**Tipos de datos en JavaScript**

* **Tipos Primitivos:** Estos tipos de datos son **inmutables** y se pasan por valor. Esto significa que una vez que se crea un valor primitivo, no se puede cambiar. Si realizas una operación sobre uno de estos tipos, obtendrás un nuevo valor en lugar de modificar el existente. **number – string – boolean – null – undefined – symbol – bigInt**

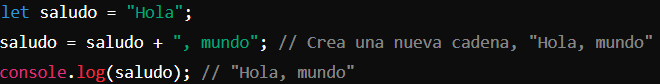
**Inmutabilidad de Referencia con “const”:** “const” hace que la referencia a un objeto o array sea inmutable, pero no impide la mutabilidad del contenido del objeto o array al que hace referencia.

* **Tipos Compuestos:** Estos tipos de datos son **mutables** y se pasan por referencia. Esto significa que se puede cambiar el contenido del objeto o array después de su creación, y cuando se asignan a una variable o se pasan a una función, se copia la referencia al valor en la memoria, no el valor en sí. **Object – Array – Function – Date – Map – Set – etc**

**Inmutabilidad (copia)**

La **inmutabilidad** implica que una vez que un valor es creado, no se puede modificar. En cambio, cualquier "modificación" resulta en la creación de un nuevo valor, dejando el original intacto. Cuando modificas un valor primitivo, creas un nuevo valor en la memoria.

Los **Tipos Primitivos se pasan por valor**,cuando asignas un tipo primitivo a una nueva variable o lo pasas a una función, se copia el valor.



Aunque parece que estamos modificando la cadena saludo, en realidad estamos creando una nueva cadena y asignándola a la variable. El valor original "Hola" permanece inmutable.

**Object.freeze()**: Congela un objeto para hacerlo inmutable.

**Mutabilidad (referencia)**

La **mutabilidad** se refiere a la capacidad de un valor ser modificado sin cambiar su referencia en la memoria. Cuando modificas un objeto o array, alteras el contenido del valor al que apunta la referencia.

Los **Tipos Compuestos se pasan por referencia,** cuando asignas un tipo de referencia a una nueva variable o lo pasas a una función, se copia la referencia, no el valor en sí.

Texto

Descripción generada automáticamente

El contenido del objeto ha cambiado sin cambiar la referencia.

**Spread operator y métodos no mutables**: Crean nuevas copias de objetos o arrays en lugar de modificarlos directamente.

**Object.freeze()**

Es un método que se utiliza para congelar un objeto. Congelar un objeto significa que se previenen cambios en el objeto, lo que implica que no se pueden agregar, eliminar, o modificar las propiedades del objeto. Además, no se pueden cambiar los atributos de las propiedades existentes (como la configuración de lectura/escritura).

Sintaxis: **Object.freeze(objeto);**

* Las propiedades del objeto no pueden ser modificadas, añadidas, ni eliminadas.
* No puedes cambiar la configuración de las propiedades existentes.
* No realiza una congelación profunda. Es decir, si el objeto tiene propiedades que son a su vez objetos, esas propiedades no se congelan automáticamente. Solo el objeto en sí mismo está congelado.