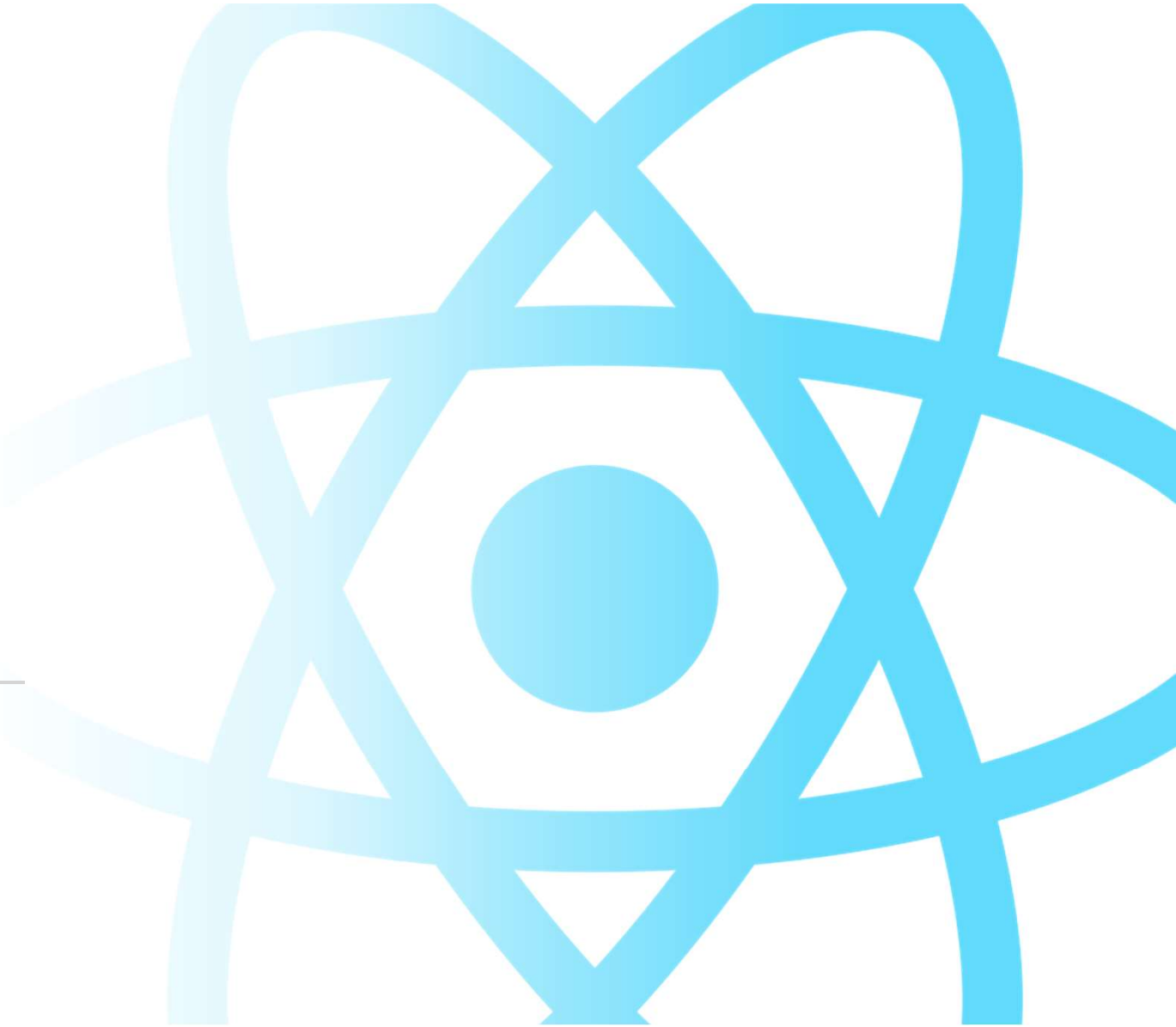
React Workshop

Let’s React!



Document Object Model (DOM)

- A programming interface for web document
- It allows you to change the document structure, style and content
- Allows languages such as JavaScript to interact with the page
- Ex: `document.getElementById("id").style.width = 100px;`

What is React?

- React is a library developed by Jorden Walke
  - Originally called “FaxJS”
- Was originally created because Walke felt that the DOM was slow.
- To fix this problem, React had a virtual DOM which tries to find the most efficient way to update the browser’s content.

