# Atelier 02.06 Routage

# Routage Moderne avec React Router DOM v7

### Objectif général

Apprendre à créer une application React multi-pages avec :

- Un layout avec menu
- Des routes imbriquées
- Une route par défaut (index)
- Page 404
- Navigation interne (Link)
- Paramètres d'URL
- Routes dynamiques
- Pages animées avec texte + image
- Feuille de style globale simple

### Avant de commencer : Identifier la version actuelle

```
npm list react-router-dom
```

### Installation

```
npm install react-router-dom@latest
```

### Étape 1 — Mise en place du Routage + Page d'Accueil seulement

### 1. Créer le Layout (structure commune)

```jsx
//src/pages/Layout.jsx
import { Link, Outlet } from "react-router-dom";

export default function Layout() {
  return (
    <div>
      <header style={{ background: "#222", padding: "15px" }}>
        <nav>
          <Link to="/" style={{ color: "white", marginRight: "10px" }}>
            Accueil
          </Link>
          {/* On ajoutera d'autres liens plus tard */}
        </nav>
      </header>

      <main style={{ padding: "20px" }}>
        {/* Zone où les pages vont s'afficher */}
        <Outlet />
      </main>
    </div>
  );
}
```

### 2. Créer la page d'accueil

```
//src/pages/Home.jsx
export default function Home() {
  return (
    <div>
      <h1>Bienvenue sur notre Application React Router</h1>
      <p>Ceci est la page d'accueil. Nous allons ajouter des pages progressivement.</p>

      <img
        src="https://picsum.photos/700/250?random=10"
        alt="Accueil"
        style={{ borderRadius: "6px", marginTop: "20px" }}
      />
    </div>
  );
}
```

### 3. Configurer le Router (React Router DOM v7)

```
//src/main.jsx
import ReactDOM from "react-dom/client";
import { RouterProvider, createBrowserRouter } from "react-router-dom";

import Layout from "./pages/Layout";
import Home from "./pages/Home";

const router = createBrowserRouter([
  {
    path: "/",           // route principale
    element: <Layout />, // page Layout
    children: [
      { index: true, element: <Home /> },  // route par défaut = Page d'accueil
    ],
  },
]);

ReactDOM.createRoot(document.getElementById("root")).render(
  <RouterProvider router={router} />
);
```

### 4. Aperçu attendu

Lorsque vous lancez :

npm run build

Vous devez voir :

- Un menu (contenant uniquement *Accueil* pour le moment)
- Un titre "Bienvenue sur notre Application React Router"
- Une image
- Aucune autre navigation active pour l'instant

### Exercice 1 — Ajouter la page Blogs

**Objectif**

Introduire une **nouvelle page** et **l'ajouter au menu + au router**.

**1) Créer la page Blogs**

**Fichier : src/pages/Blogs.jsx**

```
export default function Blogs() {
  return (
    <div>
      <h2>Page Blogs</h2>
      <p>Liste d'articles et contenus informatifs seront affichés ici.</p>

      <img
        src="https://picsum.photos/700/250?random=20"
        alt="Blogs"
        style={{ borderRadius: "6px", marginTop: "20px" }}
      />
    </div>
  );
}
```

**2) Ajouter le lien dans le menu du Layout**

**Modifier src/pages/Layout.jsx**

Ajouter la ligne Link suivante :

```
<Link to="/blogs" style={{ color: "white", marginRight: "10px" }}>
  Blogs
</Link>
```

**3) Ajouter la route dans le router**

**Modifier src/main.jsx** → Ajouter :

```
import Blogs from "./pages/Blogs";
```

Puis dans le tableau children :

```
{ path: "blogs", element: <Blogs /> },
```

**Résultat attendu**

- Un nouveau lien **Blogs** apparaît dans le menu
- En cliquant dessus → une nouvelle page s'affiche

**Exercice 2 — Ajouter la page Contact**

**Objectif**

Comprendre l'ajout progressif de pages et routes.

