Módulo 4
Desarrollo de interfaces interactivas con React

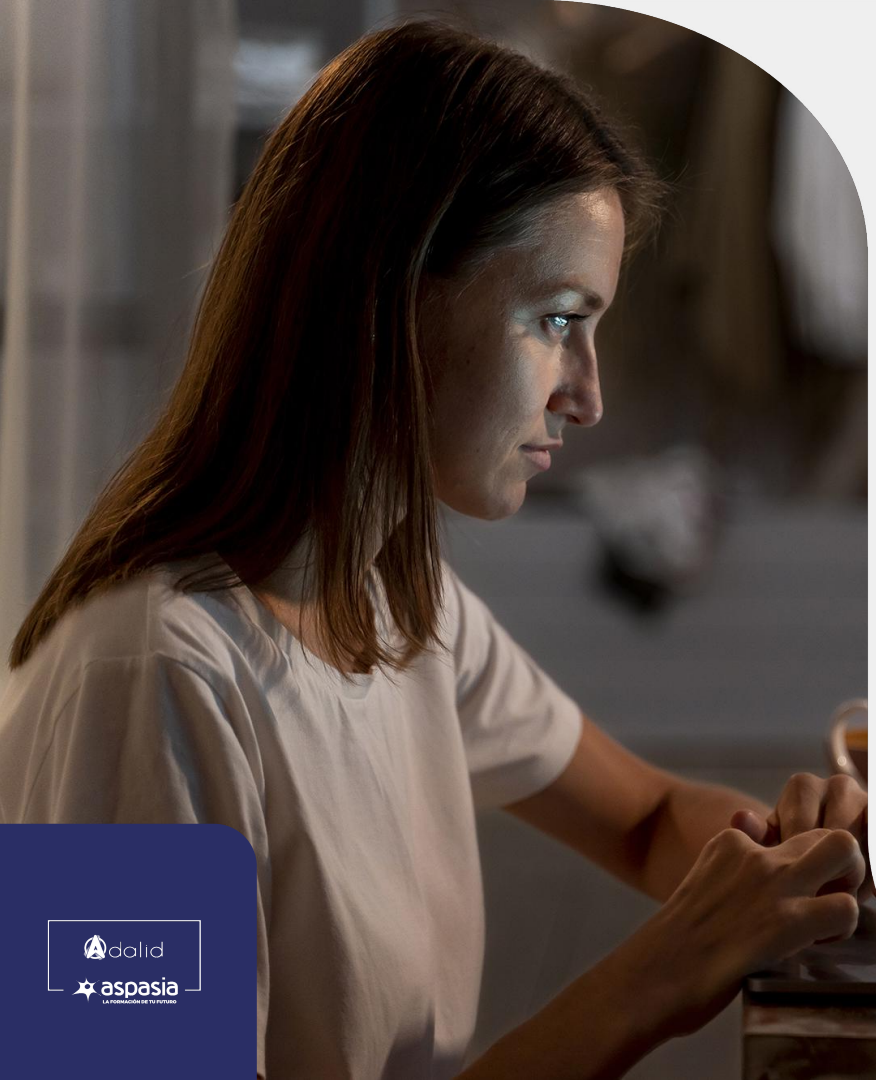
Introducción a ReactJs

Módulo 4

AE 1

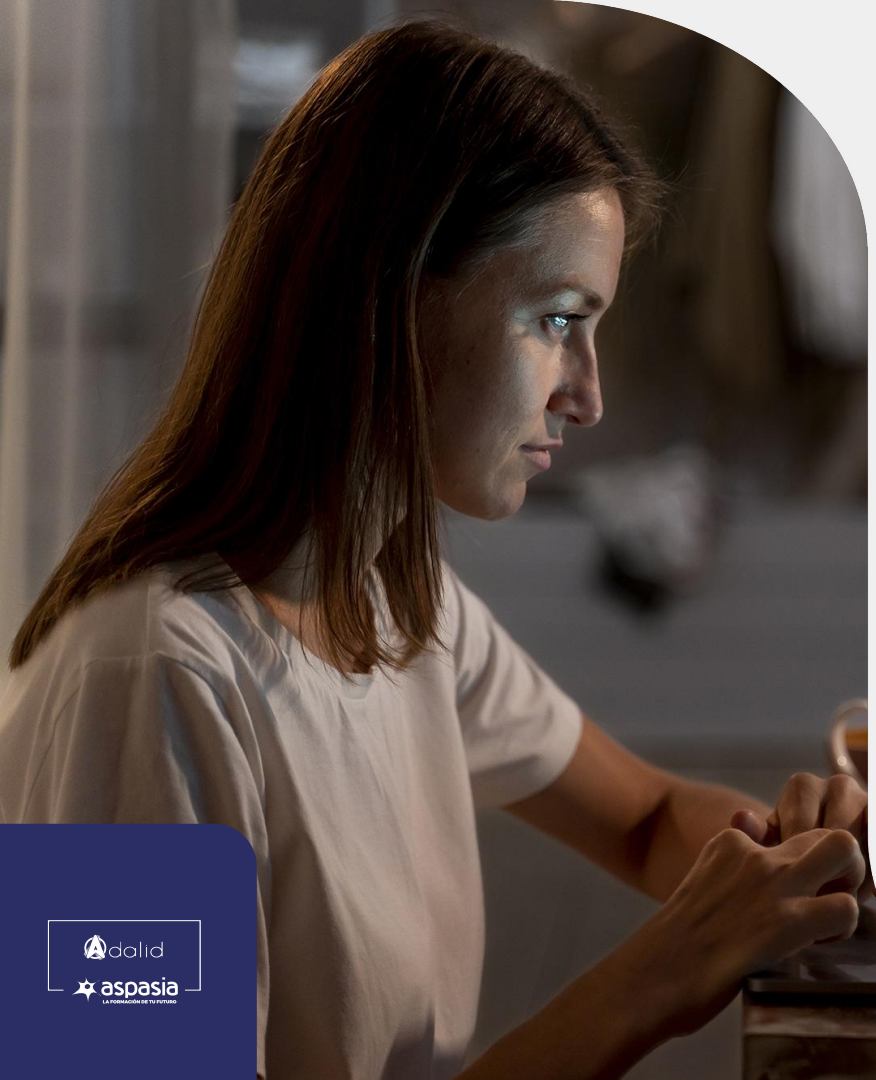
OBJETIVOS

Explorar ReactJS, sus características y entornos de ejecución, entendiendo