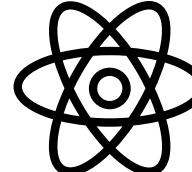


04 Octubre - 2024

Introducción a la programación y
Computación 1



Reactívate en Frontend

Tu primer encuentro con ReactJS 

Andrea María Cabrera Rosito





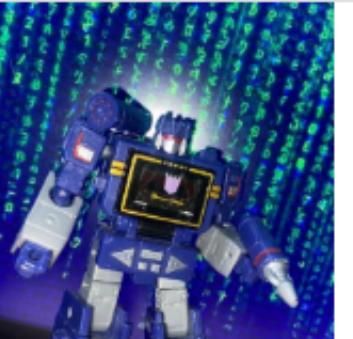
04 Octubre - 2024

Introducción a la programación y
Computación 1



CONTACTO:

Andrea María Cabrera Rosito



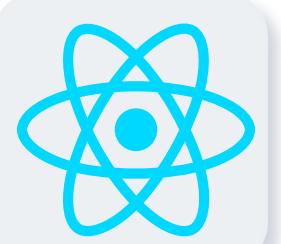
AndreaCabrera01 - Overview

AndreaCabrera01 has 12 repositories available.
Follow their code on GitHub.

 GitHub



AGENDA



¿Qué es ReactJS?



