Hema Coding School

YouTube Link: https://www.youtube.com/@HemaCodingSchool

Notes on useNavigate and useParams in React Router DOM

useNavigate

Purpose: useNavigate is a hook provided by React Router DOM that allows you to programmatically navigate to different routes within your application. This is useful for navigation actions that are not triggered by user interactions with a Link or NavLink component.

Import the Hook:

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

Initialize the Hook:

```
const navigate = useNavigate();
```

Programmatically Navigate:

```
function handleClick() {
  navigate('/target-route'); // Navigate to the specified route
}
```

Example:

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

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

  const goToAbout = () => {
    navigate('/about'); // Navigates to the /about route
  };

  return (
    <div>
      <h1>Home Page</h1>
      <button onClick={goToAbout}>Go to About Page</button>
    </div>
  );
}

export default Home;
```

Additional Options:

- **replace:** Replace the current entry in the history stack instead of adding a new one.

```
navigate('/about', { replace: true });
```

state: Pass state to the new route.

```
navigate('/about', { state: { from: 'home' } });
```

useParams

Purpose: useParams is a hook provided by React Router DOM that allows you to access URL parameters. This is useful for dynamic routes where parts of the URL can change.

Usage:

1. **Import the Hook:**

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

Extract URL Parameters:

```
const { paramName } = useParams();
```

example:

```
import React, { useEffect, useState } from 'react';
import { useParams } from 'react-router-dom';

function ProductDetails() {
  const { productId } = useParams(); // Extract the productId parameter from the URL
  const [product, setProduct] = useState(null);

  useEffect(() => {
    // Fetch product details based on the productId
    fetch(`https://fakestoreapi.com/products/${productId}`)
      .then((res) => res.json())
      .then((data) => setProduct(data))
      .catch((error) => console.error('Error fetching product:', error));
  }, [productId]);

  if (!product) return <div>Loading...</div>;
  return (
    <div>
      <h2>{product.title}</h2>
      <p>{product.description}</p>
      <p>Price: ${product.price}</p>
      <img src={product.image} alt={product.title} style={{ width: '200px' }} />
    </div>
  );
}

export default ProductDetails;
```

Dynamic Route Definition:

- Define a route with a parameter in your router configuration

```

import { BrowserRouter as Router, Routes, Route } from 'react-router-dom';
import ProductDetails from './ProductDetails';

function App() {
  return (
    <Router>
      <Routes>
        <Route path="/products/:productId" element={<ProductDetails />} />
      </Routes>
    </Router>
  );
}

export default App;
```

Navigate in React Router DOM

Purpose: Navigate is a component provided by React Router DOM that allows you to programmatically redirect users to a different route. It is often used for conditional redirects, such as after form submissions, authentication, or when a specific condition is met in the component logic.

Import the Component:

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

Use the Navigate Component:

```

<Navigate to="/target-route" replace={true} />
```

Props:

- **to:** The path to navigate to.

```
<Route path="" element={<Navigate to="electronics" />} />
```

replace: A boolean indicating whether to replace the current entry in the history stack instead of adding a new one. This is similar to the replace option in useNavigate.

```
<Navigate to="/target-route" replace={true} />
```

state: An object that you can pass to the new route, similar to the state in useNavigate.

```
<Navigate to="/home" state={{ from: 'login' }} />
```

Interview Questions:

Navigate

Question 1: What is the Navigate component in React Router DOM used for?

Answer: The Navigate component is used to programmatically redirect users to a different route within a React application. It is useful for conditional redirects, such as after form submissions, authentication, or when a specific condition is met.

Question 2: How do you use the Navigate component to redirect to a different route?

Answer: To use the Navigate component, import it from react-router-dom and include it in your component, specifying the to prop with the target route.

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

function ExampleComponent() {
  return <Navigate to="/target-route" />;
}
```

Question 3: How can you replace the current entry in the history stack when using the Navigate component?

Answer: You can replace the current entry in the history stack by setting the replace prop to true.

Example:

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

function ExampleComponent() {
  return <Navigate to="/target-route" replace={true} />;
}
```

useNavigate

Question 4: What is the useNavigate hook used for in React Router DOM?

Answer: The useNavigate hook is used to programmatically navigate to different routes within a React application. It provides a function that can be called to change the current route.

Question 5: How do you use the useNavigate hook to navigate to a different route?

Answer: To use the useNavigate hook, import it from react-router-dom, call it within a functional component to get the navigate function, and use that function to navigate to a different route.

Example:

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

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

  const handleClick = () => {
    navigate('/target-route');
  };

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

Question 6: How can you replace the current entry in the history stack when using the useNavigate hook?

Answer: You can replace the current entry in the history stack by passing an options object with replace set to true.

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

```
function ExampleComponent() {
  const navigate = useNavigate();

  const handleClick = () => {
    navigate('/target-route', { replace: true });
  };

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

useParams

Question 7: What is the useParams hook used for in React Router DOM?

Answer: The useParams hook is used to access URL parameters in a React application. It returns an object containing the key-value pairs of the URL parameters.

Question 8: How do you use the useParams hook to access URL parameters?

Answer: To use the useParams hook, import it from react-router-dom and call it within a functional component to get the URL parameters.

Example:

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

function ExampleComponent() {
  const { paramName } = useParams();

  return <div>Parameter value: {paramName}</div>;
}
```

Question 9: How would you define a route with parameters and access those parameters using the useParams hook?

Answer: To define a route with parameters, include the parameter name prefixed with a colon (:) in the route path. Use the useParams hook to access those parameters.

```
import { BrowserRouter as Router, Routes, Route } from 'react-router-dom';
import React from 'react';
import ExampleComponent from './ExampleComponent';

function App() {
  return (
    <Router>
      <Routes>
```

```
        <Route path="/example/:paramName" element={<ExampleComponent />} />
      </Routes>
    </Router>
  );
}

export default App;
```