

REACT {

<!--Bootstrap-react-->

01

<Por="Karen Marín"/>

02

<Por="Alexandra
Sánchez"/>

Contenidos

01

¿Qué es React?

02

¿Qué es un componente?

03

¿Qué es JSX?

04

¿Cuál es la función de los estados en React?

05

¿Qué son los props?

06

¿Qué son las SPA y ventajas?

07

Diferencia entre vanilla.

Javascript y React.

{

¿Qué es React?

React es una biblioteca de JavaScript creada por Facebook para construir interfaces de usuario interactivas. Se enfoca en crear componentes reutilizables que manejan su propio estado y se combinan para formar aplicaciones web complejas. React utiliza un DOM virtual para optimizar las actualizaciones y hacer que las aplicaciones sean más rápidas y eficientes.

Imagina que estás construyendo con bloques de LEGO. Cada pieza de LEGO es un componente de React. Puedes crear piezas específicas (un botón, un menú, una tarjeta) y luego combinarlas para construir algo más grande (una aplicación completa). Si necesitas cambiar algo, solo cambias esa pieza específica sin desarmar todo.

}

{

¿Qué es un componente?

Un componente es una pieza independiente y reutilizable de código que representa una parte de la interfaz de usuario. Puedes pensarlo como un bloque de construcción. Por ejemplo, un botón, un formulario o una tarjeta pueden ser componentes. Hay dos tipos:

Componentes funcionales:

Son funciones que retornan JSX (los más usados actualmente)

Componentes de clase:

Son clases que extienden de `React.Component` (menos comunes hoy)

}

{

¿Qué es JSX?

```
const elemento = <h1>Hola Mundo</h1>;  
//Esto NO es HTML (aunque lo parezca)  
  
// Es JavaScript con una sintaxis  
  especial
```

- JSX (JavaScript Syntax Extension) es una extensión de la sintaxis de JavaScript que permite escribir código similar a HTML dentro de JavaScript.
- Sirve para crear interfaces de usuario de manera más fácil y legible.
- JSX combina la lógica (JavaScript) y la estructura visual (HTML) en un solo archivo, lo que facilita el desarrollo de componentes.

}

{ ¿Cuál es la función de los estados en React?

El estado (state) es un objeto que almacena datos dinámicos de un componente. Cuando el estado cambia, React automáticamente vuelve a renderizar el componente para reflejar los cambios en la interfaz. Es la forma de hacer que los componentes sean interactivos y respondan a las acciones del usuario.

```
function Contador() { const  
  [count, setCount] = useState(0);  
  return ( <button onClick={() =>  
    setCount(count + 1)}> Clicks:  
    {count} </button> );}
```

¿Qué son los props?

```
function Saludo(props) { return  
<h1>Hola, {props.nombre}  
</h1>; } // Uso: <Saludo  
nombre="María" />
```

Los props (propiedades) son datos que se pasan de un componente padre a un componente hijo. Son inmutables (no se pueden modificar dentro del componente hijo) y permiten personalizar y reutilizar componentes con diferentes datos.

¿Qué son las SPA y sus ventajas? {

¿Qué son las SPA?

Las SPA (Single Page Applications) son aplicaciones web que cargan una única página HTML y actualizan dinámicamente el contenido sin recargar la página completa. React es ideal para crear SPAs.

Ventajas:

- **Experiencia de usuario fluida:** No hay recargas de página, transiciones suaves.
- **Mayor velocidad:** Solo se cargan los datos necesarios, no toda la página.
- **Desarrollo frontend/backend separado:** Mejor organización del código.
- **Menor carga en el servidor:** El procesamiento se hace en el cliente.
- **Interactividad mejorada:** Respuesta inmediata a las acciones del usuario

Diferencia entre Vanilla JavaScript y React {

Vanilla JavaScript:

- Manipulas directamente el DOM
- Escribes más código imperativo (cómo hacer las cosas)
- Actualizaciones manuales de la interfaz
- Más difícil de mantener en aplicaciones grandes
- No tiene estructura predefinida

```
// Vanilla JS
const button = document.getElementById('btn');
let count = 0;
button.addEventListener('click', () => {
  count++;
  document.getElementById('count').textContent =
  document.getElementById('count').textContent =
  document.getElementById('count').textContent =
  count;
});
```

React:

- Trabaja con un DOM virtual, React maneja las actualizaciones
- Código más declarativo (qué quieres mostrar)
- Actualizaciones automáticas cuando cambia el estado
- Componentización facilita el mantenimiento
- Estructura basada en componentes
- Ecosistema robusto de herramientas y librerías

```
// React
function App() {
  const [count, setCount] = useState(0);
  return <button onClick={() => setCount(count + 1)}>
{count}</button>;
}
```

Anthropic. 2023. Claude. <https://www.anthropic.com>.

}

Gracias. {