# Forms in React

- What changes with forms in React?
    - What doesn't change?
- How do we do validation?
- How to handle a11y with reusable Components?

# React has two ways to manage forms

- Controlled
    - React handles the values of field elements
        - `<input>`, `<select>`, `<textarea>`, etc
- Uncontrolled
    - Browser manages values of field elements

Both have their uses

- Controlled takes more effort
- Uncontrolled has limitations

We will do only Controlled forms

## Generated HTML is mostly the same

- You CAN skip `<form>`
  - No actual submission, after all
  - But don't want to
  - `onSubmit` is still useful

# Validation Checks

- Check form field values:
    - `onInput()`
    - `onChange()`
        - React makes `onChange()` on text input act like `onInput()`
    - `onSubmit()`
    - I suggest saving to local state variables
        - Like "in progess" shown in previous class
- Remember to `.preventDefault()` in `onSubmit()`
    - Even on success! Just save info in state

## Calculate Derived State and Render

- Local state variables have form field values
- "Should I show an error message?"
    - is a **derived state**
    - Calculate on render, do not store as state
    - Exception: Fields that default to invalid state
- Conditionally render error messages
    - based on those derived state(s)
- Error messages are just Conditionally Rendered

## Accessibility Concerns

- Components are meant to be **reusable**
    - Can use repeatedly in application
    - Can use in other applications
- Forms need labels for a11y
    - Labels often rely on `id` attribute
    - `id` are NOT reusable
- How to resolve?

## `for` in JSX

- Just like `class` attribute is changed to `className`
- `for` attribute of `<label>` is changed to `htmlFor`
    - Still refers to the id attribute of the element

## Resolving the uniqueness problem

- React has a `useId` hook
- Gives a unique value for each component
- Use as a prefix if you have multiple fields
- Can't be targeted by CSS
    - Use className for that part

```
import { useId } from 'react';

function MyComponent() {
  const id = useId();

  return (
    <form>
      <label htmlFor={`${id}-name`}>Name</label>
      <input id={`${id}-name`}/>
    </form>
  );
}
```