**1) Créer la page Contact**

```
//src/pages/Contact.jsx
export default function Contact() {
  return (
    <div>
      <h2>Contact</h2>
      <p>Pour toute demande : contact@example.com</p>
```

```
      <img
        src="https://picsum.photos/700/250?random=30"
        alt="Contact"
        style={{ borderRadius: "6px", marginTop: "20px" }}
      />
    </div>
  );
}
```

### 2) Ajouter au menu (Layout.jsx)

```
<Link to="/contact" style={{ color: "white", marginRight: "10px" }}>
  Contact
</Link>
```

### 3) Ajouter la route (main.jsx)

```
import Contact from "./pages/Contact";
```

Ajouter dans children :

```
{ path: "contact", element: <Contact /> },
```

### Résultat attendu

- Trois pages : Accueil / Blogs / Contact
- Navigation fluide sans rechargement

### Exercice 3 — Ajouter une Route Dynamique (Détail d'un Blog)

### Objectif

- Comprendre l'utilisation des **paramètres dynamiques** dans les routes
- Savoir utiliser useParams() pour lire l'ID depuis l'URL

### 1) Créer la page BlogDetail

### Fichier : src/pages/BlogDetail.jsx

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

export default function BlogDetail() {
  const { id } = useParams(); // récupère le paramètre de l'URL

  return (
    <div>
      <h2>Détail de l'article #{id}</h2>
      <p>Voici le contenu détaillé de l'article sélectionné.</p>

      <img
        src="https://picsum.photos/700/250?random=40"
        alt="Détail Blog"
        style={{ borderRadius: "6px", marginTop: "20px" }}
      />

      <p style={{ marginTop: "15px" }}>
        <Link to="/blogs">Retour à la liste des blogs</Link>
      </p>
```

```
    </div>
  );
}
```

### 2) Modifier la page Blogs pour afficher une liste cliquable

**Modifier src/pages/Blogs.jsx**

Ajouter une liste d'articles :

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

export default function Blogs() {
  const articles = [
    { id: 1, title: "Comprendre les composants React" },
    { id: 2, title: "Introduction aux Hooks" },
    { id: 3, title: "Gestion du State dans une application" },
  ];

  return (
    <div>
      <h2>Page Blogs</h2>
      <p>Choisissez un article pour afficher ses détails :</p>

      <ul>
        {articles.map((article) => (
          <li key={article.id}>
            <Link to={`/blogs/${article.id}`}>{article.title}</Link>
          </li>
        ))}
      </ul>

      <img
        src="https://picsum.photos/700/250?random=20"
        alt="Blogs"
        style={{ borderRadius: "6px", marginTop: "20px" }}
      />
    </div>
  );
}
```

### 3) Ajouter la route dynamique dans main.jsx

**Modifier src/main.jsx**

```
import BlogDetail from "./pages/BlogDetail";
```

Puis dans children :

```
{ path: "blogs/:id", element: <BlogDetail /> },
```

**Résultat attendu**

- La page **Blogs** affiche une liste de liens
- En cliquant sur un article → on se rend sur /blogs/1 ou /blogs/2 etc.
- La page affiche le détail et propose retour

### Exercice 4 — Ajouter une Page 404 (Route Catch-All)

**Objectif**

- Gérer les URLs qui n'existent pas

- Améliorer l'expérience utilisateur

### 1) Créer la page NoPage

```
//src/pages/NoPage.jsx
import { Link } from "react-router-dom";

export default function NoPage() {
  return (
    <div>
      <h2>Erreur 404</h2>
      <p>La page demandée n'existe pas.</p>

      <img
        src="https://picsum.photos/700/250?random=50"
        alt="404"
        style={{ borderRadius: "6px", marginTop: "20px" }}
      />

      <p style={{ marginTop: "15px" }}>
        <Link to="/">Retour à l'accueil</Link>
      </p>
    </div>
  );
}
```

### 2) Ajouter la route 404 dans main.jsx

### Modifier src/main.jsx

```
import NoPage from "./pages/NoPage";
```

Ajouter dans children :

```
{ path: "*", element: <NoPage /> },
```

### Résultat attendu

- Si l'utilisateur tape une URL inexistante (ex : /xyz123)
- La page **Erreur 404** s'affiche au lieu d'une page blanche