



# Análisis y desarrollo de software.

## ID: 2692929



[www.sena.edu.co](http://www.sena.edu.co)

@SENAComunica

# Instructor

---

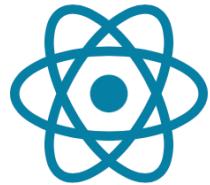
Diego Fernando Calderón Silva  
Correo: dfcalderon@sena.edu.co

# Conociendo React y

---

# Creando el primer proyecto

# ¿Qué es React-js?



## React

La biblioteca para interfaces de usuario web y nativas

**Crea interfaces de  
usuario a partir de  
componentes**

React te permite construir interfaces de usuario a partir de piezas individuales llamadas componentes. Crea tus propios componentes de React como `Thumbnail`, `LikeButton`, y `Video`. Luego combinalos para formar pantallas, páginas y aplicaciones.



# Instalemos:

- 1: <https://marketplace.visualstudio.com/items?itemName=burkeholland.simple-react-snippets>
- 2: <https://marketplace.visualstudio.com/items?itemName=dsznajder.es7-react-js-snippets>
- 3: <https://chromewebstore.google.com/detail/react-developer-tools/fmkadmapgofadoplbjbfkapdkoienihi?hl=es&authuser=1>
- 4: <https://chromewebstore.google.com/detail/redux-devtools/lmhkpmbekcpmknklioeibfkpmffibljd?hl=es>
- 5: <https://www.mongodb.com/try/download/shell>
- 6: <https://www.postman.com/downloads/>



# ¿Qué es Vite?

## Descripción general

Vite (palabra francesa que significa "rápido", pronunciada [/vit/](#) 🔊, como "veet") es una herramienta de compilación que tiene como objetivo proporcionar una experiencia de desarrollo más rápida y sencilla para proyectos web modernos. Consta de dos partes principales:

- Un servidor de desarrollo que proporciona [numerosas mejoras de funciones](#) con respecto a [los módulos ES nativos](#), por ejemplo, [reemplazo de módulo en caliente \(HMR\)](#) extremadamente rápido .
- Un comando de compilación que agrupa su código con [Rollup](#), preconfigurado para generar activos estáticos altamente optimizados para producción.

<https://vitejs.dev/guide/>



# Primer proyecto en Vite



Usamos:  
npm create vite@latest

```
[elingelindo@iMac-de-Diego Proyectos % npm create vite@latest primerProyectoReact
Need to install the following packages:
create-vite@5.2.1
[Ok to proceed? (y) y
✓ Package name: ... primer-proyecto-react
✓ Select a framework: > React
✓ Select a variant: > JavaScript + SWC

Scaffolding project in /Users/elingelindo/Library/Mobile Documents/com~apple~CloudDocs/Sena/2024/Portafolio2024-1/Portafolio AD
SO ID 2692929/Presentaciones clases/react/Proyectos/primerProyectoReact...
```

Done. Now run:

```
cd primerProyectoReact
npm install
npm run dev
```

# Primer proyecto en Vite

EXPLORADOR

PROYECTOS primerProyectoReact > package.json > ...

primerProyectoReact

- public vite.svg
- src
  - assets
  - # App.css
  - App.jsx
  - # index.css
  - main.jsx
  - .eslintrc.cjs
  - .gitignore
  - index.html
- { package.json
- README.md
- vite.config.js

```
1  {
2    "name": "primer-proyecto-react",
3    "private": true,
4    "version": "0.0.0",
5    "type": "module",
6    // Depurar
7    "scripts": {
8      "dev": "vite",
9      "build": "vite build",
10     "lint": "eslint . --ext js,jsx --report-unused-disable-directives --max-warnings 0",
11     "preview": "vite preview"
12   },
13   "dependencies": {
14     "react": "^18.2.0",
15     "react-dom": "^18.2.0"
16   },
17   "devDependencies": {
18     "@types/react": "^18.2.56",
19     "@types/react-dom": "^18.2.19",
20     "@vitejs/plugin-react-swc": "^3.5.0",
21     "eslint": "^8.56.0",
22     "eslint-plugin-react": "^7.33.2",
23     "eslint-plugin-react-hooks": "^4.6.0",
24     "eslint-plugin-react-refresh": "^0.4.5",
25     "vite": "^5.1.4"
26   }
27 }
```

```
"scripts": {
  "dev": "vite",
  "build": "vite build",
  "lint": "eslint . --ext js,jsx --report-unused-disable-directives --max-warnings 0",
  "preview": "vite preview"
},
```

# Primer proyecto en Vite



main.jsx × < index.html

```
primerProyectoReact > src > main.jsx
1 import React from 'react'
2 import ReactDOM from 'react-dom/client'
3 import App from './App.jsx'
4 import './index.css'
5
6 ReactDOM.createRoot(document.getElementById('root')).render(
7   <React.StrictMode>
8     | <App />
9   </React.StrictMode>,
10 )
```

main.jsx < index.html ×

```
primerProyectoReact > index.html
1 <!doctype html>
2 <html lang="en">
3   <head>
4     <meta charset="UTF-8" />
5     <link rel="icon" type="image/svg+xml" href="/vite.svg" />
6     <meta name="viewport" content="width=device-width, initial-scale=1.0" />
7     <title>Vite + React</title>
8   </head>
9   <body>
10    <div id="root"></div>
11    <script type="module" src="/src/main.jsx"></script>
12  </body>
13</html>
```

# Primer proyecto en Vite



App es un componente

```
main.jsx          App.jsx      index.html
primerProyectoReact > src > App.jsx > ...
1   import { useState } from 'react'
2   import reactLogo from './assets/react.svg'
3   import viteLogo from '/vite.svg'
4   import './App.css'
5
6   function App() {
7     const [count, setCount] = useState(0)
8
9     return (
10    <>
11    <div>
12      <a href="https://vitejs.dev" target="_blank">
13        <img src={viteLogo} className="logo" alt="Vite logo" />
14      </a>
15      <a href="https://react.dev" target="_blank">
16        <img src={reactLogo} className="logo react" alt="React logo" />
17      </a>
18    </div>
19    <h1>Vite + React</h1>
20    <div className="card">
21      <button onClick={() => setCount((count) => count + 1)}>
22        count is {count}
23      </button>
24      <p>
25        Edit <code>src/App.jsx</code> and save to test HMR
26      </p>
27    </div>
28    <p className="read-the-docs">
29      Click on the Vite and React logos to learn more
30    </p>
31  </>
32)
33}
34
35 export default App
```

# Resumen de lo visto hasta ahora



El index.html tiene un div root el cual es tomado por main.jsx el cual renderiza y una de las cosas que renderiza es el componente creado en App.jsx

# ¿Qué jsx?



Jsx, es una extensión de sintaxis propia de react la cual combina html con JavaScript para poder hacer componentes dinámicos y reutilizables.



