Positioning

The CSS Box model has every element as a box, contained within the parent container box.

The `position` css property allows us to change that last clause

Positions

- static
- relative
- absolute
- fixed
- sticky

position: static

`static` is easy to describe, because you've been doing it all along.

Elements are maintained within the document flow (distinct from text flow!).

The `top` / `bottom` / `left` / `right` CSS properties do nothing

position: relative

`relative` maintains the **space** for the element in the document flow, but actually positions the element relative to that space.

We will discuss `stacking context` later, but this creates a new stacking context.

You rarely want to move the element using `position: relative;` (though it does occasionally happen).

More often you want to make a containing element "positioned", so it changes the behavior of `position: absolute;`

position: absolute

`absolute` pulls the element completely out of the document flow - space is not left for it.

- element now defaults to height/width of contents, even as block element

The `top` / `bottom` / `left` / `right` CSS properties allow you to place the element `

placing an absolute element

`top` / `bottom` / `left` / `right` will place the nearest side of the element that distance from the listed side.

- e.g. `top: 5px;` will place the top of the element 5 pixels from the top of the positioned container
- e.g. `right: 10px;` will place the right side of the element 10 pixels from the right of the positioned container
    - This means you don't (normally) have to do math about the size of the container to position it

But what is the "positioned container"?

Positioned container

By default, `absolute` will be relative to the document.

`top: 0;` will position it at the top of the document (probably covering up whatever else is at the top of the document)

But if the element has an ancestor that has position set other than `static`, that ancestor is positioned, and the `absolute` element direction properties are relative to THAT ancestor.

Uses of absolute positioning

A "modal window" looks like a pop-up that covers the screen.

- Usually done as an absolute positioned element relative to the document

Putting overlays or tooltip-like effects

- Done as absolute positioned elements relative to a positioned container

position: fixed

`fixed` position elements are

- pulled from the document flow
- no space is given for the element
- placed relative to the document
    - like an absolute with no positioned container

These elements remain in that position *relative to the viewport*

- e.g. a top menu will always be at the top even if you scroll

Fixed position issues

There are issues with fixed positions

- can get in the way of other elements
    - collapsing can help, but complexity goes up
- can stutter on heavy scroll

position: sticky

`sticky` elements start normal, but then behave like fixed while their container is still in the viewport but their normal position isn't

- position is relative to a "scrolling" ancestor, which is confused between browsers
- e.g. a big table might want the header (or section header) to always be visible while part of the table is
    - but it breaks if the wrong part is horizontally scrollable