Profundizando las propiedades de un componente

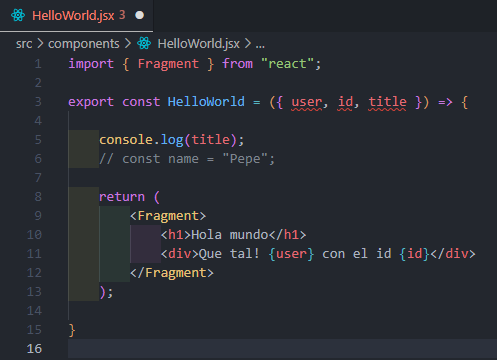
En esta sección vamos a profundizar el tema de las props (propiedades) de un componente en React y cómo manejarlas cuando se tienen props no definidas, valores por defecto en las props y también vamos a utilizar la consola de Chrome para visualizar los componentes.

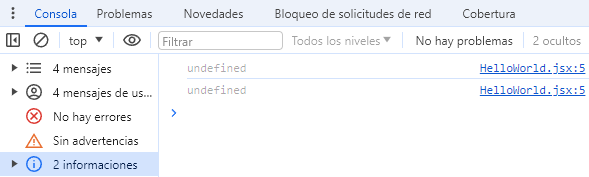
Propiedades no definidas

Los valores propiedades o props que se pasan como argumentos de la función al momento de renderizar el componente permiten la comunicación entre un componente padre (**main.jsx**) y un componente hijo (**HelloWorld.jsx**). Sin embargo, ¿qué sucede si se define una propiedad en el componente **HelloWorld.jsx**, pero no va a tener un valor asignado?

Manejo de propiedades con un valor no definido

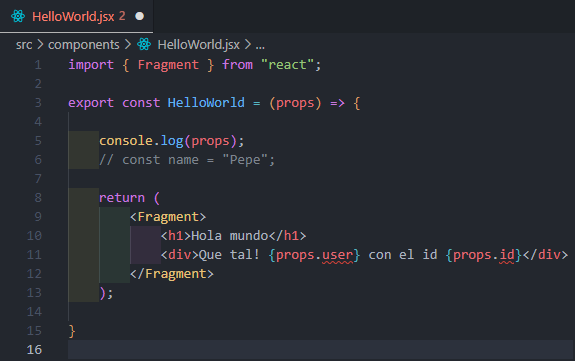
Al definir las props en el componente **HelloWorld.jsx**, como **user**, **id** y **title**, si no se asigna un valor a **title** (desde el componente padre **main.jsx**), al imprimirlo mediante **console.log**, mostrará **undefined** o **null** debido a que no está asignado.

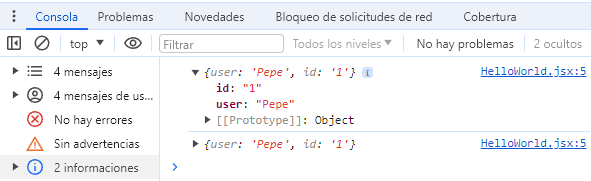




Mostrar todos los valores de las propiedades

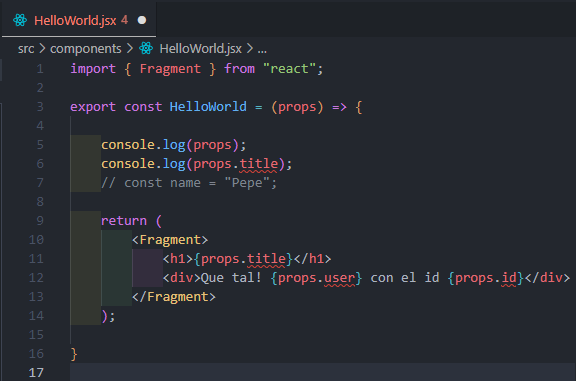
Si se imprime el objeto **props** directamente sin desestructurar, se mostrará un objeto en la consola que incluye todas las props, excepto aquellas props no definidas, como **title**.