# Primer proyecto en Vite



Instalemos el proyecto usando:  
npm install

```
elingelindo@iMac-de-Diego primerProyectoReact % npm install  
  
added 221 packages, and audited 222 packages in 23s  
  
95 packages are looking for funding  
  run `npm fund` for details  
  
found 0 vulnerabilities
```

# Primer proyecto en Vite

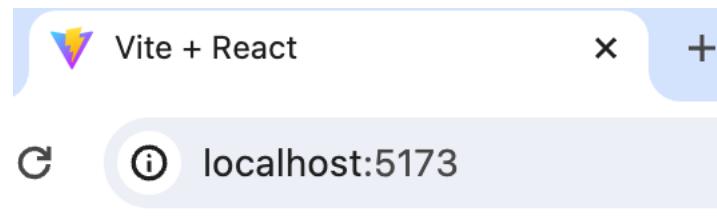
Corramos el proyecto con:  
npm run dev

```
VITE v5.1.4  ready in 308 ms
→ Local:  http://localhost:5173/
→ Network: use --host to expose
→ press h + enter to show help
```

# Primer proyecto en Vite



¿Dónde está el ícono y el título de la página?



¿Cómo puedo cambiarlo?



# Componentes

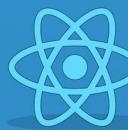
## ¿Qué son los componentes?

Los componentes son bloques de construcción de la interfaz de usuario lo cual encapsula la lógica y la estructura visual de una parte de todo lo que ve el usuario, es decir, en lugar de tener una interfaz grande y compleja, la dividimos en pequeños bloques para finalmente conformar la interfaz de usuario completa.



# Componentes

## COMPONENTES EN REACT



EDteam



Los componentes son piezas de código reutilizables.

P.ej: El componente <Cupcake/> nos permite crear cupcakes de diferentes sabores y colores.



```
<Cupcake color="rosa" sabor="vainilla" />
```

```
<Cupcake color="azul" sabor="chocolate" />
```



Color y sabor son propiedades.  
"PROPS"

Los componentes pueden tener **estado**. Valores que pueden cambiar. P.ej: {vendido: true}

### Se pueden escribir de 2 formas:

Función

```
function Cupcake({color, sabor}) {  
  return (  
    <p  
    className={color}>{sabor}</p>  
  )  
}
```

Cada vez **se usa menos**.

Clase

```
class Cupcake extends Component {  
  render() {  
    return (  
      <p  
      className={this.props.color}>  
        { this.props.sabor }  
      </p>  
    )  
  }  
}
```

**¡NUEVO CURSO!**

Domina React con EDteam en:

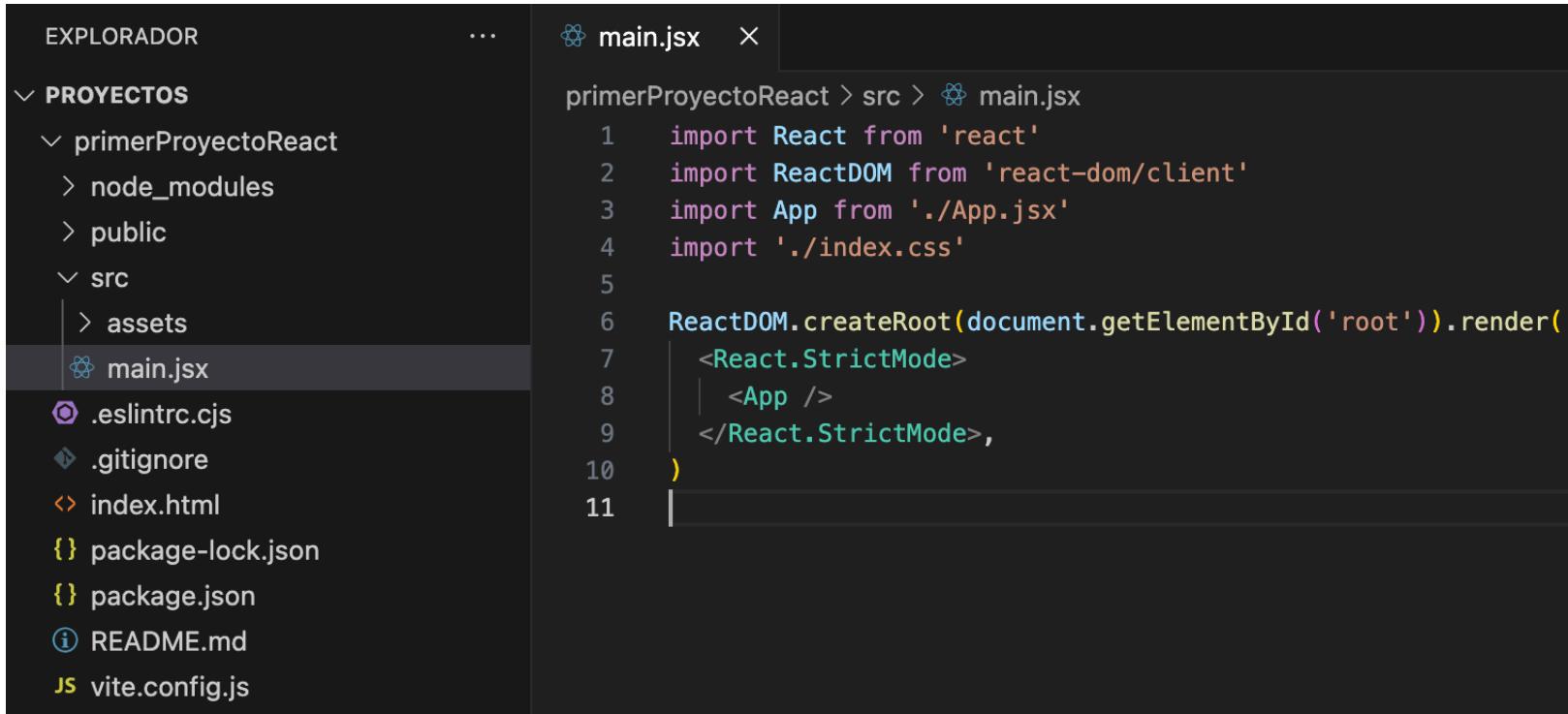


[ed.team/cursos/react](http://ed.team/cursos/react)



# Componentes

Vamos a re organizar el proyecto así:



The image shows a code editor interface with two main sections: a file explorer on the left and a code editor on the right.

