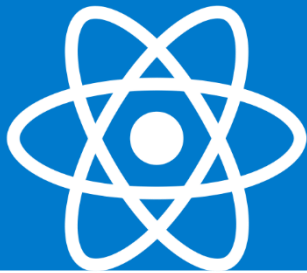
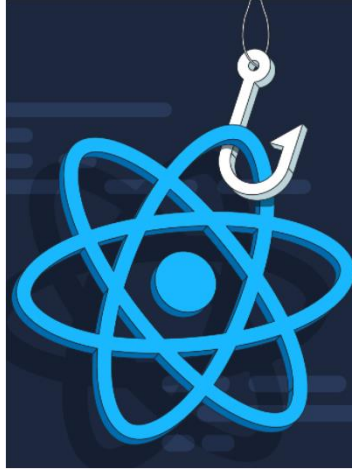




Hooks

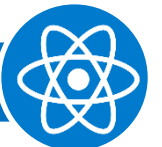
#React Notes

| Hooks



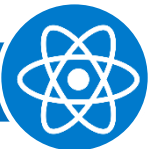
Hooks

- **React Hooks** let you **manage the state** and **react to state changes** in functional components.
- Hooks are a new addition in React 16.8.
- They let you use state and other React features without writing a class (Class Component).
- Hooks will not work inside Classes. **it will use in Functional Component Only.**
- Traditionally in React.js, class-based components were needed in order to utilize lifecycle methods and state.
- With newer releases of React.js though, programmers working on the front-end with React.js can now create functional components with hooks, which hold similarities to class-based components
- <https://reactjs.org/docs/hooks-intro.html>



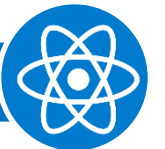
Hooks

- Hooks = Function Component + State + Lifecycle Method



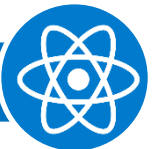
What is use of Hooks ?

- We can use State management in Functional Component
- We can use LifeCycle Method in Function Component



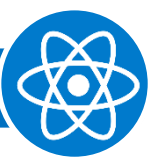
Important Rules for Hooks

- Never call Hooks from inside a loop, condition or nested function
- Hooks should sit at the top-level of your component
- Only call Hooks from React functional components
- Never call a Hook from a regular function
- Hooks can call other Hooks



Difference Between Hooks Vs Classes

React Hooks	Classes
Used in functional components in React	Used in class based components in React
Does not require the declaration of any kind of constructor	Declaration of constructor has to be made inside of the class component.
There is no need of using <i>this</i> keyword in state declaration or modification	<i>this</i> keyword is used in state declaration i.e. <code>this.state</code> and in modification - <code>this.setState()</code>
Easier to use because of the <code>useState()</code> functionality	No specify function that helps us access the state and its corresponding <code>setState</code> variable.
React Hooks can help in the implementation of Redux and context API	Due to the long setup of state declarations, class states generally not preferred.



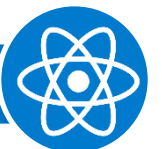
Difference Between Class and Function Hooks

- Before

- In React JS Class Component will allow you to use state and LifeCycle method such as `componentDidMount()` and other Method.
- In Functional component we can not use state and lifecycle methods.

- After

- Using React Hooks we can use **state and lifecycle method** in functional component.

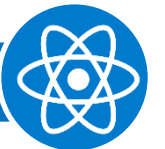


Basic Hooks

1. useState
2. useEffect
3. useContext

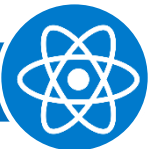
- Documentation

- <https://reactjs.org/docs/hooks-intro.html>
- <https://reactjs.org/docs/hooks-state.html>
- <https://reactjs.org/docs/hooks-effect.html>

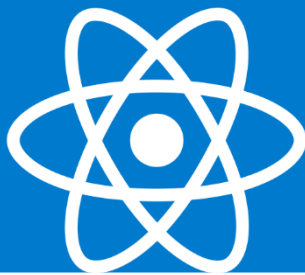


Basic Hooks

1. `useState`
 - Will use to **manage states** in functional component
2. `useEffect`
 - It will call automatically when **component will load** or update.
3. `useContext`
 - Share the Data



useState (State management in Function)



useState

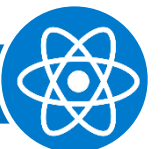
- It declares a “state variable...This is a way to “preserve” some values between the function calls — useState is a new way to use the exact same capabilities that this.state provides in a class.
- Normally, variables “disappear” when the function exits but state variables are preserved by React.

Method 1:

```
import { useState } from 'react';  
const [state, setState] = useState(initialState);
```

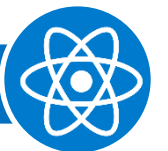
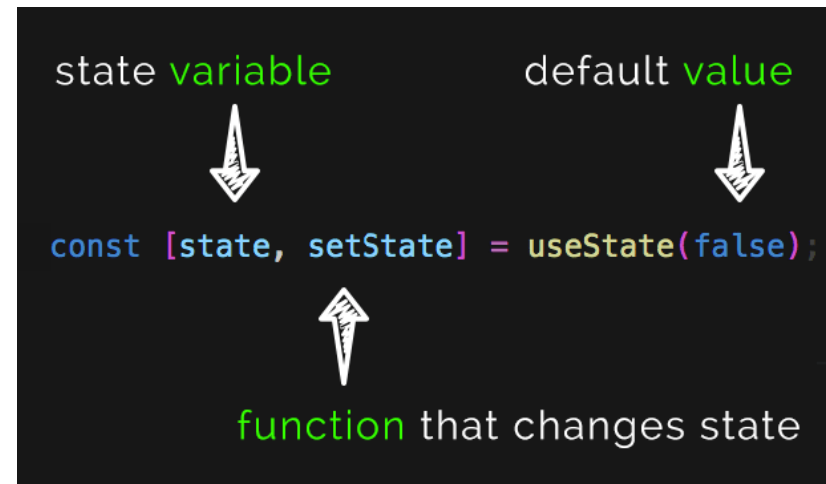
Method 2:

```
import React from "react";  
const [state, setState] = React.useState(initialState);
```



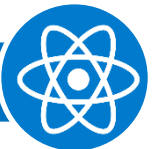
How to Define State in Hooks

- `const[state,setState] = useState(false);`
 - **State** is a **variable** name
 - **setState** is **Function** which will change value of state
 - **useState(false)** is used for Assign **Default value**
- state and setState names are customisable names you could call them whatever you like.



Example:-

- **Define and Initialized State**
 - `const[color,setColor] = useState('red');`
 - `const[no,setNo] = useState(101);`
- **Set Value :- (Update Value using Events)**
 - `setColor(color: 'yellow');`
 - `setNo(no : '99');`
- **Get Value :-**
 - `{color}`
 - `{no}`

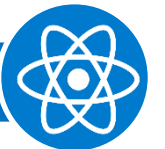


Variable vs Object

- when working with form fields, instead of keeping user info in one state like this:
- **Method 1: (Individual State Variable, Function and Default Value)**

```
const [name, setName] = useState('John');  
const [email, setEmail] = useState('john@example.com');  
const [age, setAge] = useState(25);
```
- **Method 2: (Instead of Creating Separate Variable we can store in One Object)**

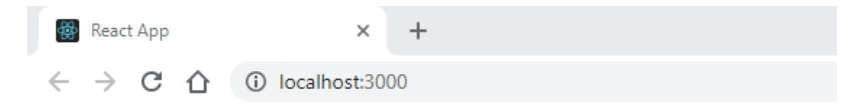
```
const [user, setUser] = useState(  
  { name: 'John', email: 'john@example.com', age: 25 }  
);
```
- **Set Value :- (Update Value)**
 - `setUser({ name: 'Harry' });`



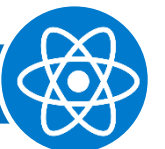
Set and Print Value

- Define Only State (Color) with Default value Red:
- Print Color on Screen

```
1  import React from 'react';
2
3  function App() {
4    const [color] = React.useState('red');
5
6    return (
7      <div>
8        Color is : {color}
9      </div>
10   );
11 }
12
13 export default App;
```



