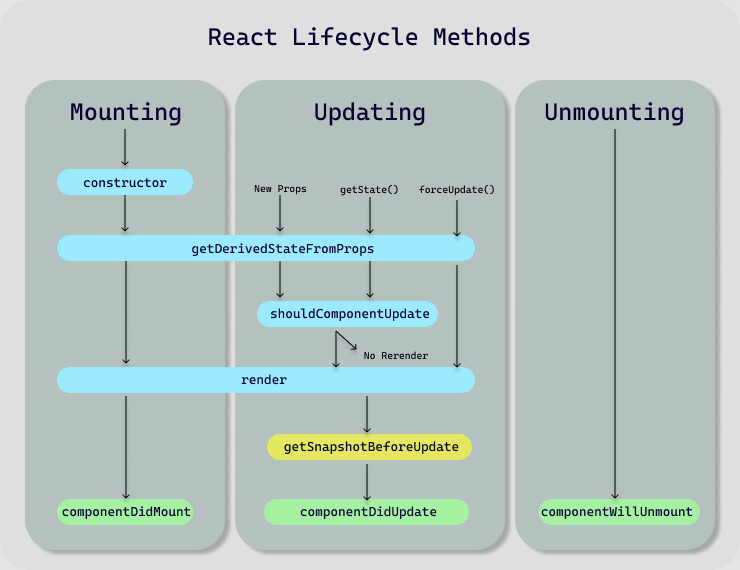
**MODULE: 4 (List and Hooks)**

Q: Explain Life cycle in Class Component and functional component with Hooks

Ans: Life cycle in Class Component undergoes three phases in its lifecycle: **mounting**, **updating**, and **unmounting**.

1. **Mounting: - "The mounting phase is when a new component is installed and secured in its designated location." It is produced and inserted into the DOM, or when a component's life begins. This can only happen once and is commonly called "initial render."**
2. **Updating: -** During the updating phase, the component is either updated or re-rendered. When the props or the state are updated, this reaction is eliminated. This phase can occur several times, which is kind of the point of React**.**
3. **Unmounting: -** The unmounting phase occurs at the end of a component's lifecycle when the component is removed from the DOM.



In a class-based component, we can call distinct methods for each phase of the lifecycle. These lifecycle methods, of course, are not relevant to functional components because they can only be written/contained within a class. React hooks, on the other hand, enable functional components to use states

import React from 'react';

import ReactDOM from 'react-dom/client';

class Test extends React.Component {

    constructor(props){

        super(props);

        this.state = { hello: "World!" };

    }

    componentDidMount(){

        console.log("componentsDidMount()");

    }

    changeState(){

        this.setState({ hello: "React" });

    }

    render(){

        return (

            <div>

              <h1>

                Hello{this.state.hello}

              </h1>

              <h2>

                <a onClick={this.changeState.bind(this)} href='/'>Press Here!</a>

              </h2>

            </div>

          )

    }

    shouldComponentUpdate(nextProps,nextState){

        console.log("shouldComponentUpdate()");

        return true;

    }

    componentDidUpdate(){

        console.log("componentDidUpdate()");

    }

}

export default Test

const root = ReactDOM.createRoot(

    document.getElementById("root")

);

root.render(<Test/>)

**Functional components: -**

Functional components are some of the more common components seen while working in React. These are simple JavaScript functions. By writing a JavaScript function, we can create a functional component for React.

import React, { useState } from 'react';

const Functionalcomponent = () => {

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

  const increase =()=>{

    setCount(count +1);

  }

  return (

    <div style={{margin: '50px'}}>

      <h3>Counter App using Functional Component : </h3>

      <h2>{count}</h2>

      <button onClick={increase}>Add</button>

    </div>

  )

}

export default Functionalcomponent;

**Class Component: -**

This is the basis of most modern web projects developed with ReactJS. These are simple classes composed of numerous functions that enhance the application's functionality.

class ClassComponent extends React.Component {

    constructor() {

        super();

        this.state = {

            count: 0

        };

        this.increase = this.increase.bind(this);

    }

    increase() {

        this.setState({ count: this.state.count + 1 });

    }

    render() {

        return (

            <div style={{ margin: '50px' }}>

                <h3>Counter App using Class Component:</h3>

                <h2>{this.state.count}</h2>

                <button onClick={this.increase}>Add</button>

            </div>

        )

    }

}

export default ClassComponent;