**EXPLORADOR**

- PROYECTOS
  - primerProyectoReact
    - node\_modules
    - public
    - src
      - assets
      - main.jsx
  - .eslintrc.cjs
  - .gitignore
  - index.html
  - package-lock.json
  - package.json
  - README.md
  - vite.config.js

... (ellipsis)

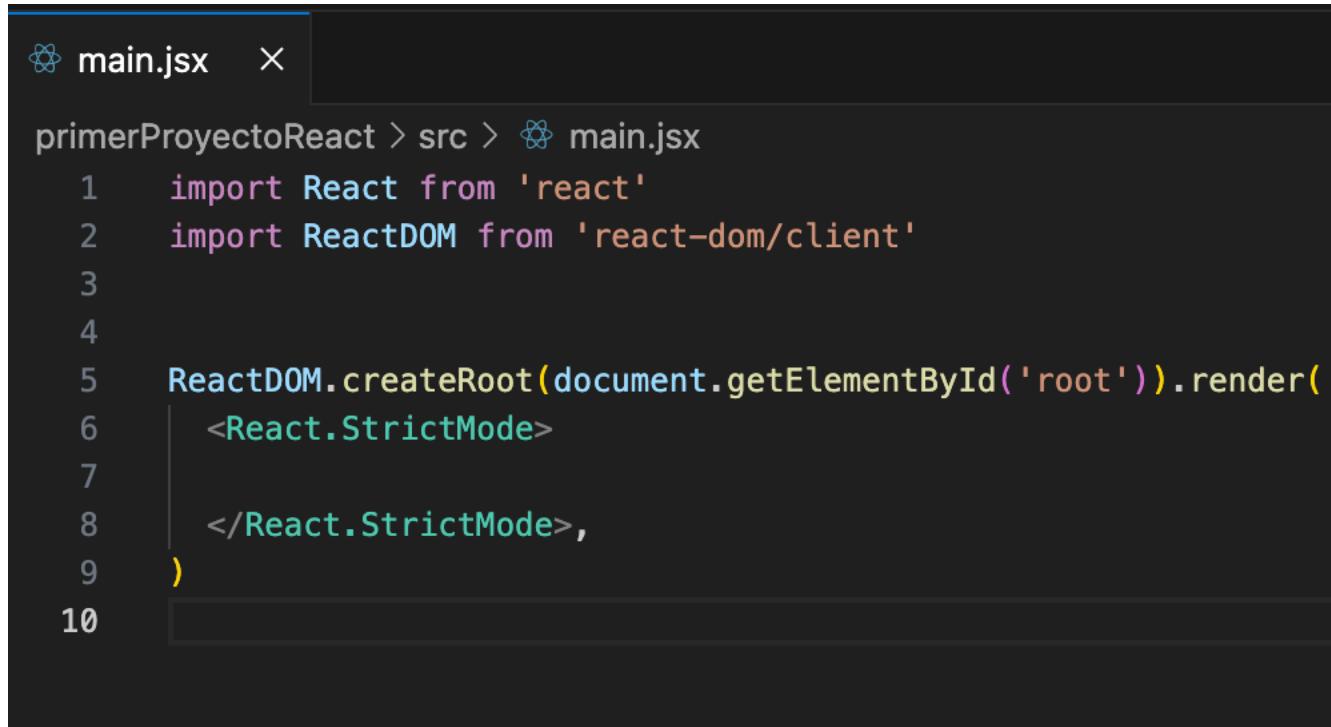
**main.jsx**

```
primerProyectoReact > src > main.jsx
1 import React from 'react'
2 import ReactDOM from 'react-dom/client'
3 import App from './App.jsx'
4 import './index.css'
5
6 ReactDOM.createRoot(document.getElementById('root')).render(
7   <React.StrictMode>
8     | <App />
9   </React.StrictMode>,
10 )
11 |
```

Eliminamos todo, menos el main.jsx incluyendo la carpeta public

# Componentes

Dentro de main eliminamos las importaciones de app e index

A screenshot of a code editor showing the main.jsx file. The file contains the following code:

```
main.jsx  X
primerProyectoReact > src > main.jsx
1 import React from 'react'
2 import ReactDOM from 'react-dom/client'
3
4
5 ReactDOM.createRoot(document.getElementById('root')).render(
6   <React.StrictMode>
7
8     </React.StrictMode>,
9
10 )
```

The code imports React and ReactDOM, creates a root element, and renders a StrictMode component.

Nuestro proyecto en la web se vera blanco.

# Componentes

Ahora creamos nuestro primer componente:

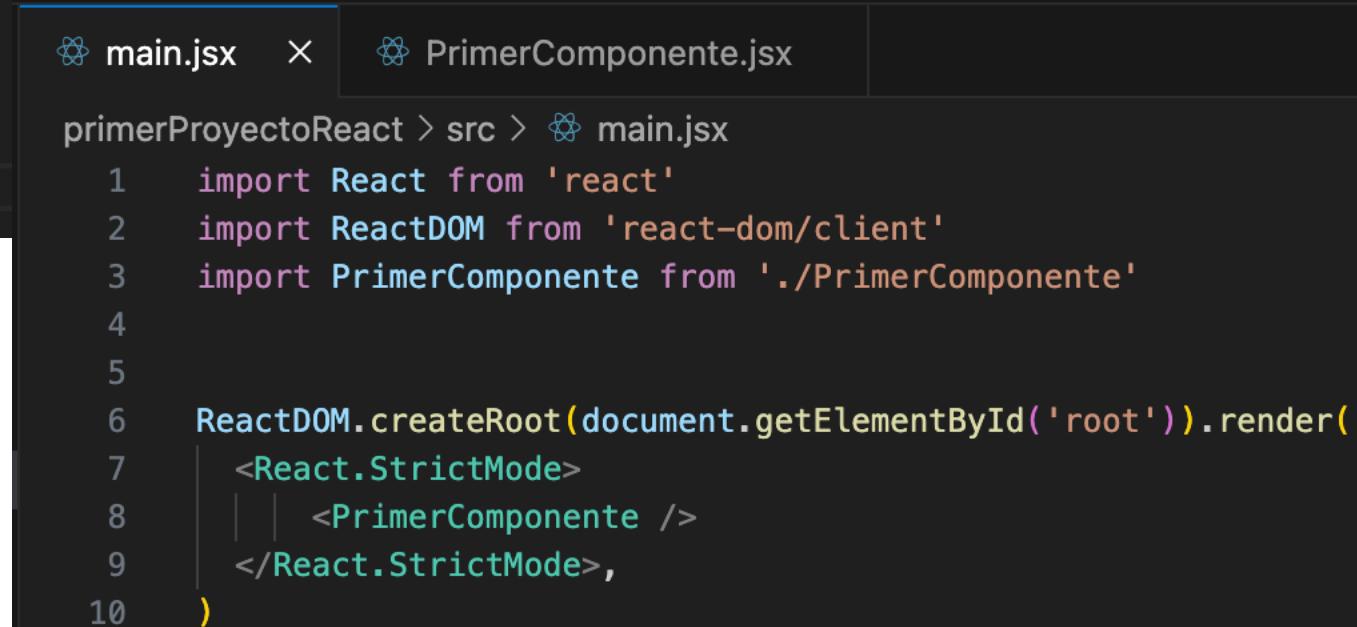
Para esto crearemos un script llamado: PrimerComponente.jsx dentro de src configurado así: con el snippet ffc creamos el primer componente



The screenshot shows a dark-themed VS Code interface. A new file named 'PrimerComponente.jsx' is being created in the 'src' directory of a project named 'primerProyectoReact'. The code editor displays the following content:

```
function PrimerComponente() {
  return ( <h1>Hola mundo</h1> );
}

export default PrimerComponente;
```



The screenshot shows the 'main.jsx' file in the same project structure. It imports the 'PrimerComponente' component from the 'src' directory and uses it within a React root element.