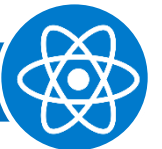
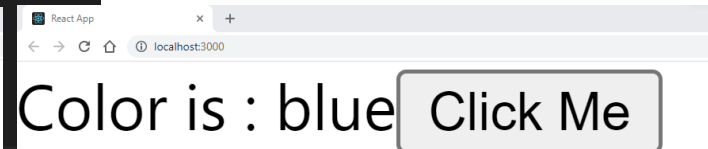
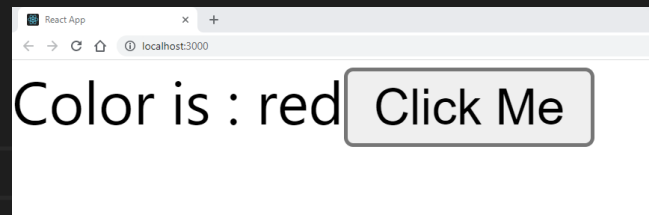
Color is : red



Update Value

- We can Change value using setColor(Function)
- Call Function with Value on ButtonClick

```
1  import React from 'react';
2
3  function App() {
4    const [color, setColor] = React.useState('red');
5
6    return (
7      <div>
8        Color is : {color}
9        <button onClick={()=>setColor("blue")}>Click Me</button>
10     </div>
11   );
12 }
13
14 export default App;
15
```

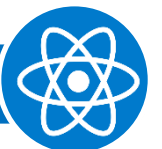


useEffect

- If you're familiar with React class lifecycle methods, you can think of useEffect Hook as
 - **componentDidMount,**
 - **componentDidUpdate,** and
 - **componentWillUnmount** combined.
- } **useEffect**
- The useEffect Hook allows us to perform side effects in our functional components.

```
import { useEffect } from "react";
```

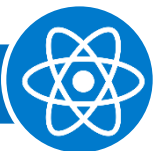
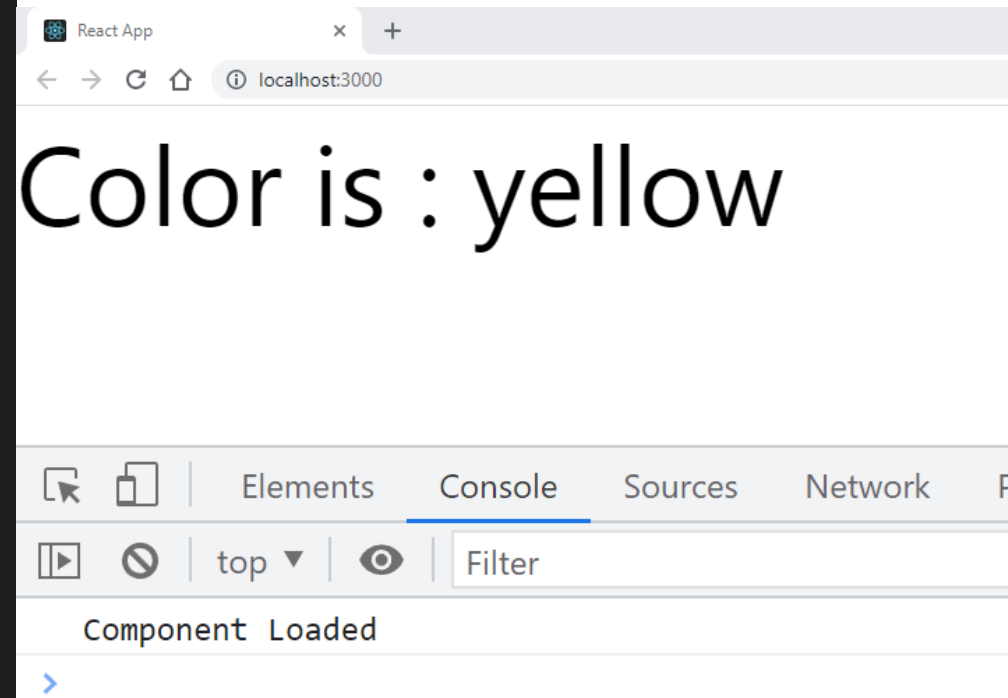
```
useEffect(() => {  
  // Inside this callback function we perform our side effects.  
});
```



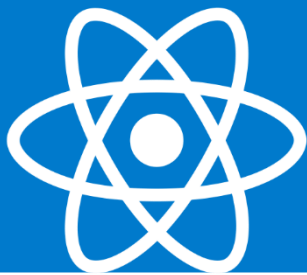
UseEffect Value Change

- useEffect will call automatically when you component will load.
- In useEffect it prints message and change color using function.

```
1 import React from 'react';
2
3 function App() {
4   const [color, setColor] = React.useState('red');
5
6
7   React.useEffect(() => {
8     console.log("Component Loaded");
9     //Update State Value
10    setColor("yellow")
11  }, [])
12
13  return (
14    <div>
15      Color is : {color}
16    </div>
17  );
18 }
19
20 export default App;
```



| useState Example



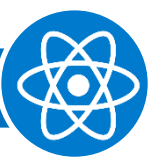
Simple State Assign and Print in Hooks

```
JS App.js 1, M ×
src > JS App.js > [default]
1  import React from 'react'
2
3  function App() {
4    //State
5    const [no, setNo] = React.useState(100);
6    return (
7      <div>
8        State Value is {no}
9      </div>
10   );
11 }
12 export default App;
```

React App × +

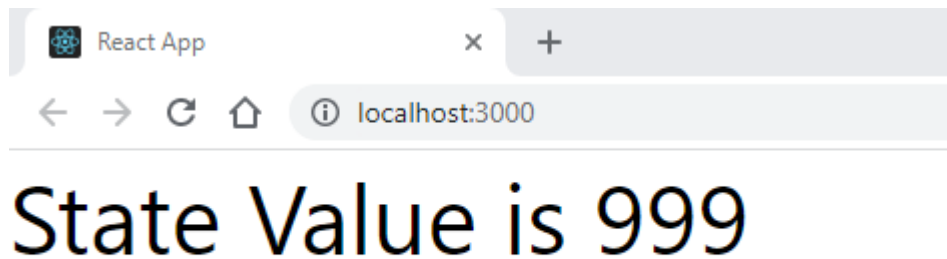
localhost:3000

State Value is 100

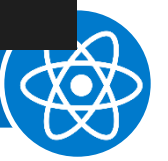


Update State Value While Loading Component

- By Default State Value will be 100
- On Loading of Component we will update Value with 999

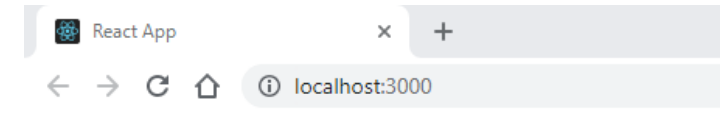


```
App.js 1, M X
src > JS App.js > ...
1  import React from 'react'
2
3  function App() {
4    //State
5    const [no,setNo] = React.useState(100);
6
7    //Component Load Event
8    React.useEffect(() => {
9      console.log("Component Loaded");
10     setNo(999); //State Update
11   });
12
13   return (
14     <div>
15       State Value is {no}
16     </div>
17   );
18 }
19 export default App;
```



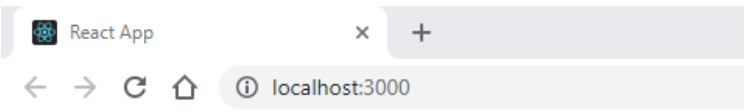
OnButton Click Change Value

```
App.js M X
src > JS App.js > App
1 import React from 'react'
2
3 function App() {
4   //State
5   const [no,setNo] = React.useState(100);
6   return (
7     <div>
8       State Value is {no}
9       <br/>
10      <button onClick={()=>{ setNo(999)}} >Click Me</button>
11    </div>
12  );
13 }
14 export default App;
```



