

# Variable en Javascript



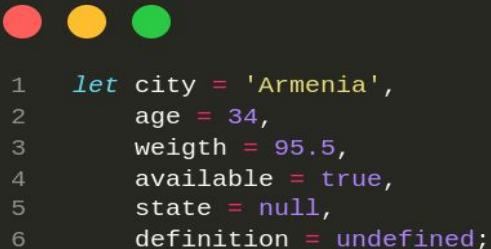
Una variable en programación es un espacio en memoria donde se almacenará determinado tipo de dato. En esa medida en javascript y pese a que es un lenguaje dinámicamente tipado, también se reconocen algunos tipos de variables, tal y como veremos a continuación

JS

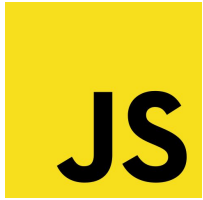
# Variable en Javascript



Tipo	asignación
String	"texto" - 'texto' - `texto`
Number	50 - 50.7
Boolean	false - true
Others	null, undefined

A code editor window with a dark background and three colored window control buttons (red, yellow, green) at the top left. It contains a JavaScript code snippet with line numbers 1 through 6.

```
1  let city = 'Armenia',  
2    age = 34,  
3    weigth = 95.5,  
4    available = true,  
5    state = null,  
6    definition = undefined;
```



# Variable en Javascript



Teniendo en cuenta que JS es un lenguaje dinámicamente(débilmente) tipado.

La manera como se nombran las variables aquí es:

var **nameValue**

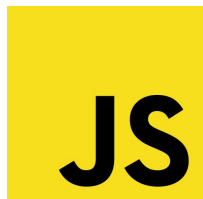
let **nameValue**

const **nameValue**

donde **nameValue** es el nombre que decides darle a la variable(es bueno que las variables comienzan en minúscula y utilizar el método [camelCase](#))

- Con el pasar del curso iremos comprendiendo un poco el uso de cada tipo, pero por lo pronto, usaremos **let** para variables y **const** para constantes

```
1  var nameUser = 'Edwin';  
2  let lastNameUser = 'Rozo';  
3  const DNI = 555555;
```



# Variable numéricas

Normalmente y dependiendo cada caso para garantizar las variables numéricas en Javascript se utilizan los siguientes tópicos:

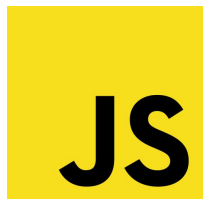
<b>Number</b>
<b>parseInt</b>
<b>parseFloat</b>



Estos 3 se aplican en caso de que el número ingrese en tipo **string**, y requiera su valor numérico.



```
1 inputEntrada = Number(input.value) // Tipo número
2 inputEntrada = parseInt(input.value) // Tipo número entero
3 inputEntrada = parseFloat(input.value) // Tipo Flotante
```



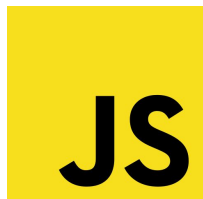
# Variables String

Para dar uso de variables tipo String en javascript, existen varias formas, todas útiles.



```
1
2  const product1 = "Tv LG de 20 Pulgadas";
3  const product2 = '  Monitor de TV.  ';
4  const product3 = String('TV');
5  const product4 = new String('TV');
6  const product5 = `Este es mi ${product1} `; //Template Literal
```

- Aunque todas son válidas, lo más usual es simplemente asignar la variable tal y como sucede en las líneas 2,3 y 6 del ejemplo,



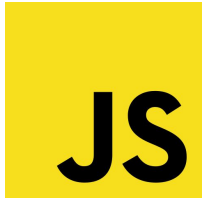
# Variables String



Vamos a dedicar un momento a mostrar la importancia de usar el template literal (comilla invertida ``) ya que por su sintaxis permite ahorrar la concatenación de cadenas y permite concatenar muchos valores (funciones, variables, cálculos, arrays entre otros)



```
1  const product5 = `Este es mi ${product1}`; //Template Literal
```



# Métodos para trabajar con Strings

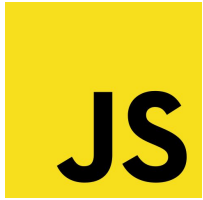


**string.length:** Imprime la dimensión de la cadena de texto.



20

```
1  const product1 = "Tv LG de 20 Pulgadas";  
2  console.log(product1.length); // Cuántos caracteres incluyendo espacios hay.
```



# Métodos para trabajar con Strings



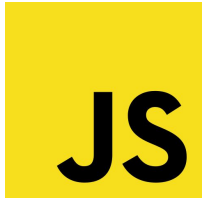
**string.indexOf(''):** Si el elemento que colocamos dentro del parámetro existe, retornará su posición. En caso de no encontrar coincidencia retornará -1



3

```
1  const product1 = "Tv LG de 20 Pulgadas";  
2  console.log(product1.indexOf('LG'));  
3  // En este caso donde encuentre la primera coincidencia LG, retorna el número donde se encuentra el caracter.  
4
```





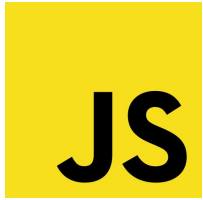
# Métodos para trabajar con Strings



**string.includes(""):** Comprobar si un elemento existe o no en la cadena de texto. En caso de no encontrar coincidencia, retornará **false**.

