React 2

CS571: Building User Interfaces

Cole Nelson

Recap of React 1

What did we like?

What did we not like?

What is React?

Definition: Also called ReactJS, React is a JS library for building user interfaces.

- Developed by Facebook, dating back to 2010.
- Started as an internal development tool, then open-sourced in 2013.

More on the history of React

What are the benefits of a Virtual DOM?

- Incredibly fast, as only what is updated in the Virtual DOM is updated in the real DOM.
- Abstracts away interactions with DOM; makes programming more *declarative*.
- Used in React and vue.js; Angular does its own thing.

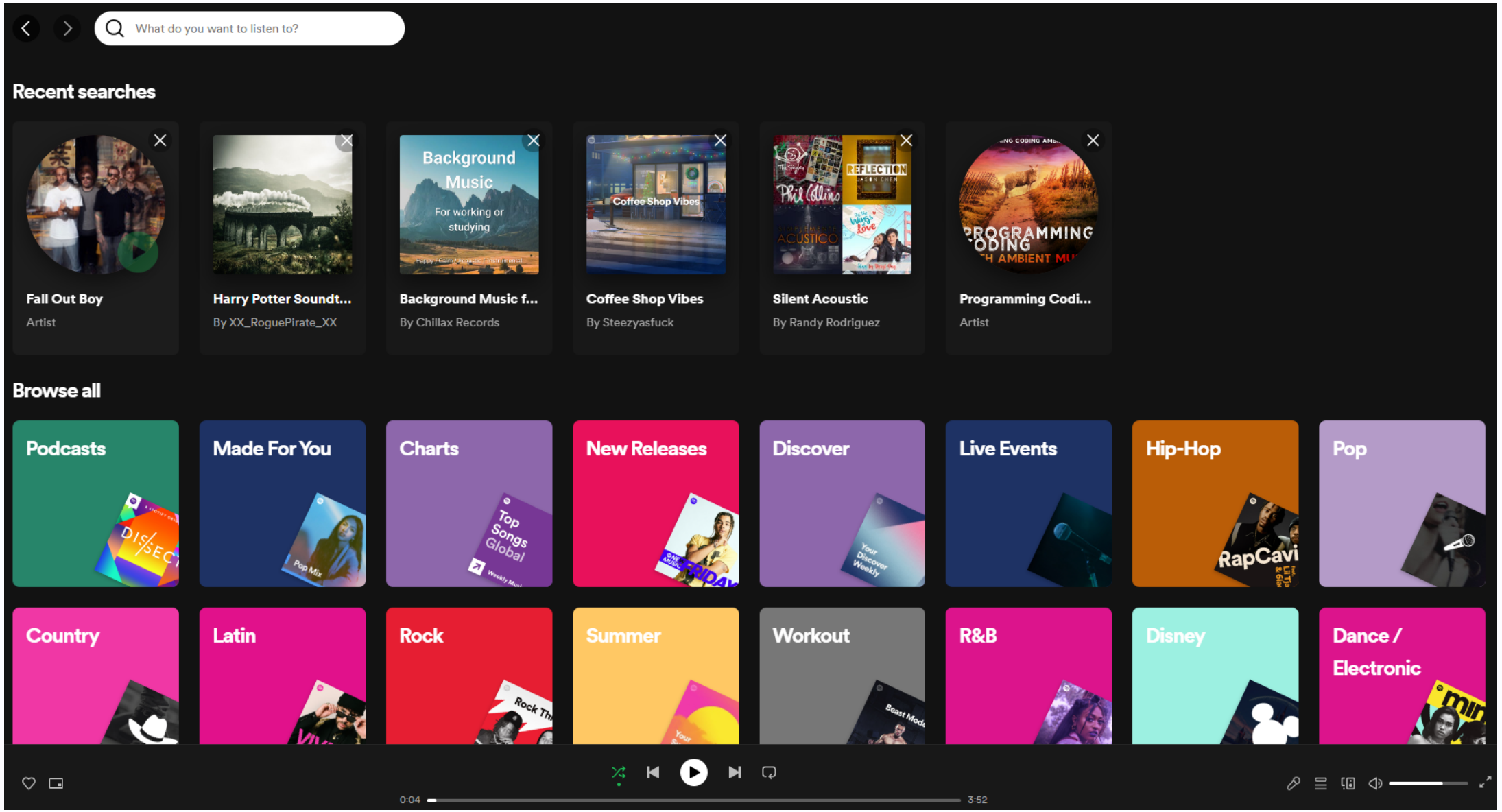
React Essentials

Every "thing" is a component.

Every component is a function, inheriting `props` and maintaining an internal `state`.

What defines a component?

- Similar question: *what defines a class in Java?*
- Some re-usable piece of the interface.
- May have many children, but only one parent.


What do you want to listen to?
Recent searches
Fall Out Boy
Artist
Harry Potter Soundt...
By XX_RoguePirate_XX
Background Music
For working or studying
Background Music f...
By Chillax Records
Coffee Shop Vibes
Coffee Shop Vibes
By Steezyasfuck
Silent Acoustic
By Randy Rodriguez
Programming Codi...
Artist
Browse all
Podcasts
Made For You
Charts
New Releases
Discover
Live Events
Hip-Hop
Pop
Country
Latin
Rock
Summer
Workout
R&B
Disney
Dance / Electronic
0:04
3:52

Example of a React Component

This React component displays Hello World on the webpage using JSX.

```
function Welcome() {
  return <h1>Hello World!</h1>;
}
```

Babel transpiles JSX into JS and HTML counterparts.

StackBlitz

`useState` Hook

Used to maintain state! Takes an initial value as an argument. Returns a pair of the *read-only* state value and a *mutator* function.

