

Class-based Components: An Alternative To Functions

Default & Most Modern Approach!

Functional Components

```
function Product(props) {  
  return <h2>A Product!</h2>  
}
```

Components are regular JavaScript functions which return renderable results (typically JSX)

Was Required In The Past

Class-based Components

```
class Product extends Component {  
  render() {  
    return <h2>A Product!</h2>  
  }  
}
```

Components can also be defined as JS classes where a render() method defines the to-be-rendered output

Traditionally (**React < 16.8**), you had to use
Class-based Components
to manage “**State**”

React 16.8 introduced “**React Hooks**” for **Functional** Components

Class-based Components **Can't Use** React Hooks!

Class-based Component Lifecycle

Side-effects in Functional Components: **useEffect()**

Class-based Components can't use React Hooks!

`componentDidMount()`

Called once component mounted
(was evaluated & rendered)

`useEffect(..., [])`

`componentDidUpdate()`

Called once component updated
(was evaluated & rendered)

`useEffect(..., [someValue])`

`componentWillUnmount()`

Called right before component is
unmounted (removed from DOM)

`useEffect(() => { return () => {...}}, [])`

You **don't have to use**
Functional Components
– it is fine to use Class-
based Ones instead

Class-based vs. Functional Components

Prefer functional components

Use class-based if...

...you prefer them

...you're working on an
existing project or in a team
where they're getting used

...you build an "Error
Boundary"