



# React y Git

clase n° 8


\*REC



recordá  
poner a grabar la clase

# gitbash

Es una herramienta que nos permite usar Git desde una línea de comandos en Windows y simula un entorno Linux. Esto es útil para ejecutar comandos de Git y trabajar con repositorios de código. Pasos para instalar GitBash:

- **Descargar GitBash:**  
Visitar el sitio oficial  <https://git-scm.com/>  
Hacer clic en el botón de descarga correspondiente a tu sistema operativo.
- **Instalar GitBash:**  
Ejecutar el instalador descargado.  
Durante la instalación, elegir configuraciones predeterminadas si no tienes experiencia previa. Opcionalmente, puedes personalizar opciones como el editor de texto predeterminado o las herramientas a integrar.
- **Verificar la instalación:**  
Abrir GitBash y escribir el comando ***git --version***.  
Si aparece un número de versión, la instalación fue exitosa.

# react con vite

**Vite** es una herramienta que facilita la creación de aplicaciones web al proporcionar una configuración inicial optimizada. Vite es ideal para proyectos con React debido a su velocidad y sencillez.

## Instalación manual

En su proyecto, puede instalar la `vite` CLI usando:

 npm  Hilo  PNP  Bollo

```
$ npm install -D vite
```

intento

Y crea un `index.html` archivo como este:

```
<p>Hello Vite!</p>
```

html

Luego ejecute el comando CLI apropiado en su terminal:

 npm  Hilo  PNP  Bollo

```
$ npx vite
```

intento

El `index.html` testamento será entregado el `http://localhost:5173`.

# react con vite

## ■ Instalar Node.js:

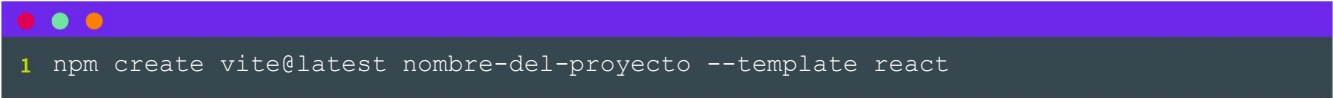
Ir al sitio oficial:  <https://nodejs.org/>

Descargar la versión recomendada para usuarios (LTS).

Instalar siguiendo los pasos del asistente, asegurándote de incluir la opción

## ■ Crear el proyecto con Vite:

Abrir GitBash o cualquier terminal. Ejecutar el siguiente comando:




```
1 npm create vite@latest nombre-del-proyecto --template react
```

Cambiar nombre-del-proyecto por el nombre deseado para tu aplicación.

## ■ Instalar dependencias:

Moverse al directorio del proyecto:



```
1 cd nombre-del-proyecto
```

# react con vite


Instalar las dependencias necesarias con:

A terminal window with a purple title bar and three colored window control buttons (red, green, orange) on the left. The terminal has a dark background and shows the command 'npm install' preceded by a yellow prompt character '1'.

```
1 npm install
```

## ■ Iniciar el servidor de desarrollo:

Ejecutar el comando:

A terminal window with a purple title bar and three colored window control buttons (red, green, orange) on the left. The terminal has a dark background and shows the command 'npm run dev' preceded by a yellow prompt character '1'.

```
1 npm run dev
```

Ver en el navegador la URL proporcionada (por ejemplo, *http://localhost:5173*) para verificar que la configuración inicial fue exitosa.

## ■ Finalizar configuración inicial:

Ahora que el proyecto está configurado, nos centraremos en el desarrollo con React, dejando Vite solo como herramienta para manejar el entorno.



**JSX** es una extensión de sintaxis para JavaScript que permite escribir estructuras declarativas similares a HTML o XML dentro del código JavaScript. Aunque se asemeja al HTML, JSX no es directamente comprensible por los navegadores; en su lugar, es transformado por herramientas como Babel en funciones de JavaScript que React utiliza para construir y manipular el DOM virtual.

### Características clave de JSX:

- Permite mezclar lógica de JavaScript con estructura visual.
- Cada etiqueta JSX debe estar correctamente cerrada.
- JSX se transpila a JavaScript puro para ser interpretado por el navegador.

Ejemplo:

```
1 const saludo = <h1>Hola, mundo!</h1>;
```

En este ejemplo, h1 es una etiqueta JSX que representa un encabezado HTML.

# componentes

Un **componente en React** es una pieza reutilizable de código que define cómo se debe ver y comportar una parte de la interfaz de usuario. Los componentes reciben datos (llamados **props**) y devuelven elementos visuales (usando JSX) que React renderiza en la página. Son como bloques de construcción que se combinan para crear aplicaciones completas.

## Tipos de componentes:

- **Componentes funcionales:** Simples y basados en funciones de JavaScript.

```
1 function Saludo() {  
2   return <h1>¡Hola, mundo</h1>;  
3 }
```

- **Componentes de clase:** (menos comunes en React moderno) Basados en clases de JavaScript.



# elementos

Un **elemento en React** es el bloque más pequeño y básico para construir interfaces. Representa una descripción de lo que se debe renderizar en la pantalla, como un nodo del DOM o un componente de React. Es un objeto inmutable que React utiliza para actualizar de manera eficiente el DOM. Se crea con JSX o con la función `React.createElement()`

Ejemplo:

```
1 function Boton() {  
2   return <button>Haz clic aquí</button>  
3 }
```

*En este ejemplo, <button> es un elemento HTML dentro del componente Boton.  
Los elementos pueden tener propiedades (props) para personalizarlos.*

# primer componente

Tu aplicación React comienza en un componente **root** (raíz), normalmente se crea automáticamente al iniciar un nuevo proyecto. La mayoría de las aplicaciones React usan componentes root. Esto significa que no solo usarás componentes para piezas reutilizables como botones, sino también para piezas más grandes como barras laterales, listas y, en última instancia, páginas completas. Los componentes son una forma práctica de organizar el código UI y el marcado, incluso si algunos de ellos solo se utilizan una vez.

## Crear un componente básico:

- **Crear un archivo para el componente:** Dentro de la carpeta `src`, crear un archivo llamado `Saludo.jsx`.

```
1 function Saludo() {  
2   return <h1>¡Hola, React!</h1>  
3 }  
4 export default Saludo;
```

# primer componente

- Importar y usar el componente: En el archivo `src/App.jsx`

```
1 import Saludo from './Saludo';
2 function App() {
3   return (
4     <div>
5       <Saludo />
6     </div>
7   );
8 }
9 export default App;
```

- Ejecutar y ver el resultado: Guardar todos los archivos.  
Ejecutar el comando `npm run dev` nuevamente.  
En el navegador, deberías ver el mensaje ¡Hola, React!.



revolución\*  
digital\_