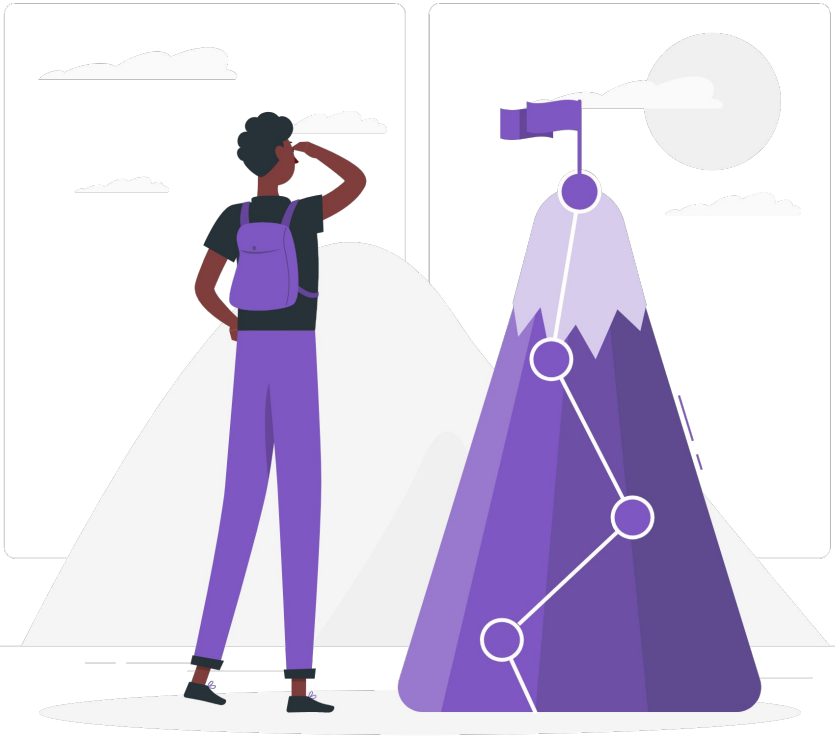
Mutabilidad & useRef

DEV.F:
DESARROLLAMOS(PERSONAS);



Objetivos

- Entender qué es la mutabilidad e inmutabilidad.
- Comprender qué problemas resuelve el hook useRef
- Aprender a implementar el hook de useRef.

DEV.F::

In/Mutabilidad

DEV.F::
DESARROLLAMOS(PERSONAS);

Inmutabilidad

Su planteamiento básico es muy sencillo: **un dato u objeto, una vez creado, no puede ser cambiado, manteniendo su estado original en todo momento**.

Si por algún motivo se tuviera que cambiar el dato, entonces se obtendría una copia con los datos modificados, pero nunca se cambian los valores originales.

DEV.F::

Immutable (Primitive Values)	Mutable (Everything Else)
undefined	Object
Boolean	Array
Number	Map
String	Set
BigInt	Date
Symbol	Function
null	Almost everything made with ‘new’ keyword

Mutabilidad

Lo contrario a la inmutabilidad es la mutabilidad, es decir, **la capacidad para cambiar el valor o el estado de los elementos de un lenguaje de programación.**

Cuando cambiamos el valor de una propiedad, o la referencia de una variable, estamos haciendo uso la mutabilidad, es decir, de la capacidad de cambiar.

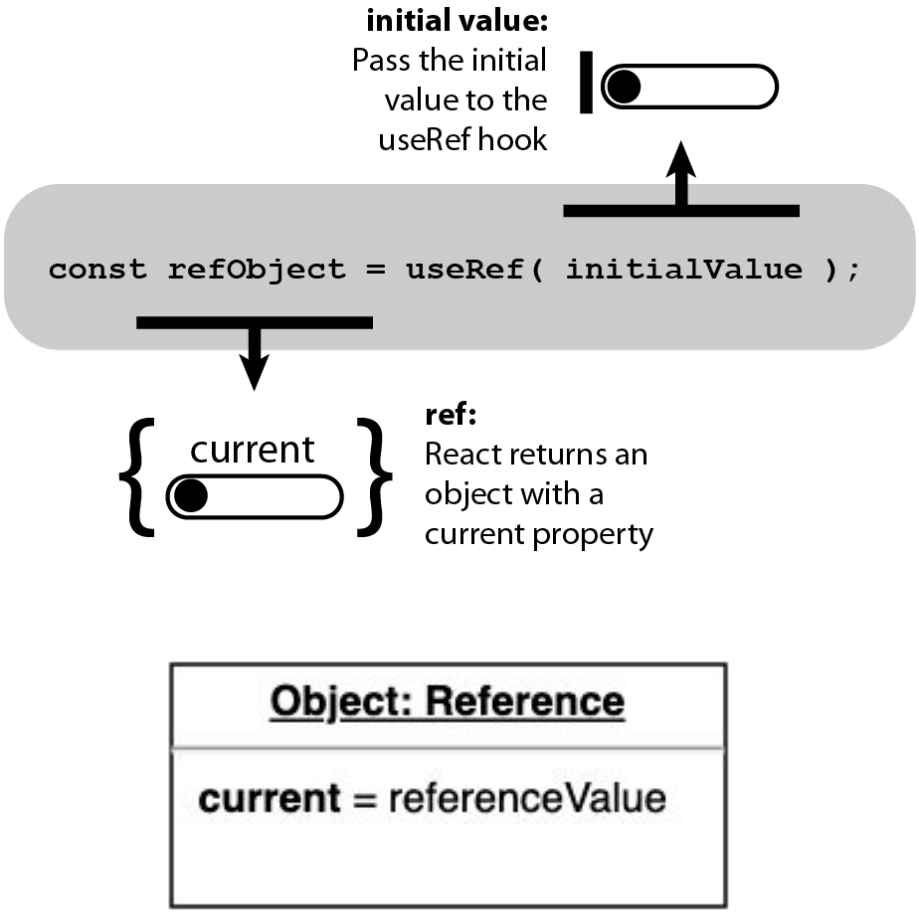
DEV.F::

Problemas de la Mutación

La mutación es una causa de los estados impredecibles en los datos y objetos, lo que exige una programación defensiva, comprobando que los datos son del tipo que esperamos, están completos o los objetos siguen en el mismo estado interno y no han sido cambiados después de haber sido pasados a otra función.

useRef

DEV.F::
DESARROLLAMOS(PERSONAS);



useRef

useRef es un hook utilizado para obtener una referencia a los datos de un objeto con información mutable.

Es decir, es como una manera de siempre poder obtener los datos más recientes mediante referencia de algún objeto de html. Por ejemplo para referenciar a los valores recientes de un formulario.

DEV.F::

Características useRef

Dos características importantes de useRef es qué:

1. Los datos son persistentes en caso de que se re-renderice el componente.
2. Actualizar los datos de esta referencia no causan el re-render.

Cabe recalcar la diferencia con useState, que la actualización de datos es síncrona, ya además como hemos mencionado, no se re-renderiza.

¡ Vamos al Código !