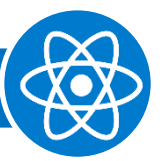
State Value is 100

Click Me



State Value is 999

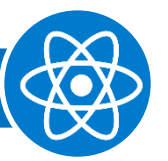
Click Me



Counter App - Class VS Function with Hooks

```
JS App.js M X
src > JS App.js > App > constructor
1  import React from 'react';
2  class App extends React.Component {
3    constructor(props) {
4      super(props);
5      this.state = {
6        count: 0,
7      };
8    }
9
10   render() {
11     return (
12       <div>
13         <p>You clicked {this.state.count} times</p>
14         <button
15           onClick={() =>
16             this.setState({ count: this.state.count + 1 })
17           }
18         >
19           Click me
20         </button>
21       </div>
22     );
23   }
24 }
25 export default App;
```

```
JS App.js M X
src > JS App.js > ...
1  import React from 'react';
2
3  // how to use the state hook in a React function component
4  function App() {
5    const [count, setCount] = React.useState(0);
6
7    return (
8      <div>
9        <p>You clicked {count} times</p>
10       <button onClick={() => setCount(count + 1)}>
11         Click me
12       </button>
13     </div>
14   );
15 }
16
17 export default App;
```



Counter App - Class VS Hooks

```
import React from 'react';
class App extends React.Component {
  constructor(props) {
    super(props);
    this.state = {
      count: 0,
    };
  }

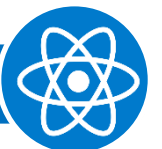
  render() {
    return (
      <div>
        <p>You clicked {this.state.count} times</p>
        <button
          onClick={() =>
            this.setState({ count: this.state.count + 1 })
          }
        >
          Click me
        </button>
      </div>
    );
  }
}
export default App;
```

```
import React from 'react';

// how to use the state hook in a React function component
function App() {
  const [count, setCount] = React.useState(0);

  return (
    <div>
      <p>You clicked {count} times</p>
      <button onClick={() => setCount(count + 1)}>
        Click me
      </button>
    </div>
  );
}

export default App;
```



ComponentDidMount in Class

```
src > JS App.js > default
1 import React from "react";
2 class App extends React.Component {
3   constructor() {
4     super();
5     this.state = {}
6   }
7
8   componentDidMount() {
9     console.log("Component Did Mount Called ");
10  }
11
12  render() {
13    return (
14      <div>
15        Component Did Mount
16      </div>
17    );
18  }
19 }
20 export default App;
```

React App

localhost:3000

Component Did Mount

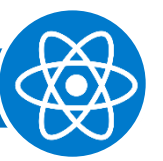
Elements Console Sources Network

top Filter

× Expression
not available

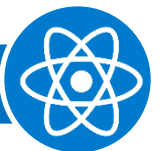
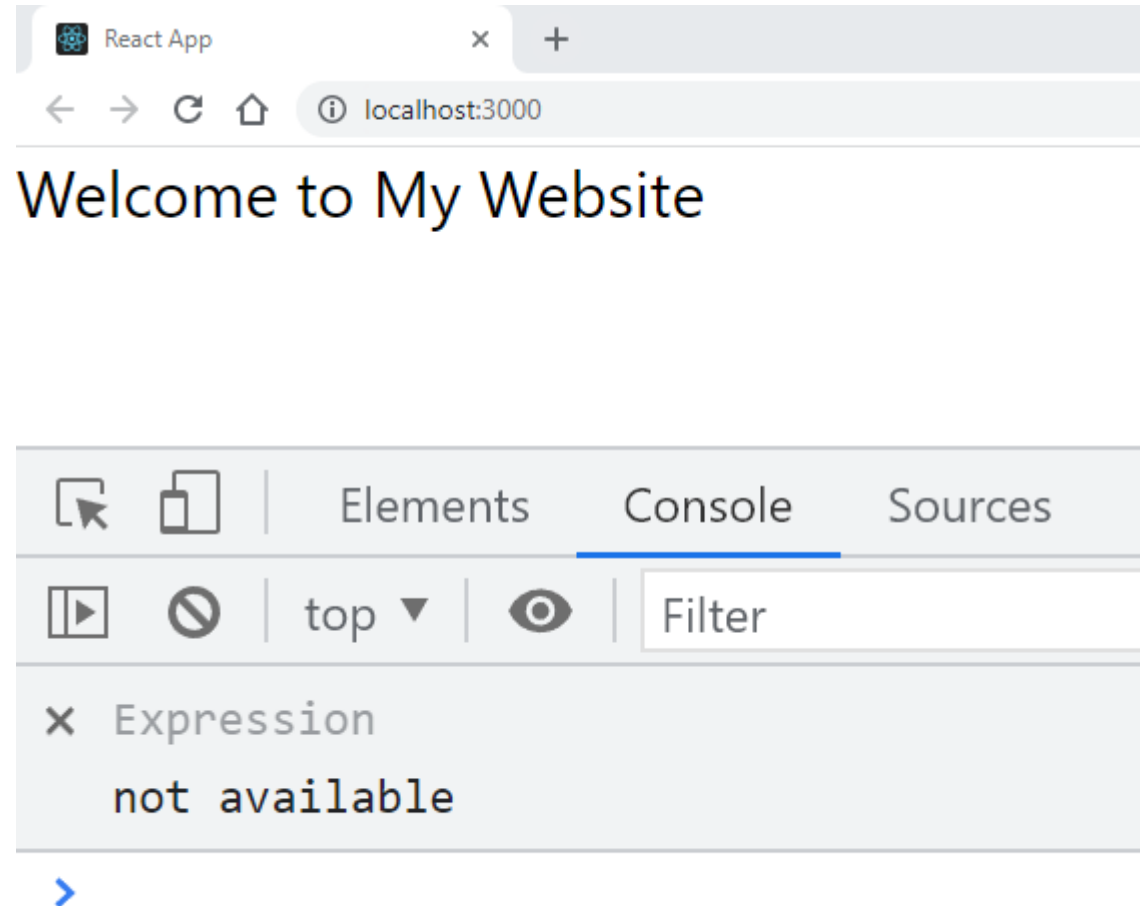
Component Did Mount Called