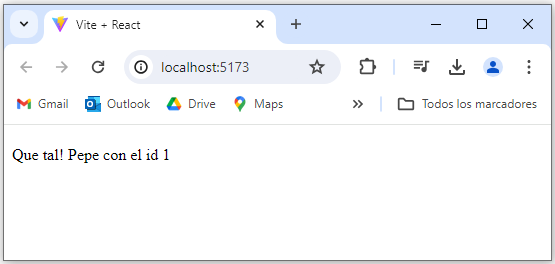


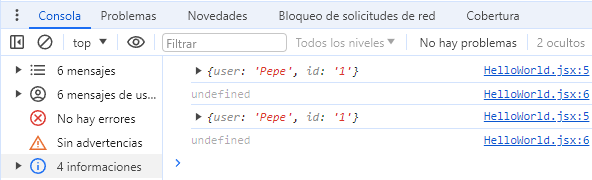


Mostrar el título no definido en el navegador

Si se intenta mostrar el valor de la prop **title** en el componente, aunque no esté definido, no causará errores en la aplicación, pero simplemente no se mostrará nada en el navegador y en la consola se imprimirá **undefined**.

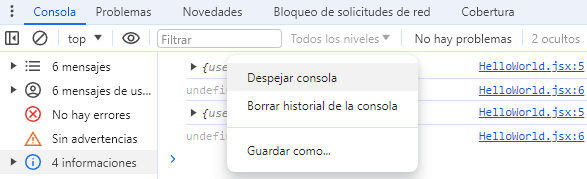






Limpiar la consola de Chrome

Una forma sencilla de limpiar la consola es haciendo clic derecho en ella y seleccionando la opción **Despejar consola** o **Clear console**. Esto borrará todos los mensajes que aparecieron en la consola hasta el momento.

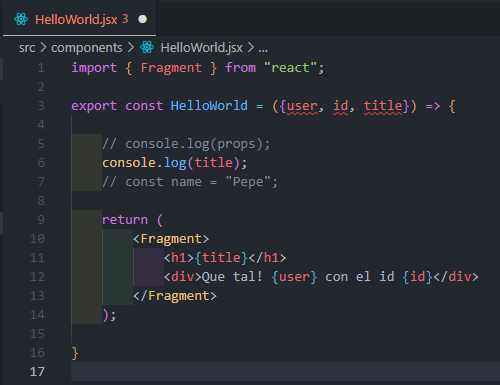


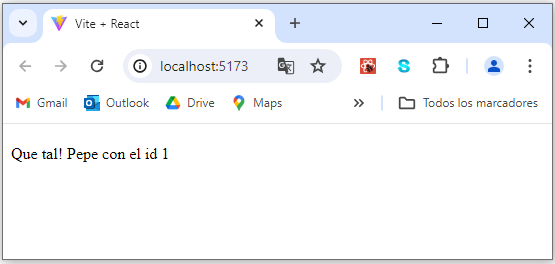
Asignando valores por defecto a las propiedades

Cuando se utiliza la desestructuración en los parámetros de una función, podemos asignar valores por defecto a esos parámetros.

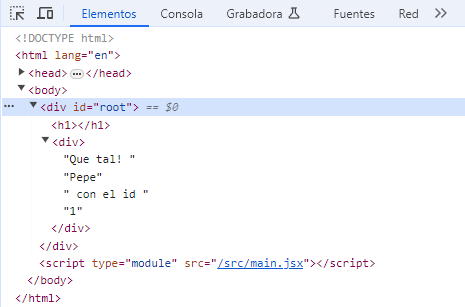
Volviendo a aplicar la desestructuración

Procede a restaurar los cambios en la función **HelloWorld** para que el título se muestre correctamente en caso de que no se defina explícitamente. Luego ve a inspeccionar el código fuente para revisar cómo se manejan los valores por defecto.



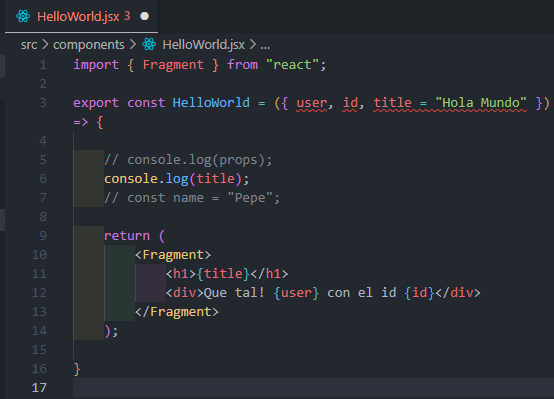


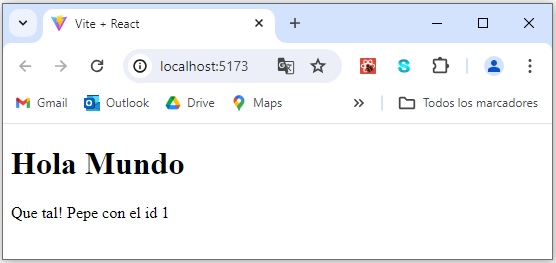
Cuando se inspecciona el código fuente generado, se puede visualizar que dentro de las etiquetas **<h1></h1>** no se muestra nada si el título no está definido, lo que resulta en **undefined**. También se tiene en cuenta que si el título es **false** (como booleano), **null** o un string vacío, tampoco se mostrará nada en la interfaz.



