

# Conceptos Basicos

# React

*juan jose zapata  
juan sebastian villamil*



02



# Que es react

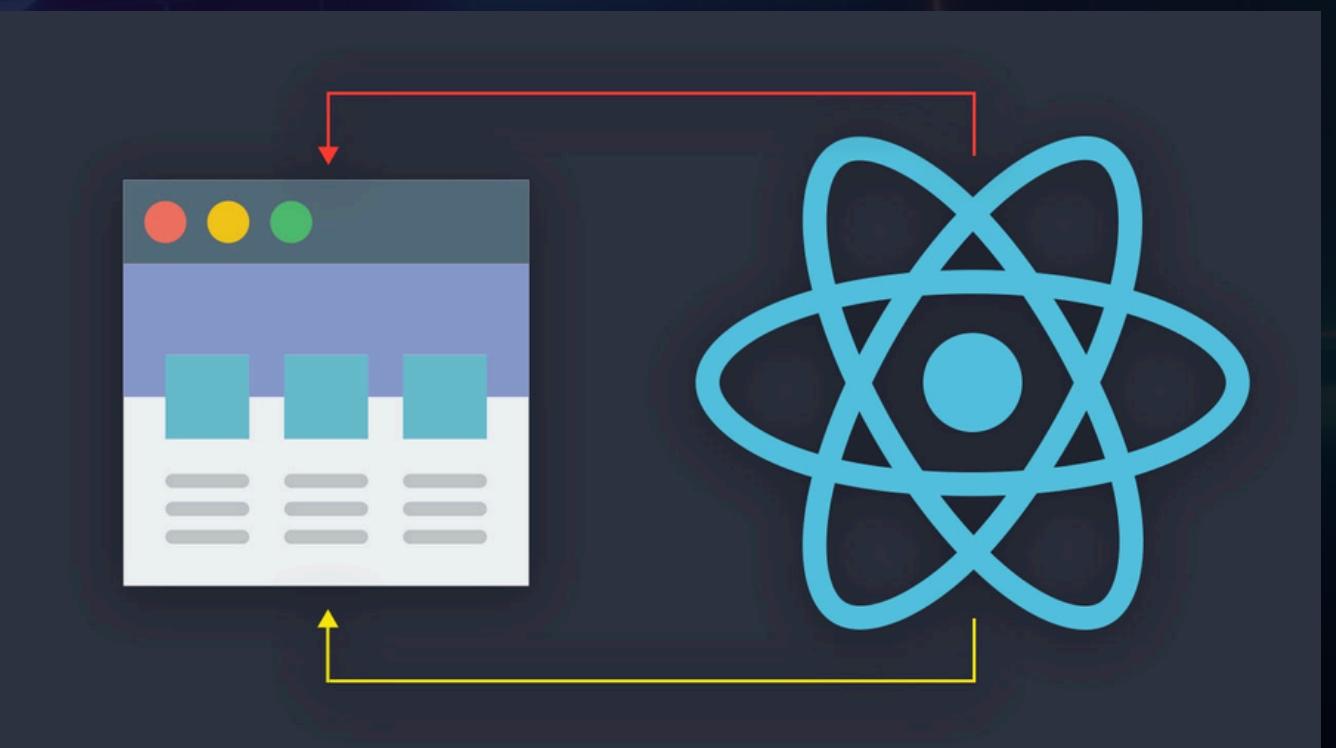
*Es una libreria de javas script que nos ayuda a construir interfacez para la web rapidamente.*

*su objetivo principal es facilitar el desarrollo de aplicaciones web complejas, permitiendo que al crear componentes de interfaz de usuario reutilizables se actualizan de manera eficiente cuando los datos cambian.*

# Que es un componente

*Es una pieza reutilizable de la interfaz de usuario que combina java script y HTML para crear elemtenos en la pantalla.*

*Los componentes son pequeñas partes que el usuario ve o interactua con ellos, se utilizan como plantillas que se pueden cambiar, reutilizar, combinar, e.t.c*



