More DOM and events

Michael Chang
Spring 2023
Plan for today

More about JS events
  Default behavior, propagation, form submit

Another example
  Adding/removing elements
  Communicating between classes
DOM events aside

Older-style events

```html
<button onclick="/* js code */">...</button>
```

```javascript
button.onclick = /* js function */;
```

Best practice: don't use these

These are older, less flexible
First version breaks separation (HTML = markup, JS = logic)
Use `addEventListener` instead
HTML forms

<form>: wrap inputs/buttons

Historically used to send data to server

Inside a form

<button> defaults to submit (use type=”button” to override)
<input name=”foo”>

Can use instead of id (unique within form but not whole page)
<button type=”reset”>
formElem.reset()

Clear all inputs in the form
Event default behavior

Some actions have default behaviors
  Clicking a link or submit button
  Typing text in an input

Event handler called before default
  Default action will happen afterward

To prevent default action
  Call event.preventDefault()
Accessing DOM Elements

document.querySelector(selector)

Find an element anywhere on page (returns first match)

selector is a CSS selector

  #foo: element with ID foo
  .bar: element with class="bar"
  li: an <li> element

We'll learn more about selectors next time
Aside: frontend frameworks/libraries

Some are very useful for large projects
E.g. React uses HTML-like syntax directly in JS and builds the nodes for you

Some are a product of an older time
jQuery used to be very popular
Provides many functions for accessing HTML/CSSS attributes, defining events, etc.
   E.g. $("selector") is like querySelector
But now unnecessary given broad support for standard techniques,

Best practice: avoid jQuery
   There's still a lot in the wild (SO answers, etc.)
Summary

So far
  Lots of JavaScript

Before next time
  assign1

Next time
  Styling elements and page layout