wawiwa
React
2022 © Wawiwa Tech | Confidential


wawiwa
15 - React Router
visual design
2022 © Wawiwa Tech | Confidential

wawiwa

Introducción

React Router es una **biblioteca popular para controlar routing (enrutamiento)** en aplicaciones de React (React no incluye enrutamiento de páginas).

Proporciona una forma de **asignar URL de forma declarativa a componentes**, lo que permite una navegación fluida y mantiene la interfaz de usuario sincronizada con la URL.

Con React Router, puedes fácilmente definir rutas anidadas, manejar su historia y ejecutar redirecciones, pasar props a componentes y más.

React Router es bueno para construir apps de una página, muchas páginas o un híbrido de las dos.

wawiwa

Introducción

React Router renderiza condicionalmente ciertos componentes dependiendo en la ruta usada en el URL.

/ para la página de inicio, /about para página “about”, etc.

Por ejemplo:

Podemos usar React Router para conectar: www.my-website.com a www.my-website.com/about

wawiwa

Instalar React Router

Para usar React Router, debemos instalarlo usando NPM:

npm install react-router-dom

```
PS C:\Users\danie\Documents\my-app> npm install react-router-dom

added 3 packages, and audited 1951 packages in 13s

102 packages are looking for funding
  run `npm fund` for details

75 vulnerabilities (13 low, 16 moderate, 42 high, 4 critical)

To address issues that do not require attention, run:
  npm audit fix

To address all issues (including breaking changes), run:
  npm audit fix --force

Run `npm audit` for details.
PS C:\Users\danie\Documents\my-app>
```

wawiwa

<BrowserRouter>

Primero, debemos configurar nuestra app para trabajar con React Router.

Envuelve tu App dentro del elemento <BrowserRouter>. Todo lo que se renderiza necesita ser envuelto.

BrowserRouter es el componente que hace toda la lógica de muestra los componentes que van a ser proporcionados.

En index.js:

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

const root = ReactDOM.createRoot(document.getElementById('root'));
root.render(
    <BrowserRouter>
        <App />
    </BrowserRouter>
);
```

wawiwa

Routes (Rutas)

Routes es usado para definir las rutas.

Las rutas se definen usando rutas. Esto se logra en la **parte superior de tu aplicación,** por ejemplo, en el componente App (puede ser donde sea que tu prefieras). Este componente va a contener todas las rutas.

App.js:

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

function App() {
  return (
    <Routes>
    </Routes>
  );
}
export default App;
```

wawiwa

Route (Ruta)

Route - nos permite **mapear la ubicación de los diferentes componentes de tu aplicación de React**.

Para crear una nueva ruta:

```
<Route path="/your_path" element={<YourComponent />}/>
```

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

cada vez un usuario navega a la ruta /about:

```
export default function App() {
  return (
    <Routes>
      <Route path="/about" element={<About />} />
    </Routes>
  );
}
```

wawiwa

¿Cómo funcionan las Rutas?

Cuando tu URL cambia, React Router va a buscar las rutas definidas en tu componente Routes y va renderizar el contenido en el prop de la Route que tiene una ruta que coincide con el URL.

En el ejemplo siguiente, si nuestro URL era /about, el componente URL será renderizado.

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

Cuando navegas entre páginas, solamente se va a actualizar el contenido dentro del componente Routes. El resto del contenido es tu página va a mantenerse igual, eso ayuda al rendimiento y la experiencia del usuario.

wawiwa

Ejemplo de Routes

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

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

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

export default function App() {
  return (
    <Routes>
      <Route path="/about" element={<AboutPage />} />
      <Route path="/" element={<HomePage />} />
    </Routes>
  );
}
```

wawiwa

Manejar Navigación <Link>

Para navegar en la aplicación, usando navbar con enlaces presionables, usa el elemento **Link** del React Router.

Para cada componente Link en el app, usa to=”url” para enlazarlos.

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

wawiwa

Ejemplo completo de React Router

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

function AboutPage() {
  return <h1>About Page</h1>;
}
function HomePage() {
  return <h1>Home Page</h1>;
}
function Navbar() {
  return (
    <>
    <Link to="/">Home</Link> <br />
    <Link to="/about">About Us</Link>
    </>
  )
}
```

```
export default function App() {
  return (
    <>
    <Navbar />
    <Routes>
      <Route path="/about" element={<AboutPage />} />
      <Route path="/" element={<HomePage />} />
    </Routes>
    </>
  );
}
```

wawiwa

Ruta no encontrada

Un * va a hacer coincidir cualquier cosa, lo que lo hace perfecto para cosas como una página 404.

