Manejo de variables y propiedades en un componente

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

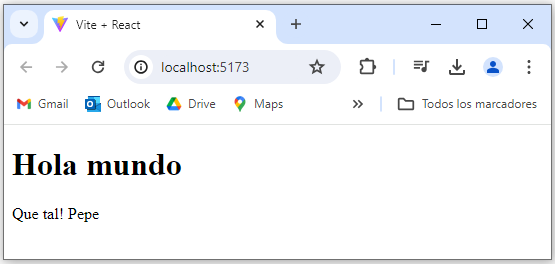
Variables internas en componentes

Los componentes en React son funciones especiales que se exportan y pueden recibir propiedades (props) a través de los parámetros de la función según la documentación de React.

Ten en cuenta que cuando utilizamos un componente en nuestra aplicación, como el **HelloWorld.jsx**, lo hacemos en el método **render** como una etiqueta HTML **<HelloWorld />** no como una función.

Para incluir variables que se pueden leer desde cualquier parte del componente, podemos simplemente definirlas como variables JavaScript dentro del cuerpo del componente. Por ejemplo, en el componente **HelloWorld.jsx** podríamos imprimir el valor de la variable **name** en el navegador insertando el nombre de la variable entre llaves en la porción JSX.





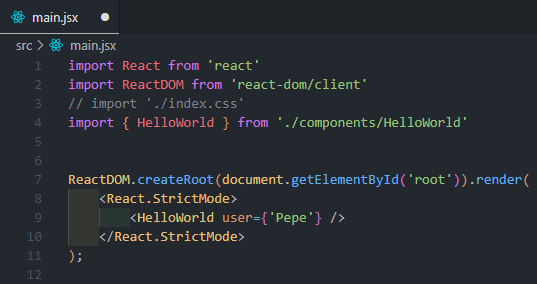
Es importante saber que los componentes pueden reutilizarse y anidarse dentro del cuerpo de otras funciones de retorno (específicamente dentro del **return**). Sin embargo, siempre necesitan tener un elemento raíz.

**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}**.

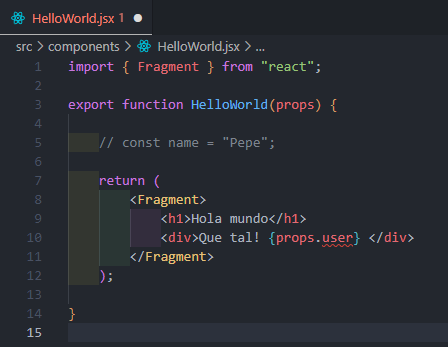
Pasar valores entre componentes

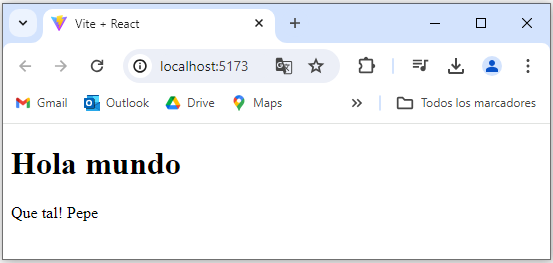
Para pasar un valor desde un componente padre a un componente hijo utilizamos las propiedades (props) definidas en el componente.

En el archivo **main.jsx**, donde se llama al componente **HelloWorld**, pasamos los valores al componente hijo; en este caso se define la propiedad **user** de la siguiente manera:



En el componente **HelloWorld.jsx**, definimos las props como un parámetro de la función del componente; este objeto contiene todos los parámetros que se van a pasar. Luego, para acceder al valor pasado desde el componente padre, usamos **props.user**.