por qué es ideal para construir aplicaciones SPA basadas en componentes y cómo React se compara con otros frameworks, incluyendo su manejo de DOM, flujo de datos unidireccional, y sintaxis JSX.



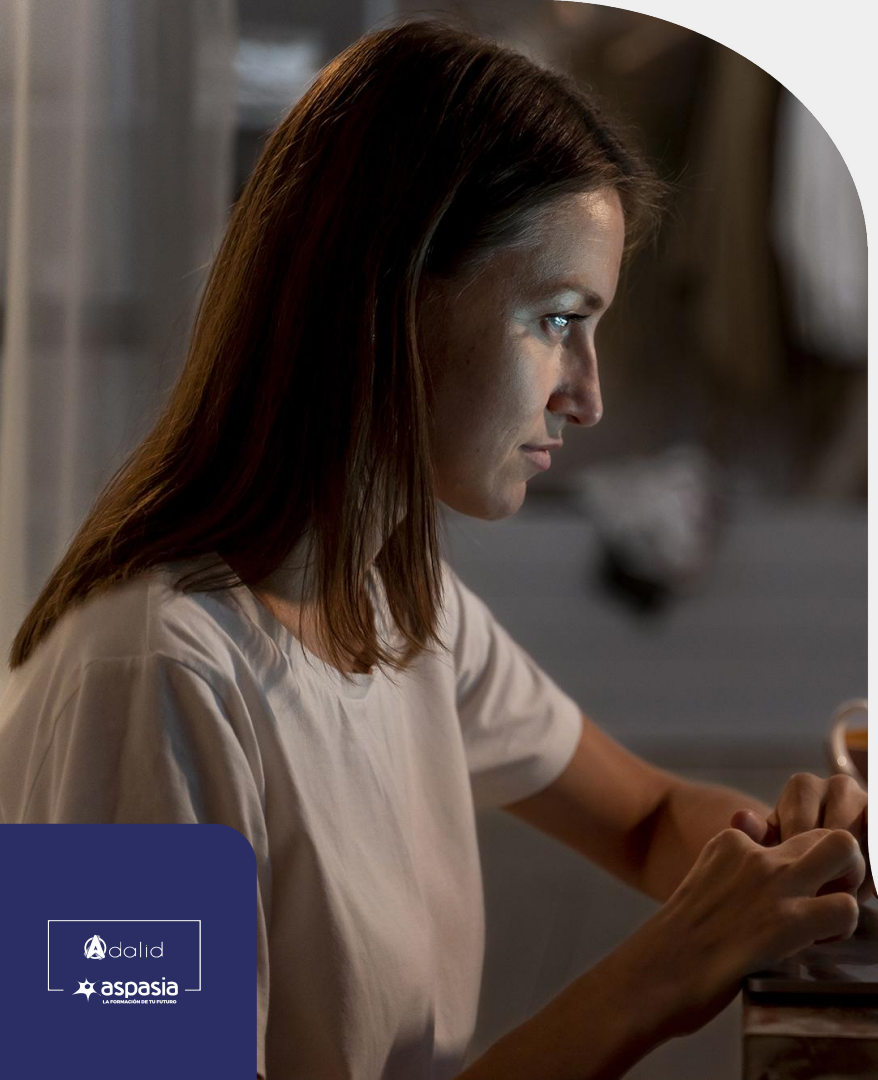
¿QUÉ VAMOS A VER?

