



REACT

React es una biblioteca de JavaScript desarrollada por Facebook para crear interfaces de usuario (UI).

Permite construir aplicaciones web rápidas, interactivas y modulares mediante el uso de componentes reutilizables.

¿QUE ES ?

- Biblioteca, no un framework completo
- Enfocada en la capa de vista (UI)
- Basada en componentes reutilizables
- Utiliza un Virtual DOM para optimizar el rendimiento
- Mantiene una arquitectura declarativa

PRINCIPALES

CARACTERISTICAS



COMPONENTE

Un componente es una parte independiente y reutilizable de la interfaz.

Cada componente puede tener su propia lógica, estructura y estilo, y se puede combinar con otros para crear una aplicación completa.

Ejemplo: un botón, una barra de navegación o una tarjeta de producto.

TIPOS

1.Componentes funcionales (modernos)

```
function Saludo( { nombre } ) {  
  return <h1>Hola, {nombre}!</h1>;  
}
```

2.Componentes de clase (legado)

```
class Saludo extends React.Component {  
  render( ) {  
    return <h1>Hola, {this.props.nombre}!</h1>;  
  }  
}
```



JSX

JSX (JavaScript XML) es una extensión de sintaxis para JavaScript que permite escribir código similar a HTML dentro de JavaScript.

ejemplo

```
const elemento = <h1>Hola Mundo</h1>;
```

```
const usuario = {  
  nombre: 'María',  
  edad: 25  
};
```

```
const perfil = (  
  <div>  
    <h2>{usuario.nombre}</h2>  
    <p>Edad: {usuario.edad}</p>  
  </div>  
)
```

¿PARA QUE SIRVE?

Hace el código más
legible e intuitivo

Permite usar
expresiones JavaScript
dentro de {}

Combina la lógica de
renderizado con la UI

Se transpila a JavaScript
puro mediante Babel

Previene inyecciones
XSS por defecto

FUNCIÓN DE LOS ESTADOS EN REACT

El estado (state) es un objeto que guarda datos que pueden cambiar con el tiempo.

Cuando el estado cambia, React vuelve a renderizar el componente automáticamente, mostrando la nueva información sin recargar la página.

CARACTERISTICAS

```
import { useState } from 'react';

function Contador() {
  const [contador, setContador] = useState(0);

  return (
    <div>
      <p>Contador: {contador}</p>
      <button onClick={() => setContador(contador + 1)}>
        Incrementar
      </button>
    </div>
  );
}
```

- Es privado y controlado por el componente
- Cuando cambia, el componente se re-renderiza
- Es inmutable (no se modifica directamente)
- Puede ser de cualquier tipo: string, number, array, object

PROPS

Las props (properties) son argumentos que se pasan de un componente padre a un componente hijo, similar a los parámetros de una función.

-EJEMPLO- -CARACTERÍSTICAS-

```
javascript
// Componente hijo
function Tarjeta({ titulo, descripcion, imagen }) {
  return (
    <div className="tarjeta">
      <img src={imagen} alt={titulo} />
      <h3>{titulo}</h3>
      <p>{descripcion}</p>
    </div>
  );
}
```

```
// Componente padre
function App() {
  return (
    <Tarjeta
      titulo="React"
      descripcion="Biblioteca de JavaScript"
      imagen="react-logo.png"
    />
  );
}
```

- Son de solo lectura (inmutables)

- Flujo de datos unidireccional (padre → hijo)