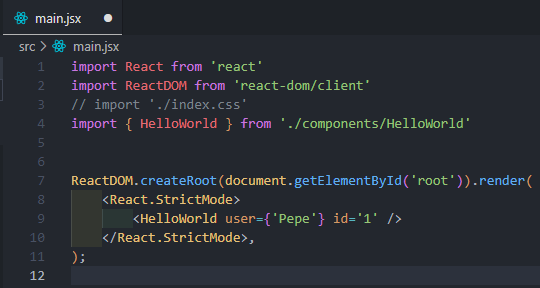


Se tiene en cuenta este mecanismo que permite pasar datos desde un componente padre a un componente hijo a través de las props.

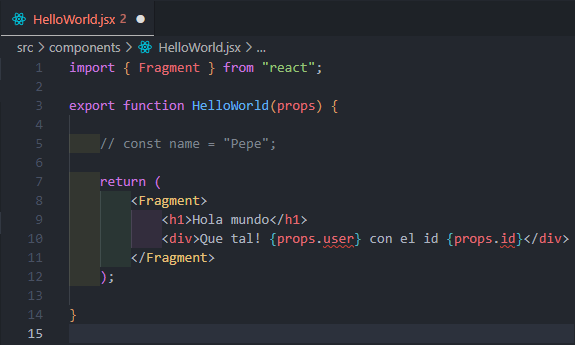
Múltiples valores entre componentes

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

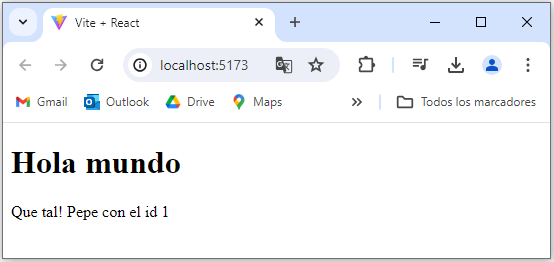
La diferencia entre pasar el valor de **user** usando llaves y el valor de **id** sin llaves es simplemente porque **user** es una expresión de JavaScript que necesita ser evaluada, mientras que **id** es una cadena de texto directa que se pasa como un valor estático.



En el componente **HelloWorld.jsx**, se accede a los valores de **user** e **id** a través de **props.user** y **props.id**, respectivamente.

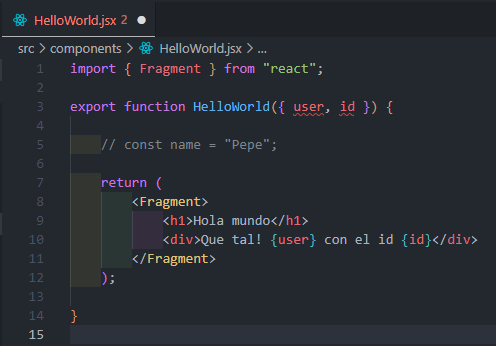


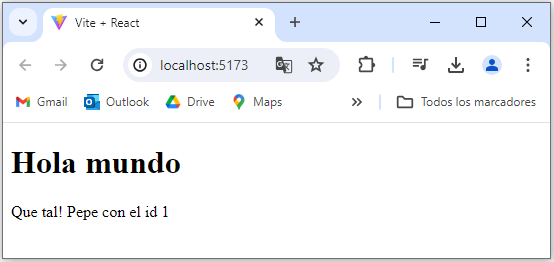
De esta manera, el componente **HelloWorld** podrá acceder a los valores de **user** e **id** que le son pasados desde **main.jsx**.



Uso de la desestructuración

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



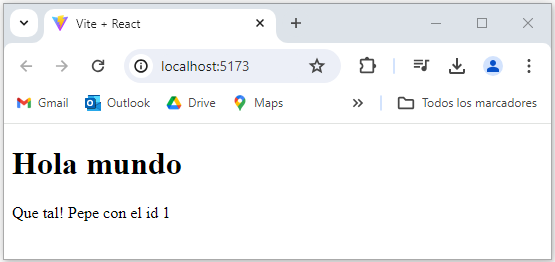


Obtenemos el mismo resultado en el navegador, pero el código es más limpio y legible.

Uso de la función de flecha

También podemos 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**.