Estado y Props



Rutas

JSX y su sintaxis



Hooks



AGENDA



Tailwind

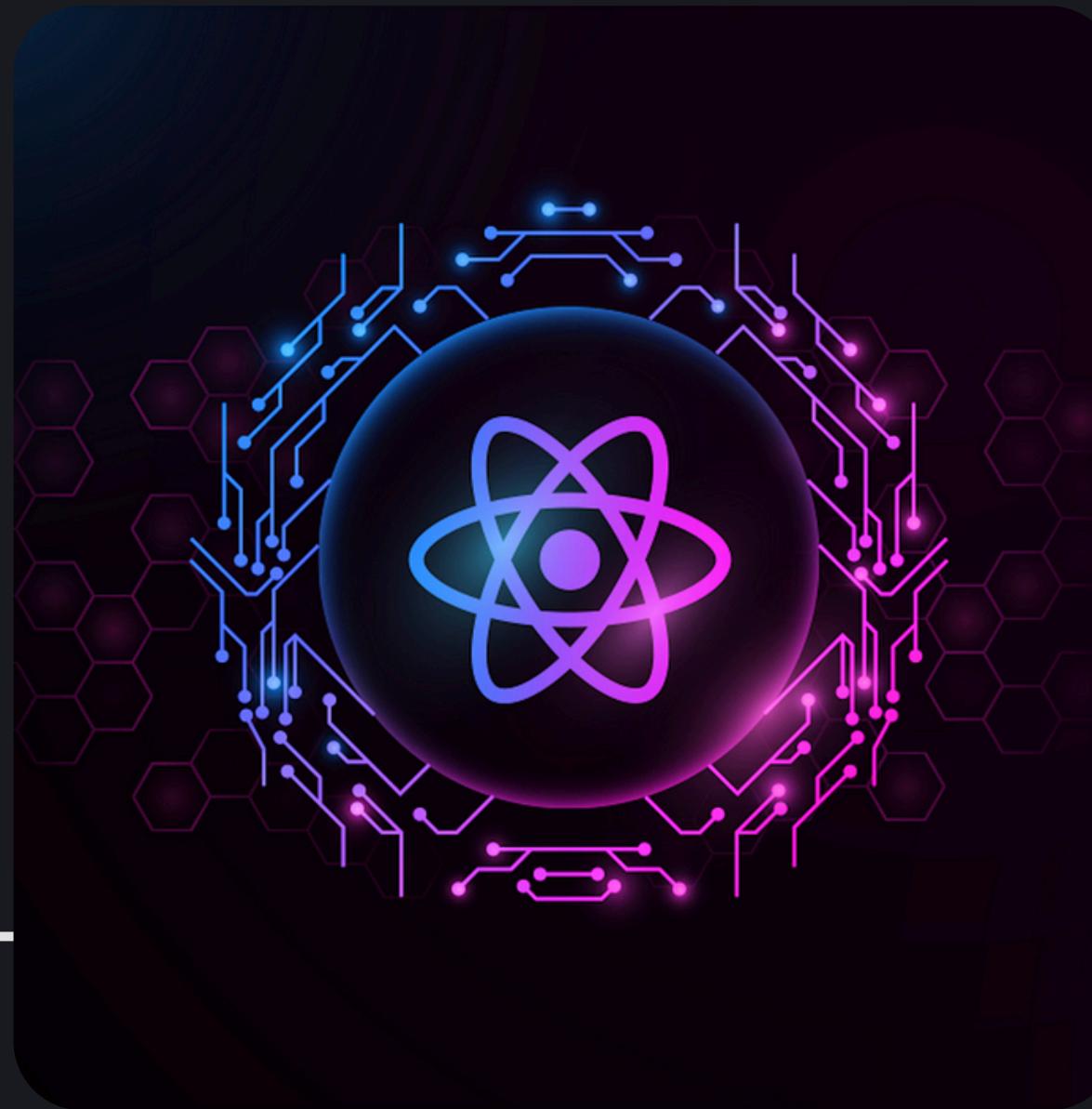
Conexión
Backend/Frontend



Ejemplo práctico



¿Qué es ReactJS?

