

Functional component & Class component

Objective

- Functional component
- Class component
- Life cycle methods

Functional component

- ❖ The simplest way to define a component is to write a JavaScript function:

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

Class component

- ❖ You can also use an ES6 class to define a component:

```
class Welcome extends React.Component {  
  render() { return <h1>Hello, {this.props.name}</h1>; }  
}
```

TUWAIQ ACADEMY



Life cycle methods

`componentDidMount()`

- ❖ This function is invoked immediately after the component is mounted to the DOM.
- ❖ This method is a good place to set up any subscriptions. If you do that, don't forget to unsubscribe in `componentWillUnmount()`.

`componentDidUpdate()`

- ❖ `componentDidUpdate()` is invoked immediately after updating occurs. This method is not called for the initial render.
- ❖ Use this as an opportunity to operate on the DOM when the component has been updated.

Life cycle methods

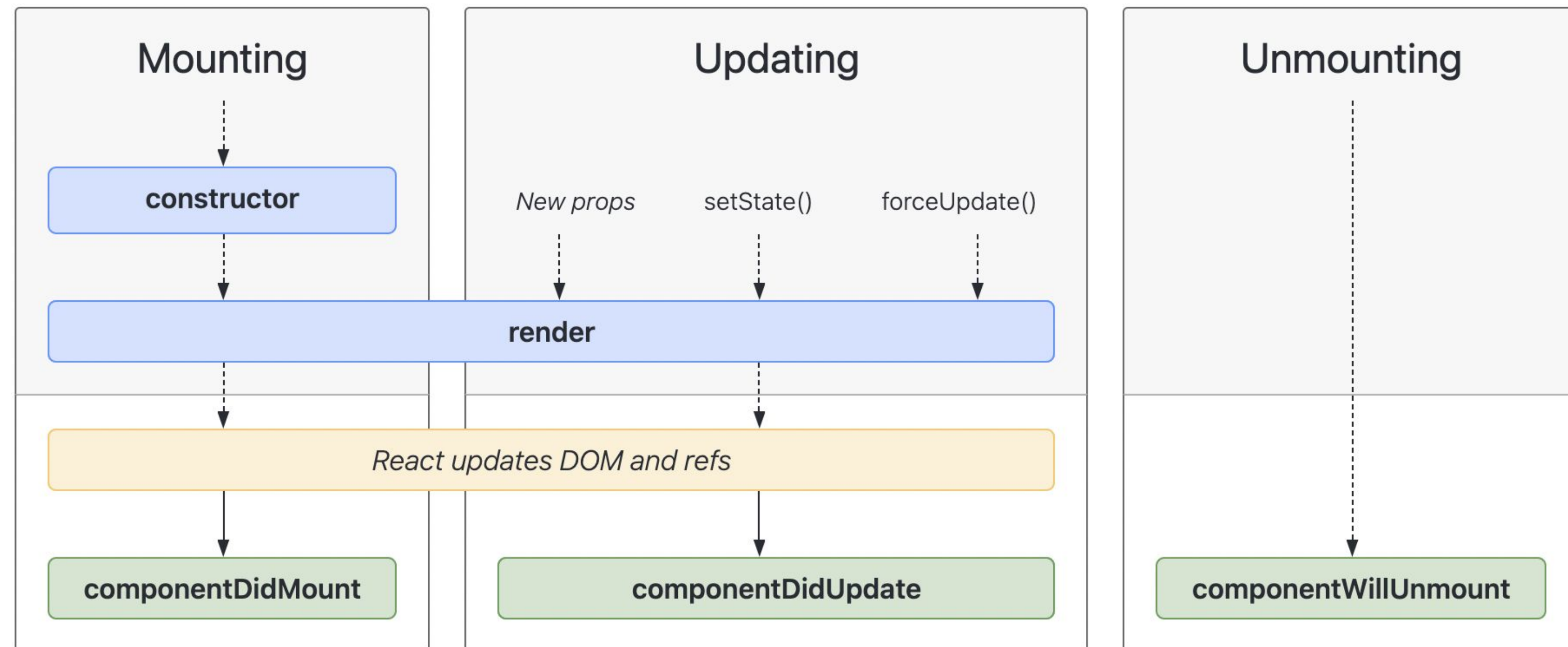
`componentWillUnmount()`

- ❖ The `componentWillUnmount()` is invoked immediately before a component is unmounted and destroyed.
- ❖ Perform any necessary cleanup in this method, such as invalidating timers, canceling network requests, or cleaning up any subscriptions that were created in `componentDidMount()`.

Life cycle methods

"Render phase"
Pure and has no side effects. May be paused, aborted or restarted by React.

"Commit phase"
Can work with DOM, run side effects, schedule updates.





Resources

- <https://reactjs.org/docs/react-component.html#componentdidmount>
- <https://reactjs.org/docs/react-component.html#componentdidupdate>
- <https://reactjs.org/docs/react-component.html#componentwillunmount>

Summary

- Functional component
- Class component
- Life cycle methods