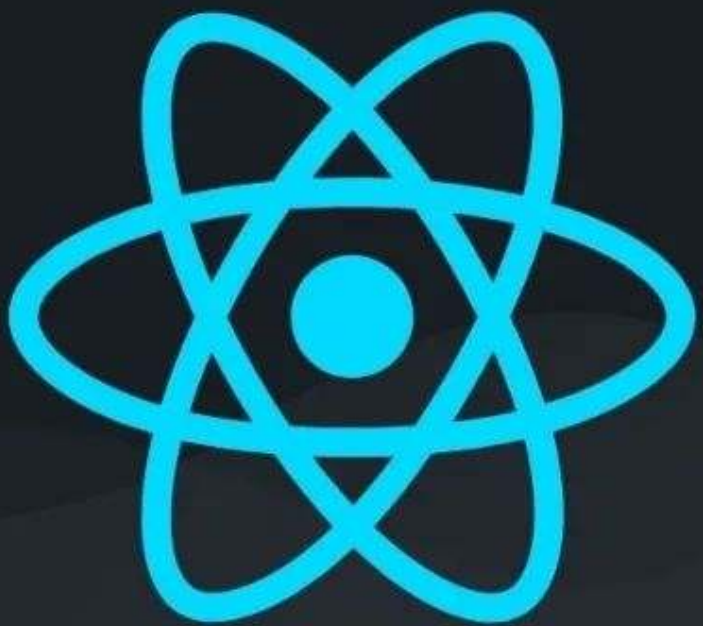
React Router Hooks

In Detail



Mallikarjun | @CodeBustler

Router Components

React Router provides several hooks that allow you to interact with the router and access information about the current navigation state. Here are some commonly used React Router hooks

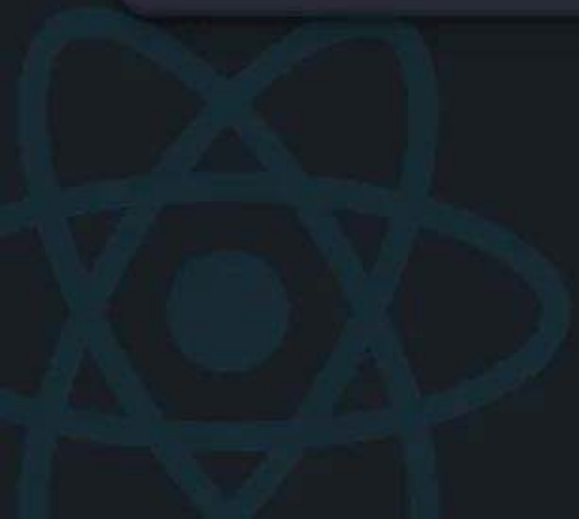
- useNavigate( )
- useLocation( )
- useParams( )
- useOutlet( )
- useRoutes( )

1. useNavigate( )

Provides the navigate function to programmatically **navigate to a different route**.

Example

```
import { useNavigate } from 'react-router-dom';

function MyComponent() {
  const navigate = useNavigate();

  const handleClick = () => {
    // Navigate to the "/about" route
    navigate('/about');
  };

  return <button onClick={handleClick}>Go to About</button>;
}
```



2. useLocation( )

Returns the current location object, representing **where the app is now.**

```
import { useLocation } from 'react-router-dom';

function CurrentPath() {
  const location = useLocation();

  return <p>Current path is {location.pathname}</p>;
}
```

Example

3. useParams( )

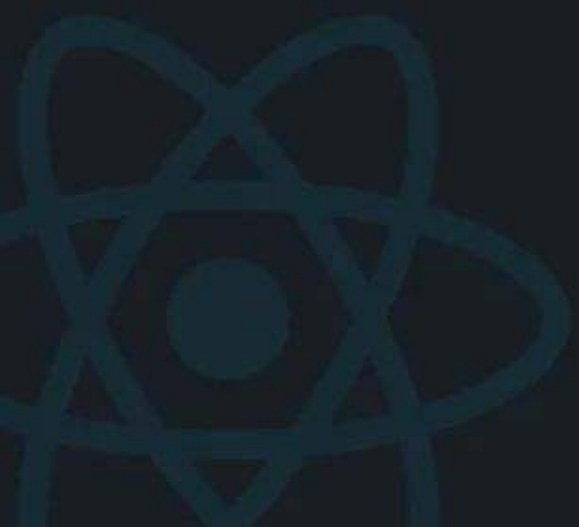
Retrieves the parameters from the current route's URL.

```
import { useParams } from 'react-router-dom';

function UserProfile() {
  const { username } = useParams();

  return <p>User profile of {username}</p>;
}


// <Route path="/users/:userId" element={<UserDetails />} />
```