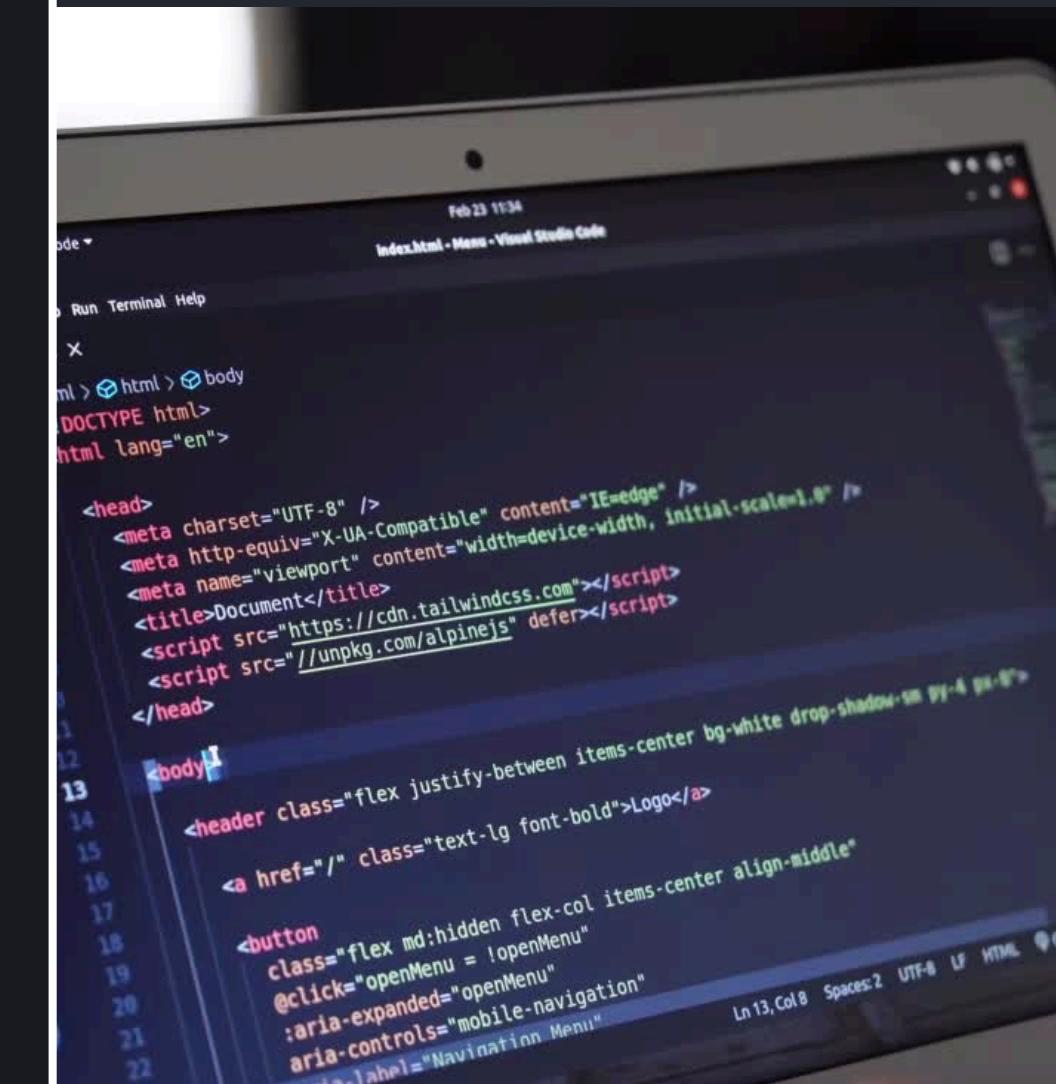


React JS

ReactJS es una biblioteca de JavaScript de código abierto desarrollada por Facebook para construir interfaces de usuario (UI) de manera eficiente y flexible. Se utiliza principalmente para desarrollar aplicaciones web de una sola página (Single Page Applications) donde es crucial manejar cambios dinámicos en la interfaz de manera rápida y eficiente.

Como características principales tiene:

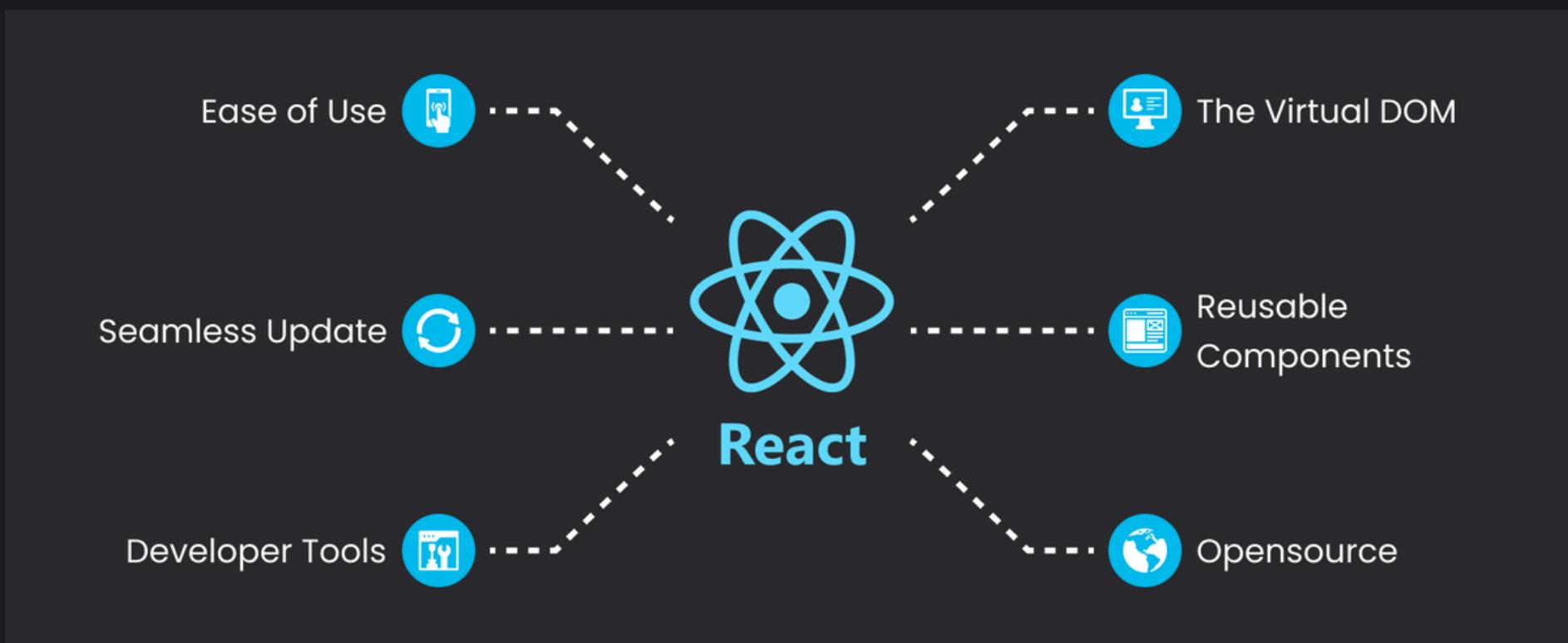
- Componentes Reutilizables
- JSX (Javascript XML)
- Virtual DOM
- Flujo de datos unidireccional
- Amplio ecosistemas y herramientas



```
Feb 23 11:34  
Index.html - New - Visual Studio Code  
File Edit View Insert Terminal Help  
x  
ml > html > body  
DOCTYPE html>  
html lang="en">  
  
<head>  
<meta charset="UTF-8" />  
<meta http-equiv="X-UA-Compatible" content="IE=edge" />  
<meta name="viewport" content="width=device-width, initial-scale=1.0" />  
<title>Document</title>  
<script src="https://cdn.tailwindcss.com"></script>  
<script src="https://unpkg.com/alpinejs" defer></script>  
</head>  
  
<body>  
<header class="flex justify-between items-center bg-white drop-shadow-sm py-4 px-6">  
<a href="/" class="text-lg font-bold">Logo</a>  
<button class="flex md:hidden flex-col items-center align-middle" @click="openMenu = !openMenu" :aria-expanded="openMenu" aria-controls="mobile-navigation" data-label="Navigation Menu">
```