Una ruta que contiene * va a ser menos específica que cualquier otra cosa entonces nunca vas a coincidir accidentalmente una ruta * cuando otra ruta debe ser coincidida.

```
<Route path="*" element={<NotFound />} />
```

Esta línea va a ser el último Route:

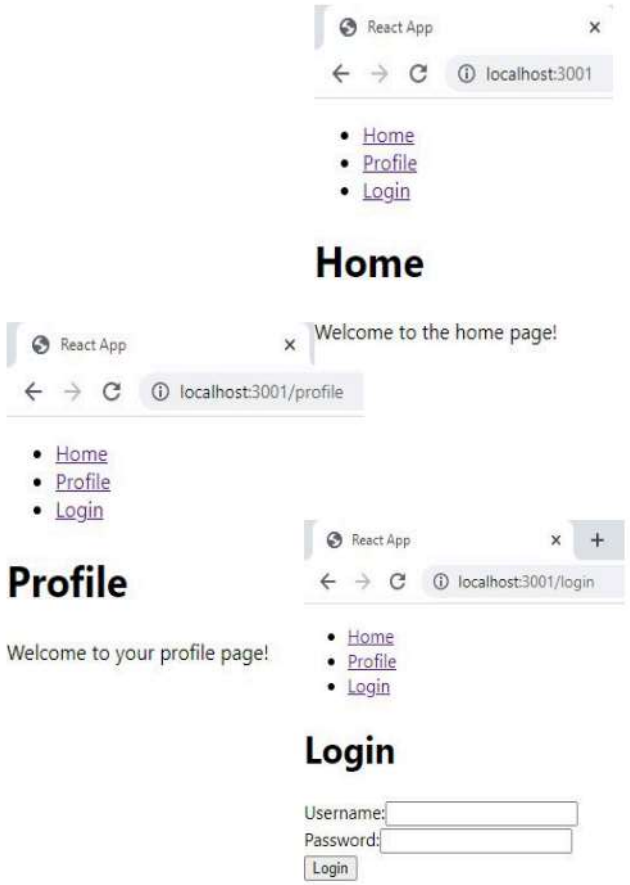
```
<Route exact path="/" component={Home} />
<Route exact path="/about" component={About} />
<Route exact path="/contact" component={Contact} />
<Route path="*" component={NotFound} />
```

wawiwa

Ejercicio 16

En este ejercicio, vas a construir un sitio web usando React Router.

1. Crea tres componentes: Home, Profile, y Login. Cada componente debe renderizar algo de contento, como un título y un párrafo.
2. Crea un componente Navbar que muestre enlaces para los componentes Home, Login y Profile.
3. Usa React Router para crear rutas para cada uno de los componentes. El componente Home debe ser la ruta por defecto.
4. El componente Login va a contener inputs para username y password.



wawiwa

useNavigate()

useNavigate() es un hook proporcionado por React Router v6 que da acceso a los objeto “navegation”. El objeto “navegation” es usado para **navegar a páginas diferentes dentro de la aplicación**.

En contraste con usar **useHistory() en React Router v5**, que da acceso al objeto “history” del navegador, useNavigate() en React Router v6 da acceso al objeto “navigation” que está diseñado específicamente para navegar dentro de la aplicación. El objeto “navegación” proporciona un conjunto de propiedades y métodos como el objeto “history” pero con comportamientos y usos diferentes.

wawiwa

Métodos de useNavigate()

El hook useNavigate() retorna un objeto que contiene muchas propiedades y métodos, incluyendo:

- **navigate(to, { replace, state }) -** Navega a una ubicación específica. Puedes pasar un objeto como segundo argumento para especificar si quieres reemplazar la entrada actual o una nueva entrada de la historia de la navegación a la nueva ubicación.
- **goBack() -** Mueve la historia de la navegación una entrada atrás.
- **goForward() -** Mueve la historia de la navegación una entrada adelante.
- **go(n) -** Mueve la historia de la navegación “n” entradas adelante o atras, dependiendo del signo de “n”.

wawiwa

Sintaxis de useNavigate()

Para usar el hook en tus componentes:

1. Importa:

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

1. Llama el hook usando una variable (para un uso más conveniente):

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

1. Llama a la variable con un método, por ejemplo:

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

wawiwa

Ejercicio 17

Crea una aplicación simple con dos páginas: una página “Home” y una página “About”. La página “Home” debe mostrar un botón que navegue a la página “About” cuando se le haga click. La página “About” debe mostrar un botón que navegue atrás a la página “Home” cuando se le haga click.