Explain Life cycle in Class Component and functional component with Hooks

Mounting Phase : The mounting phase refers to the period when a component is being created and inserted into the DOM.

Constructor()

The constructor is where the initial state and the values are set in a React component. This method is called before the component is mounted.

Render()

This is the only required method in React components. It is the most widely used one as well. This method is used in both mounting and updating phases.

Updating Phase : The next phase in react life cycle is updating phase . A component is updated whenever there is a change in the component state and props.

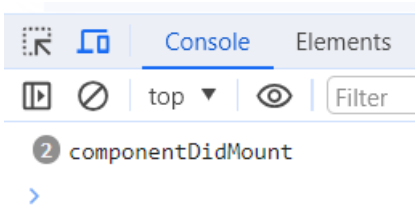
Unmounting Phase : The next phase in the lifecycle is when a component is removed from the DOM, or unmounting as React likes to call it.

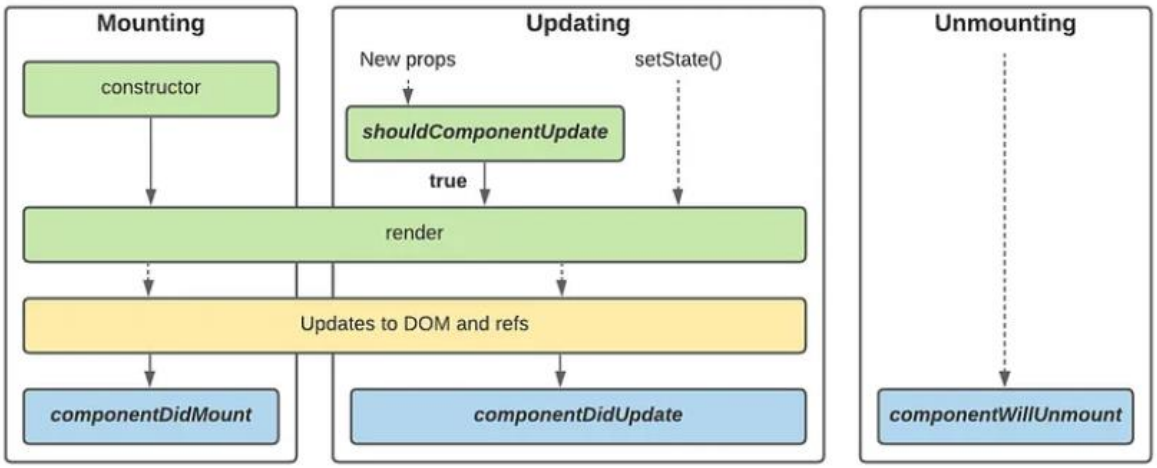
```
import React, { Component } from 'react'

class Lifecycle extends React.Component{
    constructor(props){
```

```
        super(props);
        this.state = { hello : "world"}
    }
    componentDidMount(){
        console.log("componentDidMount")
    }
    changeState(){
        this.setState({hello : "react"})
    }

    render(){
        return (
            <div>
                <h1>
                    hello
                    {this.state.hello}
                </h1>
                <h2>
                    <a onClick={this.changeState.bind(this)}> Click here</a>
                </h2>
            </div>
        )
    }
    shouldComponentUpdate(nextProps,nextstate){
        console.log("shouldComponentUpdate")
        return true
    }
    componentDidUpdate(){
        console.log("componentDidUpdate")
    }
}
export default Lifecycle;
```





Function Component

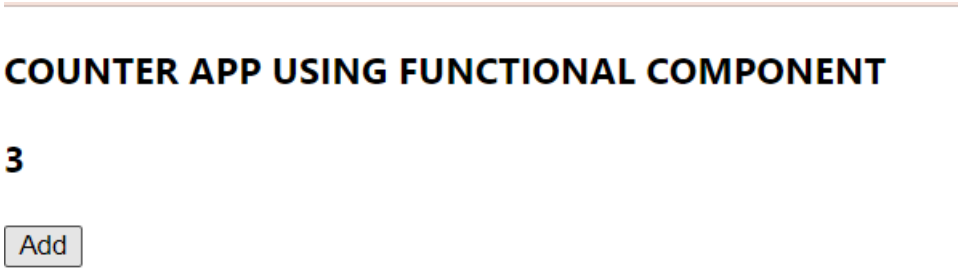
Function component are some of the more common component that will come across while working in react . These are simply javascript functions . we can create a functional component by writing a javascript function .

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

const FunctionlComponent = () => {
    const  [ count , setCount] = useState(0)
    const Increase = () => {
        setCount(count + 1)
    }
    return(
        <>
        <div>
            <h3>COUNTER APP USING FUNCTIONAL COMPONENT</h3>
            <h3>{count}</h3>
            <button onClick={Increase}> Add </button>
```

```
        </div>
        </>
    )
}

export default FunctionlComponent
```



Class Component

React Class components have a built-in state object . The state object is where you store property values that belongs to the component. When the state object changes, the component re-renders.

```
import { render } from "@testing-library/react";
import React, {Component} from "react";

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>
        <h3>Counter app using class component</h3>
        <h2> {this.state.count}</h2>
        <button onClick={this.increase}>Add</button>
    </div>
    )
}
}
export default Classcomponent;
```