

Lesson 04 Demo 05

Create a React Application Using the Custom Hooks

Objective: To develop a React application that demonstrates the working of custom Hooks

Tools required: Node terminal, React App, and Visual Studio Code

Prerequisites: Knowledge of creating a React app and understanding of the folder structure

Steps to be followed:

1. Create a new **React** app
2. Create a new file called **useCounter.js** in the **src** directory
3. Import **useCounter** from the **useCounter.js** file in **App.js**
4. Run the app and view it in the browser

Step 1: Create a new React app

1.1 Open your terminal and run the command:

```
npx create-react-app custom-hook-demo
```

```
shreemayeebhatt@ip-172-31-22-250:~$ npx create-react-app custom-hook-demo
```

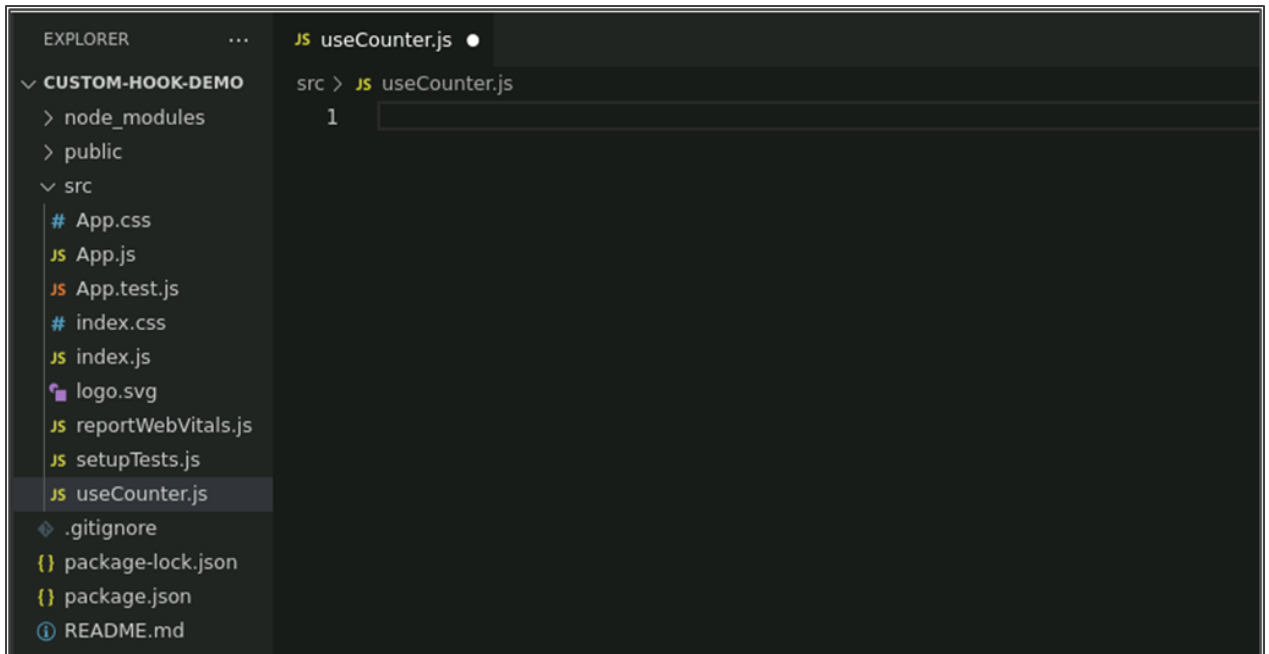
This command will create a new React app with the name **custom-hook-demo**

1.2 Run the **cd custom-hook-demo** command in the terminal to change the current directory to the newly created React app directory

```
Happy hacking!  
shreemayeebhatt@ip-172-31-22-250:~$ cd custom-hook-demo/
```

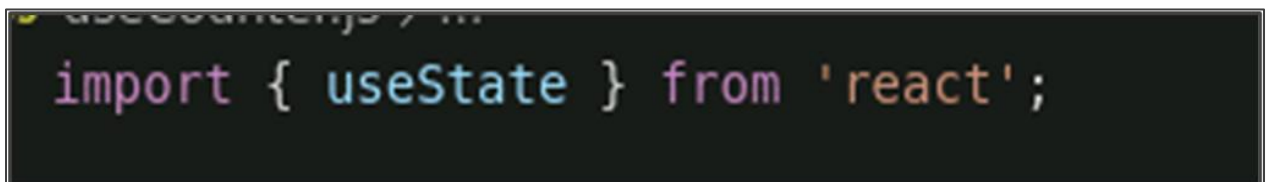
Step 2: Create a new file called `useCounter.js` in the `src` directory

2.1 Open **Visual Studio Code** and navigate to the **custom-hook-demo** directory. In the **src** directory, create the **`useCounter.js`** file

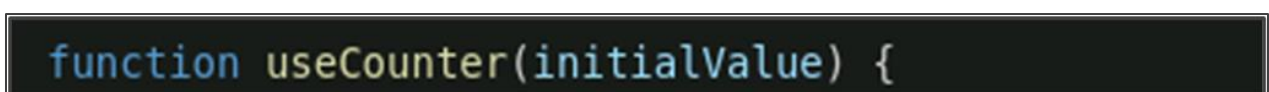


2.2 Inside the **`useCounter.js`** file, import the **`useState`** Hook from the React library using the following code:

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



2.3 Define the **`useCounter`** function that takes an **`initialValue`** as a parameter as shown in the screenshot:



2.4 Inside the **useCounter** function, use the **useState** Hook to create the **count** state variable and **setCount** function to update the state

```
const [count, setCount] = useState(initialValue);
```

