

# Understanding Higher-Order Components (HOCs)

SWIPE LEFT



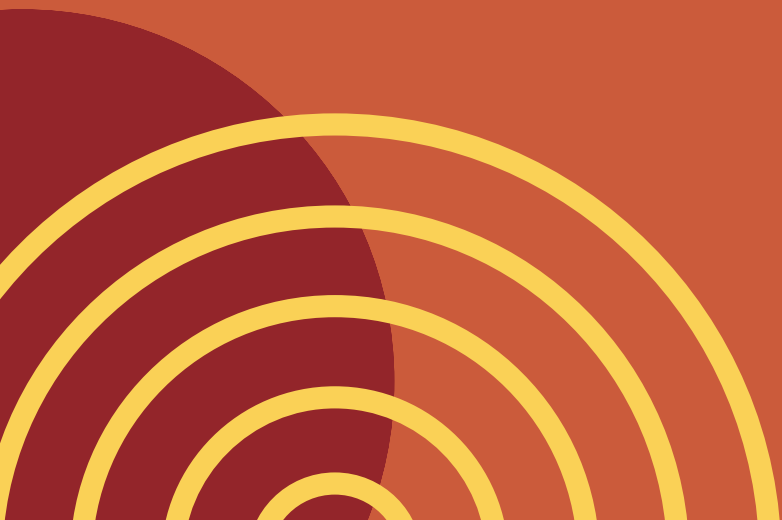
Abhijeet Chakane

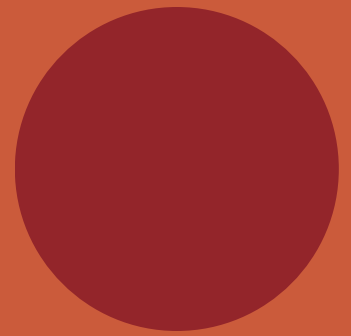




# What Are Higher-Order Components (HOCs)?

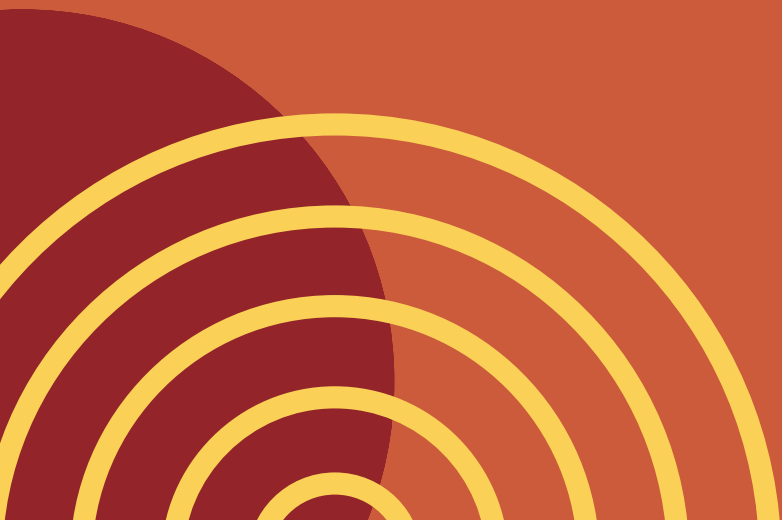
Higher-Order Components (HOCs) are functions that take a component and return a new component. They are a pattern in React for reusing component logic and enhancing components with additional functionalities.

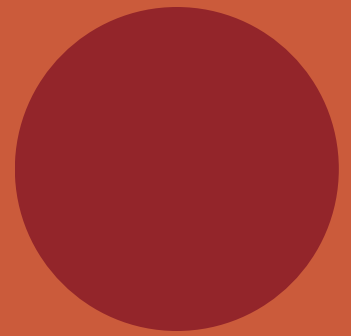




# Why Use HOCs?

- Code Reusability: Encapsulate reusable logic in one place.
- Separation of Concerns: Separate the "what" (UI) from the "how" (behavior).
- Component Composition: Build new components by wrapping them in HOCs.



