Scrimba React

import ReactDOM from ‘react-dom’

- whenever that particular file will render something to the page

import React from ‘react’

- in any file where JSX will be used

convention to have each file contain one React component

export default ‘MyFileName’

- any time that file needs to be used by another file in the project
- to import another file in same directory: import MyFileName from “./MyFileName”

App component mainly just has a tree with components nested

- not really HTML, should just render components from other component files

index.js can just look kind of like this:

```
import React from "react"
import ReactDOM from "react-dom"
import App from "./App"
ReactDOM.render(<App />, document.getElementById("root"))
```

to style elements in JSX, use “className=“myclassname”” inside element tags

- only for React elements (not components)

wrap anything in an element you want interpreted as JS in curly braces { }

- ex: <p> hello {myConst } </p>

class based components must have a render method

class based components reference props with this.props.myProps; functional components do not

props is immutable, state is not

constructor method → needed in class based component to maintain state

- initializes parts of class
- values are initialized
- super() → goes to parent class (i.e. React.Component) & grabs a bunch of shit so we can use it

all event handlers in html elements need to be camelCase

every time you use a class method that involves setState, need to bind the method to the class

- ex: `this.handleClick = this.handleClick.bind(this)`

Lifecycle Methods

componentDidMount() → runs when the component hits the DOM

- useful for waiting until mounted to throw data in

shouldComponentUpdate() → helps React decide if it’s worth re-rendering

componentWillUnmount() → ex: get rid of event listener; clean up code before code disappears

getDerivedStateFromProps() → need to write static before

- maybe dont need to know

getSnapshotBeforeUpdate() → create backup of current data of whats going on now

wrap string property name inside of object in square brackets

- ex:
  ```
  this.setState({
      [event.target.name]: event.target.value
  })
  ```

methods inside a class component can use arrow function, negating need to bind methods to state

React Context API can be used instead of Redux often times; simpler