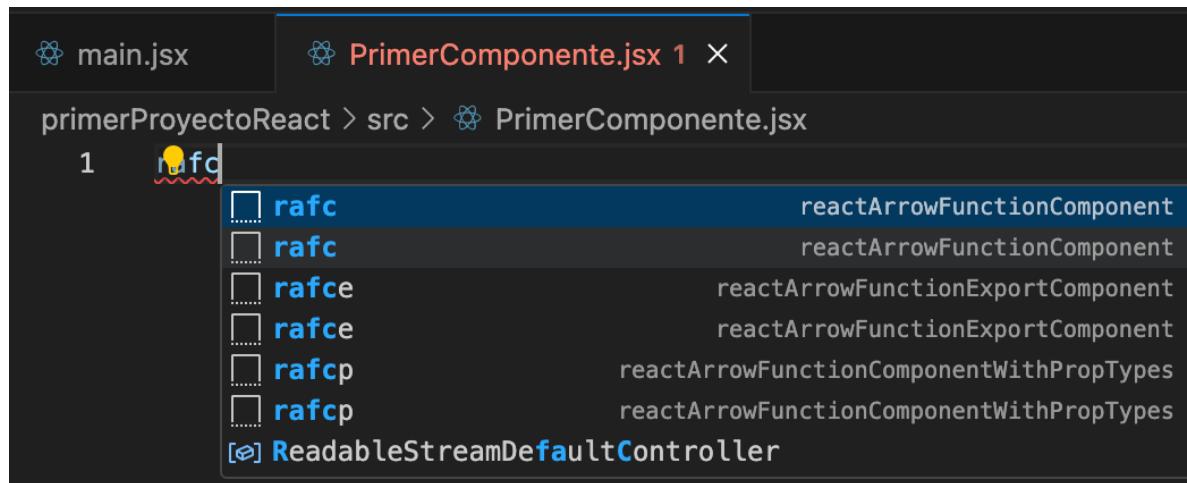
```
import React from 'react'
import ReactDOM from 'react-dom/client'
import PrimerComponente from './PrimerComponente'

ReactDOM.createRoot(document.getElementById('root')).render(
  <React.StrictMode>
    <PrimerComponente />
  </React.StrictMode>,
)
```

<https://github.com/r5n-dev/vscode-react-javascript-snippets/blob/HEAD/docs/Snippets.md>

# Componentes

Siguiendo el repo de snippets <https://github.com/r5n-dev/vscode-react-javascript-snippets/blob/HEAD/docs/Snippets.md> nos encontramos con **rafc** el cual nos crea un componente como función completo.



The screenshot shows the VS Code interface with two files open: 'main.jsx' and 'PrimerComponente.jsx'. The 'main.jsx' file contains the code: 'import React from 'react'' followed by a snippet placeholder. A snippet dropdown menu is open at the cursor position, showing several options starting with 'rafc'. The first option, 'rafc', is highlighted in blue. Other options include 'rafc', 'rafce', 'rafce', 'rafc', 'rafc', and 'ReadableStreamDefaultController'. The 'PrimerComponente.jsx' file is shown in the background.



The screenshot shows the 'PrimerComponente.jsx' file in VS Code. The code is as follows:

```
1 import React from 'react'
2
3 const PrimerComponente = () => {
4     return (
5         <div>
6
7             </div>
8     )
9 }
10
11 export default PrimerComponente
12 |
```

# Componentes

Ahora, si por algún motivo se crea con la exportación diferente, así mismo se debe importar en el main de forma diferente, por ej:



```
main.jsx
PrimerComponente.jsx 1 ×

primerProyectoReact > src > PrimerComponente.jsx >
1  import React from 'react'
2
3  export const PrimerComponente = () => {
4    return (
5      <div>
6        | Hola mundo
7      </div>
8    )
9  }
```



```
main.jsx × PrimerComponente.jsx 1

primerProyectoReact > src > main.jsx
1  import React from 'react'
2  import ReactDOM from 'react-dom/client'
3  import {PrimerComponente} from './PrimerComponente'
4
5
6  ReactDOM.createRoot(document.getElementById('root')).render(
7    <React.StrictMode>
8      <PrimerComponente />
9    </React.StrictMode>,
10  )
```

La importación debe contener {} como si fuera una variable, mientras que al exportar se hace directamente desde la constante.

# Variables en JSX



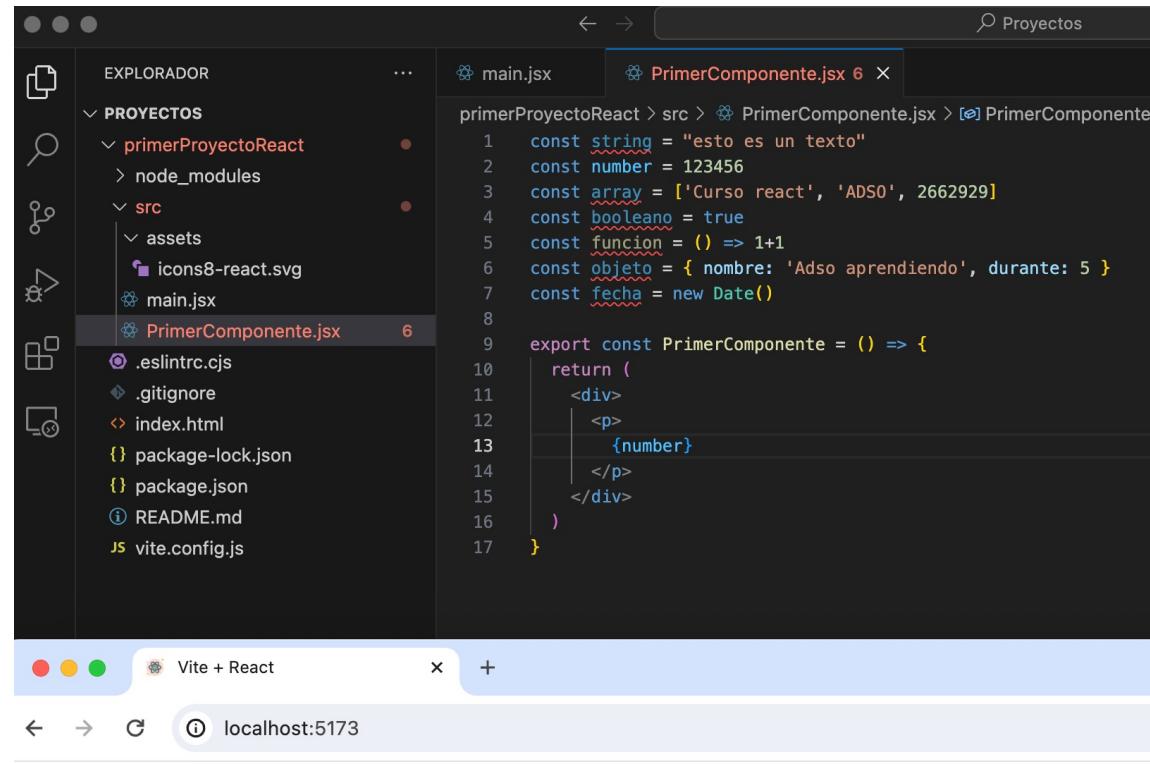
Recordemos que jsx es una fusión de html, con xml, fusión con JavaScript y básicamente se mira como el componente que acabamos de crear.

