



O que é o useContext?

É um hook que permite **compartilhar dados** entre componentes **sem precisar ficar passando props** manualmente por várias camadas.

Serve para:

- tema (dark/light)
 - usuário logado
 - carrinho de compras
 - configurações
 - idioma
 - etc.
-



Como usar o useContext (passo a passo)

1 Criar o Context

Você cria um contexto que vai guardar o valor a ser compartilhado.

```
import { createContext } from "react";  
  
export const TemaContext = createContext();
```

2 Criar um Provider

O Provider é quem “entrega” os dados para os componentes filhos.

```
// TemaProvider.jsx
import { useState, createContext } from "react";

export const TemaContext = createContext();

export function TemaProvider({ children }) {
  const [tema, setTema] = useState("light");

  const alternarTema = () => {
    setTema(tema === "light" ? "dark" : "light");
  };

  return (
    <TemaContext.Provider value={{ tema, alternarTema }}>
      {children}
    </TemaContext.Provider>
  );
}
```

3 Envolver a aplicação com o Provider

Normalmente no main.jsx ou App.jsx.

```
import React from "react";
import ReactDOM from "react-dom/client";
import App from "./App";
import { TemaProvider } from "./TemaProvider";

ReactDOM.createRoot(document.getElementById("root")).render(
  <TemaProvider>
    <App />
  </TemaProvider>
);
```

4 Consumir o contexto com useContext

Agora qualquer componente pode acessar o tema **sem receber props**:

```
import { useContext } from "react";
import { TemaContext } from "../TemaProvider";

function BotaoTema() {
  const { tema, alternarTema } = useContext(TemaContext);

  return (
    <>
      <p>Tema atual: {tema}</p>
      <button onClick={alternarTema}>Alternar Tema</button>
    </>
  );
}

export default BotaoTema;
```

Criar um **UserContext** para disponibilizar o **usuário logado** em toda a aplicação — sem precisar ficar passando props.



Estrutura recomendada

```
src/
├── context/
│   └── UserContext.jsx
├── App.jsx
├── main.jsx
└── components/
    ├── Header.jsx
    └── Perfil.jsx
```



1 Criar o *UserContext* + Provider

```
src/context/UserContext.jsx
```

```
import { createContext, useState } from "react";

export const UserContext = createContext();

export function UserProvider({ children }) {
  // Aqui você guarda o usuário logado
  const [usuario, setUsuario] = useState(null);

  // Função para simular login
  const login = (nome, email) => {
    setUsuario({ nome, email });
  };

  // Função para logout
  const logout = () => {
    setUsuario(null);
  };

  return (
    <UserContext.Provider value={{ usuario, login, logout }}>
      {children}
    </UserContext.Provider>
  );
}
```

2 Envolver toda a aplicação com o Provider

main.jsx

```
import React from "react";
import ReactDOM from "react-dom/client";
import App from "../App";
import { UserProvider } from "../context/UserContext";
```

```
ReactDOM.createRoot(document.getElementById("root")).render(  
  <UserProvider>  
    <App />  
  </UserProvider>  
);
```

Agora **qualquer componente** dentro de <UserProvider> pode usar o usuário logado.

3 Consumir o usuário com useContext

Exemplo 1 — Header mostrando usuário logado

components/Header.jsx

```
import { useContext } from "react";  
import { UserContext } from "../context/UserContext";  
  
function Header() {  
  const { usuario, logout } = useContext(UserContext);  
  
  return (  
    <header>  
      <h2>Sistema</h2>  
  
      {usuario ? (  
        <>  
          <span>Bem-vindo, {usuario.nome}</span>  
          <button onClick={logout}>Sair</button>  
        </>  
      ) : (  
        <span>Nenhum usuário logado</span>  
      )}  
    </header>  
  );  
}
```

```
}
```

```
export default Header;
```

Exemplo 2 — Componente com botão para login

App.jsx

```
import { useContext } from "react";
import { UserContext } from "../context/UserContext";
import Header from "../components/Header";

function App() {
  const { login } = useContext(UserContext);

  return (
    <>
      <Header />

      <button onClick={() => login("Vilson Moro",
"vilson@email.com")}>
        Fazer Login
      </button>
    </>
  );
}

export default App;
```

Exemplo 3 — Outro componente usando o mesmo usuário

components/Perfil.jsx

```
import { useContext } from "react";
import { UserContext } from "../../context/UserContext";
```

```
function Perfil() {  
  const { usuario } = useContext(UserContext);  
  
  if (!usuario) return <p>Você não está logado</p>;  
  
  return (  
    <div>  
      <h3>Perfil do Usuário</h3>  
      <p>Nome: {usuario.nome}</p>  
      <p>Email: {usuario.email}</p>  
    </div>  
  );  
}  
  
export default Perfil;
```