React Context

React Context es como un mensajero que lleva información importante a todos los rincones de tu aplicación.

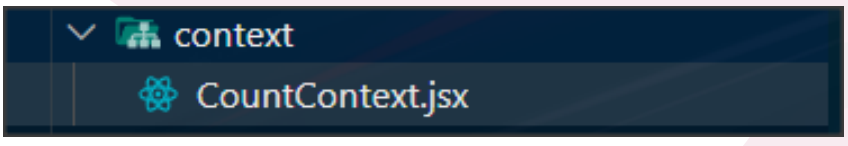
Imagina que estás organizando una fiesta y necesitas decirle a todos los invitados dónde se encuentra la comida. En lugar de ir uno por uno, puedes enviar un mensajero con esa información a todos los invitados al mismo tiempo. Eso es lo que hace React Context: permite compartir datos entre componentes sin tener que pasar la información manualmente a través de cada uno de ellos.

Ahora, en términos más técnicos, React Context es una característica que te permite compartir datos entre componentes sin tener que pasar explícitamente props a través de cada nivel de la jerarquía de componentes. Funciona creando un "proveedor" que contiene los datos que quieres compartir y luego haciendo que los componentes "consumidores" accedan a esos datos. Esto simplifica mucho el manejo de datos en aplicaciones grandes y complejas, evitando la llamada "prop drilling", que es cuando tienes que pasar props a través de varios niveles de componentes para que lleguen a donde los necesitas.

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

export const CountContext = createContext(); // Creamos el contexto y lo guardamos en la variable

const CountContextProvider = ({ children }) => {
  const [count, setCount] = useState(0);

  useEffect(() => {…
  }, []);

  function sumar() {…
  }

  function restar() {…
  }

  function reset() {…
  }

  const data = {
    count,
    sumar,
    restar,
    reset,
  };

  return <CountContext.Provider value={data}>{children}</CountContext.Provider>;
};

export default CountContextProvider;
```

Algunos casos de uso comunes para React Context son:

- **Temas y Estilos: Puedes usar React Context para proporcionar temas o estilos globales a tu aplicación para que todos los componentes tengan acceso a ellos.**
- **Autenticación y Autorización: Puedes usar Context para manejar el estado de autenticación de un usuario y proporcionar esa información a lo largo de la aplicación.**
- **Internacionalización: Si estás creando una aplicación multilingüe, puedes usar Context para proporcionar el idioma seleccionado a todos los componentes.**
- **Gestión del Estado Global: En lugar de depender de bibliotecas de gestión de estado externas como Redux, puedes usar Context para gestionar el estado global de tu aplicación.**
- **Favoritos: Si estás construyendo una aplicación donde los usuarios pueden marcar elementos como favoritos (por ejemplo, publicaciones, productos, imágenes, etc.), puedes usar React Context para administrar el estado de los favoritos y proporcionar esa información a través de la aplicación para que los componentes puedan mostrar y manejar los favoritos de manera consistente.**
- **Carrito de Compras: En una tienda en línea, el carrito de compras es un elemento fundamental. Con React Context, puedes almacenar la información del carrito de compras y proporcionarla a lo largo de la aplicación para que los usuarios puedan ver y actualizar su carrito desde cualquier página sin tener que pasar el estado del carrito a través de múltiples componentes manualmente.**



En resumen, React Context es una herramienta poderosa que te permite compartir datos de manera eficiente entre componentes en una aplicación de React, lo que facilita el desarrollo y mantenimiento de aplicaciones más grandes y complejas.