

◆ *Voila*

`#wideblock(reverse: true)[...]`: The `reverse` option makes the block extend to the inside margin instead. Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt.

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

## 4 Headers and Page 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 ever try to `.update` the `marginalia._config` directly, this might easily break the code.

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
        #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(
    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%))))
})
```

## 5 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

## A.1 as-note

Format a counter like the note icons. Default numbering for notes.

```
notecounter.display(as-note)
```

◆

```
let i = 1
while i < 12 {
  [ #as-note(i) ]
  i = i + 1
}
```

◆ ● ■ ▲ ◇ ○ □ △ 1 2 3

**Parameters**

```
as-note(..counter: int ) -> content
```

## A.2 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 reccomended to change at least one of either `inner` or `outer` to be wide enough to actually contain text.

**Parameters**

```
configure(
  inner: dictionary ,
  outer: dictionary ,
  top: length ,
  bottom: length ,
  book: boolean ,
  flush_numbers: boolean ,
  numbering: str  function
)
```

**inner**　`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.

Default: (`far`: 5mm, `width`: 15mm, `sep`: 5mm)

**outer**    `dictionary`

Outside/right margins. Analogous to `inner`.

Default: `(far: `5mm`, width: `15mm`, sep: `5mm`)`

**top**    `length`

Top margin.

Default: `2.5cm`

**bottom**    `length`

Bottom margin.

These are not used for any of the Marginalia-functionality, they are only used when passed to `page_setup()`.

Default: `2.5cm`

**book**    `boolean`

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

Default: `false`

**flush_numbers**    `boolean`

Disallow note icons hanging into the whitespace.

Default: `false`

**numbering**    `str` or `function`

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

Default: `as-note`

## A.3 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
```

Page 6 of 7 MARGINALIA 18. November 2024

## A.4 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.5 note

Create a marginnote. Will adjust it's position downwards to avoid previously placed notes, to a limit. Typst starts to complain about the layout not converging at three notes that are in conflict, and the fourth note will overlap.

**Parameters**

```
note(
    numbered: boolean ,
    reverse: boolean ,
    dy: length ,
    body: content
)
```

**numbered**  `boolean`

Whether to put a mark.

Default: `true`

**reverse**  `boolean`

Whether to put it in the opposite (inner/left) margin.

Default: `false`

**dy**  `length`

Vertical offset of the note.

Default: `0pt`

## A.6 page_setup

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`

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

## A.7 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**   `boolean`

Whether to extend into the inside/left margin instead.

Default: `false`

**double**   `boolean`

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

Default: `false`

## A.8 notecounter   `counter`

The counter used for the note icons.

*Mostly internal.*

It is reccomended to reset this counter regularly if the default symbols are used, as after eight notes it will start to number them.

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