Características Principales



- **Componentes Reutilizables:** React permite dividir la interfaz de usuario en componentes independientes y reutilizables, lo que facilita el desarrollo y el mantenimiento del código.
- **JSX (JavaScript XML):** Es una extensión de sintaxis que permite escribir código similar a HTML dentro de JavaScript, lo que hace que la creación de componentes sea más intuitiva y legible.
- **Virtual DOM:** React utiliza un DOM virtual para optimizar las actualizaciones del DOM real del navegador.
- **Flujo de Datos Unidireccional:** Los datos en React fluyen de los componentes padres a los hijos, lo que facilita el seguimiento y la depuración de la aplicación.
- **Ecosistema y Herramientas:** React cuenta con un amplio ecosistema de bibliotecas y herramientas complementarias, como Redux para la gestión del estado, React Router para el enrutamiento, y Create React App para la configuración rápida de proyectos.



JSX y su sintaxis

JSX es una extensión de sintaxis para JavaScript que permite escribir marcado similar a HTML dentro de una archivo JavaScript. Aunque hay otras formas de escribir componentes, la mayoría de los desarrolladores de React prefieren la concisión de JSX, y la mayoría de las bases de código lo usan.

La web ha sido construida con HTML, CSS y Javascript, siendo el HTML encargado del contenido, CSS sobre el diseño y lógica en Javascript - separando estos 3. Pero ya que se determinó que a este punto, javascript se encuentra a cargo del HTML, por lo que en react la lógica del renderizado y marcado se manejan juntos: **Componentes**.



JSX y su sintaxis

```
<div>  
  <p></p>  
  <form>  
    </form>  
</div>
```

HTML

```
isLoggedIn() {...}  
onClick() {...}  
onSubmit() {...}
```

JavaScript

```
Sidebar() {  
  if (isLoggedIn()) {  
    <p>Welcome</p>  
  } else {  
    <Form />  
  }  
}
```

Componente de React Sidebar.js

```
Form() {  
  onClick() {...}  
  onSubmit() {...}  
  
<form onSubmit>  
  <input onClick />  
  <input onClick />  
</form>  
}
```

Componente de React Form.js



JSX y su sintaxis

- Los componentes de React agrupan la lógica de renderización junto con el marcado porque están relacionados.
- JSX es similar a HTML, con algunas diferencias. Puede usar un [convertidor](#) si lo necesita.
- Los mensajes de error a menudo te guiarán en la dirección correcta para corregir tu marcado.



React

Learn React

REACT.DEV/LEARN

Escribir marcado con JSX – React

The library for web and native user interfaces



Estados y Props



En React, el **estado** (o state) es un objeto que representa las partes variables de una aplicación que pueden cambiar con el tiempo. Es una de las características fundamentales que permite a los componentes de React ser dinámicos y reactivos.