- Pueden ser de cualquier tipo de dato

- Permiten la reutilización de componentes

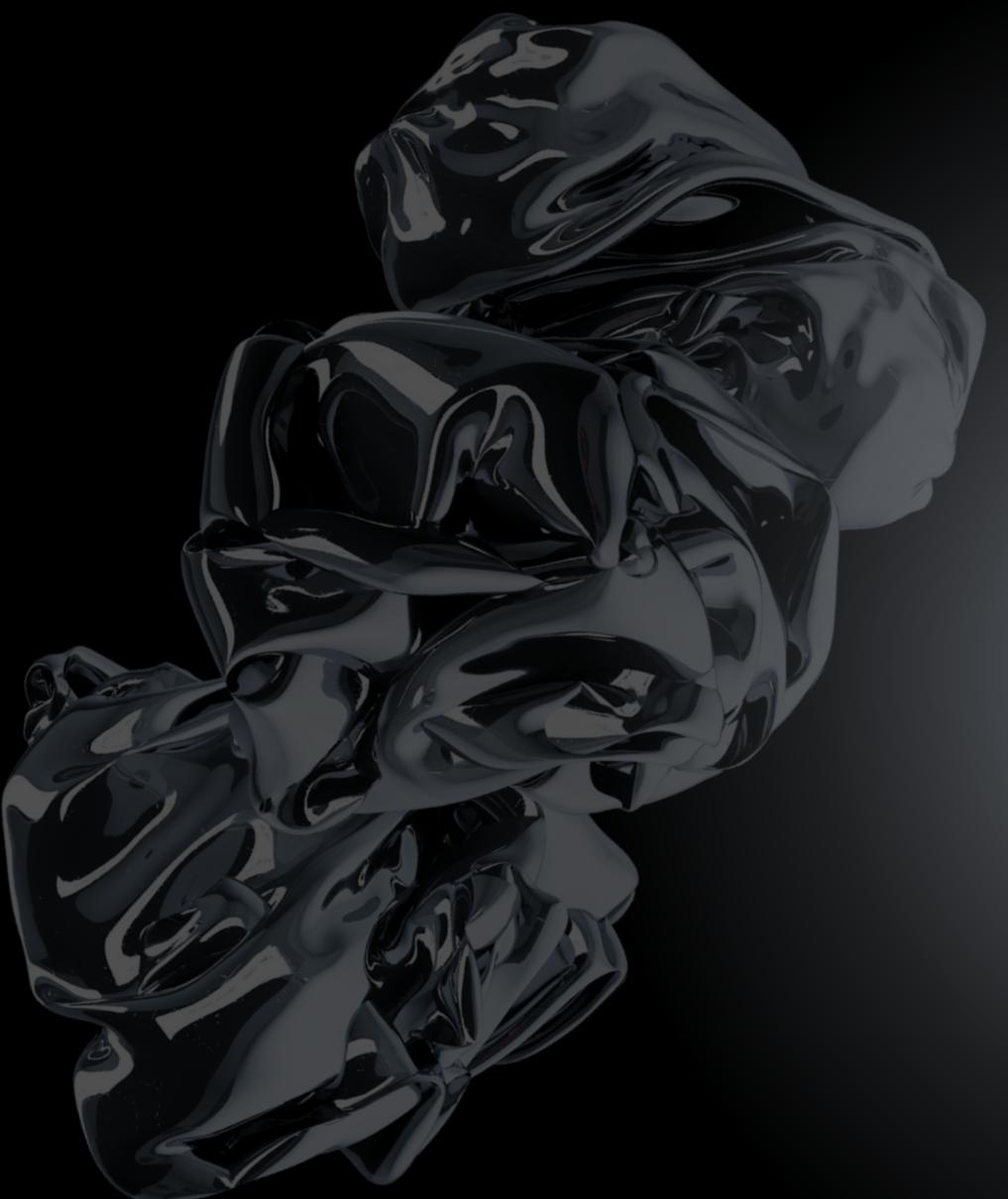
SPA

(Single Page Application)

Es una aplicación web que carga una única página HTML y actualiza dinámicamente el contenido sin recargar la página completa.

¿CÓMO FUNCIONA?

- Se carga inicialmente todo el JavaScript necesario
- La navegación ocurre en el cliente mediante JavaScript
- Solo se intercambian datos con el servidor (JSON)
- El enrutamiento se maneja del lado del cliente



VENTAJAS

- Transiciones rápidas sin recargas
 - Interacciones instantáneas
- Después de la carga inicial, es muy rápida
 - Menor carga en el servidor
- Separación clara entre frontend y backend
 - Reutilización de código
- Puede usar cache para trabajar sin conexión
 - Progressive Web Apps (PWA)

DESVENTAJAS

- Carga inicial más lenta
- Requiere más JavaScript
- SEO puede ser complicado (se soluciona con SSR)



DIFERENCIA ENTRE VANILLA JAVASCRIPT Y REACT

CLAUDE.2025

VANILLA JAVASCRIPT

```
javascript
// Manipulación directa del DOM
const boton =
document.getElementById('miBoton');
const contador =
document.getElementById('contador');
let count = 0;

boton.addEventListener('click', () => {
  count++;
  contador.textContent = count;
});
```

REACT

```
javascript
function Contador() {
  const [count, setCount] = useState(0);

  return (
    <div>
      <p id="contador">{count}</p>
      <button onClick={() => setCount(count + 1)}>
        Incrementar
      </button>
    </div>
  );
}
```

VANILLA JAVASCRIPT

Características

- Manipulación manual del DOM
 - Código más verboso
 - Difícil de mantener en aplicaciones grandes
 - Control total pero más complejidad

REACT

Características

- No manipulas el DOM directamente
 - React se encarga de actualizar la UI
 - Código más limpio y mantenible
- Componentes reutilizables