- Introducción a ReactJs.
- ¿Qué es ReactJs y cuál es su origen?.
- ¿Porqué usar ReactJs?.
- Acerca de las aplicaciones SPA.
- Renderización de elementos del DOM.
- ReactJs y ReactJs Native.
- Descripción y uso de ReactJs en aplicaciones web.
- ¿Es ReactJs un framework?.
- ReactJs versus otros Frameworks basados en Javascript.
- ¿Qué NO es ReactJs?.



¿QUÉ VAMOS A VER?

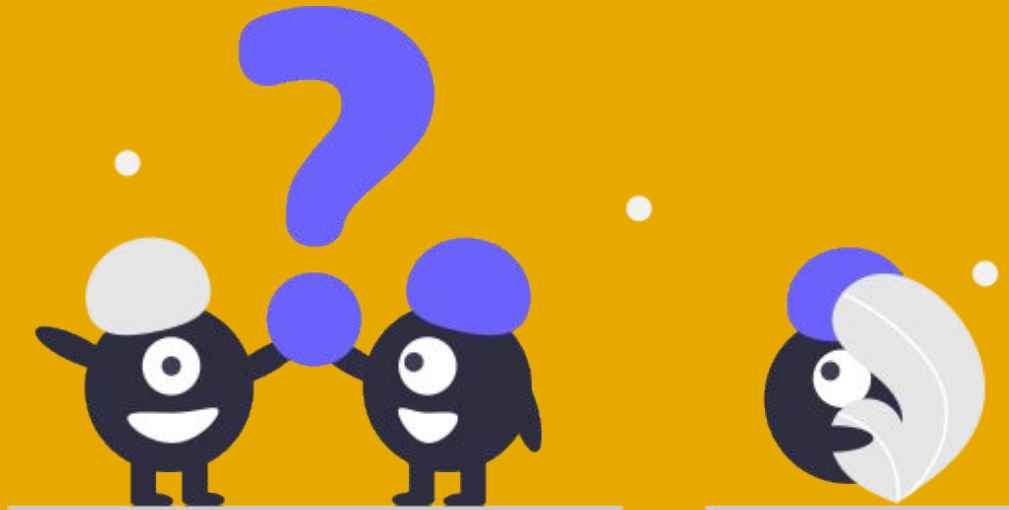
- Características de ReactJs.
- Librería basada en Javascript.
- Lenguaje declarativo.
- Basado en componentes.
- Manejo del DOM.
- Isomorfismo (renderización en cliente y servidor).
- Sintaxis JSX.
- ReactJs sin JSX.
- Flujo de datos unidireccional.

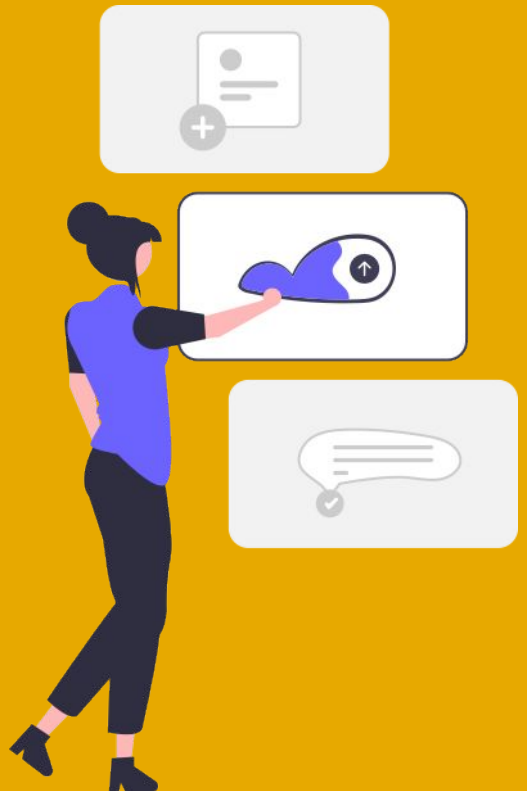


¿QUÉ VAMOS A VER?

