

Introduction to React Components

A comprehensive guide to building React applications

What are React Components?

React components are the building blocks of any React application. They allow you to split the UI into independent, reusable pieces that can be thought about in isolation.

Types of Components

There are two main types of React components:

1. Functional Components

- Simple JavaScript functions that return JSX
- Can use hooks like useState and useEffect
- Recommended for modern React development
- Easier to read, test, and maintain

2. Class Components

- ES6 classes that extend React.Component
- Have lifecycle methods
- Still supported but less common in new code

Example: Functional Component

```
function Welcome(props) {  
  return <h1>Hello, {props.name}!</h1>;  
}  
  
// Usage  
<Welcome name="CareerQuest Student" />
```

Key Concepts

- Components receive data through props
- Components can maintain internal state
- Components can be composed together
- Components should be pure and predictable
- Each component should have a single responsibility

Best Practices

- Keep components small and focused
- Use descriptive component names
- Extract reusable logic into custom hooks

- Avoid deeply nested component hierarchies
- Use PropTypes or TypeScript for type checking

Component Lifecycle

In functional components with hooks, the lifecycle is managed through:

useEffect Hook

The useEffect hook lets you perform side effects in functional components. It serves the same purpose as componentDidMount, componentDidUpdate, and componentWillUnmount combined in class components.

```
useEffect(() => {
  // This runs after every render
  document.title = 'Welcome!';

  // Cleanup function (optional)
  return () => {
    document.title = 'React App';
  };
}, []); // Empty array = run once on mount
```

Conclusion

React components are essential for building modern web applications. By understanding how to create and use components effectively, you can build scalable, maintainable applications. Practice creating different types of components and experiment with props, state, and hooks to deepen your understanding.