2.5 Define the **increment** and **decrement** functions that update the **count** state by increasing or decreasing it as shown in the screenshot:

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

2.6 Define a **return** object with the **count**, **increment**, and **decrement** values as shown in the screenshot

```
return {  
  count,  
  increment,  
  decrement,  
};  
  
export default useCounter;
```

Note: Refer to the following code to configure the **useCounter.js** file:

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

function useCounter(initialValue) {
  const [count, setCount] = useState(initialValue);

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

  const decrement = () => {
    setCount(count - 1);
  };

  return {
    count,
    increment,
    decrement,
  };
}

export default useCounter;
```

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

function useCounter(initialValue) {
  const [count, setCount] = useState(initialValue);

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

  const decrement = () => {
    setCount(count - 1);
  };

  return {
    count,
    increment,
    decrement,
  };
}

export default useCounter;
```

Step 3: Import useCounter from the useCounter.js file in App.js

- 3.1 Open the **App.js** file in your code editor and modify it by importing the **useCounter** Hook from the **useCounter.js** file

```
import useCounter from './useCounter';
```

3.2 Inside the **App** function component, call the **useCounter** Hook and assign the returned values to the variables

```
function App() {  
  const { count, increment, decrement } = useCounter(0);
```

3.3 Use the **count**, **increment**, and **decrement** variables in the **JSX** to display the count and handle the increment and decrement functionalities

```
return (  
  <div className="App">  
    <h1>Custom Hook Demo</h1>  
    <p>Count: {count}</p>  
    <button onClick={increment}>+</button>  
    <button onClick={decrement}>-</button>  
  </div>  
);  
}
```

Note: Refer to the following code to configure the App.js file:

```
import React from 'react';  
import useCounter from './useCounter';  
import './App.css';
```

```
function App() {  
  const { count, increment, decrement } = useCounter(0);
```

```
return (  
  <div className="App">  
    <h1>Custom Hook Demo</h1>  
    <p>Count: {count}</p>  
    <button onClick={increment}>+</button>  
    <button onClick={decrement}>-</button>  
  </div>  
);  
}
```

```
export default App;
```

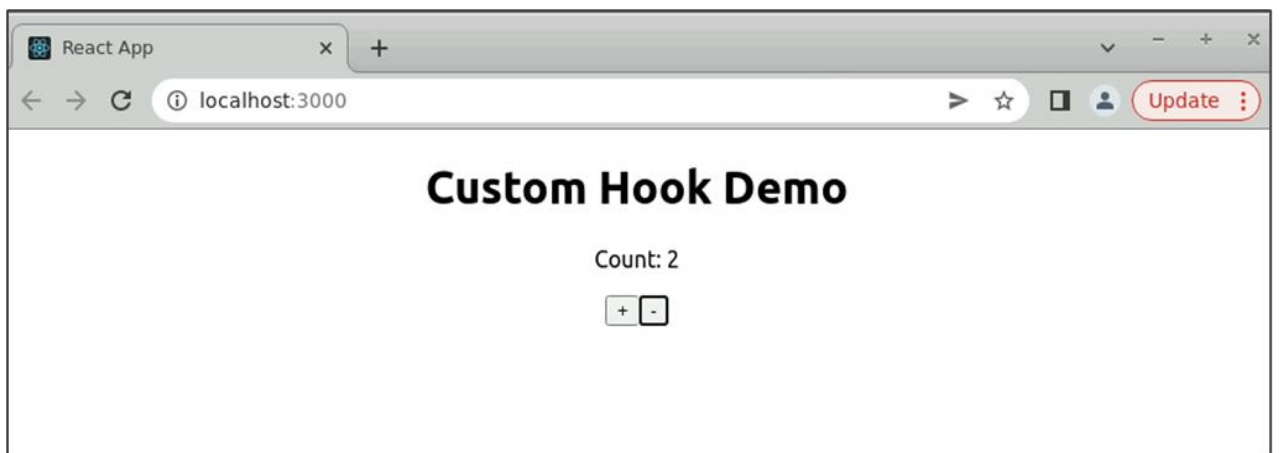
```
s App.js > ...  
import logo from './logo.svg';  
import React from 'react';  
import useCounter from './useCounter';  
import './App.css';  
  
function App() {  
  const { count, increment, decrement } = useCounter(0);  
  
  return (  
    <div className="App">  
      <h1>Custom Hook Demo</h1>  
      <p>Count: {count}</p>  
      <button onClick={increment}>+</button>  
      <button onClick={decrement}>-</button>  
    </div>  
  );  
}  
  
export default App;
```

Step 4: Run the app and view it in the browser

4.1 In the terminal, run the command **npm start** to start the app

```
shreemayeebhatt@ip-172-31-22-250:~/custom-hook-demo$ npm start
```

4.2 Open your browser and navigate to <http://localhost:3000> to view the final output



You will see a simple app with a counter displayed with two buttons that increment and decrement the count using the custom **useCounter** Hook.