








Le router

Thèmes	 React
Description	Créer un routage dynamique dans une application React.
Date de création	@16 mars 2025 19:18
Créée par	 Kévin Wolff

Chapitres

-  [Introduction](#)
-  [Mise en place](#)
-  [Navigation](#)
-  [Accéder aux paramètres](#)
-  [Accéder aux informations d'URL](#)
-  [Les états de navigation](#)
-  [Accéder aux queries](#)

Sur le même sujet

-  [Premier composant React](#)
-  [Approche atomic design](#)
-  [Cycle de vie d'un composant](#)

Introduction

React Router est la bibliothèque de référence pour gérer la navigation dans une application React. Elle permet de définir des routes et de naviguer entre différentes pages sans recharger la page, ce qui est essentiel pour les Single Page Applications (SPA).

Mise en place

Installez React Router.

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

Dans un projet React, le routing est souvent défini dans un fichier comme `App.js`.

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

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

```

    <Route path="*" element={<NotFound />} />
  </Routes>
</Router>
);
}

```

`BrowserRouter` est le conteneur principal qui active le routage dans l'application.

`Routes` regroupe toutes les routes définies.

`Route` définit une route et le composant associé.

Navigation

Pour naviguer entre vos pages, utilisez le composant `<Link>` de React Router.

```

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

export default function Navbar() {
  return (
    <nav>
      <Link to="/">Accueil</Link>
      <Link to="/about">À propos</Link>
    </nav>
  );
};

```

Accéder aux paramètres

Le hook `useParams()` permet d'extraire les valeurs des paramètres dynamiques de l'URL. Par exemple, pour l'URL `/user/42`, `useParams()` retournera `{ id: "42" }`.

Déclarez un paramètre dans la définition de l'URL, ici l'ID de l'utilisateur pour afficher sa page de profil.

```

import { BrowserRouter as Router, Routes, Route } from "react-router-dom";
import Home from "./pages/Home";
import UserProfile from "./pages/UserProfile";

export default function App() {
  return (
    <Router>
      <Routes>
        <Route path="/" element={<Home />} />
        <Route path="/user/:id" element={<UserProfile />} />
      </Routes>
    </Router>
  );
}

```

Accédez au paramètre `id` de l'URL grâce au hook `useParams` dans le composant `UserProfile` représentant la page de profil de l'utilisateur.

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

export default function UserProfile() {
  const { id } = useParams();

  return (
    <div>
      <h1>Profil de l'utilisateur {id}</h1>
    </div>
  );
};
```

URL	Hook	Résultat
<code>/user/17</code>	<code>const { id } = useParams();</code>	17

Accéder aux informations d'URL

Le hook `useLocation` permet de récupérer le chemin actuel et l'état de la navigation.

Le hook `useLocation()` permet d'accéder aux informations sur l'URL en cours, comme le chemin (`pathname`), les paramètres de requête (`search`), et l'état passé lors de la navigation (`state`).

```
import { BrowserRouter as Router, Routes, Route } from "react-router-dom";
import Home from "../pages/Home";
import PageInfo from "../pages/PageInfo";

export default function App() {
  return (
    <Router>
      <Routes>
        <Route path="/" element={<Home />} />
        <Route path="/info" element={<PageInfo />} />
      </Routes>
    </Router>
  );
}
```

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

export default function PageInfo() {
  const location = useLocation();

  return (
    <div>
      <h1>Hello world</h1>
    </div>
  );
}
```

```

    </div>
  );
};

```

URL	Hook	Résultat
/info?name=John	const { pathname } = useLocation();	/info
/info?name=John	const { search } = useLocation();	?name=John
/info?name=John	const { state } = useLocation();	null

Les états de navigation

Lorsque vous naviguez à l'aide de la fonction `navigate` de React Router, vous avez la possibilité d'ajouter un second paramètre sous forme d'objet avec une clé `state` déclarant elle même un objet où vous êtes libre de déclarer des paires de clés/valeurs que vous pourrez exploiter sur la page de destination grâce au hook `useLocation`.

```

import { useNavigate } from "react-router-dom";

export default function GoToInfo() {
  const navigate = useNavigate();

  return (
    <button onClick={() => navigate("/info", { state: { from: "Accueil" } })}>
      Aller à Info avec état
    </button>
  );
};

```

```

import { useLocation } from "react-router-dom";

export default function PageInfo() {
  const location = useLocation();

  console.log(location.state); // Affichera l'objet : { from: "accueil" }

  return (
    <div>
      <h1>Hello world</h1>
    </div>
  );
};

```

Cette fonctionnalité est utile pour véhiculer des informations d'une page à une autre sans passer par le paramètre GET de l'URL.

Accéder aux queries

Le hook `useSearchParams` permet de récupérer les paramètres de l'URL.

Déclarez un paramètre dans la définition de l'URL, ici l'ID de l'utilisateur pour afficher sa page de profil.

```
import { BrowserRouter as Router, Routes, Route } from "react-router-dom";
import SearchPage from "./pages/SearchPage";

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

Accédez au paramètre de l'URL grâce au hook `searchParams` dans le composant `SearchPage` représentant la page de résultats d'une recherche.

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

export default function SearchPage() {
  const [searchParams] = useSearchParams();
  const name = searchParams.get("name");

  return (
    <div>
      <h1>Recherche</h1>
      <p>Nom recherché : {name || "Aucun"}</p>
    </div>
  );
}
```

URL	Hook	Résultat
<code>/search?name=John</code>	<code>const name = searchParams.get("name");</code>	John
<code>/search?country=France&name=John</code>	<code>const country = searchParams.get("country");</code>	France