

# **CSS Layouts**

**Michael Chang**  
**Spring 2023**

# Plan for today

## **Last time: styling elements with CSS**

Selectors, fonts, colors

## **CSS for page layouts**

block vs. inline

The box model

## **Intro to Flexbox**

Using Flexbox for more interesting layouts

# Pseudoclasses

**Classes automatically applied to elements**

**Example: styling links**

`a`: all links

`a:link`: unvisited links

`a:visited`: visited links

`a:hover`: links when hovered

# Page flow

## Recall: some elements take up full width

`<p>`, `<h1>`

## Others lay out left to right

`<a>`, `<strong>`

## Block vs. inline

block: has width and height, defaults to full width

inline: can't set width or height, can't have block children

## Exceptions

`<img>`: can be sized, but it's inline

# display CSS property

## Override default flow

**Values: block, inline,**

none: hide the element completely

inline-block: inline, but has width/height (like `<img>`)

## **`<div>` and `<span>`**

`<div>`: generic block element

`<span>`: generic inline element

No semantics or default presentation

Useful for custom styling, layouts

...But don't use when more precise element exists

# Box model

## Add spacing within and between elements

**margin**: outside of border/background

**padding**: within border

## Shorthand properties

Can have 1, 2, or 4 values

1: all

2: top+bottom, left+right

4: top, right, bottom, left

Can set separately

E.g. margin-top

### Box Model



# Box model quirks

## What do width and height define?

Default: content box: doesn't count padding or margin

Can also use border box: include padding + border, not margin

```
box-sizing: border-box;
```

## Change globally

```
:root {  
  box-sizing: border-box;  
}  
* {  
  box-sizing: inherit;  
}
```

▼ Box Model



# Box model quirks

## Inline elements

Vertical margin/padding don't work

Use `line-height` to set vertical spacing

## Margin collapsing

Vertical margins of adjacent elements "collapse"

Uses largest margin

## Default margins

Many elements have default margins

E.g. `<p>`, `<h1>`, `<body>`



# More margin stuff

## margin can be negative

Overlap with previous element

Counteract another element's margin

## auto margin

Horizontally center element in container

Can't vertically center

E.g. `margin: 0 auto;` doesn't work

# So far

## **We can**

- Set font and text styles, colors

- Control element spacing

- Create simple layouts

## **But some things are hard or impossible**

- Vertically center elements

- "Navbars" with left and right sections

- Grid layouts (e.g. news sites)

- Static footer at bottom of window

# So far



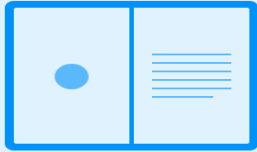
Block elements



Inline elements

# Flexbox

**Flexbox** can solve all of these problems



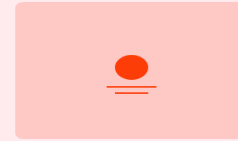
**Split-screen**



**Sidebar**



**Sticky footer**



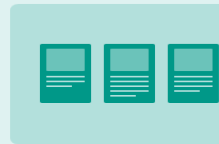
**Centering**



**Fluid grid**



**Collection grid**



**Equal-height modules**

Credit to Victoria for this slide

# display: flex

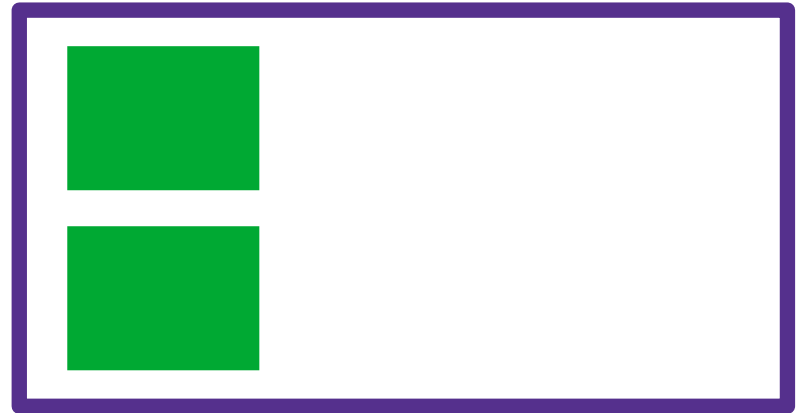
## Completely changes how element is laid out

The element becomes a "flex container"

Its (direct) children become "flex items"

## Lays out flex items in a row or column

Default: row. Use `flex-direction: column;` to change



# Flexbox properties

## **justify-content: layout along the "main axis"**

main axis = flex-direction

flex-start, flex-end, center

space-between: equal space between flex items

space-around: also leave space on the ends

## **align-items: layout along the cross axis**

cross axis = opposite of flex-direction

flex-start, flex-end, center

# Summary

## **CSS for page layouts**

block vs. inline

The box model

## **Intro to Flexbox**

Using Flexbox for more interesting layouts

## **Next time: more CSS**

More Flexbox properties

position property

CSS odds and ends