Floats.

Objectives:

By the end of this chapter, you should be able to:

- Use `float` to build simple layouts
- Define what document flow is and how floating changes the document flow
- Use `clear` to bring elements back into the document flow

Floats + Clearing

Another way to lay out a page, used specifically for moving elements to a certain side of the page, involves floats. Floats are used a bit less now that there are more advanced (and easier) means of laying out a page, but they are important to know about.

In order to understand floats, we first need to understand the idea of `document flow`. From here:

> The document flow is the model by which elements are rendered by default in the CSS specifications. In this model, elements are rendered according by their default display rule. In other words, block-level elements are displayed on a new line and inline elements on the same line. Everything is stacked in an ordered way from top to bottom.

When we float an element, we remove it from the document flow, let's check out this example:

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title>Document</title>
    <style>
        header {
            background: red;
        }

        article {
            background: green;
            width: 65%;
        }

        aside {
            float: left;
            background: yellow;
            width: 35%;
        }
```

```
        footer {
            background: brown;
        }
    </style>
</head>
<body>
    <header>
        header
    </header>
    <section>
        <aside>
            Sidebar
        </aside>
        <article>
            <div>Lorem ipsum dolor sit amet, consectetur adipisicing elit. Deserunt enim porro praesentium in adipisci iste debitis reiciendis ipsum. Minima harum, quisquam consequuntur nihil, error beatae ea dolorem blanditiis molestias veniam.</div>
            <div>Quasi fuga provident neque repudiandae quo sed reprehenderit vitae vel voluptatibus laboriosam quae eveniet quibusdam molestiae distinctio quia saepe aspernatur, fugiat ipsum expedita repellendus, molestias harum? Sed ex nostrum id.</div>
            <div>Fugiat neque, veritatis soluta modi voluptate hic vel inventore, quod quaerat ad itaque enim aut sint quo earum! Magnam possimus, autem dolorum laboriosam culpa quos, amet nostrum? At, maiores repellat.
</div>
            <div>Ipsam, nulla. Laboriosam quam vero quas molestiae maiores. Soluta distinctio officiis placeat necessitatibus fuga porro quis odio, odit earum cupiditate natus iusto, ut alias, voluptatem amet maxime repudiandae architecto deleniti.</div>
            <div>Dolorum debitis saepe, numquam sunt, et eum atque deleniti dolore, culpa provident dolor nesciunt dignissimos. Corrupti quaerat dicta aliquam voluptatum? Quaerat cumque odio, maiores, nesciunt dolores enim suscipit earum sequi!</div>
            <div>Quos sit non deleniti fuga expedita quisquam unde asperiores. Accusantium assumenda consequuntur ad dicta odit molestiae praesentium facere impedit illo delectus nisi, quia asperiores dolorum necessitatibus saepe a deserunt vero!</div>
        </article>
    </section>
    <footer>
        Footer
    </footer>
```

```
</body>
</html>
```

Open this page up in your browser, and you should see that the sidebar is literally floated to the left. While the content in the `article` respects the sidebar, the `article` itself does not; instead, it looks as though the side bar is sitting right on top of the `article`.

Sometimes this is what you want, but sometimes you might want an element adjacent to a float to move down to the next line. In other words, you might want the element to be **cleared** below the float. To see what this looks like, try setting `clear: both;` on the `article`. How does this change the layout?

Next, let's make a two-column layout by floating the `article`. The `aside` is floated to the left, so let's remove the `clear` property on the `article` and instead set `float: right;` on it.

When you do this, you'll see that you've got two columns, but there's a problem: the `aside` is much shorter than the `article`, because it has less content! Consequently, the yellow background on the left is much shorter than the green background on the right. How can we fix this? If you know the exact height you need the `aside` to be you could set the `height` property, but very often you don't know what the height needs to be, as it may depend on the screen size. In this case, there are a couple of workarounds. Here's the approach we'll use:

1. Set the `padding-bottom` to `1000px` (or some other large number) on the `aside`. This makes the `aside` tall enough: in fact, now it's too tall!
2. Set the `margin-bottom` to `-1000px` (or some other large number) on the `aside`.
3. Set the `overflow` property to `hidden` on the `section`.

(These last two steps ensure that the footer is in the right place, and that the extra height from the `aside` gets hidden.)

If you think this feels like a bit of a hack, you're not alone. With the advent of Flexbox (which will get to very soon), some people argue that floats should be avoided. However, they're still quite common, so it's important to be familiar with them and know how to deal with some of their quirks.