# ReactJS: Destructuring Props & State

**What is Destructuring?**

Destructuring is a JavaScript ES6 feature that lets you unpack values from objects into variables.  
In React, we use it to clean up our code and make it more readable.

**Why Use Destructuring?**

Example before:  
<h1>Hello, {props.name}, you are {props.age} years old.</h1>  
  
Example after:  
const { name, age } = props;  
<h1>Hello, {name}, you are {age} years old.</h1>

**1. Destructuring Props in Functional Components**

Before:  
const Welcome = (props) => {  
 return <h2>Hello, {props.name} from {props.city}</h2>;  
};  
  
After:  
const Welcome = ({ name, city }) => {  
 return <h2>Hello, {name} from {city}</h2>;  
};

**2. Destructuring Props in Class Components**

Before:  
class Welcome extends React.Component {  
 render() {  
 return <h2>Hello, {this.props.name}</h2>;  
 }  
}  
  
After:  
class Welcome extends React.Component {  
 render() {  
 const { name } = this.props;  
 return <h2>Hello, {name}</h2>;  
 }  
}

**3. Destructuring State in Class Components**

Before:  
render() {  
 return <h2>Count: {this.state.count}</h2>;  
}  
  
After:  
render() {  
 const { count } = this.state;  
 return <h2>Count: {count}</h2>;  
}

**Summary Table**

| Use Case | Syntax |  
|----------------------|----------------------------------|  
| Functional Props | ({ name }) => {} |  
| Class Props | const { name } = this.props |  
| Class State | const { count } = this.state |