true

```
1  const product1 = "Tv LG de 20 Pulgadas";  
2  console.log(product1.includes('LG'));
```



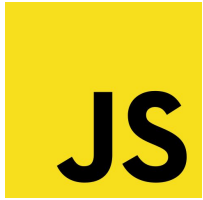
# Métodos para trabajar con Strings



**string.trimStart() - string.trimEnd() - string.trim()**

Eliminar espacios de la cadena de texto al principio, al final o a ambos lados.

```
1  const product1 = "      Tv LG de 20 Pulgadas      ";
2
3  // Elimina espacios al principio de la cadena
4  console.log(product1.trimStart());
5  // Elimina espacios al final de la cadena
6  console.log(product1.trimEnd());
7  // Elimina espacios a ambos lados
8  console.log(product1.trimStart().trimEnd());
9  // Elimina espacios a ambos lados
10 console.log(product1.trim());
```

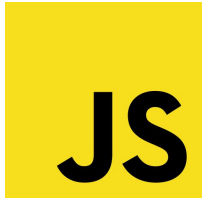


# Métodos para trabajar con Strings



**string.replace():** Reemplazar un fragmento de la cadena por otro.

```
1 // Reemplazar elemento en una cadena de texto
2 const product1 = "Tv LG de 20 Pulgadas";
3 // En este caso se va a reemplazar la palabra Pulgadas por ''
4 console.log(product1.replace('Pulgadas', ''));
```



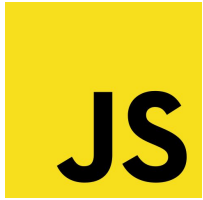
# Métodos para trabajar con Strings



**string.slice(valueInit,valueEnd):** Cortar la cadena de texto desde el valor inicial indicado hasta el valor final indicado.

**string.substring(valueInit,valueEnd):** Cortar la cadena de texto desde el valor inicial indicado hasta el valor final indicado.

```
1  const product1 = "Tv LG de 20 Pulgadas";
2  // Cortar desde la letra 3 hasta la letra 11
3  console.log(product1.slice(3,11));
4  console.log(product1.substring(3,11));
5  // El método substring tiene la capacidad de reordenar los parámetros en caso de que el valor inicial sea mayor que el valor final
6  console.log(product1.substring(11,3));
```



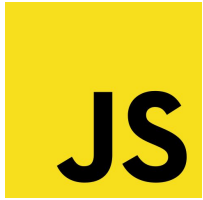
# Métodos para trabajar con Strings



**string.charAt(posicion en la cadena):** Traer el valor solicitado de la cadena de texto.



```
1  const product1 = "Tv LG de 20 Pulgadas";  
2  // En este caso devolverá el número 2 que es lo que encuentra en la posición 9 de la cadena  
3  console.log(product1.charAt(9));
```

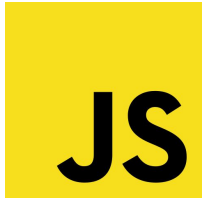


# Métodos para trabajar con Strings



**string.repeat(número de veces):** Repetir la cadena de texto de acuerdo al número ingresado en el parámetro..

```
1  const product1 = "Tv LG de 20 Pulgadas";  
2  // En este caso retornaría:  
3  // Tv LG de 20 PulgadasTv LG de 20 PulgadasTv LG de 20 Pulgadas  
4  console.log(product1.repeat(3));
```



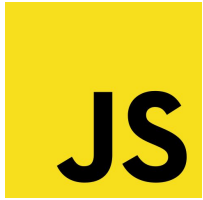
# Métodos para trabajar con Strings



**string.split("Caracter ingresado"):** Dividir la cadena de texto por en la cantidad de elementos donde aparezca el caracter ingresado en el parámetro.

La particularidad de este método es que convierte la respuesta en un arreglo.

```
1
2  const product1 = "Tv LG de 20 Pulgadas";
3
4  // En este caso estamos separando por un espacio vacío " "
5  // para lo cual obtendremos un Array
6  ["Tv", "LG", "de", "20", "Pulgadas"]
7  // console.log(product1.split(' '));
```



# Métodos para trabajar con Strings

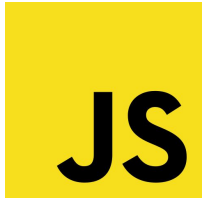


**string.toUpperCase() - string.toLowerCase()**

Convertir la cadena de texto a mayúscula - Convertir la cadena de texto a minúscula

```
1  const product1 = "Tv LG de 20 Pulgadas";
2
3  // Convertir cadena de texto a mayúscula.
4  console.log(product1.toUpperCase());
5  // Convertir cadena de texto a minúscula.
6  console.log(product1.toLowerCase());
```





# Métodos para trabajar con Strings



## **number.toString()**

Convertir un valor numérico a string.

```
1  const price = 500;  
2  // Convertir el valor 500 a cadena de Texto  
3  console.log(price.toString());
```

Plus: Recordemos que **type of** value, retorna el tipo de dato con el que estoy trabajando.

**Edwin Rozo Gómez -**  
**edwin.rozo1@misena.edu.co**