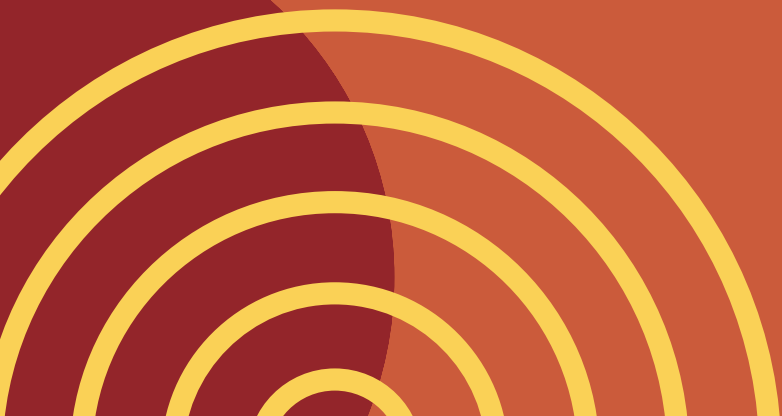


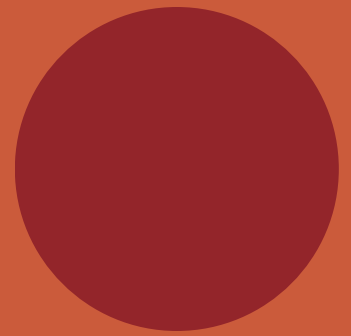
# Creating an HOC

To create an HOC, define a function that takes a component and returns another component.



```
1  function withExtraInfo(WrappedComponent) {  
2    return function EnhancedComponent(props) {  
3      return (  
4        <div>  
5          <WrappedComponent {...props} />  
6          <p>Extra information added by HOC!</p>  
7        </div>  
8      );  
9    };  
10 }  
11
```



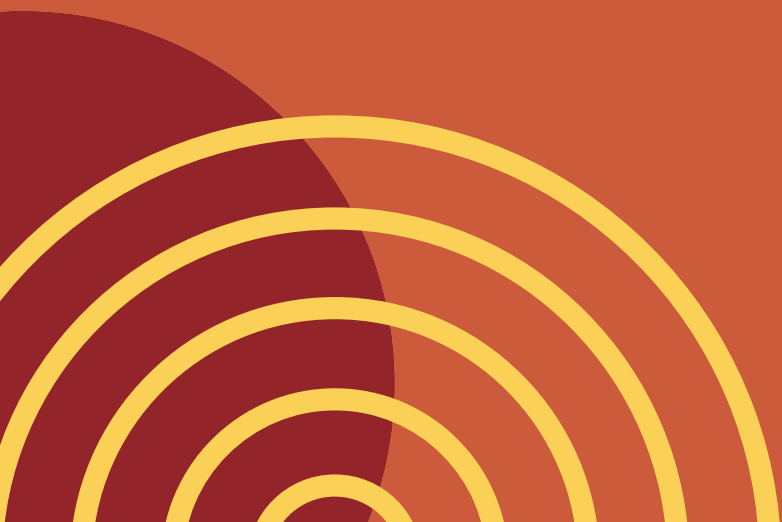


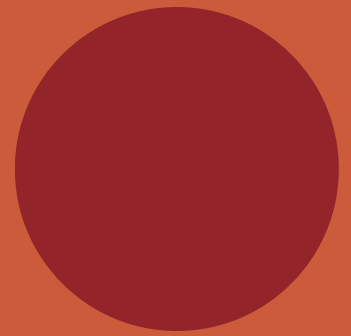
# Using an HOC

Wrap your original component with the HOC to create an enhanced component.



```
1  function MyComponent(props) {  
2    return <h1>Hello, {props.name}!</h1>;  
3  }  
4  
5  const MyComponentWithExtraInfo = withExtraInfo(MyComponent);  
6  
7  // Usage  
8  <MyComponentWithExtraInfo name="React Learner" />;  
9
```





# Using an HOC

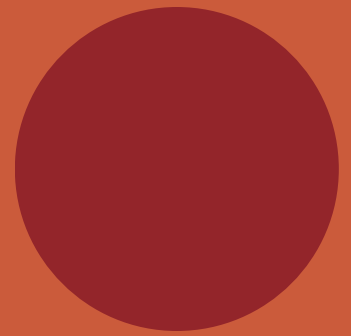
Wrap your original component with the HOC to create an enhanced component.



```
1 function MyComponent(props) {  
2   return <h1>Hello, {props.name}!</h1>;  
3 }  
4  
5 const MyComponentWithExtraInfo = withExtraInfo(MyComponent);  
6  
7 // Usage  
8 <MyComponentWithExtraInfo name="React Learner" />;  
9
```

The HOC withExtraInfo adds an extra paragraph to MyComponent.





# Using an HOC

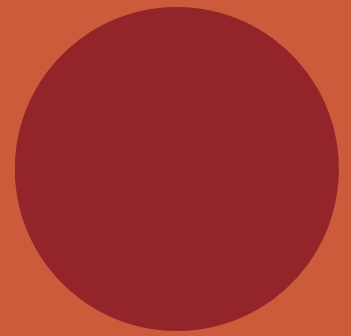
Wrap your original component with the HOC to create an enhanced component.



```
1 function MyComponent(props) {  
2   return <h1>Hello, {props.name}!</h1>;  
3 }  
4  
5 const MyComponentWithExtraInfo = withExtraInfo(MyComponent);  
6  
7 // Usage  
8 <MyComponentWithExtraInfo name="React Learner" />;  
9
```

The HOC withExtraInfo adds an extra paragraph to MyComponent.





# Key Benefits & Best Practices

- Easy Logic Sharing: Share non-UI logic across multiple components.
- Do Not Mutate: Never mutate the original component; always return a new one.
- Pass Props Through: Ensure HOCs pass all relevant props down to the wrapped component.





# Thank You



Abhijeet Chakane.