Asignar un valor por defecto en el parámetro

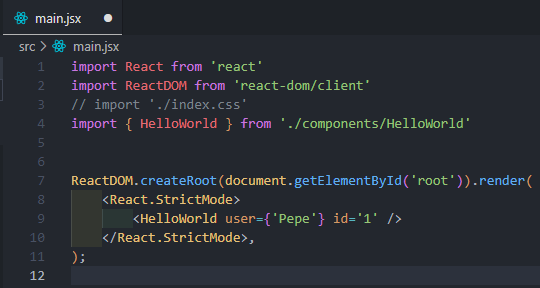
Para asignar un valor por defecto al título, podemos hacerlo directamente en la desestructuración de los parámetros. De esta manera, si no se proporciona un título al componente **HelloWorld**, se mostrará el valor definido por defecto.





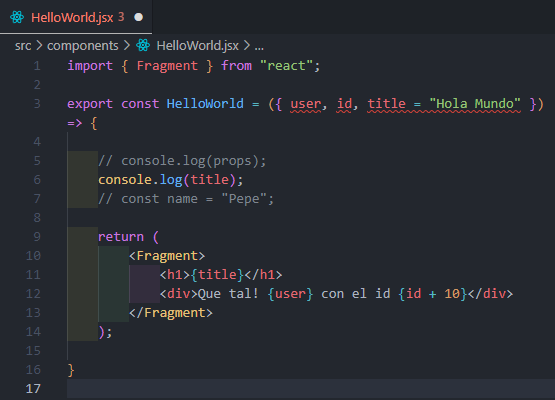
Valores por defecto de tipo String y number

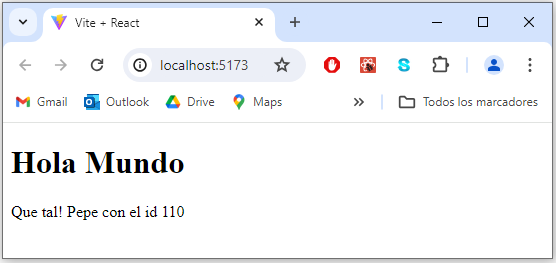
Es importante tener en cuenta los tipos de datos al manipular props en React. Por ejemplo, el atributo **id** debe ser de tipo numérico, pero en el archivo **main.jsx** se está pasando como **String**. Sin embargo, las props en React pueden manejar varios tipos de datos según cómo se asignen.



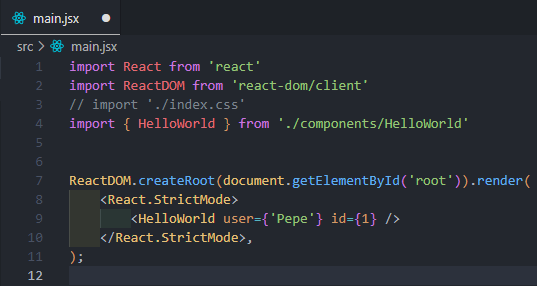
Para manipular el **id** en el archivo **HelloWorld.jsx** de manera numérica (por ejemplo, si se quiere sumar o multiplicar), primero se debe asegurar que el **id** sea tratado como un número en lugar de una cadena.

En el componente **HelloWorld**, al sumar 10 al **id**, si el **id** se pasa como una cadena, se va a concatenar en lugar de sumarse.

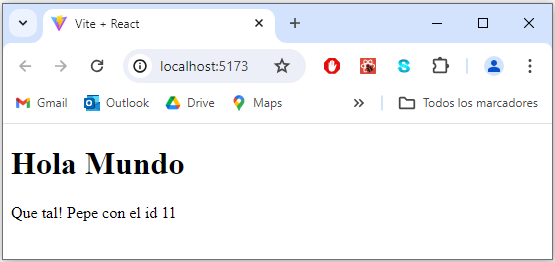




Por lo tanto, para tratarlo como un número, se debe asegurar que se interprete como tal mediante el uso de una expresión encerrada entre llaves.