Características del Estado:

- Mutable: A diferencia de las propiedades (props), el estado puede cambiar.
- Local: Cada componente puede tener su propio estado.
- Asíncrono: Las actualizaciones de estado pueden ser asíncronas.



Estados y Props



Ejemplo State:

```
● ● ●

import React, { useState } from 'react';

const Counter = () => {
  const [count, setCount] = useState(0);
  const increment = () => {
    setCount(count + 1);
  };
  const decrement = () => {
    setCount(count - 1);
  };
  return (
    <div style={{ textAlign: 'center', marginTop: '50px' }}>
      <h1>Counter: {count}</h1>
      <button onClick={increment} style={{ marginRight: '10px' }}>
        Increment
      </button>
      <button onClick={decrement}>
        Decrement
      </button>
    </div>
  );
};

export default Counter;
```



Estados y Props



En React, las **props** (abreviatura de "properties") son una forma de pasar datos y funciones desde un componente padre a un componente hijo. Son **inmutables**, lo que significa que un componente hijo no puede modificar las props que recibe; en su lugar, puede utilizarlas para renderizar información o comunicarse con el componente padre.

Características de las Props:

- Inmutables: Un componente no puede cambiar sus propias props.
- Unidireccionales: El flujo de datos va de arriba hacia abajo (de padre a hijo).
- Personalizables: Permiten reutilizar componentes con diferentes datos.



Estados y Props



Ejemplo Props:

```
● ● ●  
import React from 'react';
import ReactDOM from 'react-dom';
import Greeting from './Greeting';

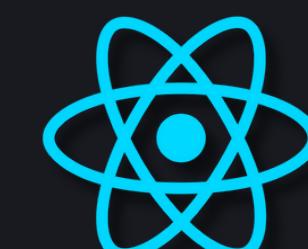
const App = () => {
  return (
    <div>
      <h1>Welcome to the Greeting App</h1>
      {/* Pasando los nombres como props */}
      <Greeting name="Alice" />
      <Greeting name="Bob" />
      <Greeting name="Charlie" />
    </div>
  );
};

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



```
import React from 'react';
const Greeting = ({ name }) => {
  return <h1>Hello, {name}!</h1>;
};