Para declarar variables loaremos inicialmente fuera de la función así:

```
const string = "esto es un texto"
const number = 123456
const array = ['Curso react', 'ADSO', 2662929]
const booleano = true
const funcion = () => 1+1
const objeto = { nombre: 'Adso aprendiendo', durante: 5 }
const fecha = new Date()
```

# Variables en JSX

Y para utilizar las variables lo haremos dentro de la función en unas etiquetas <p> por ejemplo:



The screenshot shows a development environment with the following details:

- Project Structure:** The project is named "primerProyectoReact". It contains a "src" folder which includes an "assets" folder with "icons8-react.svg" and a "main.jsx" file.
- Code Editor:** The "PrimerComponente.jsx" file is open, showing the following code:

```
primerProyectoReact > src > PrimerComponente.jsx > [e] PrimerComponente
1 const string = "esto es un texto"
2 const number = 123456
3 const array = ['Curso react', 'ADSO', 2662929]
4 const booleano = true
5 const funcion = () => 1+1
6 const objeto = { nombre: 'Adso aprendiendo', durante: 5 }
7 const fecha = new Date()

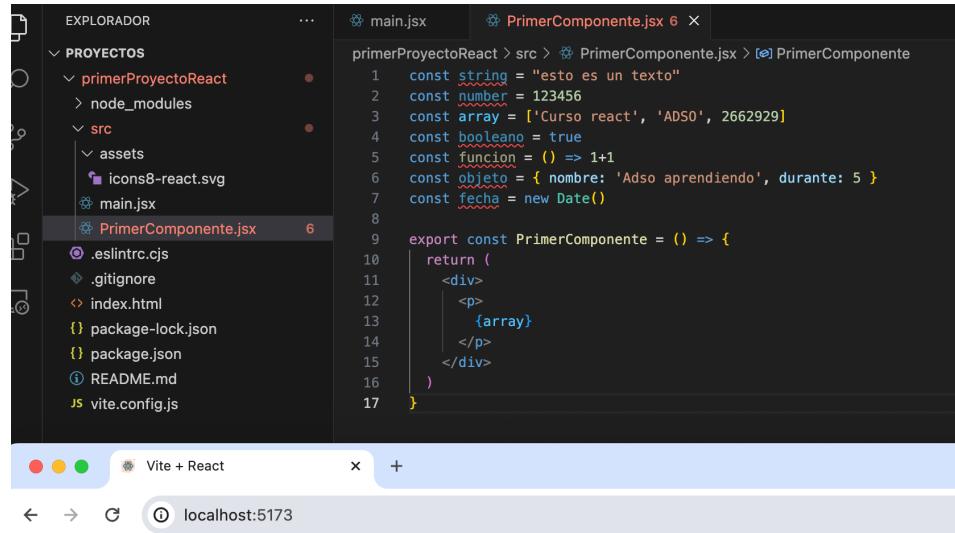
8
9 export const PrimerComponente = () => {
  return (
    <div>
      <p>
        {number}
      </p>
    </div>
  )
}
```
- Browser Preview:** The browser window shows the output of the component. The value of the variable "number" is displayed as "123456".

123456

Si se fijan la forma de llamar la variable es como si fuera en javaScript

# Variables en JSX

Ahora, que pasa si ponemos el array:



The screenshot shows a development setup with two code editors and a browser preview.

**Code Editors:**

- main.jsx:** Contains standard variable declarations (string, number, boolean, function, object, date).
- PrimerComponente.jsx:** Contains a single-line JSX return statement where the array is passed directly to the `<array>` tag.

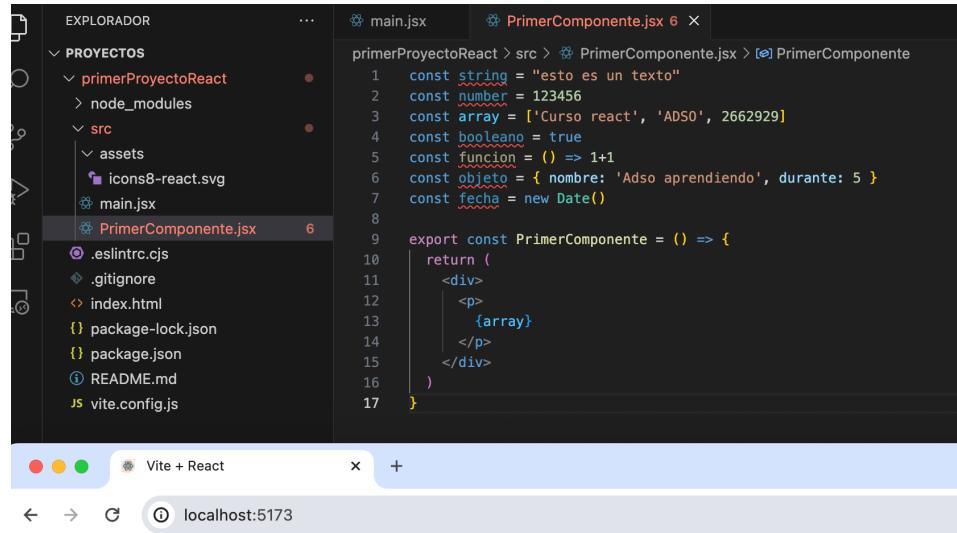
**Browser Preview:**

The browser window displays the rendered output: "Curso reactADSO2662929".

Nos sale el contenido completo, por lo que deberíamos pasar los parámetros correctos y analicemos un poco la consola.

# Variables en JSX

Ahora, que pasa si ponemos el array:



The screenshot shows a code editor interface with a dark theme. On the left is the file explorer, showing a project structure with files like `main.jsx`, `PrimerComponente.jsx`, `.eslintrc.cjs`, `.gitignore`, `index.html`, `package-lock.json`, `package.json`, `README.md`, and `vite.config.js`. The right side has two tabs open: `main.jsx` and `PrimerComponente.jsx`. The `PrimerComponente.jsx` tab is active, displaying the following code:

```
const string = "esto es un texto"
const number = 123456
const array = ['Curso react', 'ADSO', 2662929]
const booleano = true
const funcion = () => 1+1
const objeto = { nombre: 'Adso aprendiendo', durante: 5 }

export const PrimerComponente = () => {
  return (
    <div>
      <p>
        {array}
      </p>
    </div>
  )
}
```

The browser tab at the bottom shows the URL `localhost:5173`.

Nos sale el contenido completo, por lo que deberíamos pasar los parámetros correctos y analicemos un poco la consola.