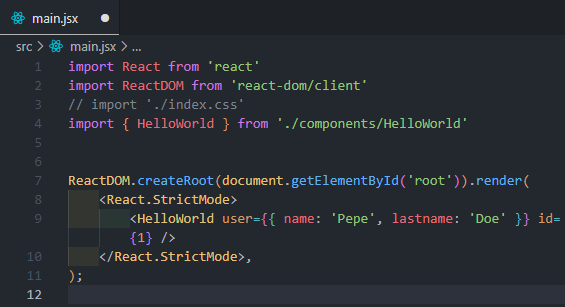


Ahora, al guardar y recargar el navegador, se mostrará la suma del **id** correctamente, ya que se está tratando como un número.



Valores por defecto de tipo objeto

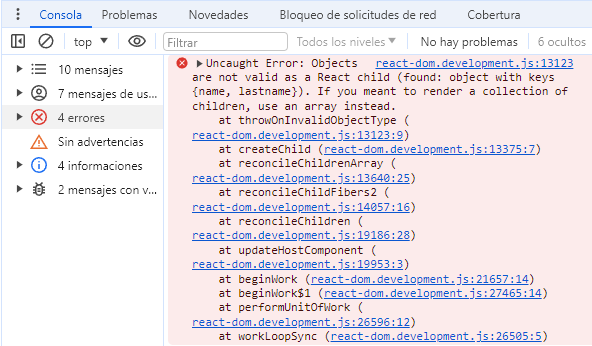
Es importante tener en cuenta cómo manejar los valores por defecto cuando se trata de atributos de objeto en React. Por ejemplo, en el archivo **main.jsx** se define el usuario (**user**) como un objeto con los campos **name** y **lastname**.



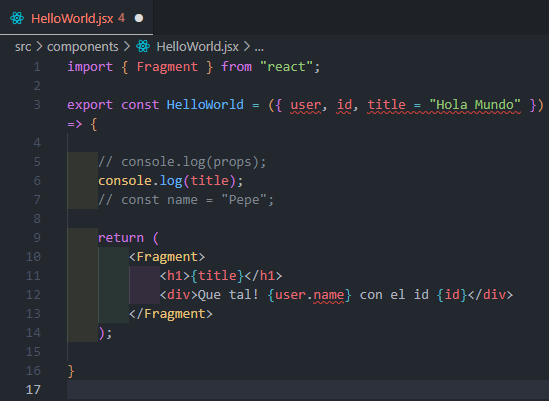
Luego, en el componente **HelloWorld.jsx**, al trabajar con un objeto como **user**, se debe accede a sus atributos de la misma manera que se accedería a los campos de un objeto en JavaScript.

Sin embargo, si se intenta imprimir el objeto directamente, se producirá un error ya que no se puede imprimir un objeto directamente como un componente de React.

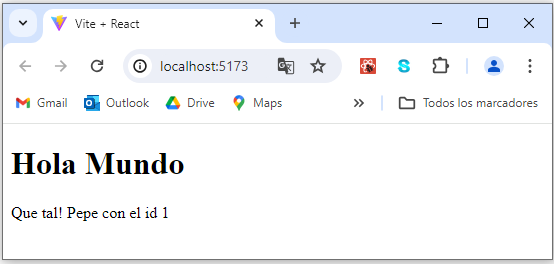




En el siguiente ejemplo, se imprime el nombre del usuario (**user.name**) en lugar del objeto **user** en sí. Esto garantiza que no se produzca un error y se muestra el valor del nombre del usuario correctamente.



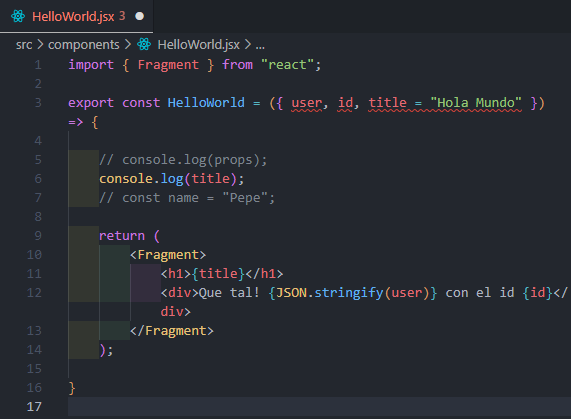
Al guardar y recargar el navegador, se verá el nombre del usuario junto con el **id** correspondiente en la aplicación