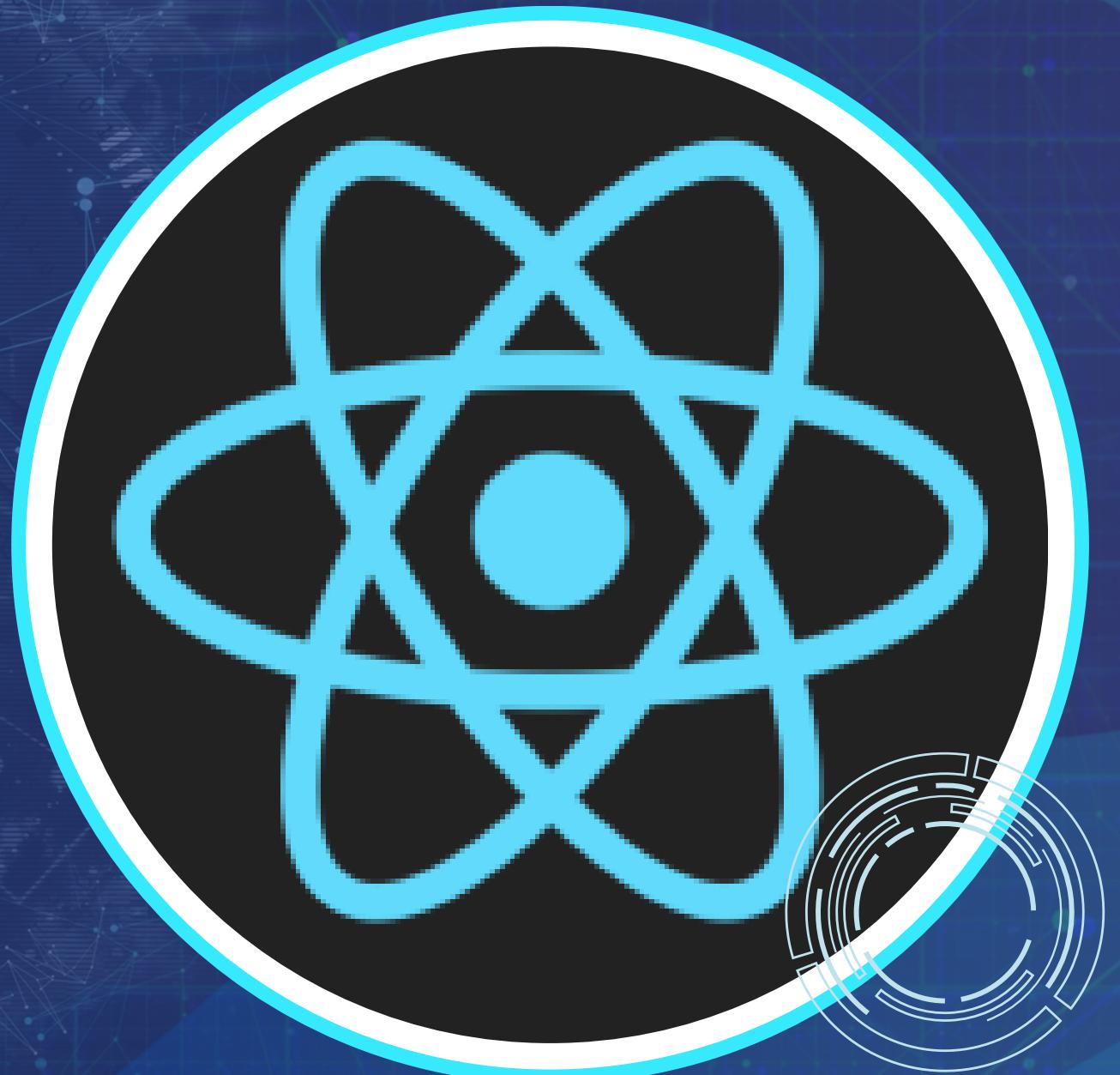


'thynk unlimited'

# QUE ES JSX

*es una union de HTML + CCS + JS en la que podemos crear contenido con bastantes funcionalidades , colocando todo en un mismo archivo.*

*El objetivo de JSX es hacer que la creación y visualización de la estructura de la interfaz de usuario sea más intuitiva y legible para los desarrolladores, otra de sus ventajas es que puede reutilizar el código y su productividad.*



