React SPA

Router

~ by tunalee

React Router의 원리

- React는 SPA(Single Page Application)
- 페이지 간 이동 없이 URL 변화만으로 컴포넌트 교체
- React Router가 URL과 컴포넌트를 매핑하여 라우팅 담당

React Router 원리

- **React Router**: 리액트 애플리케이션에서 페이지 간의 내비게이션을 가능하게 해주는 라이브러리
- **원리**:
  - URL 변화 감지: 브라우저의 주소 표시줄이 변경될 때 URL 경로에 따라 컴포넌트를 렌더링
  - 시맨틱 마크업: `<Routes>`와 `<Route>`를 사용하여 페이지 구조를 정의

React Router Dom 설치

```
$ npm install react-router-dom react-router
```

예제

```
import { BrowserRouter, Route, Routes } from 'react-router-dom';

function Home() {
  return <h1>Home Page</h1>;
}

function About() {
  return <h1>About Page</h1>;
}

function NotFound() {
  return <h1>404 - Page Not Found</h1>;
}

function App() {
  return (
    <BrowserRouter>
      <Routes>
        <Route path="/" element={<Home />} />
        <Route path="/about" element={<About />} />
        <Route path="*" element={<NotFound />} />
      </Routes>
    </BrowserRouter>
  );
}
```

Link, NavLink 태그

- Link: 단순한 라우팅 링크
- NavLink: 활성화된 링크에 스타일을 추가 가능

GNB(Global Navigation Bar)

- 여러 페이지로 이동할 수 있는 바
- 여러 페이지에서 공통으로 보이는 바

예제

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

function Home() {
  return <h2>홈 페이지</h2>;
}

function Contents() {
  return <h2>컨텐츠 페이지</h2>;
}

function Login() {
  return <h2>로그인 페이지</h2>;
}

function Layout() {
  const navigate = useNavigate();
  return (
    <div>
      <header>
        <div className="gnb">
          <ul>
            <li>
              <a className="nav-link" onClick={() => navigate('/')}>
                홈
              </a>
            </li>
            <li>
              <NavLink
                to="/contents"
                className="nav-link"
                activeclassname="active"
              >
                컨텐츠
              </NavLink>
            </li>
            <li style={{ marginLeft: 'auto' }}>
              <NavLink to="/login" className="nav-link">
                로그인
              </NavLink>
            </li>
          </ul>
        </div>
      </header>
    </div>
  );
}

function App() {
  return (
    <Router>
      <Layout />
      <Routes>
          <Route path="/" element={<Home />} />
          <Route path="/contents" element={<Contents />} />
          <Route path="/Login" element={<Login />} />
        </Routes>
    </Router>
  );
}

export default App;
```

CSS

```css
body {
  font-family: Arial, sans-serif;
  margin: 0;
  padding: 0;
}

header {
  background-color: gray;
  padding: 10px;
}

.gnb ul {
  list-style: none;
  padding: 0;
  margin: 0;
  display: flex;
}

.gnb li {
  margin-right: 20px;
}

.nav-link {
  color: white;
  text-decoration: none;
  font-size: 18px;
}

.nav-link:hover {
  color: yellow;
}

.active {
  color: yellow;
}

main {
  padding: 20px;
}
```

useNavigate

- Link, NavLink와 다르게 함수형 컴포넌트에서 호출
- 프로그래밍적 라우팅 및 동적 처리 가능

```
function Layout() {
  const navigate = useNavigate();
  return (
    <div>
      <ul>
        <li>
          <a className="nav-link" onClick={() => navigate('/')}>
        </li>
      </ul>
    <div>
  )
}
```

React Router 심화

- Parameter 사용
- Nested Router

Parameter를 사용한 라우팅

- URL Parameter: 경로에 동적 데이터 전달
- useParams: 전달받은 파라미터를 받아 사용

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

function User() {
  let { id } = useParams();
  return <h1>User ID: {id}</h1>;
}

<Route path="/user/:id" component={User} />
```

Query Parameter

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

function User() {
  let { id } = useParams();
  const location = useLocation();

  const queryParams = new URLSearchParams(location.search);
  const name = queryParams.get('name');

  return (
    <div>
      <h1>User Page</h1>
      <p>User ID: {id}</p>
      <p>Name: {name}</p>
    </div>
  );}
```

```
navigate(`/user/${id}?name=John`);
```

Nested Route

- 레이아웃을 재사용: 상위 페이지가 유지된 채 하위 페이지가 바뀌는 구조, 레이아웃을 공유하면서 특정 부분만 다른 페이지를 렌더링
- 경로의 계층적 구조 표현: URL 경로와 관련된 계층 구조

```
function Users() {
  return (
    <div>
      <h2>Users List</h2>
      <ul>
        <li>
          <NavLink to="1">User 1</NavLink>
        </li>
        <li>
          <NavLink to="2">User 2</NavLink>
        </li>
        <li>
          <NavLink to="3">User 3</NavLink>
        </li>
      </ul>
    </div>
  );
}

function UserDetail() {
  const { id } = useParams();
  return <h3>User Detail for User {id}</h3>;
}
```

```
function Layout() {
  return (
    <div>
      <header>
        <ul>
          <li>
            <NavLink to="/">Home </NavLink>
          </li>
          <li>
            <NavLink to="/users">Users </NavLink>
          </li>
        </ul>
      </header>
      <main></main>
    </div>
  );
}
```

```
function App() {
  return (
    <Router>
      <Layout />

      <Routes>
        <Route path="users" element={<Users />}>
          <Route path=":id" element={<UserDetail />} />
        </Route>
      </Routes>
    </Router>
  );
}
```

실습

홈 컨텐츠 로그인

홈 페이지

홈 컨텐츠 로그인

컨텐츠 1
컨텐츠 2
컨텐츠 3

홈 컨텐츠 로그인

컨텐츠 1
컨텐츠 2
컨텐츠 3

id: 1, content: 안녕하세요

홈 컨텐츠 로그인

컨텐츠 1
컨텐츠 2
컨텐츠 3

id: 2, content: 저는 코린이입니다.

홈 컨텐츠 로그인

컨텐츠 1
컨텐츠 2
컨텐츠 3

d: 3, content: 프론트엔드 고수가 되고 싶습니다.

로그인 페이지