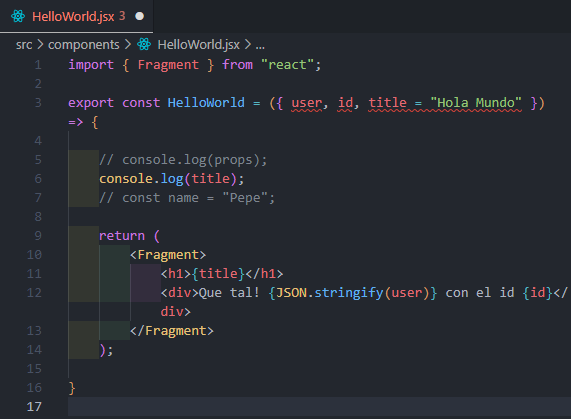


JSON Stringify

En React, cuando se trabaja con props de tipo objeto y se necesita imprimir el objeto completo como una cadena de texto, se puede utilizar el método **JSON.stringify**. Este método del objeto global JSON convierte un objeto JavaScript en una cadena de texto JSON.

En este ejemplo, **{JSON.stringify(user)}** convierte y muestra el objeto **user** como una cadena de texto JSON en el navegador.

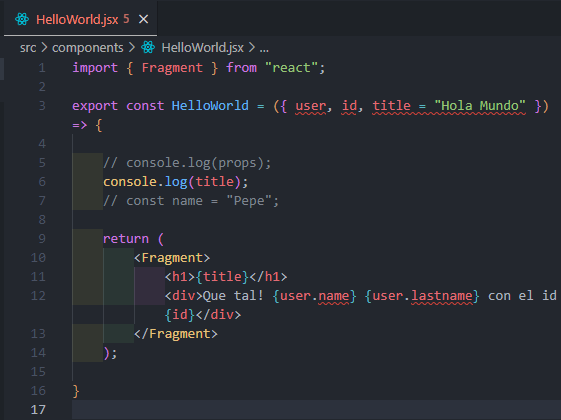


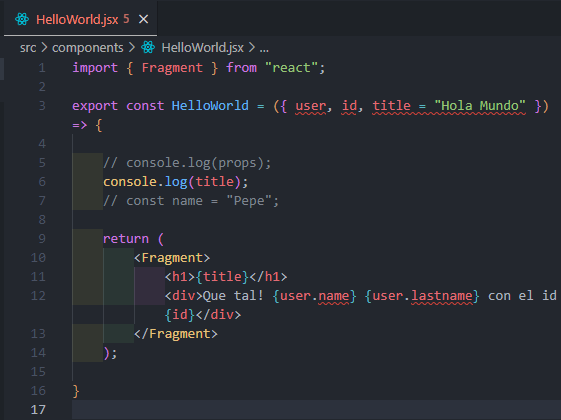


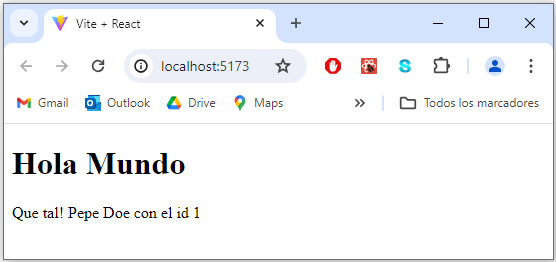


Ver los componentes de React en la consola

En el componente **HelloWorld.jsx**, asegúrate de incluir todas las props que deseas inspeccionar. Se elimina el método **JSON.stringify**, y se agrega **user.name** y **user.lastname** para mostrar el nombre y apellido del usuario.

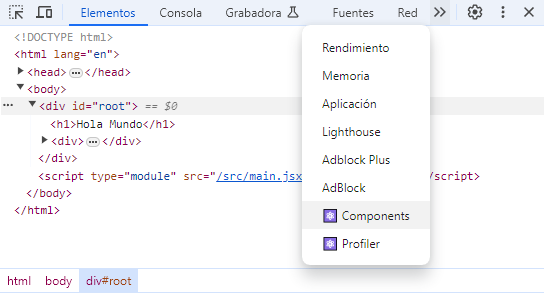






Si tienes instalada la extensión **React Developer Tools** en Google Chrome, podrás inspeccionar los componentes de React en la consola del navegador.

En la consola del navegador, busca la sección **Components** después de cargar la página. Deberías ver una lista de los componentes de React que están presentes en la página.



Haz clic en el componente **HelloWorld** para expandirlo y ver todas sus props. Aquí se puede encontrar las props **user**, **id** y cualquier otra prop que se haya pasado al componente desde el archivo **main.jsx**.

El título (**title**) no se muestra en la consola porque no se está pasando como una prop al componente **HelloWorld** desde el archivo **main.jsx**.