>



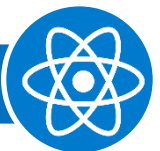
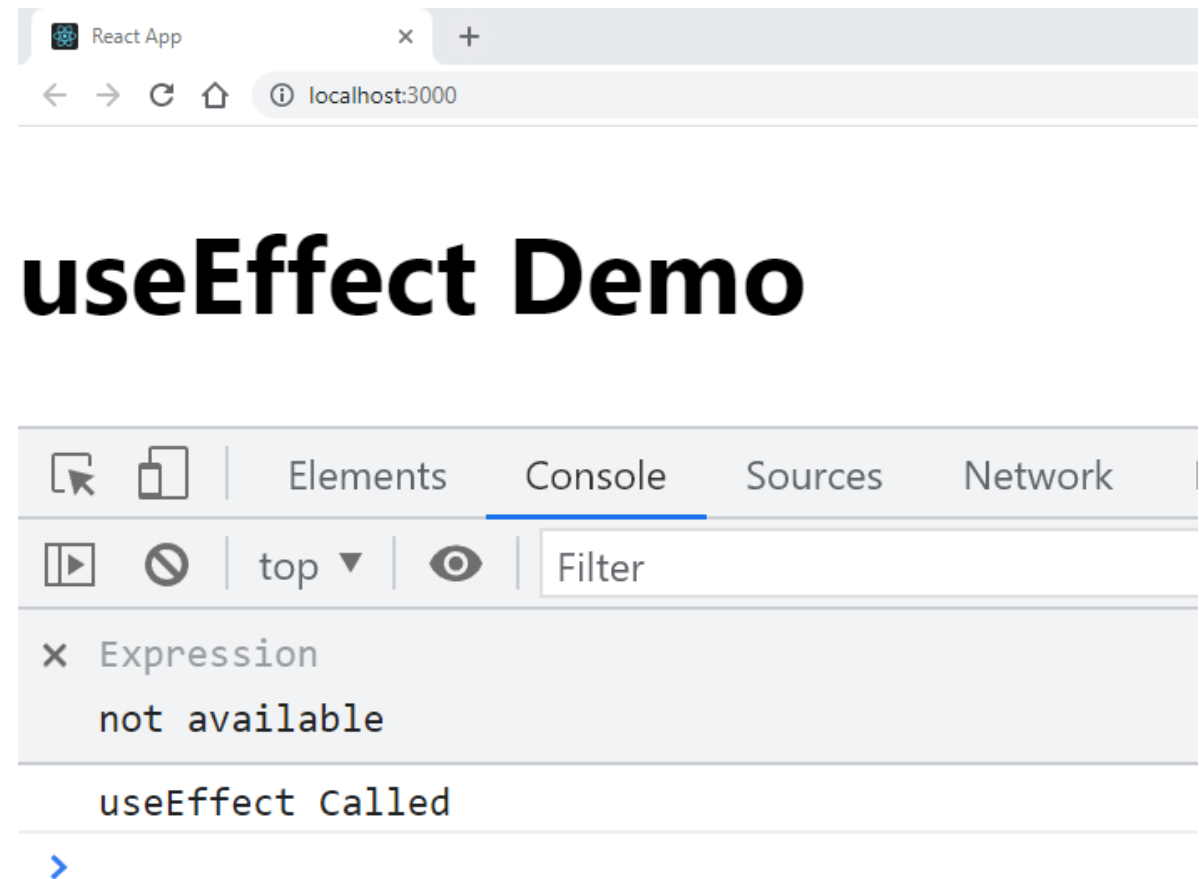
Component Did Mount Update State Value

```
JS App.js M X
src > JS App.js > App > componentDidMount
1 import React from "react";
2 class App extends React.Component {
3   constructor() {
4     super();
5     this.state = {}
6   }
7
8   componentDidMount() {
9     this.setState({
10      txt1: "Welcome to My Website"
11    })
12  }
13
14  render() {
15    return (
16      <div>
17        {this.state.txt1}
18      </div>
19    );
20  }
21 }
22 export default App;
23
```



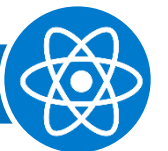
useEffect Demo

```
JS App.js M X
src > JS App.js > App
1  import React from "react";
2
3  function App() {
4    React.useEffect(() => {
5      console.log("useEffect Called");
6    });
7
8    return (
9      <div>
10       <h1>useEffect Demo</h1>
11     </div>
12   );
13 }
14 export default App;
15
```



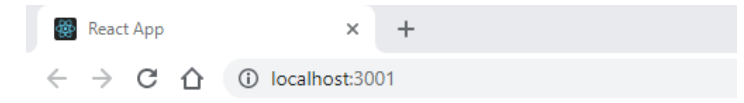
Define State and Print

```
JS App.js 1, M X
src > JS App.js > App
1  import React from "react";
2
3  function App() {
4    const [count, setCount] = React.useState(0);
5
6    React.useEffect(() => {
7      console.log("useEffect Called");
8    });
9
10   return (
11     <div>
12       <h1>Hooks Demo</h1>
13       Count Value is {count}
14     </div>
15   );
16 }
17 export default App;
18
```



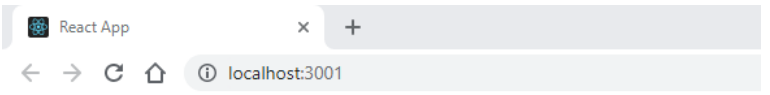
Print and Update State Value

```
JS App.js M X
src > JS App.js > App
1 import React from 'react';
2
3 function App() {
4
5   const [no1, setValue] = React.useState(0);
6
7   return (
8     <div>
9       State Value is : {no1}
10      <br/>
11      <button onClick={() => setValue(100)} >Click Me</button>
12    </div>
13  );
14 }
15 export default App;
```



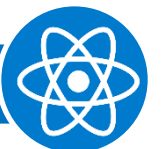
State Value is : 0

Click Me



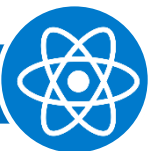
State Value is : 100

Click Me



OnButton Click Change State Value

```
JS App.js M X
src > JS App.js > [🔗] default
1  import React from "react";
2
3  function App() {
4    const [count, setCount] = React.useState(0);
5
6    React.useEffect(() => {
7      console.log("useEffect Called / Refreshed everytime the count changes");
8    });
9
10   return (
11     <div>
12       <h1>Hooks Demo</h1>
13       Count Value is {count} <br/>
14       <button onClick={() => setCount(count + 1)}> Click me </button>
15     </div>
16   );
17 }
18 export default App;
```



Hooks Demo

Count Value is 0

Click me

Elements Console Sources Network Performance

top Filter

× Expression
not available

useEffect Called / Refreshed everytime the count changes

Hooks Demo

Count Value is 2

Click me

Elements Console Sources Network Performance

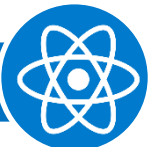
top Filter

× Expression
not available

useEffect Called / Refreshed everytime the count changes

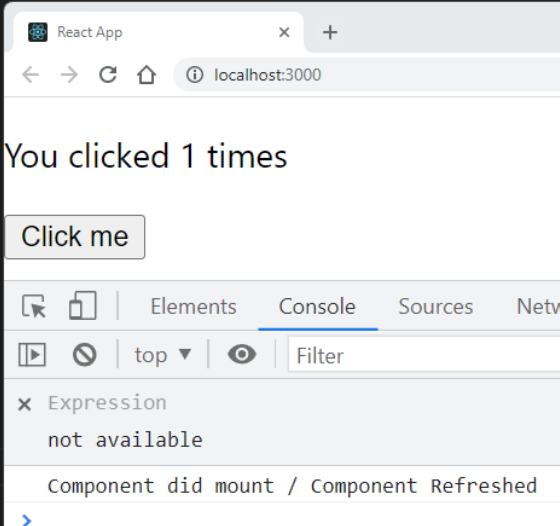
useEffect Called / Refreshed everytime the count changes

useEffect Called / Refreshed everytime the count changes

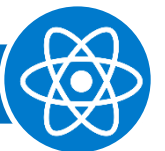


useState and useEffect (React.useState)

```
App.js M X
src > JS App.js > [default]
1  import React from 'react';
2
3  function App() {
4    const [count, setCount] = React.useState(0);
5
6    React.useEffect(() => {
7      console.log("Component did mount / Component Refreshed");
8    });
9
10   return (
11     <div>
12       <p>You clicked {count} times</p>
13       <button onClick={() => setCount(count + 1)}>
14         Click me
15       </button>
16     </div>
17   );
18 }
19
20 export default App;
21
```



```
import React from 'react';
function App() {
  const [count, setCount] = React.useState(0);
  React.useEffect(() => {
    console.log("Component did mount / Component Refreshed");
  });
  return (
    <div>
      <p>You clicked {count} times</p>
      <button onClick={() => setCount(count + 1)}>
        Click me
      </button>
    </div>
  );
}
export default App;
```



Method 2

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

```
App.js M X
src > App.js > App
1 import {useState,useEffect} from 'react';
2 function App() {
3
4   const [count, setCount] = useState(0);
5
6   useEffect(()=>{
7     console.log("Component did mount / Component Refreshed");
8   });
9
10  return (
11    <div>
12      <p>You clicked {count} times</p>
13      <button onClick={() => setCount(count + 1)}>
14        Click me
15      </button>
16    </div>
17  );
18 }
19 export default App;
20
```