4. useOutlet( )

useOutlet in React Router v6 lets you dynamically **render nested routes** within a parent component without explicitly listing them in the render method.

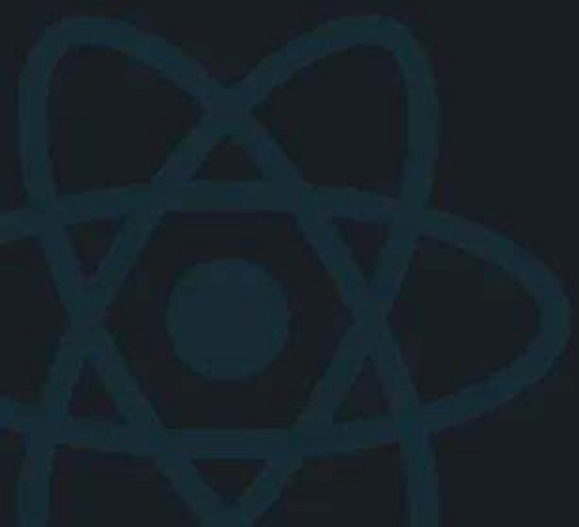
It's an **alternative to <Outlet>,** streamlining the parent component's structure for more dynamic nested route handling.



@CodeBustler

useOutlet( ) | Example

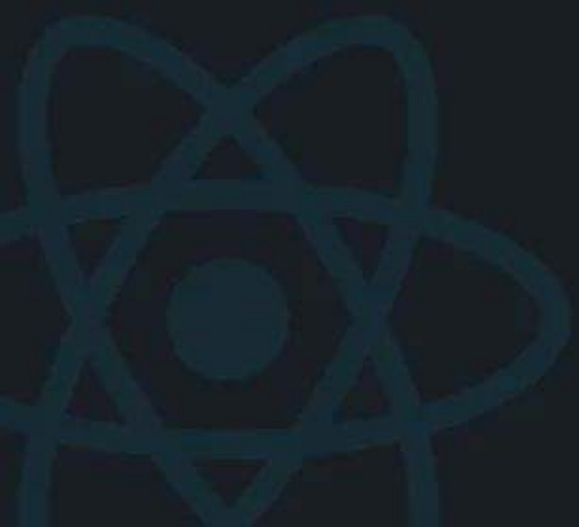
```jsx
import React from 'react';
import { useOutlet } from 'react-router-dom';

const ParentComponent = () => {
  const outlet = useOutlet();

  return (
    <div>
      <h1>Parent Component</h1>
      {outlet}
    </div>
  );
};
```



5. useRoutes( )

the useRoutes hook is a functional **alternative** to the traditional <Routes> ... <Route> approach for defining routes.

It allows you to **create route objects** using **JavaScript objects**, offering a more procedural and dynamic way to manage navigation within your application



useRoutes( ) | Example

```jsx
import React from 'react';
import { useRoutes } from 'react-router-dom';

const routes = [
  {
    path: '/',
    element: <Home />,
  },
  {
    path: '/about',
    element: <About />,
  },
  {
    path: '/users',
    element: <Users />,
    children: [
      {
        path: ':userId',
        element: <UserDetail />,
      },
    ],
  },
];

const App = () => {
  const routeElements = useRoutes(routes);

  return (
    <div>
      <h1>My App</h1>
      {routeElements}
    </div>
  );
};
```

@CodeBustler

Did you find it Useful?

Leave a comment!

Alamin CodePapa

@CodePapa360

FOLLOW FOR MORE

Like Comment Repost