- Flujo de datos unidireccional.
- Entornos de Ejecución.
- Prerrequisitos para la utilización de ReactJS.
- Versiones de Javascript compatibles con ReactJs.
- Ejecución desde CDN.
- Ejecución desde un ambiente local.
- Librerías usadas por ReactJs (Webpack, Babel, Next.js, Redux).

¿Qué es ReactJs?

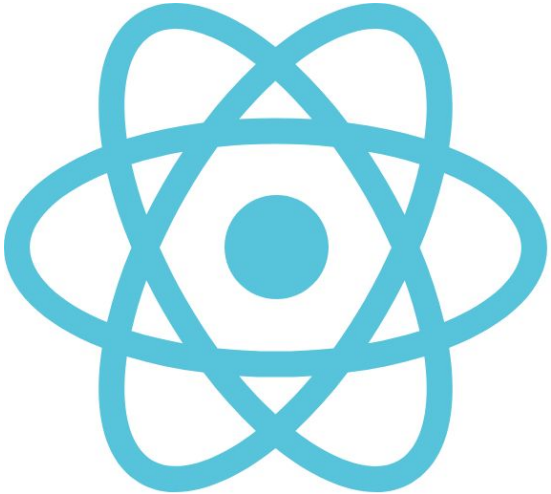




Introducción a ReactJS

Adalid
aspasia
LA FORMACIÓN DE TU FUTURO

¿Qué es ReactJS y cuál es su origen?



ReactJS es una **biblioteca de JavaScript** de código abierto **desarrollada por Facebook en 2013** para crear interfaces de usuario (UI) interactivas y reutilizables. Su principal objetivo es facilitar la **construcción de aplicaciones dinámicas** mediante el uso de componentes y el Virtual DOM.

¿Por qué usar ReactJS?

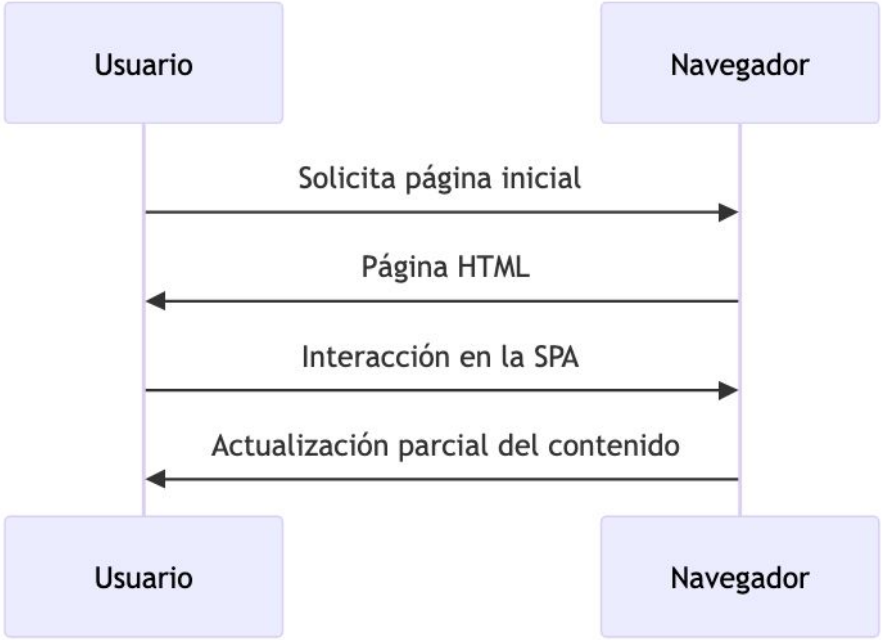
ReactJS es popular por su **rendimiento optimizado**, facilidad para crear **aplicaciones complejas** con menos código, y su soporte para **componentes reutilizables** y gestionados mediante un **flujo de datos unidireccional**.

Beneficios Clave:

- Virtual DOM para renderización eficiente.
- Amplia comunidad y ecosistema.
- Flexibilidad en proyectos de Front-End.

Acerca de las aplicaciones SPA

Las SPA (Single Page Applications) son aplicaciones web que **cargan una sola página HTML** y **actualizan dinámicamente el contenido** sin recargar completamente la página. React es ideal para SPAs debido a su manejo eficiente del Virtual DOM.



