React Props and State

Please try and sit near the front!

Agenda

- Modules
- Props
- State Intro

Warm Up!

Create a Functional Component to render the following:

Things that happen when the UMD Alert test happens: *(Use h1)*

- I panic
- I look outside
- I remember I have been at UMD for 5 years and this happens every month

Warm Up!

Create a Class Component to render the following:

Things that happen when the UMD Alert test happens: *(Use h1)*

- I panic
- I look outside
- I remember I have been at UMD for 5+ years and this happens every month

Let’s Create an actual Module with our code in it.

Why do we like this better?

Rendering with Functions

- More formally called “Functional Components”

- Functions **must** be “pure”

Pro tip: Use self closing tags for now! <tag />

Rendering with Classes

class ComponentName extends React.Component{}

Make sure you have a render() method, where your return the JSX you would like to use.

Props

Now, when we call our components we can pass data through the properties (props).

They are read only!

<Item data = “Data”></Item> or <Item data = “Data” />

Props with Functional Components

Very similar to props with Class Components:

```
function PrintData(props){
    return ( <div>
            <p>{props.data}</p>
        </div>);
```

When to use *this*?

Functions have no concept of *this* unless you force them to!

- This is the reason your function should have a props parameter

With classes, always use this.props.data

Using Props for Images

You must first import the image, much like we do with other modules.

import <A Variable Name> from “./path”;

Then, you may use it with the variable name!

Arrays as Props

You may pass in arrays into your props, just as long as you surround it with {} as we do with all data:

```
<PrintData arr = {[1,2,3,4]}/>
```

Problems with Props

- You cannot change your props!

- Let’s try to update the date on the screen and see if props will update.
  - Spoiler: props will not automatically trigger changes to the view!

Trying to Update the DOM

```
function Tick(props) {
    return (
      <div>
        <h1>Hello, world!</h1>
        <h2>It is {props.date.toLocaleTimeString()}.</h2>
      </div>
    );

  }
  ReactDOM.render(
    <Tick date = {new Date()}/>,
    document.getElementById('root')
  );
  //setInterval(tick, 1000);
```

Try #2

```
function tick() {
  const element = (
    <div>
      <h1>Hello, world!</h1>
      <h2>It is {new Date().toLocaleTimeString()}.</h2>
    </div>
  );
  ReactDOM.render(
    element,
    document.getElementById('root')
  );
}

setInterval(tick, 1000);
```

Example from: ReactJS.org

State

this.state and the setState() method will actually dynamically update the view -- without us having to force the DOM to update.

By updating the view, React will see changes to the virtual DOM and update the actual DOM.

Rules:

- Do not directly modify the state data, use setState()
- Changes to state are merged
- Treat changes asynchronously! (We'll talk about this more next class)

State vs Props

State: When we want to have data that is mutable

Props: To pass data to children Components (read only)

WTWAW (What To Walk Away With)

- Use props to pass information
- Use an image in react
- Use Arrays in props
- Create a state in a class component