React Component Lifecycle Methods

■ React Class Component Lifecycle (with Example)

The lifecycle has 3 main phases:
1. Mounting
- constructor()
- static getDerivedStateFromProps()
- render()
- componentDidMount()

2. Updating
- static getDerivedStateFromProps()
- shouldComponentUpdate()
- render()
- getSnapshotBeforeUpdate()
- componentDidUpdate()

3. Unmounting
- componentWillUnmount()

■ Full Working Example

```
import React from 'react';

class LifecycleDemo extends React.Component {
  constructor(props) {
    super(props);
    this.state = { count: 0 };
    console.log('1. constructor');
  }

  static getDerivedStateFromProps(props, state) {
    console.log('2. getDerivedStateFromProps');
    return null;
  }

  componentDidMount() {
    console.log('4. componentDidMount');
  }

  shouldComponentUpdate(nextProps, nextState) {
    console.log('5. shouldComponentUpdate');
    return true;
  }

  getSnapshotBeforeUpdate(prevProps, prevState) {
    console.log('6. getSnapshotBeforeUpdate');
    return null;
  }

  componentDidUpdate(prevProps, prevState, snapshot) {
    console.log('7. componentDidUpdate');
  }

  componentWillUnmount() {
    console.log('8. componentWillUnmount');
  }

  increment = () => {
    this.setState((prevState) => ({ count: prevState.count + 1 }));
  };

  render() {
    console.log('3 or 6. render');
    return (
      <div>
```

```
        <h1>Count: {this.state.count}</h1>
        <button onClick={this.increment}>Increment</button>
      </div>
    );
  }
}

export default LifecycleDemo;
```

■ Output Log in Console

```
When the component loads:
1. constructor
2. getDerivedStateFromProps
3. render
4. componentDidMount

When the button is clicked:
2. getDerivedStateFromProps
5. shouldComponentUpdate
3 or 6. render
6. getSnapshotBeforeUpdate
7. componentDidUpdate

When the component is removed:
8. componentWillUnmount
```

■ Bonus: Functional Component Equivalent with Hooks

```
import React, { useState, useEffect } from 'react';

function LifecycleWithHooks() {
  const [count, setCount] = useState(0);

  useEffect(() => {
    console.log('Component mounted or updated');
    return () => {
      console.log('Component will unmount');
    };
  }, [count]);

  return (
    <div>
      <h1>Count: {count}</h1>
      <button onClick={() => setCount(prev => prev + 1)}>Increment</button>
    </div>
  );
}
```