React App
localhost:3000

You clicked 1 times

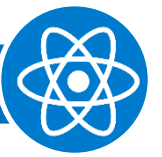
Click me

Elements Console Sources Netwo

top Filter

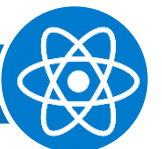
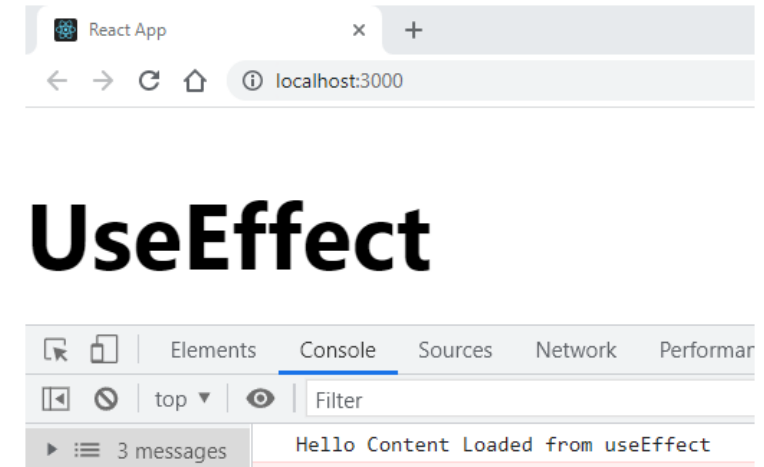
Expression
not available

Component did mount / Component Refreshed



useEffect Example

```
JS App.js M X
src > JS App.js > ...
1 | import React, { useEffect } from "react";
2 |
3 | function App(){
4 |   // useEffect - replacing lifecycle methods
5 |   useEffect(() => {
6 |     console.log(`Hello Content Loaded from useEffect`);
7 |   });
8 |   return (
9 |     <div>
10 |       <h1>UseEffect</h1>
11 |     </div>
12 |   );
13 | };
14 | export default App;
15 |
```



componentDidMount

Before (class-based component):

```
import React from "react";

class Component extends React.Component {

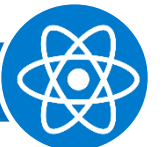
  componentDidMount() {
    console.log("Behavior before the component is
added to the DOM");
  }

  render() {
    return <h1>Hello World</h1>;
  }
};
```

After (functional component using Hooks):

```
import React, { useEffect } from "react";

function App(){
  // useEffect - replacing lifecycle methods
  useEffect(() => {
    console.log(`Hello Content Loaded from useEffect`);
  });
  return (
    <div>
      <h1>UseEffect</h1>
    </div>
  );
};
export default App;
```



useEffect Counter Example

```
JS App.js M X
src > JS App.js > [🔗] default
1 | import React, { useState, useEffect } from "react";
2 |
3 | const App = () => {
4 |   const [count, setCount] = useState(0);
5 |   // useEffect - replacing lifecycle methods
6 |   // takes arrow function as argument
7 |   // as the state changes, useEffect runs
8 |   useEffect(() => {
9 |     document.title = `Clicked ${count} times`;
10 |    console.log(`Clicked ${count} times`);
11 |  });
12 |
13 |   const increment = () => {
14 |     setCount(count + 1);
15 |   };
16 |
17 |   return (
18 |     <div>
19 |       <button onClick={increment}>Clicked {count} times</button>
20 |     </div>
21 |   );
22 | };
23 | export default App;
24 |
```

Clicked 7 times

localhost:3000

Clicked 7 times

Elements Console Sources Network

top Filter

7 messages

7 user mess...

No errors

No warnings

7 info

No verbose

Clicked 1 times

Clicked 2 times

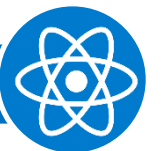
Clicked 3 times

Clicked 4 times

Clicked 5 times

Clicked 6 times

Clicked 7 times



Using useEffect Counter Example

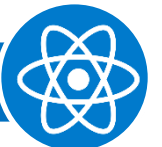
```
import React, { useState, useEffect } from "react";

const App = () => {
  const [count, setCount] = useState(0);
  // useEffect - replacing lifecycle methods
  // takes arrow function as argument
  // as the state changes, useEffect runs
  useEffect(() => {
    document.title = `Clicked ${count} times`;
    console.log(`Clicked ${count} times`);
  });

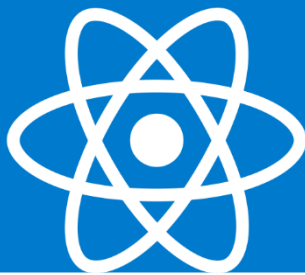
  const increment = () => {
    setCount(count + 1);
  };

  return (
    <div>
      <button onClick={increment}>Clicked {count} times</button>
    </div>
  );
};

export default App;
```



| Sum using Hooks



Sum

```
App.js 1, M x
src > App.js > ...
1 import React from 'react';
2
3 function App() {
4   const [no1, setNo1Value] = React.useState(0);
5   const [no2, setNo2Value] = React.useState(0);
6   const [ans, setAnsValue] = React.useState(0);
7
8   const submitValue = (event) => {
9     var a = no1;
10    var b = no2;
11    var c = parseInt(a) + parseInt(b);
12    setAnsValue(c);
13    alert("Sum is " + c);
14    event.preventDefault(); //Function Will Print Data as it is
15  }
16
17  return (
18    <div>
19      <form onSubmit={submitValue}>
20        No1 : <input type="text" onChange={e => setNo1Value(e.target.value)} />
21        No2 : <input type="text" onChange={e => setNo2Value(e.target.value)} />
22        <br/>
23        No1 is {no1} <br/>
24        No2 is {no2} <br/>
25        <input type="submit"/>
26      </form>
27    </div>
28  );
29
30 export default App;
31
```

React App

127.0.0.1:3000/?

No1 : No2 :

No1 is 0
No2 is 0

React App

127.0.0.1:3000/?

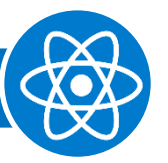
No1 : No2 :

No1 is 10
No2 is 20

127.0.0.1:3000 says

Sum is 30

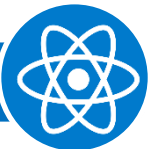
OK




```
import React from 'react';
```

```
function App() {  
  const [no1, setNo1Value] = React.useState(0);  
  const [no2, setNo2Value] = React.useState(0);  
  const [ans, setAnsValue] = React.useState(0);  
  
  const submitValue = (event) => {  
    var a = no1;  
    var b = no2;  
    var c = parseInt(a) + parseInt(b);  
    setAnsValue(c);  
    alert("Sum is " + c);  
    event.preventDefault(); //Function Will Print Data as it is  
  }  
  return (  
    <div>  
      <form onSubmit={submitValue}>  
        No1 : <input type="text" onChange={e => setNo1Value(e.target.value)} />  
        No2 : <input type="text" onChange={e => setNo2Value(e.target.value)} />  
        <br/>  
        No1 is {no1} <br/>  
        No2 is {no2} <br/>  
        <input type="submit"/>  
      </form>  
    </div>  
  );  
}
```

```
export default App;
```



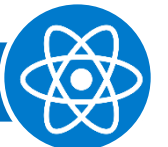
Multiple Value Using Object

```
Js App.js 1, M X
src > Js App.js > [default]
1  import React from 'react';
2  function App() {
3    const [myData, myDataUpdate] = React.useState({});
4    const [ans, setAnsValue] = React.useState(0)
5
6    const onChangeEvent = (e) => {
7      myDataUpdate((myData) => ({
8        ...myData,
9        [e.target.name]: e.target.value
10      }));
11    }
12
13    const submitValue = (event) => {
14      var c = parseInt(myData.txt1) + parseInt(myData.txt2);
15      setAnsValue(c);
16      alert("Answer is " + c);
17    }
18    return (
19      <div>
20        <form onSubmit={submitValue}>
21          No1 : <input type="text" name='txt1' onChange={onChangeEvent} />
22          No2 : <input type="text" name='txt2' onChange={onChangeEvent} />
23          <input type="submit" />
24        </form>
25      </div>
26    );
27  }
28  export default App;
```