export default Greeting;
```



React

Hooks

Los Hooks son funciones especiales que permiten "enganchar" (hook into) las características de React, como el estado y el ciclo de vida de los componentes, desde componentes funcionales. Antes de los Hooks, para manejar estado y efectos secundarios, era necesario utilizar componentes de clase. Los Hooks simplifican y potencian el desarrollo de componentes reutilizables y fáciles de mantener.

Los tipos de hooks que se pueden usar son:

- State Hooks
- Context Hooks
- Ref Hooks
- Effect Hooks
- Performance Hooks
- Customizables (Funciones Javascript)



API Reference

REACT.DEV/REFERENCE/REACT

Built-in React Hooks

The library for web and native user interfaces

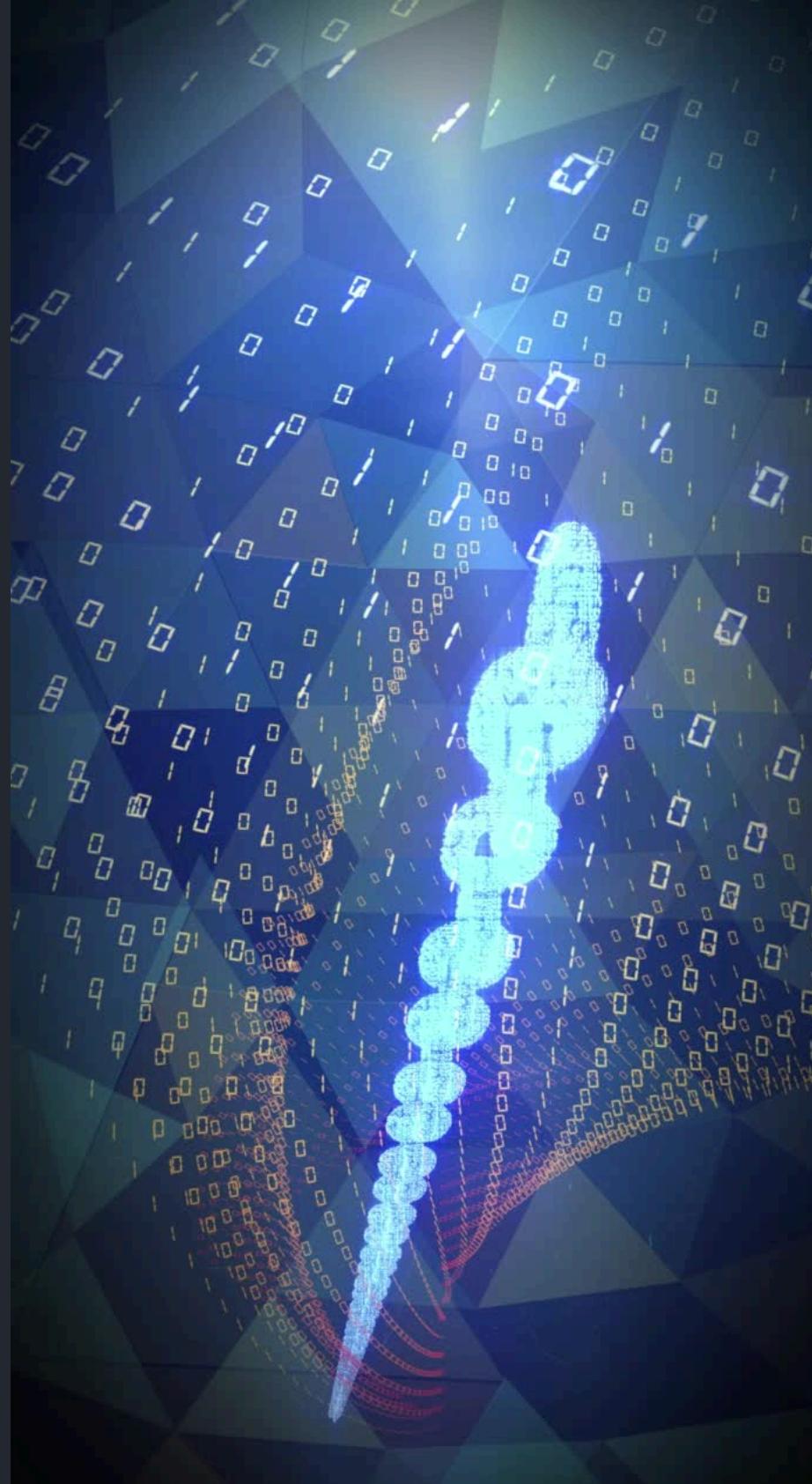


Rutas



El Routing en React permite gestionar la navegación entre diferentes componentes o vistas dentro de una aplicación sin recargar la página completa. Esto proporciona una experiencia de usuario fluida y rápida.

La biblioteca más utilizada para implementar routing en React es **React Router**. Ofrece una solución completa y flexible para manejar rutas en aplicaciones React, soportando características avanzadas como rutas anidadas, rutas protegidas, redirecciones, y más.

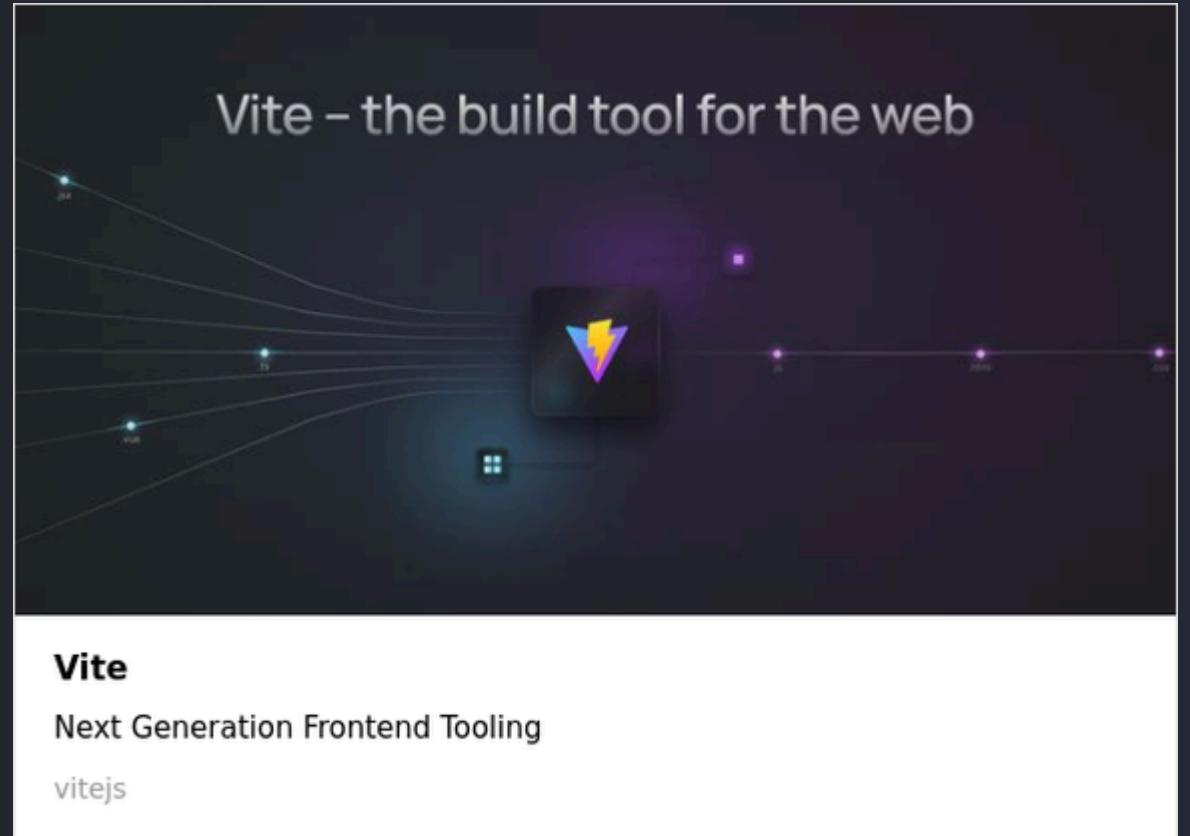




Vite es una herramienta de construcción rápida y ligera que se utiliza para proyectos de desarrollo web modernos, incluyendo aplicaciones React.