¿Cómo hago que muestre el resultado de la const función?

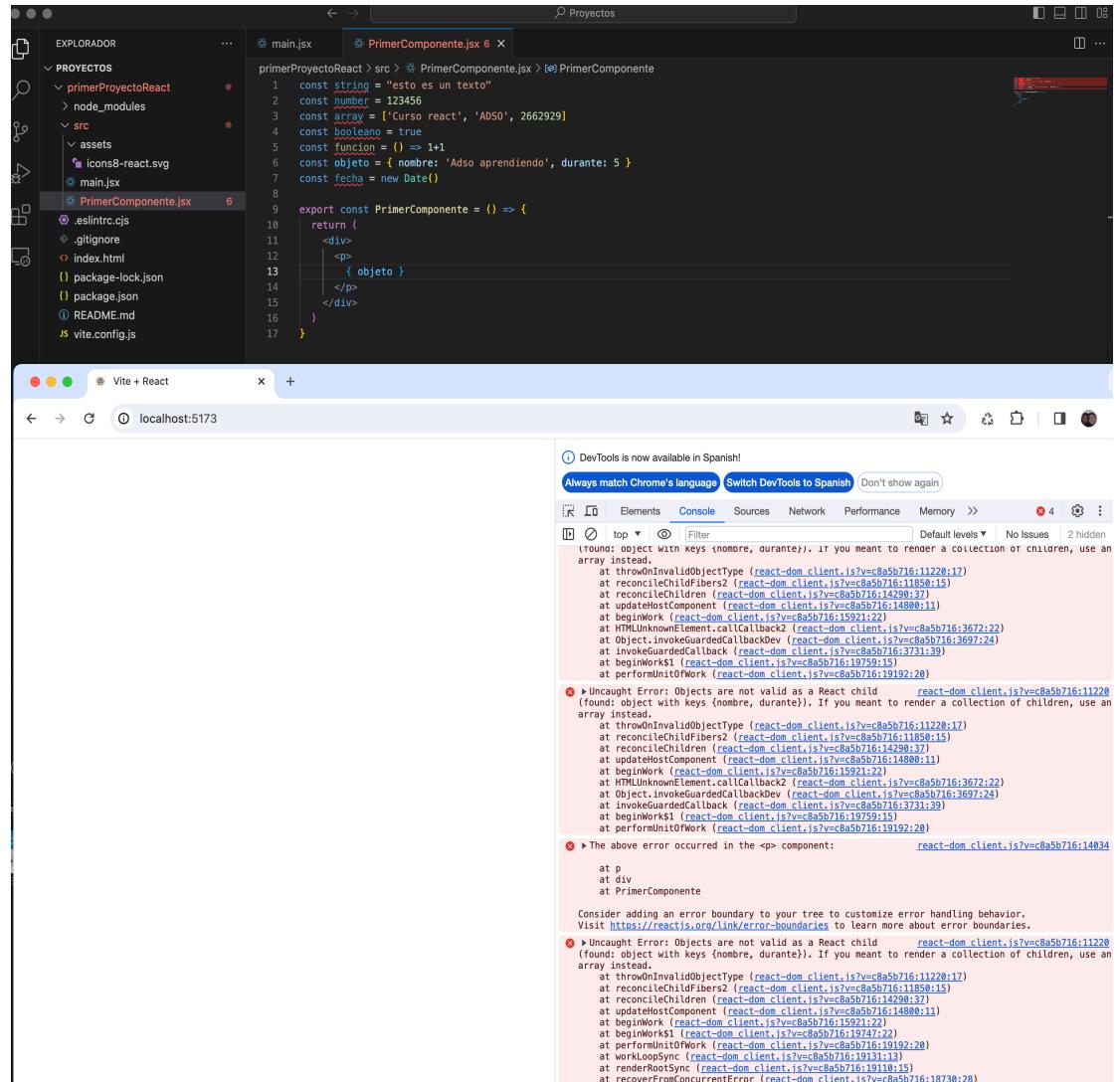
# Variables en JSX

Luego de pasar todas las variables vamos a centrarnos en el objeto:

Nos dice que los objetos no son hijos de react, por lo tanto no deberíamos pasarsel ningn objeto.

Ya que en php creamos una instancia con new, será que en react pasa lo mismo?

Descubranlo...



The screenshot shows a development setup with several windows open:

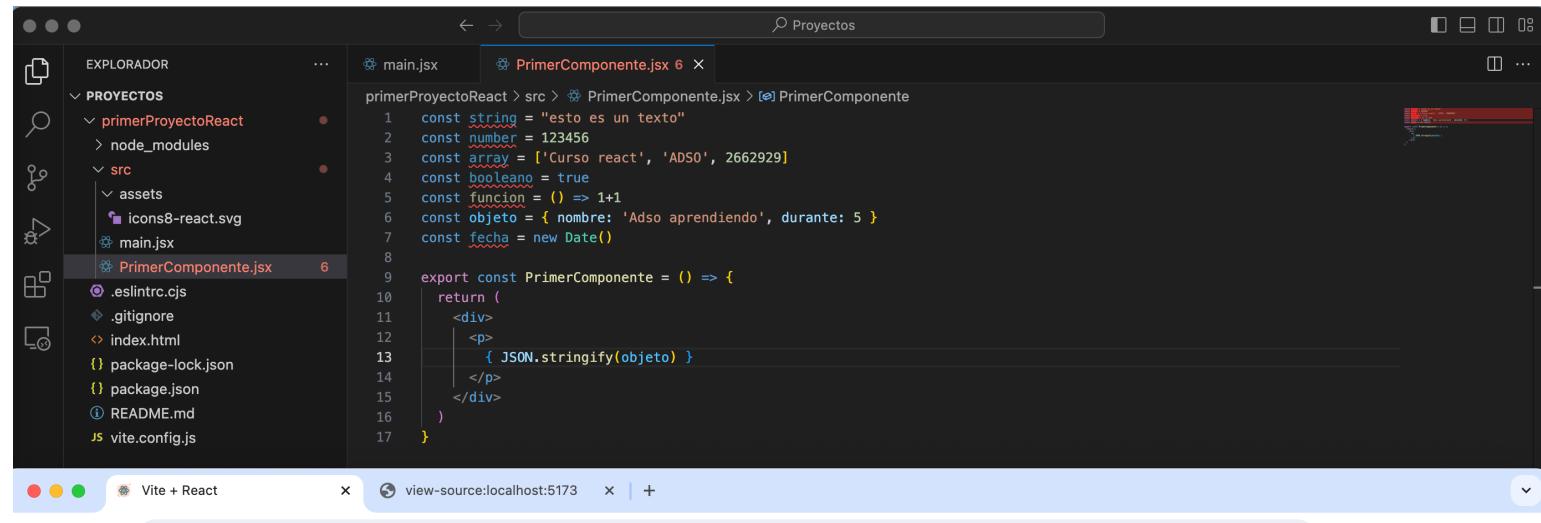
- An Explorer sidebar showing a project structure with files like `main.jsx`, `PrimerComponente.jsx`, and `index.html`.
- A code editor window displaying `PrimerComponente.jsx` containing the following code:

```
const string = "esto es un texto"
const number = 123456
const array = ['Curso react', 'ADS01', 2662929]
const booleano = true
const funcion = () => 1+1
const objeto = { nombre: 'Adso aprendiendo', durante: 5 }
const fecha = new Date()

export const PrimerComponente = () => {
  return (
    <div>
      <p>{string}</p>
      <p>{number}</p>
      <p>{array}</p>
      <p>{booleano}</p>
      <p>{funcion}</p>
      <p>{objeto}</p>
    </div>
  )
}
```
- A browser window at `localhost:5173` showing the rendered output of the component.
- DevTools open in the browser, showing the component tree and various inspection tools.

