

| Title: Implementation of react hooks |
| --- |

AIM : To demonstrate the use of the following React hooks :

* useState
* useEffect
* useContext
* useRef
* useCallback
* useReducer
* useMemo

**Expected OUTCOME of Experiment:**

**CO-2 :** Illustrate the concepts of various front-end, back-end web application development technologies & frameworks using different web development tools.

**Books/ Journals/ Websites referred :** <https://react.dev/learn>



**Code**

1. **useState**

<!DOCTYPE html>

<html>

<head>

<script src="https://unpkg.com/react@18/umd/react.development.js" crossorigin></script>

<script src="https://unpkg.com/react-dom@18/umd/react-dom.development.js" crossorigin></script>

<script src="https://unpkg.com/@babel/standalone/babel.min.js"></script>

</head>

<body>

<div id="root"></div>

<script type="text/babel">

function FavoriteColor() {

const [state, setState] = React.useState("Off");

return (<><h1>current state is {state}</h1><button onClick={()=>setState("on")}>{state}</button></>)

}

const root = ReactDOM.createRoot(document.getElementById('root'));

root.render(<FavoriteColor />);

</script>

</body>

</html>

****

1. **Useeffect**

**<!DOCTYPE html>**

**<html>**

**<head>**

**<script src="https://unpkg.com/react@18/umd/react.development.js" crossorigin></script>**

**<script src="https://unpkg.com/react-dom@18/umd/react-dom.development.js" crossorigin></script>**

**<script src="https://unpkg.com/@babel/standalone/babel.min.js"></script>**

**</head>**

**<body>**

**<div id="root"></div>**

**<script type="text/babel">**

**function Stateset() {**

**const [state, setState] = React.useState("Off");**

**React.useEffect(() => {**

**if(state=="off"){**

**setState("on");**

**}**

**else if(state=="on"){**

**setState("off");**

**}**

**});**

**return (<><h1>current state is {state}</h1><button onClick={()=>setState("on")}>{state}</button></>)**

**}**

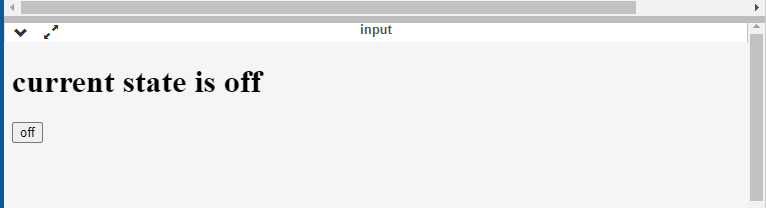
**const root = ReactDOM.createRoot(document.getElementById('root'));**

**root.render(<Stateset />);**

**</script>**

**</body>**

**</html>**

****

This will run infinite loop and toggle between on and off.

1. **Use Ref**

<!DOCTYPE html>

<html>

<head>

<script src="https://unpkg.com/react@18/umd/react.development.js" crossorigin></script>

<script src="https://unpkg.com/react-dom@18/umd/react-dom.development.js" crossorigin></script>

<script src="https://unpkg.com/@babel/standalone/babel.min.js"></script>

</head>

<body>

<div id="root"></div>

<script type="text/babel">

function Stateset() {

const [state, setState] = React.useState("Off");

const val = React.useRef(0).current

return (<><h1>number is is {val}</h1><button onClick={()=>setState("on")}>{state}</button></>)

}

const root = ReactDOM.createRoot(document.getElementById('root'));

root.render(<Stateset />);

</script>

</body>

</html>



**Explanation**

1. **useState:** This hook is used to create a state variable count and a function to update it setCount. The initial value of count is 0.
2. **useEffect:** This hook is used to perform side effects in the component. In this case, it logs a message when the component mounts and unmounts. The empty array [] as the second argument means the effect runs only once after the first render (mount) and before the component is unmounted.
3. **useContext:** This hook is used to access the value of a React context. In this case, it's trying to access the value of MyContext, but MyContext is not provided by any parent component, so value will be undefined.
4. **useRef:** This hook is used to create a ref that can be attached to React elements. In this case, it's attached to an input element.
5. **useCallback:** This hook is used to memoize a function.
6. **useReducer:** This hook is used to manage complex state logic. It's similar to useState but it accepts a reducer function. In this case, it's used to manage a state with increment and decrement actions.
7. **useMemo:** This hook is used to memoize expensive calculations. In this case, it's used to memoize the result of multiplying count by 2.

**Conclusion**

In this experiment, we were expected to learn how to use different predefined React hooks to make development using React more efficient for different scenarios.