Introduction to React Hooks

Introduction

What are React Hooks?

- They let us use features of React without having to define ES6 classes
- They use regular JavaScript functions
- Can access state and component lifecycle methods
- **Main Idea:** Complex components that couldn't otherwise be broken down because logic is stateful and can't be extracted to another function/component now can be broken down using Hooks

Simple Example of useState

```
4   const HooksExample = () => {
5       const [counter, setCount] = useState(0);
6
7       return (
8           <div className="App">
9               <header className="App-header">
10                  The button is pressed: { counter } times.
11                  <button
12                      onClick={() => setCount(counter + 1)}
13                      style={{ padding: '1em 2em', margin: 10 }}
14                  >
15                      Click me!
16                  </button>
17              </header>
18          </div>
19      )
20  }
21
22  export default HooksExample;
```

Some Common Built-in Hooks

useEffect

- Single API call that serves the purpose of componentDidMount, componentDidUpdate, and componentWillUnmount
- Then, when you use useEffect, you can use useState within it to set the state

Simple Example of useEffect

```
const HooksExample = () => {
    const [data, setData] = useState();

    useEffect(() => {
        const fetchGithubData = async (username) => {
            const result = await axios(`https://api.github.com/users/${username}/events`)
            setData(result.data)
        }
        fetchGithubData('lsurasani')
    }, [data])

    return (
        <div className="App">
            <header className="App-header">
                {data && (
                    data.map(item => <p>{item.repo.name}</p>)
                )}
            </header>
        </div>
    )
}

export default HooksExample;
```

Custom Hooks

- You can create your own hooks too!
- Standardize and encapsulate your code
- Create reusable logic and components

Simple Example of Custom Hook

```
1  import { useState } from 'react';
2
3  export const useInputValue = (initial) => {
4      const [value, setValue] = useState(initial)
5      return { value, onChange: e => setValue(e.target.value) }
6  }
```

useReducer

- An alternative to useState when there's complex state logic
- Allows you to update state given a certain type and new data to put in the state
- Takes in a specified reducer and an initial state
- Gives back a current state and dispatch function

```
3   const reducer = (state, action) => {
4       switch (action.type) {
5           case 'increment': {
6               return { ...state, count: state.count + 1, loading: false };
7               }
8           case 'decrement': {
9               return { ...state, count: state.count - 1, loading: false };
10              }
11          case 'loading': {
12              return { ...state, loading: true };
13              }
14          default: {
15              return state;
16          }
17      }
18  };
```

```
28  function Counter() {
29      const [{ count, loading }, dispatch] = useReducer(reducer, {loading: false, count: 0});
30      const onHandleIncrement = async () => {
31          dispatch({ type: 'loading' });
32          await delay(500);
33          dispatch({ type: 'increment' });
34      };
35      const onHandleDecrement = async () => {
36          dispatch({ type: 'loading' });
37          await delay(500);
38          dispatch({ type: 'decrement' });
39      };
```

Simple Example of useReducer

Usage with Forms

- Forms suddenly seem crazy easier to handle & read

```
4   const Form = () => {
5       const firstName = useInputValue('')
6       const lastName = useInputValue('')
7       const email = useInputValue('')
8       const password = useInputValue('')
9       const confirmPassword = useInputValue('')
10
11      return (
12          <div className="App">
13              <header className="App-header">
14                  <div>
15                      <input type="text" placeholder="First Name" {...firstName} />
16                  </div>
17                  <div>
18                      <input type="text" placeholder="Last Name" {...lastName} />
19                  </div>
20                  <div>
21                      <input type="text" placeholder="Email" {...email} />
22                  </div>
23                  <div>
24                      <input type="password" placeholder="Password" {...password} />
25                  </div>
26                  <div>
27                      <input type="password" placeholder="Confirm Password" {...confirmPassword} />
28                  </div>
29              </header>
30          </div>
31      );
32  }
33
34  export default Form;
```

Instead of this...

```jsx
export default class OldForm extends React.Component {
    constructor(props) {
        super(props);
        this.state = {
            firstName: '',
            lastName: '',
            email: '',
            password: '',
            confirmPassword: '',
        }
        this.handleChange = this.handleChange.bind(this)
    }

    handleChange(event) {
        this.setState({
            [event.target.name]: event.target.value
        })
    }

    render() {
        return (
            <div className="App">
            <header className="App-header">
                <div>
                    <input type="text" name="firstName" placeholder="First Name" onChange={this.handleChange} />
                </div>
                <div>
                    <input type="text" name="lastName" placeholder="Last Name" onChange={this.handleChange} />
                </div>
                <div>
                    <input type="text" name="email" placeholder="Email" onChange={this.handleChange} />
                </div>
                <div>
                    <input type="password" name="password" placeholder="Password" onChange={this.handleChange} />
                </div>
                <div>
                    <input type="password" name="confirmPassword" placeholder="Confirm Password" onChange={this.handleChange} />
                </div>
            </header>
        </div>
        )
    }
}
```

Resources

- Dan Abramov's initial introduction to React Hooks
- ReactJS docs
- A collection of React Hooks resources
- One of the best beginner tutorials/explanations I found