



# Essential ReactJS Shorthand Techniques

Swipe



# Arrow Functions in Event Handlers

Use arrow functions directly in JSX for event handling to avoid binding this in the constructor.



```
<button onClick={
  this.handleClick.bind(this)
}> Click </button>;
```



```
<button onClick={
  () => this.handleClick()
}> Click </button>;
```





# Conditional Rendering with Logical &&

Render components conditionally using the && operator for cleaner, more readable code.



```
{  
  isLoggedIn ? <LogoutButton /> : null  
}
```



```
{  
  isLoggedIn && <LogoutButton />  
}
```



# Destructuring Props and State

Use object destructuring to extract values from props and state for more concise and readable component code.

```
const value = this.props.value
```



```
const { value } = this.props
```







# Fragment Short Syntax

Utilize the short syntax `<>...</>` to group a list of children without adding extra nodes to the DOM.



```
<React.Fragment>  
  <ComponentA />  
  <ComponentB />  
</React.Fragment>
```



```
<>  
  <ComponentA />  
  <ComponentB />  
</>
```



# Spread Attributes

Spread attributes to pass all properties of an object as props to a component, simplifying the passing of props.



```
<MyComponent  
  prop1={this.props.prop1}  
  prop2={this.props.prop2}  
/>
```



```
<MyComponent {...this.props} />
```





# Function Component Declaration

Define functional components using arrow functions for a more concise syntax.



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



```
const Welcome = ({ name }) =>  
  <h1>Hello, {name}</h1>
```



# Optional Chaining for Property Access

Use optional chaining (?.) to safely access nested object properties without checking each level.



```
const name = this.props.user &&  
this.props.user.name
```



```
const name = this.props.user?.name
```





# State Initialization in Constructor

Initialize state directly using class property syntax in class components, avoiding the need for a constructor.

```
constructor(props) {  
  super(props);  
  this.state = { count: 0 };  
}
```



```
state = { count: 0 }
```





Add to Favourites

# How many do you use?



Leave a comment

Save for later