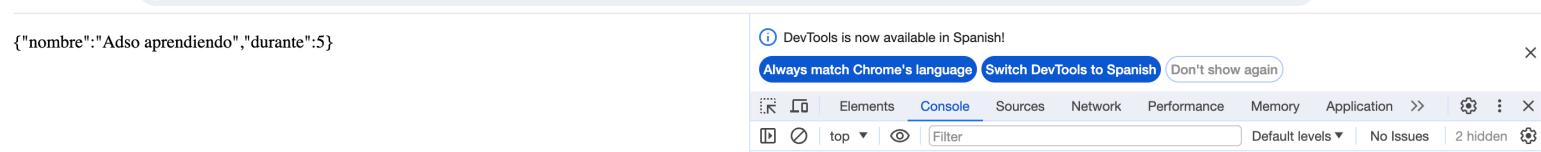
# Variables en JSX

Tal como se logra observar, es imposible mostrar objetos directamente en react, pero como meta piensa en todo, hace uso de JSON para lograr trabajar los objetos así:



```
const string = "esto es un texto"
const number = 123456
const array = ['Curso react', 'ADSO', 2662929]
const booleano = true
const funcion = () => 1+1
const objeto = { nombre: 'Adso aprendiendo', durante: 5 }
const fecha = new Date()

export const PrimerComponente = () => {
  return (
    <div>
      <p>
        { JSON.stringify(objeto) }
      </p>
    </div>
  )
}
```



```
{"nombre": "Adso aprendiendo", "durante": 5}
```

DevTools is now available in Spanish!  
Always match Chrome's language [Switch DevTools to Spanish](#) [Don't show again](#)

# Variables en JSX

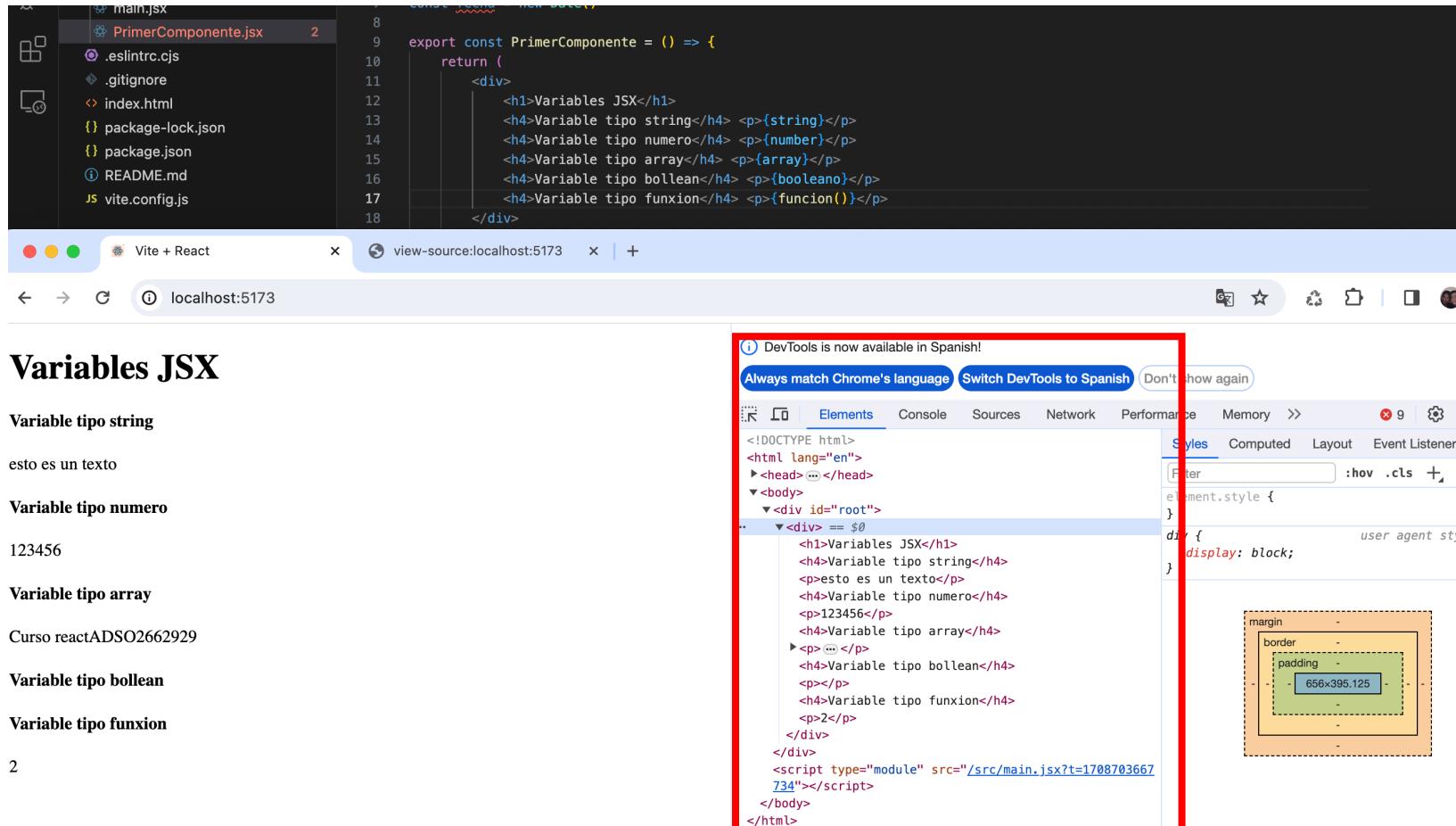


¿Qué pasa si escribo todas las variables una debajo de la otra?

```
export const PrimerComponente = () => {
  return (
    <h1>Variables en JSX</h1>
    <h4>Variable tipo String:</h4> <p>{string}</p>
    <h4>Variable tipo number:</h4> <p>{number}</p>
    <h4>Variable tipo array:</h4> <p>{array}</p>
    <h4>Variable tipo boolean:</h4> <p>{boolean}</p>
    <h4>Variable tipo funcion:</h4> <p>{string}</p>
    <h4>Variable tipo String:</h4> <p>{funcion()}</p>
  )
}
```

# Variables en JSX

Si observamos al escribir las variables solas, react nos indica que es imposible mostrarlas ya que solo funcionaran si tienen un padre único, y en este caso, solo hay hermanos. Por lo tanto, se deben contener dentro de un div pero...