Renderización de elementos del DOM

React permite actualizar sólo los **elementos necesarios del DOM** en lugar de toda la página, lo que mejora significativamente el rendimiento.

```
const element = <h2>React es eficiente</h2>;
ReactDOM.render(element, document.getElementById('demo'));
```

React JS y React Native

ReactJS se utiliza para **aplicaciones web**, mientras que React Native permite desarrollar **aplicaciones móviles nativas** usando los mismos principios. Ambos comparten la misma base: componentes reutilizables y el Virtual DOM.

- https://es.react.dev/
- https://reactnative.dev/

¿Es React JS un framework?

React **no es un framework** completo como Angular o Vue, sino una biblioteca centrada en el “V” (vista) del patrón MVC. Esto le da flexibilidad para integrarse con otras tecnologías.

Comparación:

- **ReactJS:** Biblioteca para interfaces de usuario.
- **Angular/Vue:** Frameworks completos con más funcionalidades integradas.

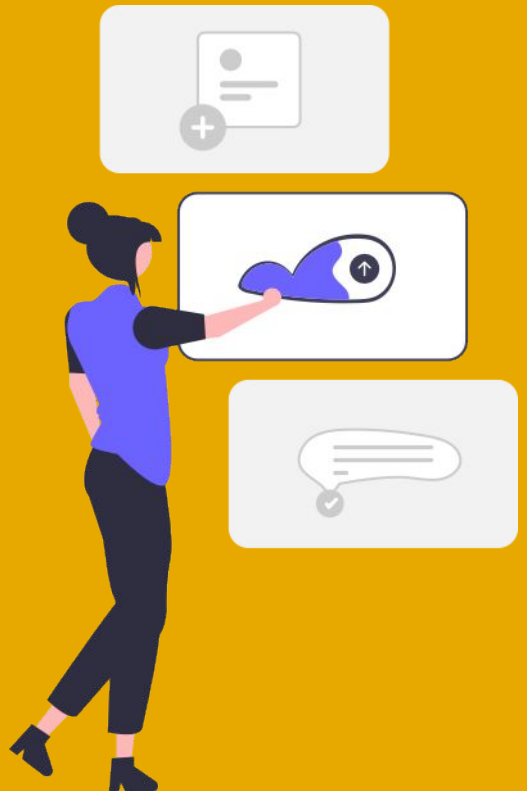
ReactJS VS otros Frameworks

React se distingue por su **enfoque en componentes** y el **Virtual DOM**. Angular y Vue, aunque más estructurados, pueden ser más complejos para principiantes.

Característica	**ReactJS**	**Angular**	**Vue**
Renderizado	Virtual DOM	Real DOM	Virtual DOM
Aprendizaje	Fácil de iniciar	Complejo	Intermedio
Flujo de datos	Unidireccional	Bidireccional	Bidireccional

¿Qué NO es ReactJS?

- **No es un framework completo**: Requiere herramientas adicionales como Redux para gestión de estado o React Router para enrutamiento.
- **No es mágico**: Aunque facilita el desarrollo, el diseño adecuado de los componentes es clave.



Características de ReactJs

adalid
aspasia

Librería basada en JavaScript

ReactJS es una librería escrita en JavaScript que permite construir **interfaces de usuario de manera eficiente y modular**. Utiliza JavaScript como su base para la lógica, lo que facilita la integración con otras tecnologías web.

Lenguaje Declarativo

React usa un enfoque declarativo, donde los desarrolladores especifican **qué quieren que se muestre** en lugar de cómo construirlo. Esto facilita la creación de interfaces complejas al centrarse en el resultado.

```
// Declarativo (React)
const App = () => <h2>Hola Declarativo</h2>;

// Imperativo (Vanilla JS)
const h2 = document.createElement('h2');
h2.textContent = 'Hola Imperativo';
document.body.appendChild(h2);
```

Basado en Componentes

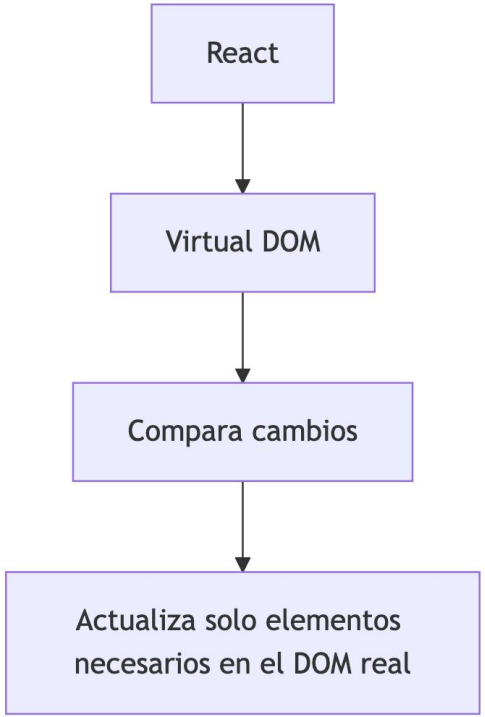
React divide la interfaz de usuario en **componentes reutilizables** y **auto-contenidos**. Cada componente es una pieza modular que puede contener lógica, estilos y estructura.

```
function Boton() {
    return <button>¡Clic aquí!</button>;
}

ReactDOM.render(<Boton />, document.getElementById('root'));
```

Manejo del DOM

React usa el **Virtual DOM** para actualizar solo las partes necesarias del DOM real, mejorando el rendimiento y la experiencia del usuario.



