CSS Refresher

Display properties: determine how elements are displayed in the browser

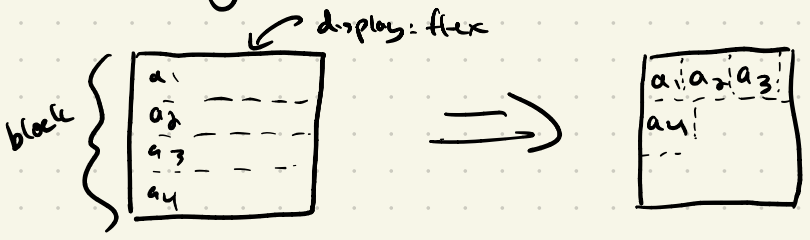
- ↳ **block** → Takes up full width of parent container
- ↳ **inline-block** → inline, but can use width and height properties
- ↳ inline → Only take up exact amount of space it needs
- ↳ none → Takes it out of the document flow

* Html elements have default display properties

* display: inline does not respect top and bottom margin / paddings

Flex

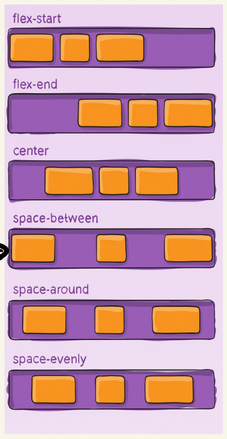
↳ display and create a flex container as the element. The container acts as block



* By adding display: flex, child elements act as "inline" despite themselves being block display

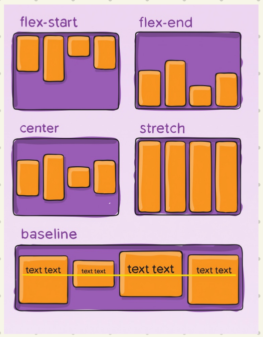
- **justify-content**: Defines the layout along the main axis. Must be on a flex container. The default is flex-start.

↳



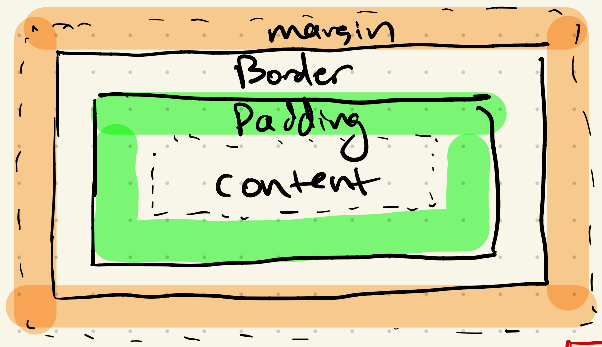
- **align-items**: Defines the layout for how items are placed on the cross axis

↳

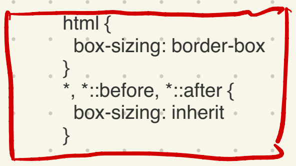


good for menus, toolbars

CSS Box Model



⇒ box-sizing:
- **border-box**: padding / border folded in.
- content-box: default. Padding and border included in size.



```
html {
  box-sizing: border-box
}
*, *::before, *::after {
  box-sizing: inherit
}
```

* avoid B.S

CSS Refresher (cont.)

- some fonts require anti-aliasing to smooth them out. To add this, use the following (often in body)

↳
```
body {
    -webkit-font-smoothing  : antialiased
    -moz-osx-font-smoothing : grayscale
}
```

- **position:** an element is said to be positioned if anything other than static
    - static: the default. Ignores top, right, left, bottom
    - absolute: element removed from original layout and positioned relative to nearest positioned parent by the positioning properties
    - fixed: element removed from original layout and positioned relative to the window
    - relative: stays in original position. Top, right, left, bottom properties nudge it from original (the element is positioned "relative" to its normal). This creates a stacking context when z-index is not auto!