Beneficios:

- Arranque rápido:** Vite utiliza un servidor de desarrollo que arranca extremadamente rápido en comparación con herramientas tradicionales como Webpack.
- Hot Module Replacement (HMR):** Vite tiene un HMR altamente eficiente, lo que permite actualizar partes de la aplicación React en tiempo real sin recargar la página completa.
- Optimización de la construcción:** Vite utiliza esbuild bajo el capó para la construcción, lo que lo hace mucho más rápido al empaquetar archivos para producción.
- Soporte nativo de ES Modules:** En lugar de agrupar todos los módulos, Vite entrega los archivos JavaScript en módulos ES nativos en el navegador, lo que lo hace más eficiente en entornos de desarrollo.
- Configuración mínima:** Vite viene con soporte listo para React, lo que significa que no se necesita mucha configuración extra para empezar un proyecto con React.

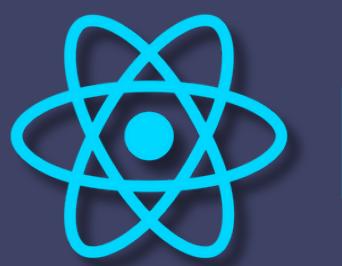




Para empezar, se debe tener **Node.js** y **npm** (Node Package Manager) instalados.

- Si todavía no se tienen, pueden descargarse desde: <https://nodejs.org/en/download/>
- Descarga el instalador para tu sistema.
- Una vez completada la instalación, puedes comprobar que Node.js y npm están instalados abriendo un símbolo del sistema y ejecutando los siguientes comandos:
 - `node -v npm -v`





React



Vite

Inicializando nuestro proyecto:

```
npm create vite@latest <<nombre_proyecto>> -- --template react
```

Navegamos al directorio creado:

```
cd <<nombre_proyecto>>
```

Instalamos las dependencias:

```
npm install
```

Iniciamos el servidor de desarrollo:

```
npm run dev
```



Tailwind CSS



Tailwind CSS es un **framework de CSS** que proporciona una amplia gama de clases de utilidad predefinidas para construir diseños directamente en el marcado HTML. A diferencia de otros frameworks como Bootstrap o Foundation, Tailwind no ofrece componentes predefinidos, sino que se enfoca en proporcionar las herramientas necesarias para diseñar cualquier tipo de interfaz de manera personalizada y eficiente.

Ventajas de Usar Tailwind CSS

1. *Productividad Mejorada:* Al tener una amplia gama de clases de utilidad, puedes diseñar rápidamente sin salir del HTML.
2. *Consistencia en el Diseño:* Facilita mantener una coherencia en los estilos a lo largo de toda la aplicación.
3. *Menor Dependencia de CSS Personalizado:* Reduce la necesidad de escribir CSS desde cero, lo que disminuye la complejidad y el tiempo de desarrollo.
4. *Facilidad para Realizar Cambios:* Modificar estilos es tan sencillo como cambiar o agregar clases en el HTML.
5. *Gran Comunidad y Ecosistema:* Amplia cantidad de recursos, tutoriales y plugins disponibles.



Tailwind CSS



INSTALACIÓN CON VITE:

The screenshot shows a section of the Tailwind CSS documentation titled "Install Tailwind CSS with Vite". The title is bold and dark blue. Below it is a subtitle: "Setting up Tailwind CSS in a Vite project." At the bottom of the screenshot, there is a small footer with the Tailwind CSS logo icon and the text "tailwindcss".

Getting Started

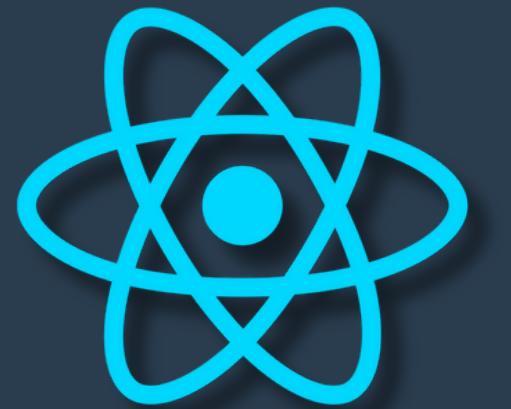
Install Tailwind CSS with Vite

Setting up Tailwind CSS in a Vite project.

Install Tailwind CSS with Vite - Tailwind CSS

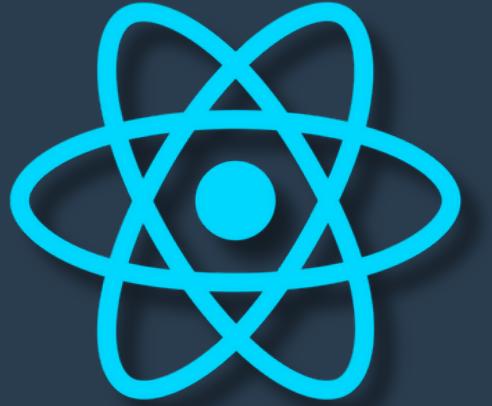
Setting up Tailwind CSS in a Vite project.

tailwindcss



EJEMPLO PRÁCTICO

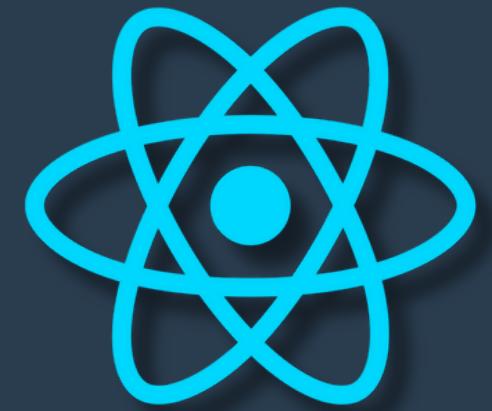




PLATAFORMA DE GESTIÓN DE EVENTOS

- Manejo de Forms (CRUD)
- Conexión con Backend
- Manejo de Sesión
- Rutas





¿DUDAS?

