**Tipos de datos primitivos y no primitivos en JavaScript**

**Datos Primitivos:** Son **inmutables**, lo que significa que sus valores no pueden ser cambiados una vez que han sido creados. Cada vez que se realiza una operación que parece "modificar" un valor primitivo, se crea un nuevo valor en lugar de modificar el original.

* **number:** Representa números enteros y de punto flotante, ejemplo: **42**, **3.14**, **-7**
* **string:** Representa secuencia de caracteres, ejemplo: **"Hola"**, **'JavaScript'** y **`Plantilla con ${variable}`**
* **boolean:** Representa un valor lógico, puede ser true o false.
* **undefined:** Representa una variable que ha sido declarada pero no ha sido inicializada.
* **null:** Es un tipo especial que indica que una variable no tiene ningún valor.
* **symbol:** Representa un valor único y anónimo, útil para crear identificadores únicos, los símbolos son inmutables y se crean con la función Symbol()

Una captura de pantalla de un celular con texto e imagen

Descripción generada automáticamente con confianza media

* **bigInt:** Representa números enteros de precisión arbitraria que no pueden ser representados por el tipo Number. Se crea agregando una n al final de un número entero o utilizando la función BigInt().

**Texto

Descripción generada automáticamente**

**Datos NO Primitivos (referencias):** Son **mutables**, lo que significa que sus valores pueden ser modificados después de ser creados. Además, cuando se asigna o se pasa una referencia, se está copiando la referencia al objeto, no el valor en sí.

* **Object literal:** Un “Object” es una colección de pares clave-valor. Las claves (propiedades) son cadenas o símbolos, y los valores pueden ser de cualquier tipo, incluidos otros objetos, arrays, funciones, etc.

Texto

Descripción generada automáticamente

* **Array:** Un “Array” es una lista ordenada de elementos, que pueden ser de cualquier tipo. Los elementos están indexados numéricamente, comenzando desde 0



* **Function:** - Una “Function” es un bloque de código diseñado para realizar una tarea específica. En JavaScript, las funciones son objetos de primera clase, lo que significa que pueden ser asignadas a variables, pasadas como argumentos y devueltas por otras funciones.

Texto

Descripción generada automáticamente

* **Date:** Representa fechas y horas. Se utiliza para trabajar con fechas y horas.

Interfaz de usuario gráfica, Texto

Descripción generada automáticamente

* **RegExp:** Representa una expresión regular, que es una secuencia de caracteres que forma un patrón de búsqueda. Es un patrón utilizado para hacer coincidir cadenas de texto con una cierta estructura.

Texto

Descripción generada automáticamente

* **Set**: Representa una colección de valores únicos. A diferencia de los arrays, no permite elementos duplicados.

Interfaz de usuario gráfica, Texto, Aplicación, Chat o mensaje de texto

Descripción generada automáticamente

* **Map**: Representa una colección de pares clave-valor donde las claves pueden ser de cualquier tipo.

Texto

Descripción generada automáticamente

* **WeakMap:** Similar a “Map”, pero las claves deben ser objetos, y las referencias a esas claves son "débiles", lo que permite que se eliminen de la memoria si no se referencian en otro lugar.Texto

  Descripción generada automáticamente
* **WeakSet:** - Similar a “Set”, pero solo acepta objetos como valores y, como en “WeakMap”, las referencias a los objetos son "débiles"

Interfaz de usuario gráfica, Texto

Descripción generada automáticamente