

React JS



¿Qué necesito para aprender ReactJS?

Hay muchas comunidades de código para el aprendizaje sobre React

Bases: Html , Css, JavaScript

Extras: Git & Terminal o línea de comandos.



¿Por qué aprender ReactJS?

React es una librería o biblioteca que se destaca por ser simple y por tener una api muy amigable que la podemos entender como si fuera JavaScript

Empresas y Startups como, Airbnb, Facebook, Instagram, Uber utilizan React.



React

Una biblioteca de código abierto de JavaScript para construir interfaces de usuario.

Conceptos

Código imperativo (Vainilla javascript):

Es decir, como una secuencia de operaciones a realizar.

Código declarativo (React):

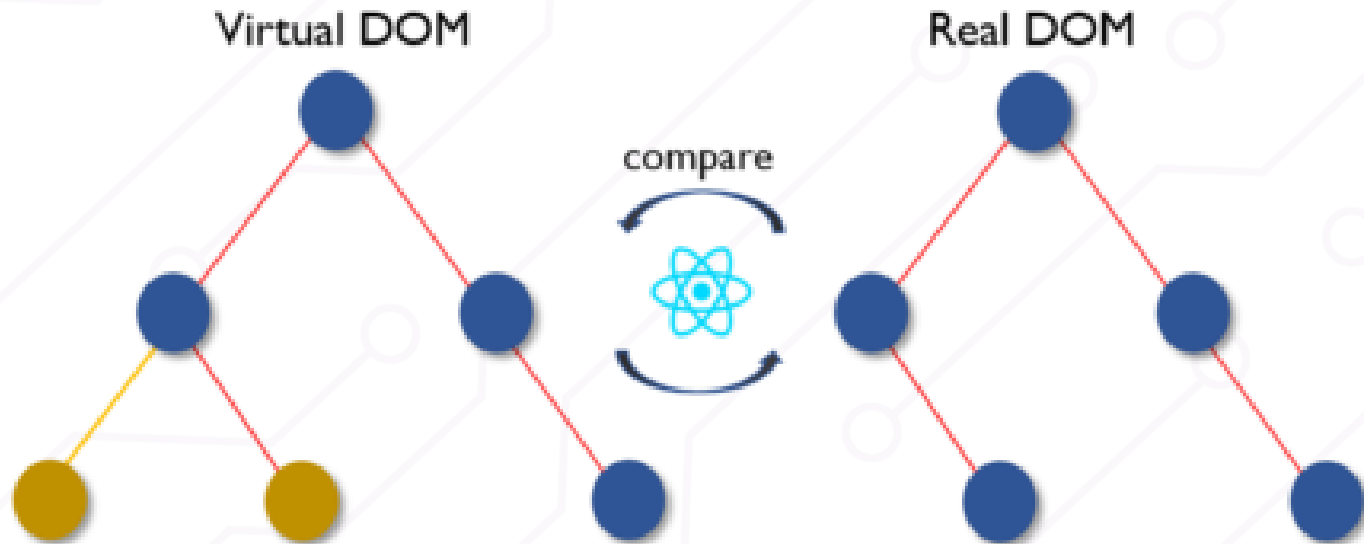
Es decir, se especifica el resultado deseado, no cómo lograrlo.

¿Cómo funciona?

*React crea un **DOM VIRTUAL** en la memoria. En lugar de manipular el DOM del navegador directamente, React crea un DOM virtual en la memoria, donde realiza toda la manipulación necesaria antes de realizar los cambios en el DOM del navegador.*

¡React solo cambia lo que necesita ser cambiado!

Virtual DOM





Biblioteca

Conjunto de implementaciones o subprogramas que podemos usar en nuestro programa.