Always use the mutator function to modify state.

Never modify the state directly.

```
const [name, setName] = useState("James");
```

`useState` Hook

```
const [name, setName] = useState("James");
```

We can use `name` to *read* the name and `setName` to *change* the name...

```
console.log(name);
setName("Jim");
console.log(name); // still James???
```

`setName` happens *asynchronously*. See `useEffect`.

StackBlitz

`useEffect` Hook

Used to perform an action on component load or state change. Takes a callback function and an array of state dependencies as arguments.

```
useEffect(() => {
  alert("The page has been reloaded!");
}, [])
```

```
useEffect(() => {
  alert("You changed your name to " + name);
}, [name])
```

Imports and Exports

Functions must be exported to be used in other files, e.g. `export default FindMyBadgers`.

This can then be imported, e.g. `import FindMyBadgers from "./components/FindMyBadgers"`

Imports and Exports

Functions can export one object as default, other exports can be non-default, e.g. `export HelperFunc2`.

These can then be imported, e.g.

```
import { HelperFunc2 } from "./utils/HelperFuncs"
```

Imports and Exports

Imports from 3rd party libraries do *not* use relative pathing, e.g.

```
import React from 'react'
```

```
import { Container } from 'react-bootstrap'
```

We Made a React App!

Find my Badgers using randomuser.me

StackBlitz Solution

Your turn!

Create an app to display the U.S. presidents! A `WallOfPresidents` should display many `President`.

Each `President` should display their name and political affiliation.

Use data from `https://cs571.org/s23/week3/api/data`

Clone from here.

What will we learn today?

- What are the differences between HTML and React-Bootstrap components?
- How does React render, actually?
- How does the `key` property work?
- Why do we use `useEffect`, actually?
- What are the differences between controlled and uncontrolled components?
  - How can we use `useRef`?
- What is the React, NPM, and Node community?

HTML vs React-Bootstrap Components

Think of it like `int` vs `Integer` in Java!

HTML components (e.g. `button`, `input`) are...

- From the 90s!
- Highlighted in *dark blue*.

3rd-party components (e.g. `Button`, `Form.Control`)...

- Require importing.
- Highlighted in *light blue*.

React Component Rendering

A render consists of running the function and updating the return value in the virtual DOM.

- A component re-renders every time its state changes.
- All child components will also re-render.

Diffing reduces changes made to the real DOM.

We can reduce re-renders using *memoization* discussed later in the semester.

React `key` Prop

The `key` prop is used by React to speed up rendering.

- Always use a *unique* key for the *parent-most* element rendered in a list.
- This key needs to be *unique among siblings*.
- This key should *not* be the index of the item (e.g. what if the order changes?)

Learn More

Loading Data -- The INCORRECT Way

```
export default function App() {
  const [presidents, setPresidents] = useState([]);
  fetch('https://cs571.org/s23/week3/api/data', {
      headers: {
        "X-CS571-ID": "bid_00000000000000000000"
      }
    })
  .then(res => res.json())
  .then(prezz => {
    setPresidents(prezz);
  });
  return // ...
```

StackBlitz

Loading Data -- The Correct Way

```
export default function App() {
  const [presidents, setPresidents] = useState([]);
  useEffect(() => {
    fetch('https://cs571.org/s23/week3/api/data', {
      headers: {
        "X-CS571-ID": "bid_00000000000000000000"
      }
    })
    .then(res => res.json())
    .then(prezz => {
      setPresidents(prezz);
    })
  }, []);
  return // ...
```

Handling Text Input

We can get user input using the HTML `input` tag or the React-Bootstrap `Form.Control` component.

We can get user input...

- in a *controlled* way using its `value` and tracking `onChange` events
- in an *uncontrolled* manner using `useRef`.

`useRef` Hook

Usually used to "reference" an input element.

```
const inputVal = useRef();
return (
  <div>
    <label for="myInput">Type something here!</label>
    <input id="myInput" ref={inputVal}></input>
  </div>
);
```

The value of a ref can be retrieved via its `current` property, e.g. `inputVal.current.value`

Controlled vs Uncontrolled Components

`useRef` is normally used to create a reference for an *uncontrolled* input component.

However, we can *control* an input component via its `value` and `onChange` properties.

Example of an uncontrolled input component.

Example of a controlled input component.

Controlled vs Uncontrolled: Pros & Cons

React generally recommends controlled components.

Controlled components can cause many re-renders, however uncontrolled components give you no control over the `onChange` property.

We'll practice using both.

Your turn!

Using the code you worked on at the beginning of lecture, add input fields for name and political party.

When an "Add" button is clicked, add that president to the wall of presidents.

How can we *permanently* add a president? We'll discuss this in React 3!

NPM & NodeJS Projects

React is a *library*. We use NPM & Node!

The `package.json` specifies details about the project.
The `package-lock.json` specifies specific details about dependencies.

`npm install` installs dependencies in `node_modules`.
`npm start` runs your webserver w/ hot-reloading!

Library Usage Assignment

Students must incorporate at least one additional, meaningful third-party library into at least one of their submissions and provide a short description about its use. Such third-party libraries could be jquery, typescript, axios, jest, eslint, and lodash; other meaningful third-party libraries are also acceptable.

Let's Explore NPM!

What makes a *good* package?

What did we learn today?

- What are the differences between HTML and React-Bootstrap components?
- How does React render, actually?
- How does the `key` property work?
- Why do we use `useEffect`, actually?
- What are the differences between controlled and uncontrolled components?
  - How can we use `useRef`?
- What is the React, NPM, and Node community?

Questions?