**¿Qué son los props?**

En React, los props (abreviatura de "properties" o propiedades) son una forma de pasar datos desde un componente padre a un componente hijo. Son como atributos en HTML, pero para componentes en React. Los props hacen que los componentes sean reutilizables y flexibles, permitiéndote crear componentes genéricos que se adaptan a diferentes datos.

**¿Cómo funcionan los props?**

1. **Declaración:** En el componente padre, se declaran los props que se quieren pasar al componente hijo dentro de las etiquetas de apertura del componente hijo.
2. **Acceso:** En el componente hijo, se accede a los props a través de this.props.
3. **Uso:** Los props se utilizan dentro del componente hijo para renderizar contenido o controlar el comportamiento.

**Ejemplo:**

JavaScript

// Componente Padrefunction Greeting(props) { return <Welcome name={props.name} />; } // Componente Hijofunction Welcome(props) { return <h1>Hola, {props.name}!</h1>; }

Use code [with caution.](file:///C:/faq#coding)

En este ejemplo, el componente Greeting pasa el nombre como prop al componente Welcome. El componente Welcome recibe el nombre a través de props.name y lo utiliza para personalizar el saludo.

**Características importantes de los props:**

* **Inmutables:** Los props no se pueden modificar dentro del componente hijo. Si necesitas actualizar un valor, debes hacerlo desde el componente padre.
* **Unidireccionales:** La información fluye desde el padre hacia el hijo. No se puede modificar un prop desde un componente hijo y esperar que ese cambio se refleje en el padre.
* **Opcionales:** Los props son opcionales. Puedes definir valores por defecto si un prop no se proporciona.
* **Tipos de datos:** Los props pueden ser de cualquier tipo de dato: strings, números, booleanos, objetos, arrays, funciones, etc.

**¿Cuándo usar props?**

* **Pasar datos estáticos:** Por ejemplo, un nombre de usuario, un título, una imagen, etc.
* **Configurar componentes:** Por ejemplo, pasar un color, un tamaño o un estilo a un componente.
* **Crear componentes reutilizables:** Los props permiten crear componentes genéricos que se pueden adaptar a diferentes escenarios.

**Ejemplo de props con tipos de datos diferentes:**

JavaScript

function Product(props) { return ( <div> <h2>{props.name}</h2> <p>Precio: ${props.price}</p> <button onClick={props.onAddToCart}>Añadir al carrito</button> </div> ); }

Use code [with caution.](file:///C:/faq#coding)

En este ejemplo, name y price son strings, y onAddToCart es una función que se pasa como prop para manejar el evento de clic.

**Resumen:**

Los props son una herramienta fundamental en React para crear componentes reutilizables y flexibles. Al entender cómo funcionan los props, podrás construir aplicaciones React más organizadas y mantenibles.