Manejo de variables y propiedades en un componente

En esta sección vamos a definir un componente con propiedades para pasar valores entre componentes; además se utilizara la desestructuración de propiedades y las funciones de flecha para escribir un código más limpio y legible.

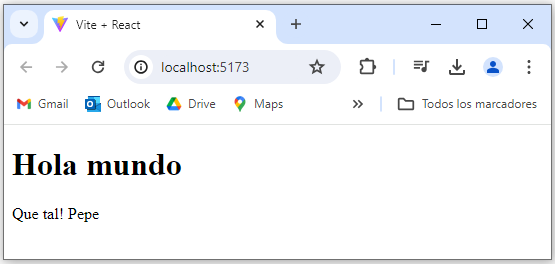
Variables internas en componentes

Los componentes en React pueden ser funciones (conocidos como componentes funcionales) que reciben propiedades (props) a través de sus parámetros.

Ten en cuenta que cuando se utiliza un componente en la aplicación, como el **HelloWorld.jsx**, se define en el método **render** como una etiqueta HTML **<HelloWorld />** no como una función.

Para incluir variables que se puedan leer desde cualquier parte del componente, solamente se definen como variables JavaScript dentro del cuerpo del componente. Por ejemplo, en el componente **HelloWorld.jsx** se podría imprimir el valor de la variable **name** en el navegador al insertar el nombre de la variable entre llaves en la porción JSX.





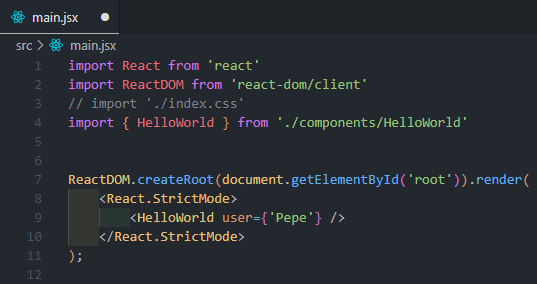
Es importante saber que los componentes pueden reutilizarse y anidarse dentro del cuerpo de otras funciones de retorno (específicamente dentro del **return**). Sin embargo, siempre necesitan tener un elemento raíz.

**Nota:** No se puede utilizar la sintaxis de Template String (plantilla de caracteres) en la parte HTML del componente para llamar a la variable: **${name}**.

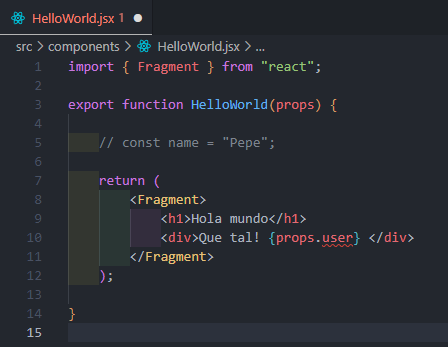
Pasar valores entre componentes

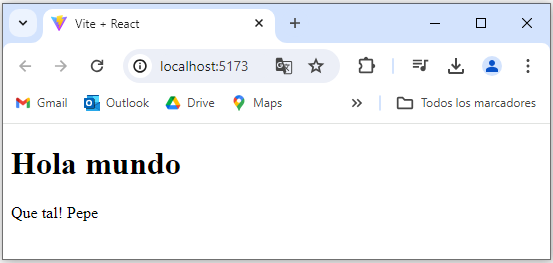
Para pasar un valor desde un componente padre a un componente hijo se utiliza las propiedades (props) definidas en el componente.

En el archivo main.jsx, donde se llama al componente **HelloWorld**, se pueden pasar valores al componente hijo definiendo propiedades. Por ejemplo: **user** es una prop que se está pasando al componente **HelloWorld**.



Dentro del componente **HelloWorld.jsx**, se define un objeto **props** como un parámetro de la función del componente. Este objeto contiene todas las propiedades que se pasan al componente. Luego, para acceder al valor pasado desde el componente padre, se utiliza **props.user**.





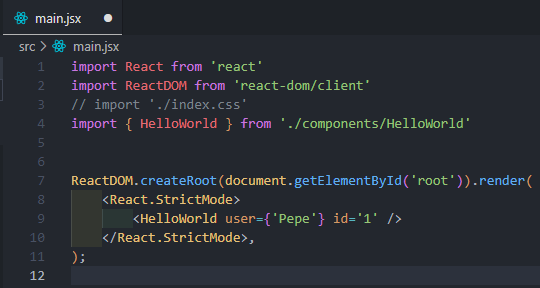
Este mecanismo que permite pasar datos desde un componente padre a un componente hijo a través de las props.