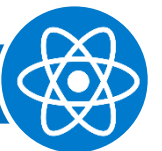
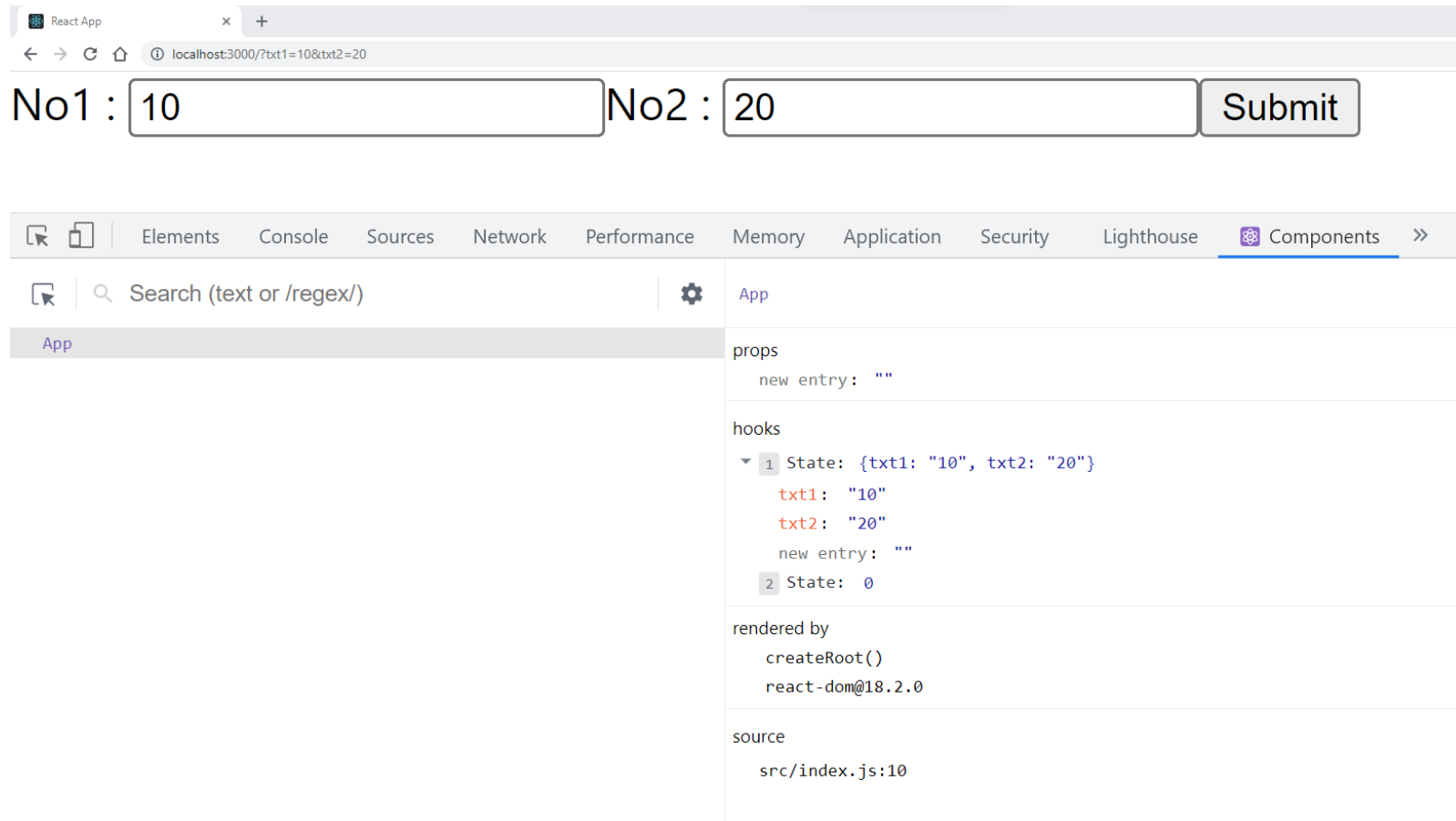
```
import React from 'react';
function App() {
  const [myData, myDataUpdate] = React.useState({});
  const [ans, setAnsValue] = React.useState(0)

  const onChangeEvent = (e) => {
    myDataUpdate((myData) => ({
      ...myData,
      [e.target.name]: e.target.value
    }));
  }

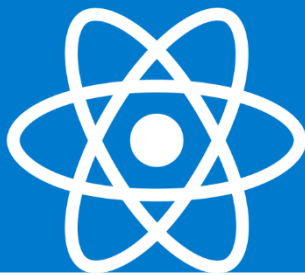
  const submitValue = (event) => {
    var c = parseInt(myData.txt1) + parseInt(myData.txt2);
    setAnsValue(c);
    alert("Answer is " + c);
  }
  return (
    <div>
      <form onSubmit={submitValue}>
        No1 : <input type="text" name='txt1' onChange={onChangeEvent} />
        No2 : <input type="text" name='txt2' onChange={onChangeEvent} />
        <input type="submit" />
      </form>
    </div>
  );
}
export default App;
```



Output

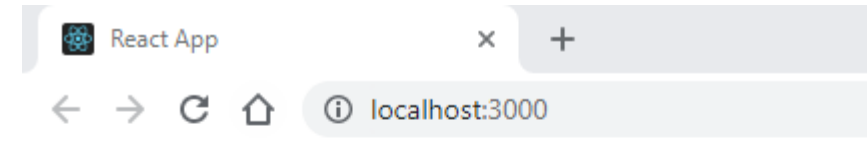


| Digital Clock

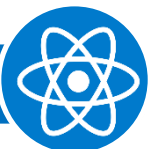


Digital Clock

```
App.js
src > App.js > default
1 import React, { useState } from 'react';
2
3 function App() {
4   let currenttime = new Date().toLocaleTimeString();
5   const [mytime, setmyTime] = useState(currenttime);
6
7   const updateTime = () => {
8     let currenttime1 = new Date().toLocaleTimeString();
9     setmyTime(currenttime1);
10  }
11  setInterval(updateTime, 1000);
12
13  return (
14    <React.Fragment>
15      <h1>{mytime}</h1>
16    </React.Fragment>
17  );
18 }
19 export default App;
20
```



21:43:16



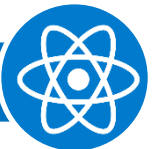
Clock Code

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

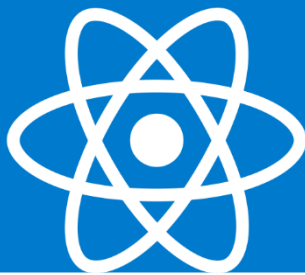
function App() {
  let currenttime = new Date().toLocaleTimeString()
  const [mytime, setmyTime] = useState(currenttime);

  const updateTime = () => {
    let currenttime1 = new Date().toLocaleTimeString();
    setmyTime(currenttime1);
  }
  setInterval(updateTime, 1000);

  return (
    <React.Fragment>
      <h1>{mytime}</h1>
    </React.Fragment>
  );
}
export default App;
```



| Fetch Data



JS App.js M X

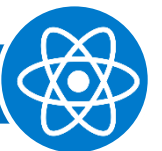
src > JS App.js > DataLoader

```
1 import React, { useState, useEffect } from "react";
2
3 export default function DataLoader() {
4   const [data, setData] = useState([]);
5
6   useEffect(() => {
7     fetch("https://jsonplaceholder.typicode.com/posts")
8       .then(response => response.json())
9       .then(data => setData(data));
10
11   });
12
13   return (
14     <div>
15       <ul>
16         {data.map(mydata => (
17           <li key={mydata.id}>{mydata.title}</li>
18         ))}
19       </ul>
20     </div>
21   );
22 }
```

React App

localhost:3000

- sunt aut facere repellat provident occaecati excepturi optio repr
- qui est esse
- ea molestias quasi exercitationem repellat qui ipsa sit aut
- eum et est occaecati
- nesciunt quas odio
- dolorem eum magni eos aperiam quia
- magnam facilis autem
- dolorem dolore est incam

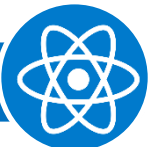



```
import React, { useState, useEffect } from "react";

export default function DataLoader() {
  const [data, setData] = useState([]);

