React Components: Functional vs Class (Without Hooks)

1. Functional Components

A Functional Component is a plain JavaScript function that returns JSX.

Example:

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

Characteristics:

- Simple and concise
- Accepts props as argument
- No access to state or lifecycle methods (without Hooks)
- Stateless before React 16.8

2. Class Components

A Class Component is an ES6 class that extends React.Component and includes a render() method.

Example:

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

Characteristics:

- Can maintain state
- Has access to lifecycle methods like componentDidMount
- Uses this.props and this.state
- More verbose and boilerplate-heavy

React Components: Functional vs Class (Without Hooks)

3. Key Differences

Feature	Functional Component	Class Component
Base Structure	JS Function	ES6 Class
Props Usage	props	this.props
State Support	No (before Hooks)	Yes
Lifecycle Methods	Not available	Available
Performance	Faster, simpler	Slightly heavier
Boilerplate Code	Less	More
Used With Hooks	Yes (after React 16.8)	Not required

4. Summary

Use Functional Components for simple UI presentation.

Use Class Components when you need to handle state or lifecycle logic.

Example Comparison:

Functional Component:

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

Class Component:

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