useContext

useContext

Permet de lire une valeur de contexte sans passer par le prop drilling. Le contexte est un état partagé accessible par tous les composants descendants.

Création d'un contexte

```
import { createContext, useContext, useState } from 'react';

// 1. Créer le contexte
const ThemeContext = createContext('light'); // valeur par défaut

// 2. Fournir le contexte
function App() {
  const [theme, setTheme] = useState('light');

  return (
    <ThemeContext.Provider value={{ theme, setTheme }}>
      <Navbar />
      <Main />
    </ThemeContext.Provider>
  );
}

// 3. Consommer le contexte
function Navbar() {
  const { theme, setTheme } = useContext(ThemeContext);

  return (
    <nav className={theme}>
      <button onClick={() => setTheme(t => t === 'light' ? 'dark' : 'light')}>
        Basculer le thème
      </button>
```

```
    </nav>
  );
}
```

Pattern : custom hook pour le contexte

```
// Bonne pratique : encapsuler dans un hook custom
function useTheme() {
  const ctx = useContext(ThemeContext);
  if (!ctx) throw new Error("useTheme doit être utilisé dans ThemeProvider");
  return ctx;
}

// Utilisation
const { theme } = useTheme();
```

Quand utiliser le contexte ?

- Thème (dark/light)
- Langue / i18n
- Utilisateur connecté
- Toute donnée vraiment globale

⚠ Le contexte re-rend **tous les consommateurs** quand la valeur change. Pour des données très fréquemment mises à jour, préférer un gestionnaire d'état dédié (Zustand, Redux).

Revision #1
Created 29 March 2026 12:57:35 by Hugo
Updated 29 March 2026 12:57:36 by Hugo