Margin Notes

Setup

`drafting` exists in the official typst package repository, so the recommended approach is to import it from the `preview` namespace:

```
#import "@preview/drafting:0.2.1"
```

Margin notes cannot lay themselves out correctly until they know your page size and margins. By default, they occupy nearly the entirety of the left or right margin, but you can provide explicit left/right bounds if desired:

```
// Example:
// Default margin in typst is 2.5cm, but we
want to use 2cm
// On the left
#set-page-properties(margin-left: 2cm)
```

The basics

```
#lorem(20)
#margin-note(side: left)[Hello, world!]
#lorem(10)
#margin-note[Hello from the other side]
#margin-note[When notes are about to overlap, they're
automatically shifted]
#margin-note(stroke: aqua + 3pt)[To avoid collision]
#lorem(25)
#margin-note(stroke: green, side: left)[You can provide
two positional arguments if you want to highlight a
phrase associated with your note.][The first is text
which should be inline-noted, and the second is the
standard margin note.]

#let caution-rect = rect.with(inset: 1em, radius: 0.5em)
#inline-note(rect: caution-rect, fill:
orange.lighten(80%))[
  Be aware that `typst` will complain when 4 notes
overlap, and stop automatically avoiding collisions when
5 or more notes
  overlap. This is because the compiler stops attempting
to reposition notes after a few attempts
  (initial layout + adjustment for each note).

  You can manually adjust the position of notes with `dy`
to silence the warning.
]
```

Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magnam aliquam quaerat. Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do. Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magnam aliquam quaerat voluptatem. Ut enim aeque doleamus. You can provide two positional arguments if you want to highlight a phrase associated with your note.

Hello, world!

Hello from the other side

When notes are about to overlap, they're automatically shifted

To avoid collision

The first is text which should be inline-noted, and the second is the standard margin note.

Be aware that `typst` will complain when 4 notes overlap, and stop automatically avoiding collisions when 5 or more notes overlap. This is because the compiler stops attempting to reposition notes after a few attempts (initial layout + adjustment for each note).

You can manually adjust the position of notes with `dy` to silence the warning.

Adjusting the default style

All function defaults are customizable through updating the module state:

```
#lorem(14) #margin-note[Default style]
#lorem(10)
#set-margin-note-defaults(stroke: orange, side: left)
#margin-note[Updated style]
#lorem(10)
```

Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut. Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do. Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do.

Default style

Updated style

Even deeper customization is possible by overriding the default `rect`:

```
#import "@preview/colorful-boxes:1.1.0": stickybox

#let default-rect(stroke: none, fill: none, width: 0pt,
content) = {
  set text(0.9em)
  stickybox(rotation: 30deg, width: width/1.5, content)
}
#set-margin-note-defaults(rect: default-rect, stroke:
none, side: right)

#lorem(20)
#margin-note(dy: -5em)[Why not use sticky notes in the
margin?]

// Undo changes from this example
#set-margin-note-defaults(rect: rect, stroke: red)
```

Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magnam aliquam quaerat.

Why not use sticky notes in the margin?

Multiple document reviewers

```
#let reviewer-a = margin-note.with(stroke: blue)
#let reviewer-b = margin-note.with(stroke: purple)
#lorem(10)
#reviewer-a[Comment from reviewer A]
#lorem(5)
#reviewer-b(side: left)[Reviewer B comment]
#lorem(10)
```

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

Comment from reviewer A

Reviewer B comment

Inline Notes

```
#lorem(10)
#inline-note[The default inline note will split the
paragraph at its location]
#lorem(10)
#inline-note(par-break: false, stroke: (paint: orange,
dash: "dashed"))[
  But you can specify `par-break: false` to prevent this
]
#lorem(10)
```

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

The default inline note will split the paragraph at its location

Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do. But you can specify `par-break: false` to prevent this Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do.

Hiding notes for print preview

```
#set-margin-note-defaults(hidden: true)

#lorem(20)
#margin-note[This will respect the global "hidden" state]
#margin-note(hidden: false, dy: -2.5em)[This note will
never be hidden]
// Undo these changes
#set-margin-note-defaults(hidden: false)
```

Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magnam aliquam quaerat.

This note will never be hidden

Outline of all notes

```
#note-outline()
```

List of Notes

Positioning

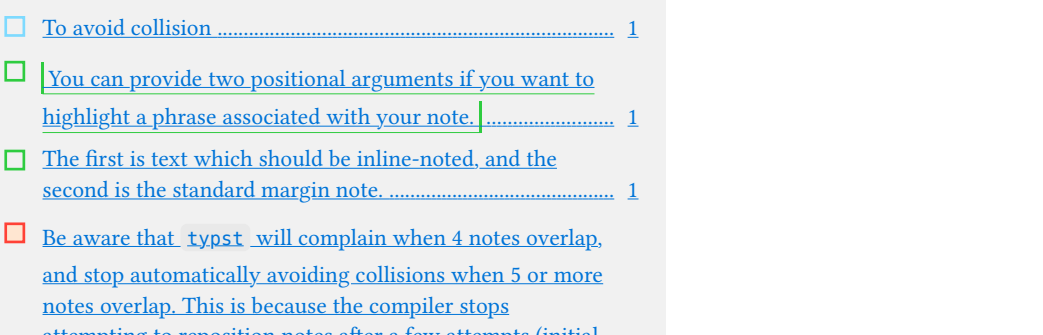
Precise placement: rule grid

Need to measure space for fine-tuned positioning? You can use `rule-grid` to cross-hatch the page with rule lines:

```
#rule-grid(width: 10cm, height: 3cm, spacing: 20pt)
#place(
  dx: 180pt,
  dy: 40pt,
  rect(fill: white, stroke: red, width: 1in, "This will
originate at (180pt, 40pt)")
)

// Optionally specify divisions of the smallest dimension
to automatically calculate
// spacing
#rule-grid(dx: 10cm + 3em, width: 3cm, height: 1.2cm,
divisions: 5, square: true,  stroke: green)

// The rule grid doesn't take up space, so add it
explicitly
#v(3cm + 1em)
```



Absolute positioning

What about absolutely positioning something regardless of margin and relative location?

`absolute-place` is your friend. You can put content anywhere:

```
#context {
  let (dx, dy) = (here().position().x,
here().position().y)
  let content-str = (
    "This absolutely-placed box will originate at (" +
repr(dx) + ", " + repr(dy) + ")"
    + " in page coordinates"
  )
  absolute-place(
    dx: dx, dy: dy,
    rect(
      fill: green.lighten(60%),
      radius: 0.5em,
      width: 2.5in,
      height: 0.5in,
      [#align(center + horizon, content-str)]
    )
  )
}
#v(0.5in)
```

This absolutely-placed box will originate at (76.4pt, 2964.67pt) in page coordinates

The "rule-grid" also supports absolute placement at the top-left of the page by passing `relative: false`. This is helpful for "rule"-ing the whole page.