The screenshot shows a development setup with a file tree on the left and code editors on the right.

**File Tree:**

- main.jsx
- PrimerComponente.jsx
- .eslintrc.cjs
- .gitignore
- index.html
- package-lock.json
- package.json
- README.md
- vite.config.js

**Code Editor (main.jsx):**

```
8
9 export const PrimerComponente = () => {
10   return (
11     <div>
12       <h1>Variables JSX</h1>
13       <h4>Variable tipo string</h4> <p>{string}</p>
14       <h4>Variable tipo numero</h4> <p>{number}</p>
15       <h4>Variable tipo array</h4> <p>[array]</p>
16       <h4>Variable tipo boolean</h4> <p>{boolean}</p>
17       <h4>Variable tipo función</h4> <p>{funcion()}</p>
18     </div>
)
```

**Code Editor (PrimerComponente.jsx):**

```
8
9 export const PrimerComponente = () => {
10   return (
11     <div>
12       <h1>Variables JSX</h1>
13       <h4>Variable tipo string</h4> <p>{string}</p>
14       <h4>Variable tipo numero</h4> <p>{number}</p>
15       <h4>Variable tipo array</h4> <p>[array]</p>
16       <h4>Variable tipo boolean</h4> <p>{boolean}</p>
17       <h4>Variable tipo función</h4> <p>{funcion()}</p>
18     </div>
)
```

**Browser View (localhost:5173):**

## Variables JSX

Variable tipo string  
esto es un texto

Variable tipo numero  
123456

Variable tipo array  
Curso reactADSO2662929

Variable tipo boolean

Variable tipo función

2

**DevTools Elements Tab:**

The DevTools Elements tab shows the rendered HTML structure. A red box highlights the element tree under the root div with id="root".

```
<!DOCTYPE html>
<html lang="en">
  <head> ... </head>
  <body>
    <div id="root">
      <h1>Variables JSX</h1>
      <h4>Variable tipo string</h4>
      <p>esto es un texto</p>
      <h4>Variable tipo numero</h4>
      <p>123456</p>
      <h4>Variable tipo array</h4>
      <p>Curso reactADSO2662929</p>
      <h4>Variable tipo boolean</h4>
      <p></p>
      <h4>Variable tipo función</h4>
      <p>2</p>
    </div>
  </body>
</html>
```

**DevTools Styles Tab:**

The DevTools Styles tab shows the computed styles for the elements. A red box highlights the styles for the root div.

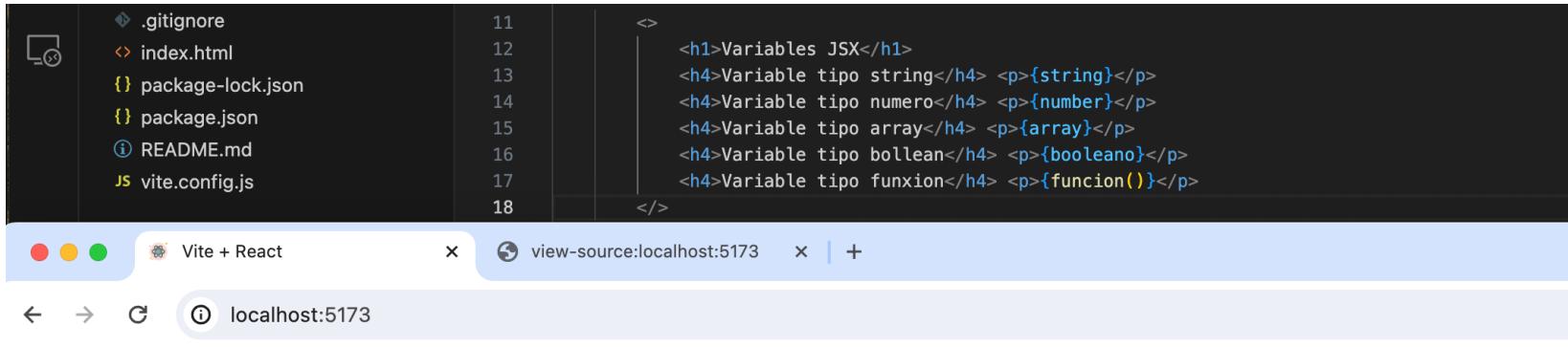
```
div {
  display: block;
}
```

**Element Inspector:**

An element from the rendered page is selected, showing its bounding box and dimensions: 656x395.125.

# Variables en JSX

La forma de eliminar este doble div, es simplemente eliminar el fragment del div y con esto...



The screenshot shows a browser window with the address bar set to 'localhost:5173'. The page content is titled 'Variables JSX' and displays several examples of JSX code. The developer tools DevTools panel is open, showing the 'Elements' tab with the DOM tree. The root element is a div with id='root'. Inside this div, there are several other elements including h1, h4, p, and script tags. The browser status bar at the bottom indicates 'Vite + React'.

## Variables JSX

### Variable tipo string

esto es un texto

### Variable tipo numero

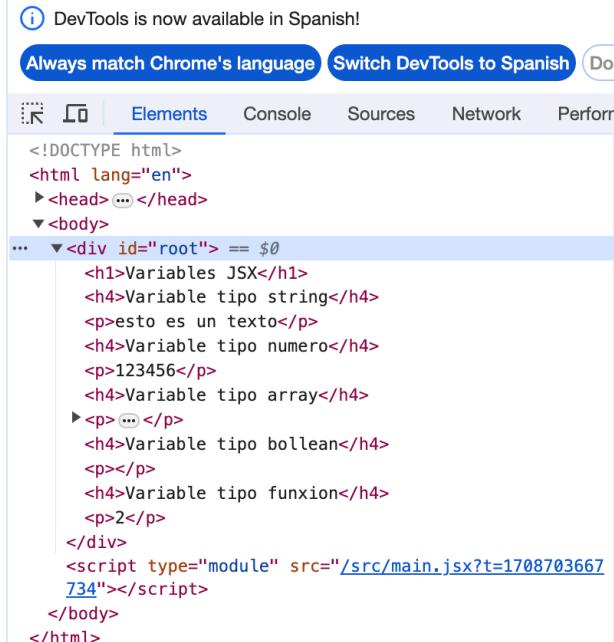
123456

### Variable tipo array

Curso reactADSO2662929

### Variable tipo boolean

### Variable tipo funxion



The screenshot shows the browser's developer tools DevTools panel with the 'Elements' tab selected. It displays the DOM structure of the page, which includes the root div with id='root'. Inside this div, there are various elements like h1, h4, p, and script tags. The browser status bar at the bottom indicates 'view-source:localhost:5173'.



# G R A C I A S

Línea de atención al ciudadano: 01 8000 910270

Línea de atención al empresario: 01 8000 910682



[www.sena.edu.co](http://www.sena.edu.co)