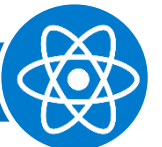
  useEffect(() => {
    fetch("https://jsonplaceholder.typicode.com/posts")
      .then(response => response.json())
      .then(data => setData(data));
  });

  return (
    <div>
      <ul>
        {data.map(mydata => (
          <li key={mydata.id}>{mydata.title}</li>
        ))}
      </ul>
    </div>
  );
}
```

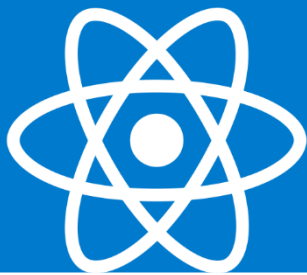


useContext

- In a typical React application, data is passed top-down (parent to child) via props, but this can be cumbersome for certain types of props (e.g. locale preference, UI theme) that are required by many components within an application.
- Context provides a way to share values like these between components without having to explicitly pass a prop through every level of the tree.

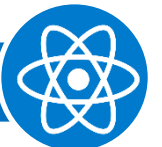


| Custom Hooks



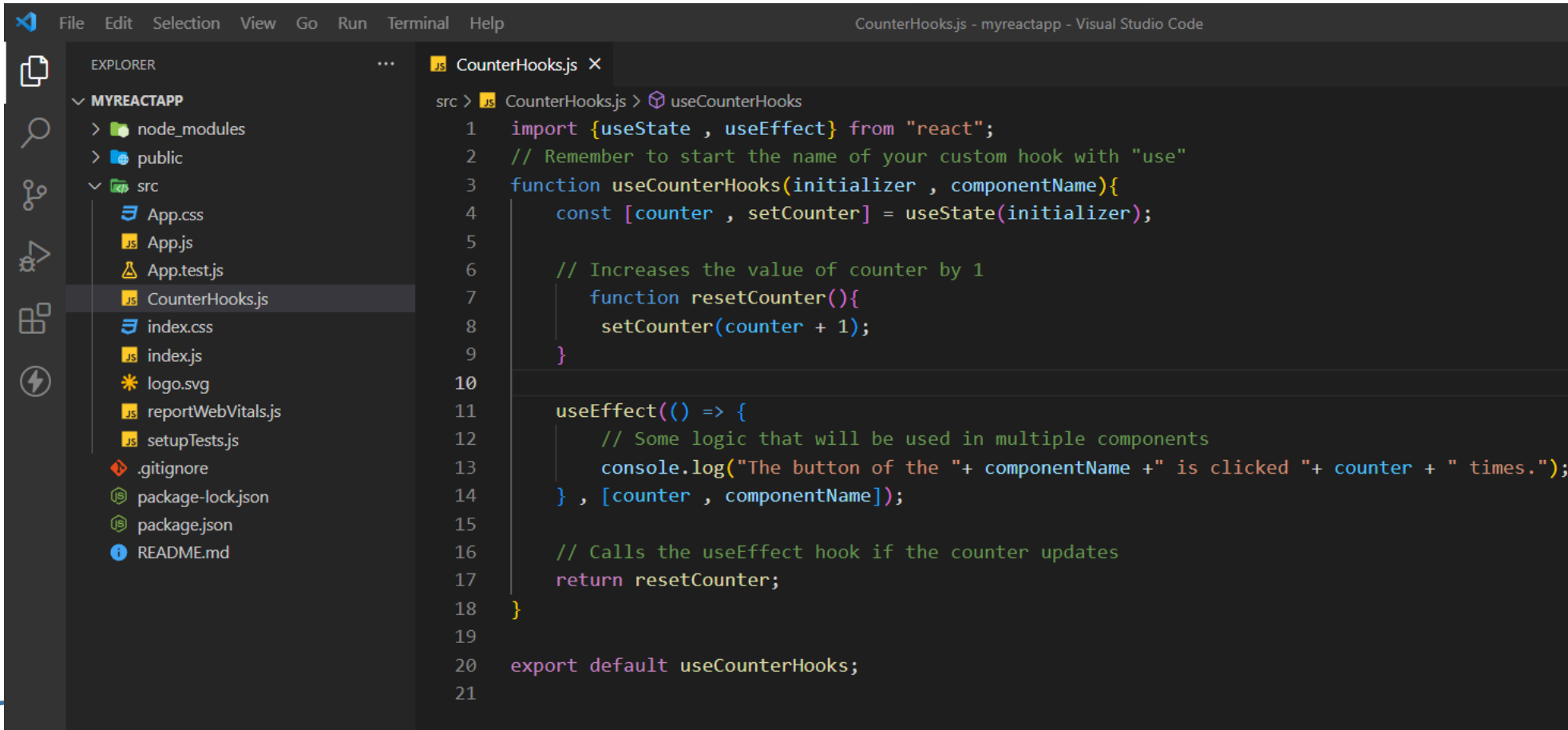
Custom hooks

- A custom hook allows you to extract some components logic into a reusable function.
- A custom hook is a Javascript function that starts with use and that call can other hooks.
- Remember that components and hooks are functions, so we are really not creating any new concepts here.
- We are just refactoring our code into another function to make it reusable.
- Since the release of the React Hooks, there has been an explosive growth of custom hooks, thousands of React devs all over the world have churned out hundreds of custom hooks that simplify most of the arduous and boring tasks we do in React projects.



Create Custom Hooks

- We can construct logic that can be reused across our applications by creating our custom hooks. It generates a lot of reusable features.

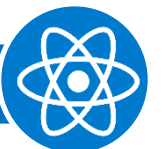


The screenshot shows the Visual Studio Code interface with a project named 'myreactapp'. The Explorer panel on the left shows the file structure, including 'src' with files like 'App.css', 'App.js', 'App.test.js', 'CounterHooks.js', 'index.css', 'index.js', 'logo.svg', 'reportWebVitals.js', 'setupTests.js', '.gitignore', 'package-lock.json', 'package.json', and 'README.md'. The 'CounterHooks.js' file is selected and open in the editor. The code in the editor is as follows:

```
src > JS CounterHooks.js > useCounterHooks
1  import {useState , useEffect} from "react";
2  // Remember to start the name of your custom hook with "use"
3  function useCounterHooks(initializer , componentName){
4      const [counter , setCounter] = useState(initializer);
5
6      // Increases the value of counter by 1
7      function resetCounter(){
8          setCounter(counter + 1);
9      }
10
11     useEffect(() => {
12         // Some logic that will be used in multiple components
13         console.log("The button of the "+ componentName + " is clicked "+ counter + " times.");
14     } , [counter , componentName]);
15
16     // Calls the useEffect hook if the counter updates
17     return resetCounter;
18 }
19
20 export default useCounterHooks;
21
```

useCounterHooks.js

```
import {useState , useEffect} from "react";  
// Remember to start the name of your custom hook with "use"  
function useCounterHooks(initializer , componentName){  
  const [counter , setCounter] = useState(initializer);  
  
  // Increases the value of counter by 1  
  function resetCounter(){  
    setCounter(counter + 1);  
  }  
  
  useEffect(() => {  
    // Some logic that will be used in multiple components  
    console.log("The button of the "+ componentName +" is clicked "+ counter + " times.");  
  } , [counter , componentName]);  
  
  // Calls the useEffect hook if the counter updates  
  return resetCounter;  
}  
  
export default useCounterHooks;
```

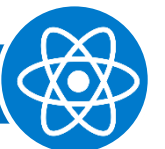


App.js

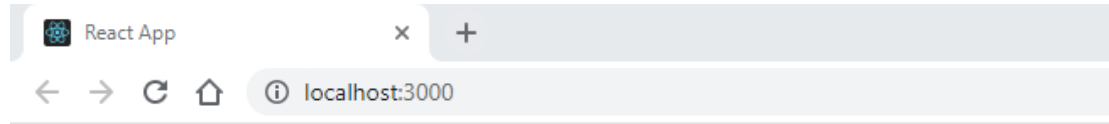
```
JS CounterHooks.js JS App.js X
src > JS App.js > App
1  import React from "react";
2  // importing the custom hook
3  import useCounterHooks from "../CounterHooks";
4  function App(props){
5      // ClickedButton = resetCounter;
6      const clickedButton = useCounterHooks(0 , "AppComponent");
7      return (
8          <div>
9              <h1> This is App Component</h1>
10             <button onClick={clickedButton}>
11                 Click here!
12             </button>
13         </div>
14     );
15 }
16
17 export default App;
18
```