Isomorfismo (Renderización en Cliente y Servidor)

React permite renderizar aplicaciones tanto en el **cliente** como en el **servidor**, lo que mejora el SEO y reduce el tiempo de carga inicial. Herramientas como **Next.js** aprovechan esta característica.

Sintaxis JSX

JSX es una **extensión de JavaScript** que combina HTML y JavaScript en un solo archivo. Facilita la creación de componentes y mejora la legibilidad.

```
const App = () => (
    <div>
        <h1>Bienvenido a React</h1>
        <p>Esta es la magia de JSX.</p>
    </div>
);
ReactDOM.render(<App />, document.getElementById('root'));
```

ReactJS sin JSX

Aunque JSX es común, no es obligatorio. Se puede usar **React.createElement** para crear componentes.

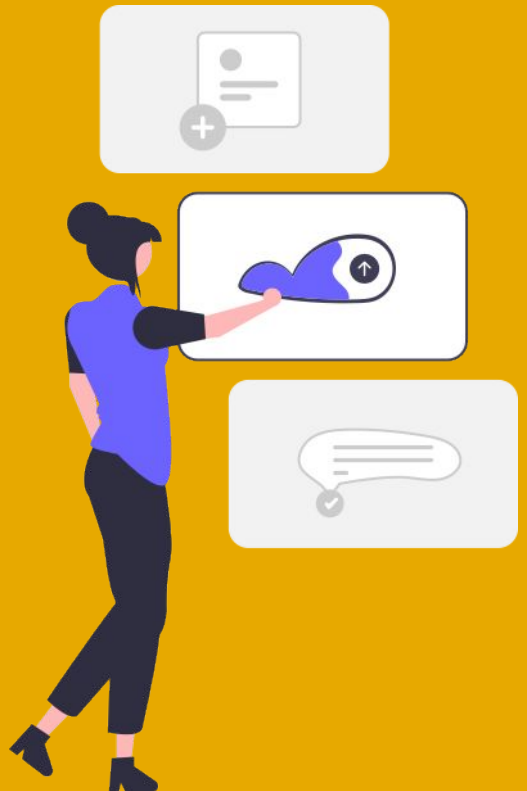
```
const element = React.createElement('h1', null, 'Hola sin JSX');
ReactDOM.render(element, document.getElementById('root'));
```

Flujo de datos unidireccional

React sigue un flujo de datos unidireccional: los datos fluyen de los **componentes padres a los hijos**, lo que facilita el rastreo de cambios y la depuración.

```
function Saludo(props) {
    return <h2>Hola, {props.nombre}</h2>;
}

ReactDOM.render(<Saludo nombre="React" />, document.getElementById('root'));
```



Entornos de Ejecución

Prerrequisitos para la utilización de ReactJS

Para trabajar con ReactJS necesitas conocimientos básicos de:

- HTML, CSS y JavaScript (ES6+).
- Familiaridad con el manejo del DOM.
- Instalación de Node.js y npm para un entorno local.

https://nodejs.org/en

Versiones de JavaScript compatibles con ReactJS

React es compatible con las últimas versiones de JavaScript (ES6 en adelante). Características clave como **arrow functions**, **template literals** y **destructuring** son ampliamente utilizadas.

```
const [nombre, setNombre] = React.useState('React');

// Destructuración
const { length } = nombre;

// Arrow function
const saludar = () => console.log(`Hola, ${nombre}`);
```

Ejecución desde CDN

React puede ejecutarse desde un CDN para proyectos pequeños o pruebas rápidas, **sin necesidad de instalar Node.js**.

