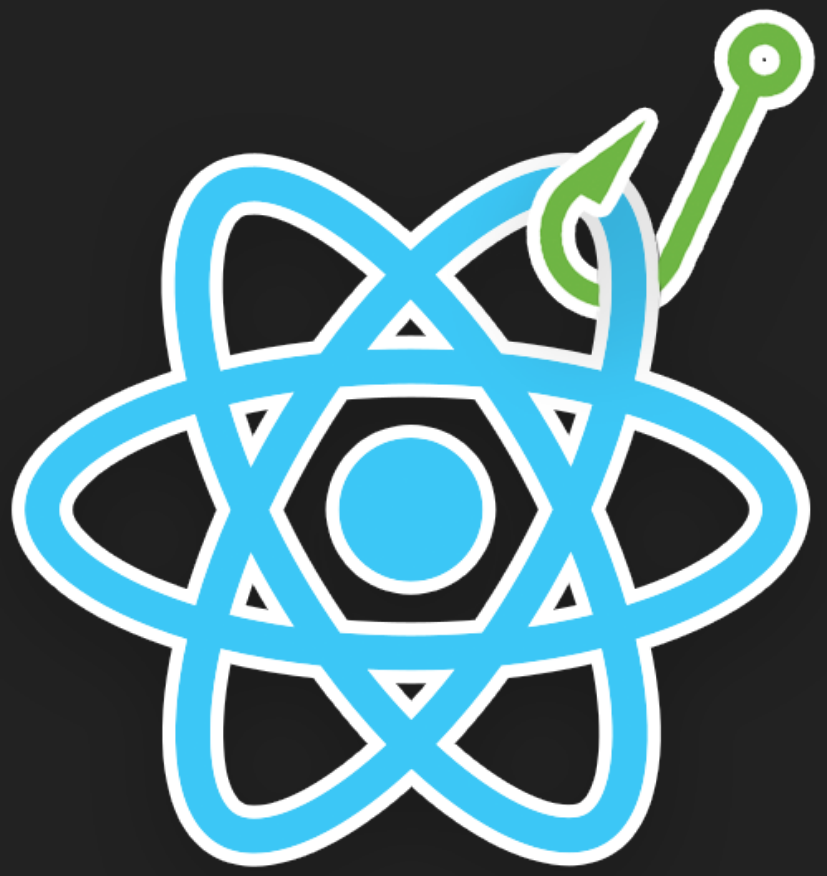
React Training

> Day Four

Understanding the context of React and Hooks



React Hooks

React Hooks

- Low level building blocks that give you special abilities
- Functions that always begin with `use`
- Requires React version > `16.8`

```
useSuperPower();
```

Rules while using Hooks

- Should only be called at the top of the **functional component**
- Doesn't work in callbacks or inside normal functions *// Exception: Custom Hooks*

Hooks Usage

```
function ComponentName() {
  useHook(); ✅

  const functionName = () => {
    useHook(); ❌
  }

  return <button onClick={() => useHook()❌ }>Submit</button>;
}
```

Basic Hooks

- useState
- useEffect
- useContext

Context

> Context provides a way to pass data through the component tree without having to pass props down manually at every level.

- Primarily used when some data needs to be accessible by many components at **different nesting levels**
- If you only want to avoid passing some props through many levels, component composition is often a simpler solution than context.

Setting up the Context

Creating a Context

```
const MyContext = React.createContext(defaultValue);
```

Subscribing to a Context

```
<MyContext.Provider value={/* some value */}>
{/* Child components who can access the context value */}
</MyContext.Provider>
```

useContext

```
const theme = useContext(MyContext);
```

Custom Hooks

```
function useCustomHookName() {
  const [state, setState] = useState(0);
  useEffect(() => {
    // fetch and update state
  });
  return state;
}
```

```
useYourImagination();
```

useRef

- Store references but don't trigger rerenders
- Grab HTML elements from the DOM

```
function App() {
  const myBtn = useRef(null);

  const clickIt = () => myBtn.current.click();

  return (
    <button ref={myBtn}>Click Me</button>
  )
}
```

Hooks I will not cover

- useReducer
  > Redux type state management
- useMemo
  > Opt in tool to optimize for performance
- useCallback
  > Opt in tool to optimize for performance
- useImperativeHandle
  > To add references to our React components (Rarely used)