Características

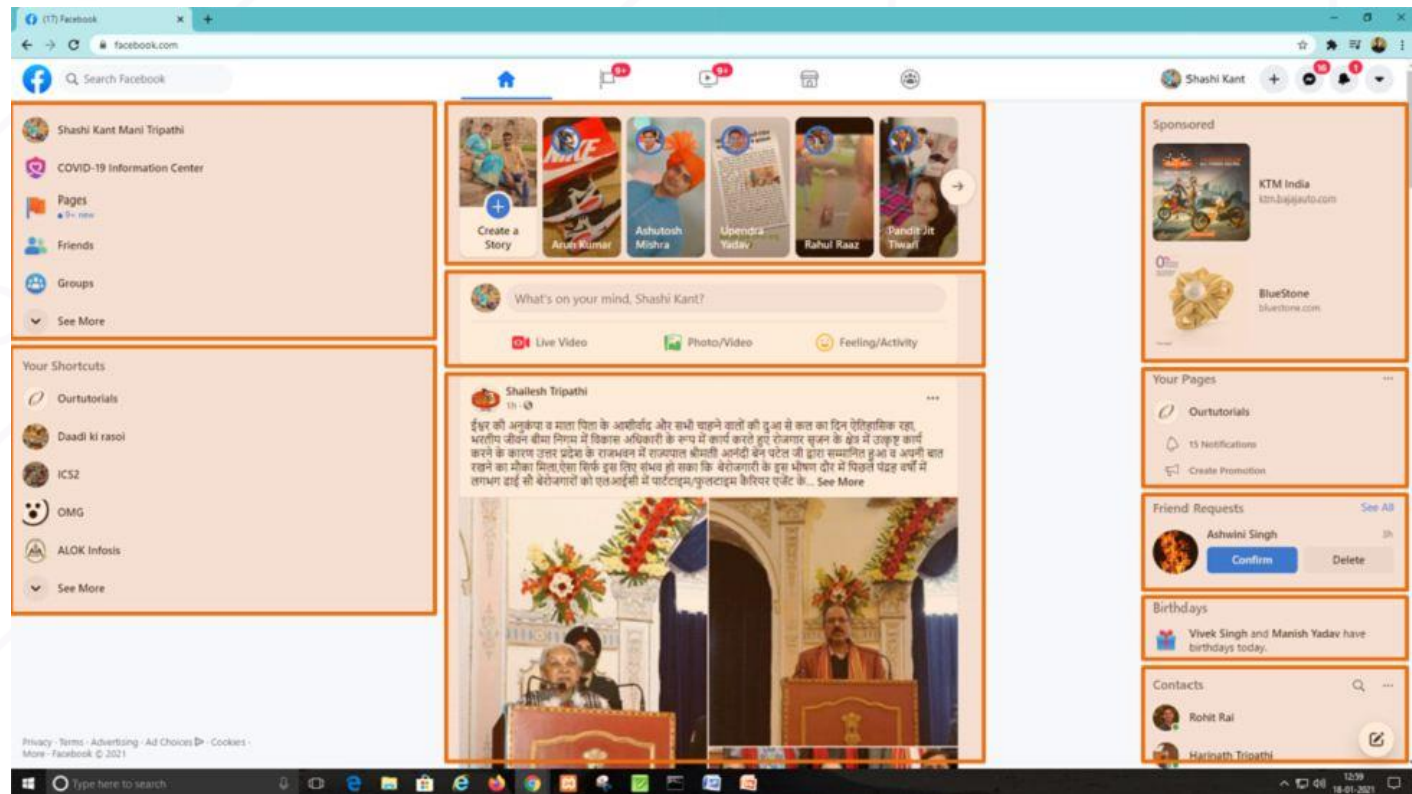
- Es relativamente fácil de aprender al manejar sintaxis de JavaScript
- Componentes reutilizables (reutilizar el código, fácil de revisar)
- Crea aplicaciones dinámicas que permita actualizar estados de la app
- Actualiza las partes de la aplicación que se debe actualizar.

Componente

Es una parte de la interfaz de usuario que es independiente y reusable. Actúan como piezas que se ensamblan para crear las aplicaciones.

- Los componentes se dividen en Funcionales y de Clases
- Los componentes de clase se usaban en versiones anteriores de React

Componente



Componente



Componente funcional

Es una función de JavaScript que Retorna un elemento de React. Un elemento que se escribirá en una versión extendida de JavaScript JSX.

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

Componente funcional

Características:

- Debe retornar elementos de React, definido con JSX.
- Los nombres de funciones deben empezar con letra mayúsculas.
- Se puede recibir parámetros. En React se llaman props (propiedades) y se pueden usar dentro de la función.

```
const Box = props => {  
  return (  
    <div className="d-flex flex-row p-4">  
      <span>{Element(1)}</span>  
      <span>{Element(2)}</span>  
      <span>{Element(3)}</span>  
    </div>  
  );  
};
```

Props

Son argumentos que reciben los componentes de React.

