upGrad ENTERPRISE

React JS Routing & SPA Concepts

By
Narasimha Rao T
Microsoft.Net FSD Trainer
Professional Development Trainer
tnrao.trainer@gmail.com

1. What is SPA (Single Page Application)?

- **Definition**:
  A Single Page Application (SPA) is a web app that loads a single HTML page and dynamically updates content as the user interacts with the app.
- **Key Points**:
  - Only **one HTML file** (`index.html`) is loaded.
  - Content updates via **JavaScript** (using frameworks like React).
  - Uses **AJAX/Fetch** for server communication.
  - Provides a **faster and smoother user experience** (no full page reload).

2. Navigation in SPA

- In a traditional multi-page app:
  - Clicking a link reloads the entire page from the server.
- In an SPA:
  - Navigation happens **client-side**.
  - The browser history is manipulated via the **History API**.
  - Components are mounted/unmounted instead of reloading the whole page.

3. What is Routing in React?

- **Definition**:
  Routing in React is the process of defining navigation paths and rendering components based on the current URL.
- **Purpose**:
  - Enables **different views/screens** in a single-page app.
  - Manages **navigation without page reloads**.
  - Helps in creating a **multi-page feel** within an SPA.

4. Why Do We Need Routing in React?

- To allow users to **navigate between pages** without refreshing.
- To create a **structured application** with different sections (Home, About, Dashboard, etc.).
- To support **deep linking** (sharing URLs for specific views).
- To manage **nested components** based on routes.

5. How to Implement Routing in React

1. Install React Router: `npm install react-router-dom`
2. Basic setup:

```
import { BrowserRouter, Routes, Route } from "react-router-dom";
import Home from "./Home";
import About from "./About";


const routing = (
     <BrowserRouter>
         <Routes>
             <Route path="/" element={<Home />} />
             <Route path="/about" element={<About />} />
         </Routes>
    </BrowserRouter>
  );
```

6. What is React Router?

- A **popular library** for routing in React apps.
- Provides components and hooks to:
  - Define routes (`<Route>`).
  - Enable navigation (`<Link>` , `<NavLink>`).
  - Handle redirects, nested routes, etc.
- Works with `react-router-dom` (for web apps).

7. Route

- **Route (`<Route>`):**
  - Maps a URL path to a React component.
  - Example:

```
<Route path="/about" element={<About />} />
```

How to Implement Routing in React

1. Install React Router: `npm install react-router-dom`
2. Basic setup:

```
import { BrowserRouter, Routes, Route } from "react-router-dom";
import Home from "./Home";
import About from "./About";


const routing = (
     <BrowserRouter>
         <Routes>
             <Route path="/" element={<Home />} />
             <Route path="/about" element={<About />} />
         </Routes>
    </BrowserRouter>
  );
```

8. Usage of `<Link>` Tag

- `<Link>` is used instead of `<a>` to avoid full-page reloads.
- Example:

```
import { Link } from "react-router-dom";

function Navbar() {
  return (
    <nav>
      <Link to="/">Home</Link>
      <Link to="/about">About</Link>
    </nav>
  );
}
```

9. Handling Nested Routes

- Useful for layouts with **sub-pages** (e.g., Dashboard → Profile/Settings).
- Example:

```
<Routes>
  <Route path="/dashboard" element={<Dashboard />}>
    <Route path="profile" element={<Profile />} />
    <Route path="settings" element={<Settings />} />
  </Route>
</Routes>
```

- Inside `Dashboard.js`, use `<Outlet />` to render nested components:

```
import { Outlet } from "react-router-dom";

function Dashboard() {
  return (
    <div>
      <h2>Dashboard</h2>
      <Outlet />
    </div>
  );
}
```

upGrad ENTERPRISE

Q & A

Narasimha Rao T
tnrao.trainer@gmail.com