```
import React from "react";
// importing the custom hook
import useCounterHooks from "../CounterHooks";
function App(props){
    // ClickedButton = resetCounter;
    const clickedButton = useCounterHooks(0 , "AppComponent");
    return (
        <div>
            <h1> This is App Component</h1>
            <button onClick={clickedButton}>
                Click here!
            </button>
        </div>
    );
}

export default App;
```

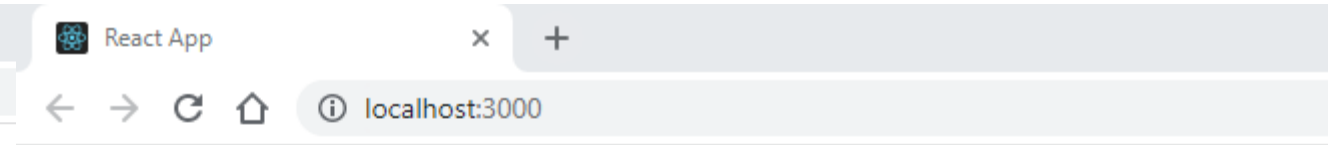
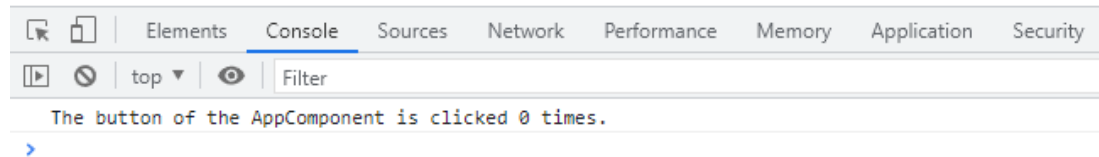


Output



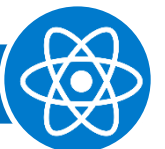
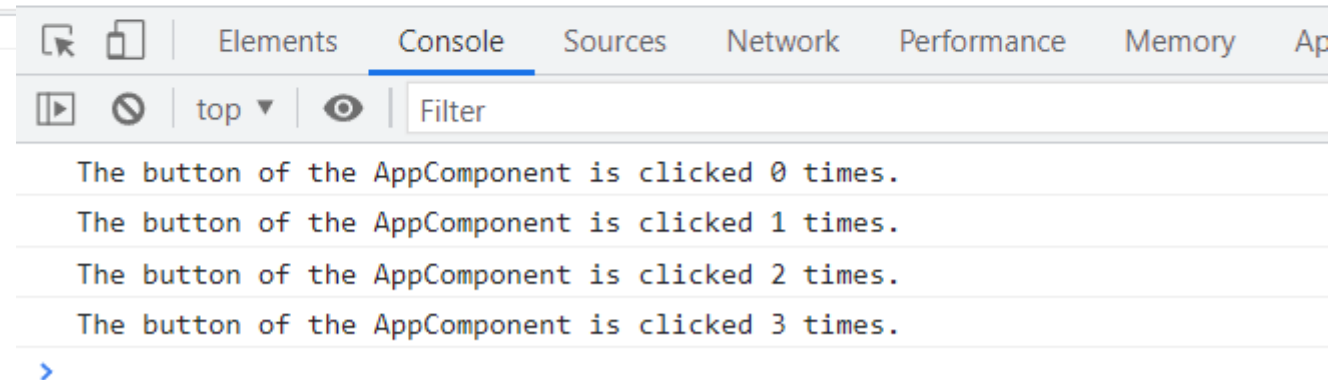
This is App Component

Click here!



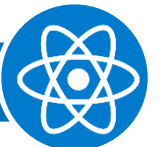
This is App Component

Click here!



Library List

- useScript
- useLocalStorage
- useIdb
- use-mouse-action
- useOnlineStatus
- useDocumentTitle
- useNetworkStatus
- useClippy

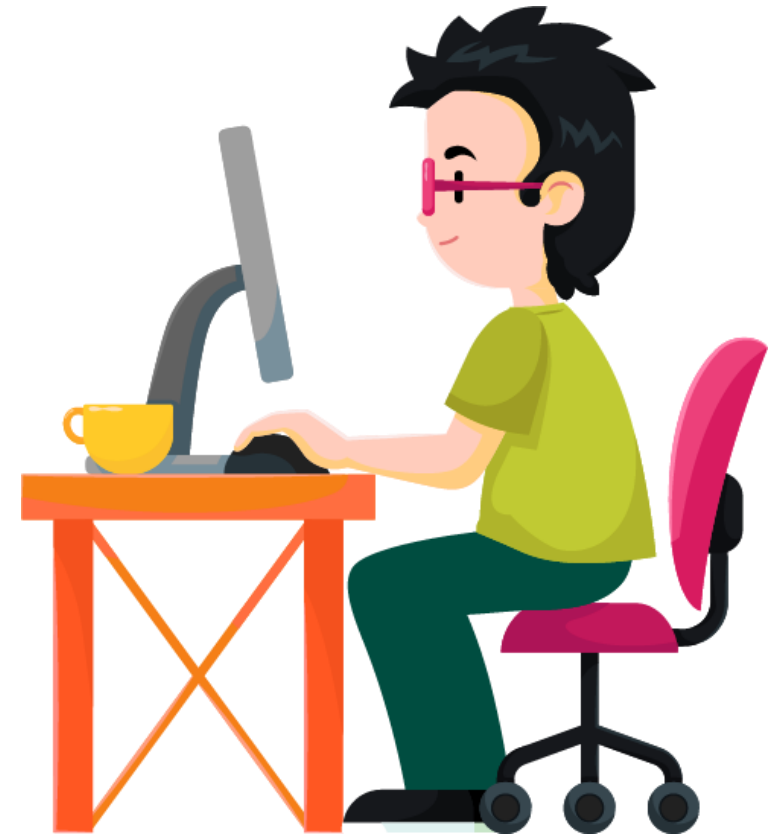


Get Exclusive Video Tutorials



www.apptutorials.com

<https://www.youtube.com/user/Akashtips>





Get More Details

www.akashsir.com



If You Liked It !

Rating Us Now



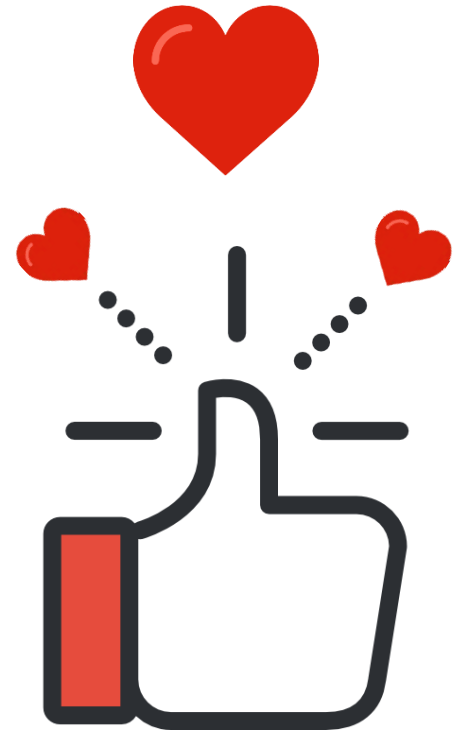
Just Dial

https://www.justdial.com/Ahmedabad/Akash-Technolabs-Navrangpura-Bus-Stop-Navrangpura/079PXX79-XX79-170615221520-S5C4_BZDET



Sulekha

<https://www.sulekha.com/akash-technolabs-navrangpura-ahmedabad-contact-address/ahmedabad>



Connect With Me



Akash Padhiyar
#AkashSir

www.akashsir.com

www.akashtechlabs.com

www.akashpadhiyar.com

www.apptutorials.com

Social Info



Akash.padhiyar



Akashpadhiyar



Akash_padhiyar



+91 99786-21654



#Akashpadhiyar

#apptutorials