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title>React desde CDN</title>
    <script crossorigin
src="https://unpkg.com/react@18/umd/react.development.js"></script>
    <script crossorigin
src="https://unpkg.com/react-dom@18/umd/react-dom.development.js"></script>
</head>
<body>
    <div id="root"></div>
    <script>
        const App = () => React.createElement('h1', null, 'Hola desde React CDN');
        ReactDOM.render(React.createElement(App), document.getElementById('root'));
    </script>
</body>
</html>
```

Ejecución desde un ambiente local

Para proyectos más grandes, se recomienda usar un **ambiente local con Node.js y npm**. React se instala como **dependencia** y puede ejecutarse mediante herramientas como **Vite** o **Create React App**.

Ejecución desde un ambiente local

Create React App

```
# Crear proyecto con Create React App
npx create-react-app mi-app
cd mi-app
npm start
```

```
mi-app/
├── node_modules/
├── public/
│   └── index.html
├── src/
│   ├── App.js
│   ├── index.js
│   └── App.css
├── .gitignore
├── package.json
└── README.md
```

Vite

```
# Crear un proyecto de React con Vite
npm create vite@latest mi-app -- --template react
cd mi-app
npm install
npm run dev
```

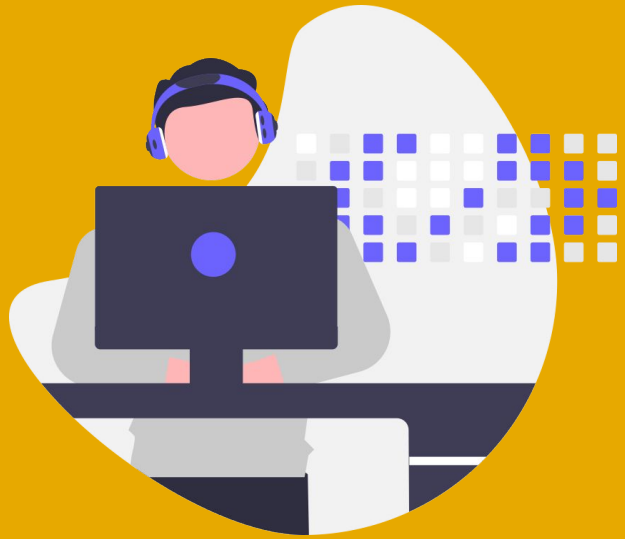
```
mi-app/
├── node_modules/
├── public/
│   └── index.html
├── src/
│   ├── App.css
│   ├── App.jsx
│   ├── index.css
│   └── main.jsx
├── .gitignore
├── package.json
├── vite.config.js
└── README.md
```

Librerías usadas por ReactJS

React utiliza diversas librerías para ampliar su funcionalidad:

- **Webpack:** Para empaquetar módulos.
- **Babel:** Transpila código ES6+ a versiones compatibles con navegadores.
- **Next.js:** Framework para renderización del lado del servidor.
- **Redux:** Administración del estado de aplicaciones complejas.

Pongamos a prueba lo aprendido 😊!!!



Adalid aspasia

Quiz: Introducción a ReactJS

Selecciona la respuesta correcta para cada pregunta. Al final del quiz, revisa tus respuestas para reforzar los conocimientos adquiridos sobre ReactJS. Toma 🗒️ y ✏️.

Quiz: Introducción a ReactJS

1 - ¿Cuál es el propósito principal de ReactJS?

A. Crear aplicaciones móviles únicamente.

B. Diseñar bases de datos.

C. Construir interfaces de usuario basadas en componentes.

D. Generar gráficos en tiempo real.

2 - ¿Qué es una SPA (Single Page Application) y cómo se relaciona con ReactJS?

A. Una aplicación que utiliza múltiples páginas HTML.

B. Una aplicación que recarga toda la página en cada cambio.

C. Una aplicación web que actualiza el contenido dinámicamente sin recargar la página.

D. Una técnica para optimizar bases de datos en tiempo real.

