Positioning

The CSS Box model has every element as a box

- contained within the parent container box
- `position` css property can change 'contained'

Positions

- static
- relative
- absolute
- fixed
- sticky

position: static

`static` is what you've been doing it all along

Elements are maintained within the document flow

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

- to an element with `position: static;`

position: relative

- maintains **space** for element in document flow
- positions the element relative to that space
- creates a new **stacking context**
    - That's for later

Rarely move elements using `position: relative;`

- it does occasionally happen

More often used to make element "positioned"

- Changes behavior of `position: absolute;` on descendants

position: absolute

`absolute` pulls the element out of the document flow

- No space is left for it
- Element now defaults to height/width of contents, even as block element

You can place the element "over" other elements

- using `top` / `bottom` / `left` / `right`

placing an absolute element

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

- Ex: `top: 5px;` places top of element 5 pixels from top of **positioned container**
- Ex: `right: 10px;` places right side of element 10 pixels from right positioned container
    - Result: Don't (normally) have to do math about size of container to position it

But what is the "positioned container"?

Positioned container

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

- Ex: `top: 0;` positions element at top of document
- probably covering up the top of the document

When an ancestor element has a non-`static` position

- that ancestor element is **positioned**
- `absolute` element direction properties are relative to THAT ancestor
- Relative to "nearest positioned ancestor"
    - "nearest" means "closest relation"
    - "parent" is closer than grandparent

Uses of absolute positioning

Show "over" other content

- Use `absolute`
- Often have to position an ancestor element too

Examples:

- Overlay menus
    - Dropdown menus
    - Slide-in menus
- Tooltip-like effects
- Older way to do "modal windows"

position: fixed

`fixed` position elements are

- Pulled from the document flow
- No space is given for the element
- Placed relative to the document
    - Like `absolute` with no positioned container

Remain in position *relative to the viewport*

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

Fixed position issues

There are issues with fixed positions

- Can get in the way of other elements
    - Ex: Hiding content because overlap
    - Collapsing can help, but complexity goes up
- Can stutter on heavy scroll

position: sticky

`sticky` elements start normal while "on screen"

- When normal position in viewport
    - `static` behavior
- When normal position out of viewport
    - And container is IN viewport
        - `fixed` behavior
    - And container OUT of viewport
        - `static` behavior (off screen)

Sticky business

- Position is relative to a "scrolling" ancestor
    - Different browsers = different behavior
- Ex: a big table wants header (or section header) always visible while part table is visible
    - Breaks if wrong part is horizontally scrollable

Summary - Practical positioning

- `static` is normal
- `relative` to create **positioned container**
- `absolute` to put "over" other content
    - Often involves positioned ancestor
- `fixed` to keep on screen when scroll
    - Can cover content unexpectedly
- `sticky` for section headers when scroll
    - Can cover content unexpectedly
    - Has issues with horizontal

Summary - relative positioning

- Element is **positioned**
  - All non-`static` are positioned
  - Relative used if that is sole point
- Keeps space for element
- Allows offset
  - using `top/right/bottom/left` properties
- Offscreen content still IN document
  - impacts a11y

Summary - absolute positioning

- Space NOT reserved in document flow
- Content will visually overlap
- Position relative to **positioned container**

Summary - fixed positioning

- Space NOT reserved in document flow
- Placed relative to viewport
    - NOT positioned container
- Used for visible headings/menus on scroll
- Can cover elements unexpectedly

Summary - sticky positioning

- Space IS reserved in document flow
- Sometimes `static`, sometimes `fixed`
- Keeps section headers onscreen while scroll
- Based on container (parent) being on screen
- Can cover elements unexpectedly
- Can get confused with horizontal scrolling