JSX y ReactDOM

Proyecto de uso básico de JSX y ReactDOM en el que trabajamos con la sintaxis básica de JSX, condicionales y arrays.

Documentación Básica sobre React, JSX y ReactDOM

¿Qué es React?

React es una biblioteca de JavaScript desarrollada por Facebook que se utiliza para construir interfaces de usuario de forma declarativa, eficiente y flexible. React se basa en componentes, pequeñas piezas de código que pueden combinarse para crear aplicaciones web completas.

¿Qué es JSX?

JSX es una extensión de la sintaxis de JavaScript que permite escribir estructuras similares a HTML directamente en el código JavaScript. Aunque parece HTML, JSX es transformado en código JavaScript por herramientas como Babel. Este enfoque facilita la creación de interfaces al combinar lógica y diseño.

Ejemplo de JSX:

```
const elemento = <h1>¡Hola, mundo!</h1>;
```

¿Qué es ReactDOM?

ReactDOM es el paquete que se encarga de renderizar elementos React en el DOM (Document Object Model) del navegador. Por ejemplo, en una aplicación React, usamos `ReactDOM.createRoot()` para especificar dónde en el DOM queremos mostrar nuestros componentes React.

Introducción a JSX, Condicionales y Arreglos en React

1. Uso de JSX

En React, JSX es el núcleo de cómo estructuramos componentes y vistas. Algunas características importantes:

- **Elementos JSX**: Los elementos JSX deben estar contenidos dentro de un único contenedor (por ejemplo, un `<div>`).
- **Propiedades en JSX**:
  - Para asignar una clase CSS, se usa `className` en lugar de `class`.
  - Los estilos se pasan como objetos en JavaScript usando `{}`.

Ejemplo del código:

```
const JSX = (
  <div>
    <h3>Aprendiendo JSX</h3>
    <p>Si queremos varias líneas</p>
    <p>Las tenemos que encerrar en un contenedor</p>
  </div>
);
```

2. Uso de Condicionales

React permite manejar condiciones en JSX de manera sencilla, utilizando operadores como el ternario `condición ? verdadero : falso` o evaluaciones cortas con `&&`. Esto es útil para renderizar elementos basados en el estado o variables.

Ejemplo del código:

```
{
  sesion2 === true ?
    <p>Variable verdadera</p>
  :
    <>
      <p>Variable falsa</p>
      <p>Más de una etiqueta</p>
      <p>Soy {nombre}</p>
      {nombre && <p>Soy {nombre}</p>}
    </>
}
```

3. Manejo de Arreglos

En React, es común trabajar con listas o arreglos y renderizarlos dinámicamente en el DOM usando el método `map`. Cada elemento renderizado necesita una clave única (`key`) para que React pueda manejarlo eficientemente.

Ejemplo del código:

```
<ul>
  {amigos.map((valor, clave) => {
    return <li key={clave} className={clave}>{valor}</li>;
  })}
</ul>
```

En este caso, el arreglo `amigos` contiene nombres que se renderizan como elementos `<li>` dentro de una lista `<ul>`.

Descripción del Código Proporcionado

1. **Declaración de Variables y JSX**:
    - Se declaran variables como `nombre`, `colorFondo` y `sesion`, que se usan dentro del JSX.
    - Se crean estructuras JSX reutilizables como `JSX` y `contenedor`.
2. **Condicionales**:
    - La función `verificarSesion` evalúa la variable `sesion` y retorna diferentes componentes según el valor de esta.
3. **Renderización**:
    - Se renderiza un saludo, información sobre el usuario, contenido condicional y una lista dinámica basada en el arreglo `amigos`.
4. **Estilos y Propiedades en JSX**:
    - Se usan `className` y estilos en línea con objetos para aplicar estilos dinámicamente.

Este código es un ejemplo claro de cómo React combina lógica y diseño para construir aplicaciones dinámicas y reactivas.