# Que es un estado en React

*a un estado nos referimos a un objeto que representa alguna parte de la aplicación o un componente que permite que esta cambie o pueda ser actualizada.*

*su principal función es almacenar y gestionar los datos que son susceptibles de cambiar con el tiempo e influir en lo que se muestra en la interfaz de usuario*

# Que son las props

*La palabra "props" viene de "properties" (propiedades), y se usa principalmente en el mundo de React –una biblioteca de JavaScript que sirve para crear interfaces de usuario*

*Las props son como los "mensajes" o "paquetes de información" que un componente le envía a otro.*

*Piensa en un componente como si fuera una cajita reutilizable (por ejemplo, una tarjeta de perfil).*

*Pero no todas las tarjetas son iguales, ¿cierto?*

*Algunas muestran el nombre "Ana", otras "Juan", otras "Carlos".*

*Entonces, cuando creas una tarjeta, tú le pasas datos específicos como:*

jsx

 Copiar código

```
<Perfil nombre="Ana" edad="25" />
```

# Que son las props

Aquí, *nombre* y *edad* son las props.

Le estás diciendo al componente *Perfil*:

"Ey, muéstrame esta cajita, pero con estos datos en particular".

Y dentro del componente, las usas así:

```
jsx Copiar código
function Perfil(props) {
  return (
    <div>
      <h2>{props.nombre}</h2>
      <p>Edad: {props.edad}</p>
    </div>
  );
}
```

# Que son las SPA y sus ventajas

*SPA significa Single Page Application o Aplicación de Página Única.*

*Es un tipo de aplicación web que carga una sola página (normalmente index.html) y luego actualiza su contenido dinámicamente, sin recargar toda la página cada vez que el usuario hace clic o navega.*

*Todo el sitio funciona como una app interactiva, usando tecnologías como JavaScript, React, Vue o Angular.*

*Una SPA se comporta más como una aplicación móvil que como una web tradicional. El usuario navega por secciones, pero sin interrupciones ni recargas, lo que hace la experiencia más rápida y fluida.*

VELOCIDAD

EXPERIENCIA FLUIDA

MEJOR XP DE  
USUARIO

REUTILIZACION DE  
COMPONENTES

# diferencias de javascript vanilla y react

*Vanilla JavaScript es el lenguaje de programación fundamental que todos los navegadores web pueden interpretar directamente. Representa el enfoque más básico y directo para el desarrollo web.*

- Manipulación del DOM: Se realiza de forma imperativa y manual. El desarrollador debe escribir código explícito para cada paso que modifica el Document Object Model (DOM), como crear un elemento, asignarle atributos, y adjuntarlo al árbol (ej., `document.createElement('div')`). Esto se vuelve tedioso y propenso a errores en aplicaciones grandes.*
- Organización: El código tiende a ser organizado por archivos separados para HTML, CSS y JS, pero la lógica de la aplicación en sí puede volverse monolítica y difícil de mantener a medida que crece.*
- Manejo de Datos y Estado: Completamente manual. Si un dato cambia, es responsabilidad del desarrollador escribir código para encontrar dónde se usa ese dato en la UI y actualizarlo manualmente.*
- Rendimiento: Puede ser lento en aplicaciones complejas, ya que la manipulación directa del DOM es una operación costosa para el navegador.*

- Manipulación del DOM: Utiliza el concepto de Virtual DOM (DOM Virtual) y un enfoque declarativo. El desarrollador solo describe cómo debe verse la UI con un conjunto de datos (usando JSX), y React se encarga de determinar el camino más eficiente para realizar esos cambios en el DOM real.*
- Organización: Se basa en una arquitectura de componentes. La UI se divide en piezas aisladas y reutilizables, donde la lógica y el marcado (escrito en JSX) conviven, promoviendo la modularidad.*
- Manejo de Datos y Estado: Es automático y reactivo. A través de los Hooks (como `useState`), React rastrea el estado y automáticamente dispara el proceso de re-renderización del componente, asegurando que la UI esté siempre sincronizada con los datos actuales.*
- Rendimiento: Es generalmente más rápido en aplicaciones dinámicas, ya que su algoritmo de "diffing" del Virtual DOM optimiza las actualizaciones, evitando manipulaciones innecesarias del DOM real.*