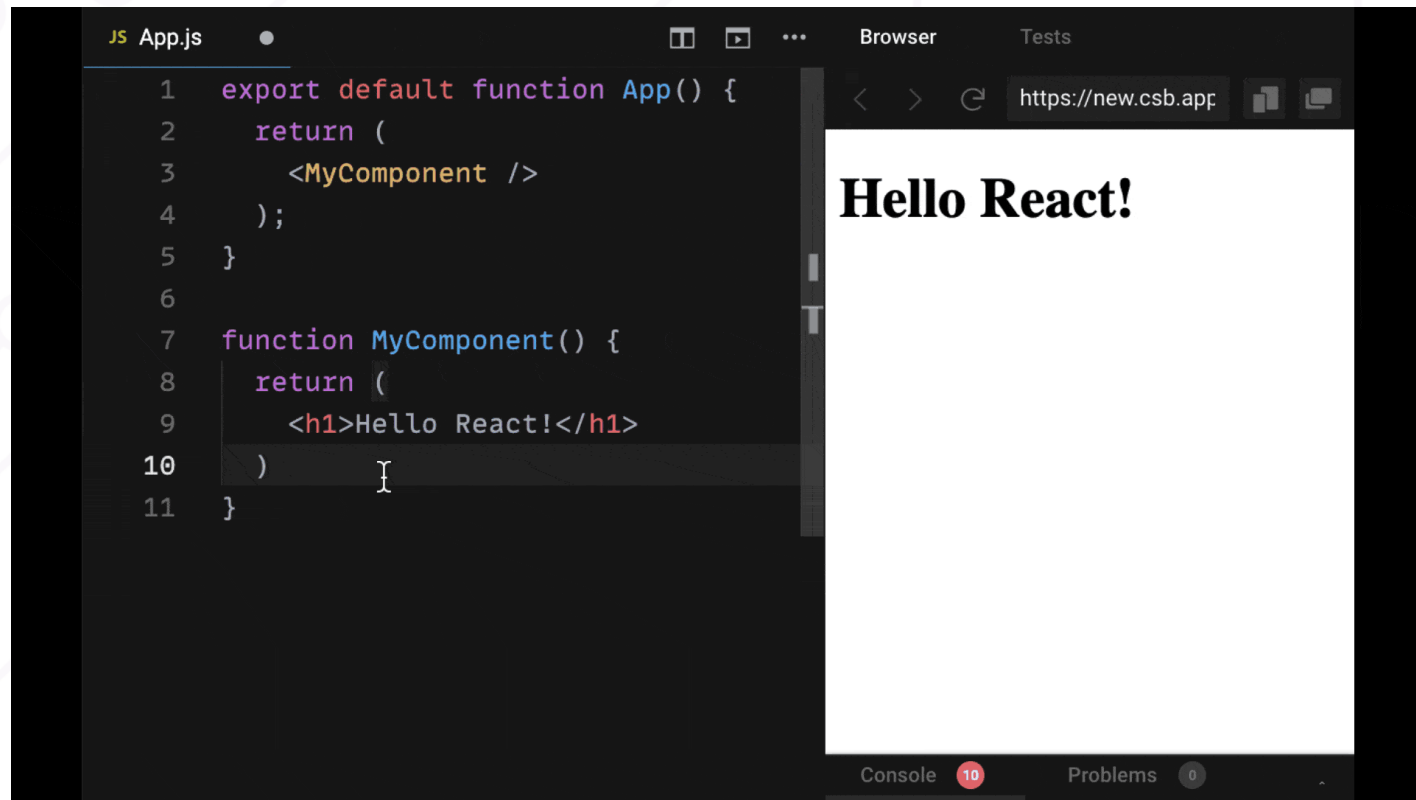
Se pueden crear componentes dentro de otros componentes y compartir props, sin embargo los props solo pueden ser enviados de padre a hijo.



```
<Hello name="Agnes" />  
  
{  
  name: 'Agnes'  
}  
  
props
```

The diagram illustrates the flow of props in React. It shows a parent component `<Hello name="Agnes" />` where the `name` prop is passed. Below it, a child component is shown with a destructured prop `name: 'Agnes'`. Arrows labeled `props` indicate the direction of data flow from the parent to the child.

