React.js

React.js es una **biblioteca de JavaScript** de código abierto desarrollada por **Facebook** para construir **interfaces de usuario** (UI). Su enfoque principal es la construcción de aplicaciones web de una sola página (SPA), donde los datos cambian dinámicamente sin necesidad de recargar la página completa. React.js es eficiente y flexible porque utiliza un modelo basado en componentes y un DOM virtual que minimiza las actualizaciones de la página, lo que mejora significativamente el rendimiento.

Componentes en React.js

Los **componentes** son la base de React. Todo lo que ves en una aplicación React está hecho de componentes. Un componente es una **unidad de código reutilizable** que define la estructura visual y el comportamiento de una sección de la UI. Pueden ser **componentes de clase** o **componentes funcionales**, aunque hoy en día se favorecen los componentes funcionales junto con los hooks.

Tipos de componentes:

1. **Componentes funcionales**: Son funciones de JavaScript que retornan elementos JSX (JavaScript XML). Son más simples y se utilizan junto con **hooks** para manejar el estado y otros ciclos de vida.

Ejemplo:

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

2. **Componentes de clase**: Son clases de ES6 que extienden de React.Component y tienen acceso a los métodos de ciclo de vida y al estado interno. Ejemplo:

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

2 Props (Propiedades)

Las **props** (abreviatura de “properties”) son un mecanismo para **pasar datos** de un componente a otro, típicamente de un **componente padre** a un **componente hijo**. Son inmutables, es decir, no se pueden cambiar dentro del componente que las recibe.

- Las props permiten que los componentes sean **reutilizables** y **dinámicos**.

Ejemplo:

```
function Tarjeta(props) {
  return (
    <div>
      <h1>{props.titulo}</h1>
      <p>{props.descripcion}</p>
    </div>
  );
}
```

En este caso, el componente **Tarjeta** puede ser reutilizado con diferentes props para mostrar distintos títulos y descripciones.

Estado (State)

El **estado** es un objeto gestionado internamente por el componente que puede **cambiar** con el tiempo. A diferencia de las props, el estado **es mutable** y puede ser modificado desde dentro del componente, lo que permite a los componentes manejar interactividad, como formularios o eventos de usuario.

En los **componentes funcionales**, el estado se gestiona con **hooks** como useState:

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

function Contador() {
  const [conteo, setConteo] = useState(0);

  return (
    <div>
      <p>Has hecho clic {conteo} veces</p>
      <button onClick={() => setConteo(conteo + 1)}>
        Aumentar contador
      </button>
    </div>
  );
}
```

En este ejemplo, useState(0) crea una variable de estado llamada conteo que inicialmente es 0, y setConteo es la función para actualizar ese estado.

¿Quién utiliza React.js?

React.js es utilizado por **grandes compañías** como **Facebook**, **Instagram**, **Airbnb**, **Netflix**, **Uber**, **Pinterest**, y muchas más para crear aplicaciones web interactivas, dinámicas y escalables. También es muy popular entre **desarrolladores independientes** y **startups** debido a su facilidad de uso y alto rendimiento.

Ventajas de React.js

1. **Rendimiento optimizado**: Gracias al **DOM virtual**, React realiza actualizaciones eficientes del DOM real, solo cambiando lo que es necesario.
2. **Componentes reutilizables**: Los componentes de React permiten crear módulos reutilizables que se pueden usar en diferentes partes de una aplicación.
3. **React Hooks**: Los hooks permiten a los desarrolladores gestionar el estado y los ciclos de vida en componentes funcionales, lo que simplifica la estructura de los componentes.
4. **SEO-friendly**: React puede ser optimizado para SEO cuando se usa junto con tecnologías de **renderizado del lado del servidor** como **Next.js**.
5. **Gran comunidad y ecosistema**: React tiene una comunidad muy activa y un ecosistema de herramientas, bibliotecas y soporte que ayuda a los desarrolladores a crear aplicaciones más rápidamente.

¿Cómo se consigue y cuánto cuesta?

- **React.js es gratuito**. Al ser una biblioteca de código abierto, puedes descargar y usar React sin ningún costo. Todo lo que necesitas hacer es instalarlo desde **npm** (Node Package Manager).

Para instalar React.js:

1. Asegúrate de tener **Node.js** y **npm** instalados en tu sistema.
2. Instala **React** creando un nuevo proyecto con el comando:

npx create-react-app nombre-de-tu-aplicacion

3. **Navega a la carpeta del proyecto:**

cd nombre-de-tu-aplicacion

4. **Inicia la aplicación:**

npm start

Esto abrirá una aplicación React básica en tu navegador local, donde podrás empezar a desarrollar.

Conclusión

React.js es una herramienta poderosa y flexible para crear aplicaciones web modernas. Sus componentes, propiedades (props) y estado permiten un flujo de trabajo estructurado y eficiente, lo que lo hace ideal tanto para desarrolladores principiantes como avanzados. Con una gran comunidad de soporte y un ecosistema en constante crecimiento, React sigue siendo una de las bibliotecas más populares para el desarrollo de interfaces de usuario.