Quiz: Introducción a ReactJS

3 - ¿Qué característica distingue a ReactJS de un framework como Angular?

A. ReactJS es una librería, no un framework.

B. ReactJS no permite el manejo del DOM.

C. ReactJS utiliza solo renderización en servidor.

D. ReactJS no admite JSX para el diseño.

4 - ¿Cuál es la diferencia clave entre ReactJS y React Native?

A. React Native es para aplicaciones web, ReactJS es para aplicaciones móviles.

B. ReactJS es para aplicaciones web, React Native es para aplicaciones móviles.

C. React Native no permite el uso de componentes.

D. ReactJS usa Java, mientras React Native usa Python.

Quiz: Introducción a ReactJS

5 - ¿Qué representa JSX en ReactJS?

A. Un lenguaje de programación alternativo.

B. Una extensión sintáctica de JavaScript para escribir componentes de UI.

C. Un formato para definir bases de datos.

D. Un lenguaje para definir servicios web.

6 - ¿Cuál de las siguientes opciones es un prerrequisito para trabajar con ReactJS?

A. Conocer Java y Python.

B. Tener conocimientos de JavaScript ES6 o superior.

C. Utilizar únicamente frameworks CSS.

D. Tener instalado un servidor físico local.

Quiz: Introducción a ReactJS

Respuestas Correctas

1. C. Construir interfaces de usuario basadas en componentes.

2. C. ReactJS utiliza solo renderización en servidor.

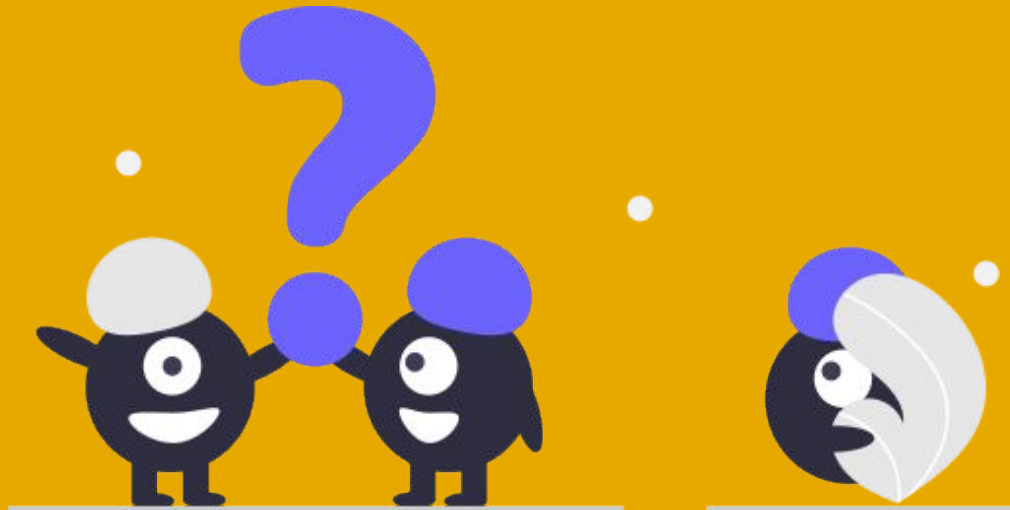
3. A. ReactJS es una librería, no un framework.

4. B. ReactJS es para aplicaciones web, React Native es para aplicaciones móviles.

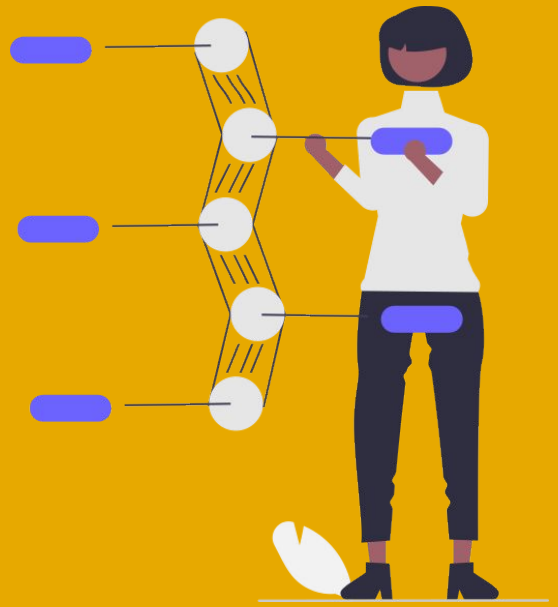
5. B. Una extensión sintáctica de JavaScript para escribir componentes de UI.

6. B. Tener conocimientos de JavaScript ES6 o superior.

¿Como te fue con el Quiz?



Resumen de lo aprendido 🧑‍🎓



Adalid aspasia

Resumen de lo aprendido

- ReactJS es una **librería de JavaScript** diseñada para construir **interfaces de usuario**, destacándose por su enfoque en **componentes reutilizables** y su capacidad para manejar el DOM de manera eficiente.
- Las aplicaciones SPA (Single Page Applications) utilizan ReactJS para **actualizar dinámicamente los contenidos** sin recargar la página, mejorando la experiencia del usuario.
- ReactJS admite características clave como **JSX** para escribir código similar a HTML, un flujo de datos unidireccional y **renderización isomórfica (cliente y servidor)**.
- Para ejecutar ReactJS, se requieren conocimientos de **JavaScript ES6+** y herramientas como Webpack, Babel o Vite, con opciones para ejecución desde CDN o entornos locales.

GRACIAS POR TU ATENCIÓN

Nos vemos en la próxima clase

Chile
Adalid
aspasia
LA FORMACIÓN DE TU FUTURO