Props



The image shows a code editor window with a file named 'App.js' and a browser window displaying the output of the code. The code defines a default export function 'App' which returns a JSX element containing a 'MyComponent'. 'MyComponent' is a function that returns an 'h1' element with the text 'Hello React!'. The browser window shows the rendered output: 'Hello React!'.

```
1 export default function App() {  
2   return (  
3     <MyComponent />  
4   );  
5 }  
6  
7 function MyComponent() {  
8   return (  
9     <h1>Hello React!</h1>  
10  )  
11 }
```

Browser: <https://new.csb.app>

Console: 10 Problems: 0

Estado (State)

Es la representación del conjunto de propiedades de un componente y sus valores actuales. El conjunto de propiedades y valores en un momento específico de la aplicación se llama Estado. En un contador el estado es cada vez que se da click a aumentar número de clics.



Hooks

Permiten agregar estado a componentes funcionales. Es una función especial que permite trabajar con estados en componentes funcionales. Es una manera mas fácil de entender.



Event Listener

Función que es ejecutada cuando ocurre un evento específico (también conocido como event handler)



JSX

(JavaScript XML)

JSX

Extensión de React para la sintaxis de JavaScript. Permite describir como se verán los componentes usando una estructura similar a HTML.

Con JSX se crean elementos. Los **elementos** son las unidades más pequeñas en React. Definen concretamente lo que se va a ver en la pantalla.



JSX

Esto podría interpretarse como un error.

```
const elemento = <h1>¡Hola, Mundo!</h1>;
```

JSX

```
1  import React from 'react';
2  import '../hojas-de-estilo/Contador.css';
3
4  function Contador({ numClicks }) {
5    return (
6      <div className="contador">
7        {numClicks}
8      </div>
9    );
10 }
11
12 export default Contador;
13
```



Atributos JSX

HTML

```
<label for="css">CSS</label>
```

JSX

```
<label htmlFor="css">CSS</label>
```


Estilos en JSX

```
1 const estiloDiv = {  
2   color: 'yellow',  
3   backgroundColor: 'black'  
4 };
```

```
<div style={estiloDiv}>¡Hola, Mundo!</div>
```

Estilos inline en JSX

```
<div style={{color: 'yellow'}}>¡Hola, Mundo!</div>
```

Objeto JavaScript



The diagram illustrates how a JavaScript object is used to define inline styles in JSX. A yellow-bordered box at the bottom contains the text 'Objeto JavaScript'. Two yellow arrows point upwards from this box to the curly braces in the JSX code snippet above, specifically to the parts of the style prop: `style={{color: 'yellow'}}`. This visualizes the concept that the object inside the braces is a JavaScript object that defines the style properties.

Componentes y Elementos

Elemento



Componente



```
return (  
  <div className="App">  
    <div className="freecodecamp-logo-contenedor">  
      <img  
        src={freeCodeCampLogo}  
        className="freecodecamp-logo"  
        alt="Logo de freeCodeCamp" />  
    </div>  
    <div className="contenedor-calculadora">  
      <Pantalla input={input} />  
      <div className="fila">  
        <Boton manejarClic={agregarInput}>1</Boton>  
        <Boton manejarClic={agregarInput}>2</Boton>  
        <Boton manejarClic={agregarInput}>3</Boton>  
        <Boton manejarClic={agregarInput}>+</Boton>  
      </div>  
    </div>  
  </div>  
)
```

Renderizar Componentes

Una vez se tienen los componentes deben mostrarse en un HTML. El elemento **div id='root'** es el elemento principal que va a contener a toda la aplicación y es manejado por **react DOM**

```
<div id="root">...</div>
```

Renderizar Componentes

React DOM es la librería que se encarga de renderizar los componentes de React para el navegador. Mientras que la biblioteca de **React**, a secas, es el motor de creación de componentes, hooks, sistema de props y estado.

```
1  const elemento = <h1>¡Hola, Mundo!</h1>;  
2  
3  ReactDOM.render(  
4    elemento, Elemento a renderizar  
5    document.getElementById('root') Contenedor para el elemento  
6  );
```

Renderizar Componentes

Para que **React DOM** pueda ser usado debe ser importado de la siguiente manera:

Paquete



```
import ReactDOM from 'react-dom';
```

Instalación React

Vite