Introduction to Components 39th Day

There are two main types of components in React. Class Components and Functional Components. The difference is pretty obvious. Class components are ES6 classes and Functional Components are functions. The only constraint for a functional component is to accept props as an argument and return valid JSX.

The simplest way to define a component in React is to write a JavaScript function.

Example:

```
function Welcome(props) {
return <h1>Hello, {props.name}</h1>;
}
```

This is just a function which accepts props and returns a React element. But you can also use the ES6 class syntax to write components.

Example:

```
class Welcome extends React.Component {
render() {
return <h1>Hello, {this.props.name}</h1>;
}
}
```

Differences between functional and class-Components

1). **Syntax**

The most obvious one difference is the syntax. A functional component is just a plain JavaScript function which accepts props as an argument and returns a React element.

A class component requires you to extend from React.Component and create a render function which returns a React element. This requires more code but will also give you some benefits which you will see later on.

2).**State**

Because a functional component is just a plain JavaScript function, you cannot use setState() in your component. That's the reason why they also get called functional stateless components. So everytime you see a functional component you can be sure that this particular component doesn't have its own state.

If you need a state in your component you will either need to create a class component or you lift the state up to the parent component and pass it down the functional component via props.

3). **LifeCycle Hook**.

Another feature which you cannot use in functional components are lifecycle hooks. The reason is the same like for state, all lifecycle hooks are coming from the React.Component which you extend from in class components.

So if you need lifecycle hooks you should probably use a class component.

You might ask yourself why you should use functional components at all, if they remove so many nice features. But there are some benefits you get by using functional components in React:

1. Functional component are much easier to read and test because they are plain JavaScript functions without state or lifecycle-hooks
2.You end up with less code
3.They help you to use best practices. It will get easier to separate container and presentational components because you need to think more about your component's state if you don't have access to setState() in your component
4.The React team mentioned that there may be a performance boost for functional component in future React versions, Which has come to pass.

More Notes on Components. **Best Wishes**. **Lux Academy**.