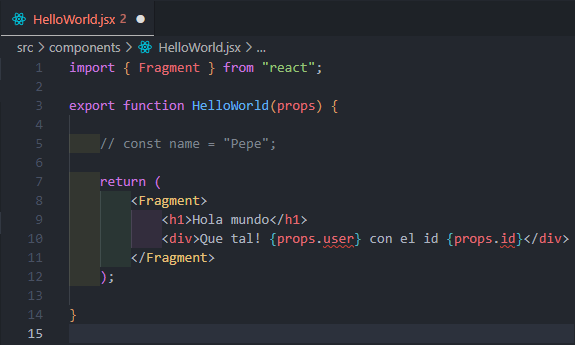
Es importante destacar que las props son de solo lectura y no deben modificarse dentro del componente hijo. Si se necesitan cambios en el valor de una prop, deberían ser gestionados por el componente padre.

Múltiples valores entre componentes

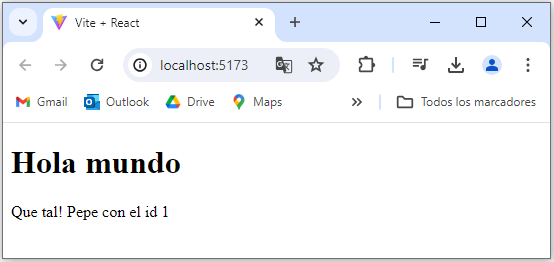
En el componente **HelloWorld.jsx**, además de imprimir el valor de la propiedad **user**, tambien podemos mostrar el valor de la propiedad **id** utilizando **props.id**. Es importante también tener en cuenta el tipo de dato que se está pasando, ya sea string, number, boolean, etc.

La diferencia entre pasar el valor de **user** usando llaves y el valor de **id** sin llaves es simplemente porque **user** es una expresión de JavaScript que necesita ser evaluada, mientras que **id** es una cadena de texto directa que se pasa como un valor estático.



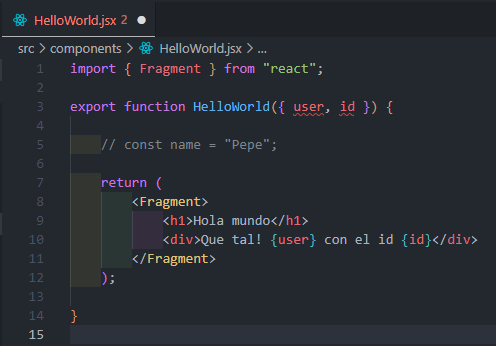


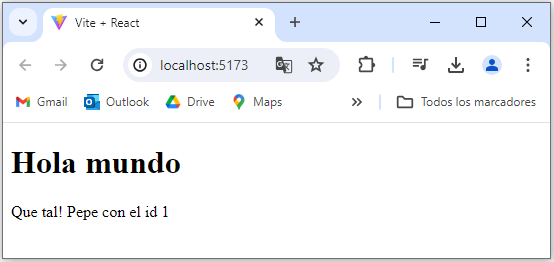
De esta manera, el componente **HelloWorld** podrá acceder a los valores de **user** e **id** que le son pasados desde **main.jsx**.



Uso de la desestructuración

Para evitar tener que acceder a las props (propiedades) cada vez que se quiera utilizar un atributo, se puede utilizar la desestructuración de objetos en los parámetros de la función. Esto permite extraer directamente los atributos requeridos.



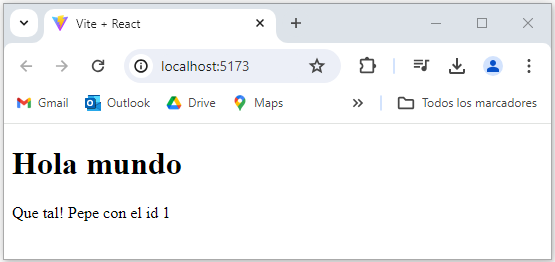


Se obtiene el mismo resultado en el navegador, pero el código es más limpio y legible.

Uso de la función de flecha

También se puede definir el componente utilizando una función de flecha, lo cual es una práctica común en React.





El uso de funciones de flecha es preferible en el contexto de React por su sintaxis más concisa y su manejo automático del contexto **this**.