How does it work?

- Everything placed in a `<div/>` in index.html
- `<div/>` acts like the body of the page
- React Components placed like HTML elements
- Components are placed in the virtual DOM, or ReactDOM
- Virtual DOM updates DOM

Components

- Reusable bits of code that are independent from everything else
- Serve as isolated JS functions that return HTML
- Can be viewed as objects
- How they look can be determined by states
- Comes in two types, Class components and Functional components

Class vs Functional

- Class based
  - To create components, you must extend Components
  - Has a render function which returns a React element
  - Uses a constructor to store states, uses functions to change states
- Functional based
  - Components made and exported as a React element with a single function
  - Before the React 16.8 Hook update, these were stateless components.
  - Uses hooks to control states

Class Based Component

```
import React from "react"
import Peach from "../assets/Peach.jpg"

class ClassComponent extends React.Component
{
    constructor(props)
    {
        super(props);
        this.state = {
            showMain: true
        };
    }

    switchMain()
    {
        this.setState({showMain: !this.state.showMain});
    }

    render()
    {
        return(
            <div style={{position: "absolute", transform: "translate(-50%, -50%)", top: "50%", left: "50%"}}>
                <h1 onClick={this.switchMain.bind(this)}>Click me to change the content below</h1>
                <div style={{width: "50rem"}}>
                    {this.state.showMain ? <p>Funny text here</p> : <img src={Peach} alt="Peach" style={{width: "100%"}}/>}
                </div>
            </div>
        );
    }
}
export default ClassComponent;
```

Functional Based Component

```
import { useState } from "react";
import Peach from "../assets/Peach.jpg"

const FunctionalComponent = () =>
{
    const[showMain, setDisplay] = useState(true);

    const switchMain = () => {
        setDisplay(!showMain);
    }

    return (
        <div style={{position: "absolute", transform: "translate(-50%, -50%)", top: "50%", left: "50%"}}>
            <h1 onClick={(switchMain)}>Click me to change the content below</h1>
            <div style={{width: "50rem"}}>
                {showMain ? <p>Funny text here</p> : <img src={Peach} alt="Peach" style={{width: "100%"}}/>}
            </div>
        </div>
    );
}
export default FunctionalComponent;
```

How are they placed?

You add them to your page like it is a HTML Component. It is as easy as that!

Class

```
import ClassComponent from './components/ClassComponent';
function App()
{
  return (
    <ClassComponent />
  );
}
export default App;
```

Functional

```
import FunctionalComponent from './components/ FunctionalComponent';
function App()
{
  return (
    <FunctionalComponent />
  );
}
export default App;
```

Component Parameters

Like functions, you can pass parameters to components as props. For class based you can think of these as parameters for a constructor.

```
import React from "react";

class ClassComponentWithParameters extends React.Component
{
    constructor(props)
    {
        super(props);
        this.state = {};
    }

    static getDerivedStateFromProps(props, state)
    {
        return {
            text: (props.text === undefined) ? "Default Text" : props.text,
            color: (props.color === undefined) ? "black" : props.color,
        };
    }

    render()
    {
        return(
            <div>
               <h1 style={{color: this.state.color}}>{this.state.text}</h1>
            </div>
        );
    }
}
export default ClassComponentWithParameters;
```

Functional Component With Parameters

```
const FunctionalComponentWithParameters = (props) =>
{
    return (
        <div>
            <h1 style={{color: (props.color === undefined) ? "black" :
        props.color}}>{(props.text === undefined) ? "Default Text" : props.text}</h1>
        </div>
    );
}
export default FunctionalComponentWithParameters;
```

How are they placed?

Components with “Hello World” in blue text

Class

```
import ClassComponentWithParameters from './components/
ClassComponentWithParameters';

function App()

{

  return (

    <ClassComponentWithParameters text="Hello World"
color="blue" />

  );

}

export default App;
```

Functional

```
import FunctionalComponentWithParameters from './components/
FunctionalComponentWithParameters';

function App()

{

  return (

    <FunctionalComponentWithParameters text="Hello World" color="blue" />

  );

}

export default App;
```

Time to code!

We are making a website using mostly class components. There will be two functional components just for exposure.

Why class components?

- It is easier to learn and understand states
- There are no performance difference between class and functional components
- Legacy code contains class components