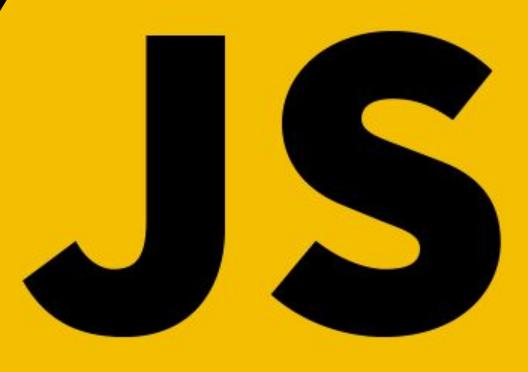
Fundamentos de Programación con JavaScript

Tipos de Datos







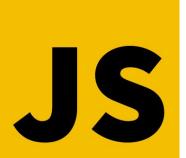
Datos básicos



MÉTODO: todo aquello que un objeto puede hacer (acción) PROPIEDAD: es una característica (atributo) que un objeto tiene

Se utilizan usando la nomenclatura del punto

- Las propiedades no usan paréntesis
- Los métodos usan paréntesis y necesitan o no parámetros



String

En cualquier lenguaje de programación, un **string** es una secuencia de caracteres usado para representar texto.

En JavaScript, un string es uno de los valores primitivos

Interactuar con cadenas:

- Concatenar
- Interpolar

El objeto **String** se utiliza para representar y manipular una secuencia de caracteres.

Las cadenas son útiles para almacenar datos que se pueden representar en forma de texto. Algunas de las operaciones más utilizadas en cadenas son verificar su length, para construirlas y concatenarlas usando operadores de cadena + y +=, verificando la existencia o ubicación de subcadenas con indexOf() o extraer subcadenas con el método substring().

`Template Strings` Los cadena literales se pueden especificar usando comillas simples o dobles, que se tratan de manera idéntica, o usando el carácter de comilla invertida . Esta última forma especifica una Plantilla literal: con esta forma puedes interpolar expresiones.

String [MÉTODOS y PROPIEDAD]

MÉTODOS Todos los métodos devuelven una cadena nueva, la cadena original no se modificada.

```
// console.log(cadena.length);
// console.log(cadena.toUpperCase());
toUpperCase() --> Devuleve la cadena a mayúsculas
toLowerCase() --> Devuelve la cadena a minúsculas
indexOf(cadena) --> Devuelve la posición en la que se encuentra la cadena, si no lo encuentra devuelve -1
replace(valor a buscar, valor nuevo) --> Remplaza el fragmento de la cadena que le digamos y pone el valor nuevo
substring(inicio [,fin]) --> Extrae los caracteres desde inicio hasta fin (el final no se incluye)
                             Si no se incluye el fin extrae hasta el final.
slice(inicio [,fin]) --> Igual que substring pero admite valores negativos,
                         si ponemos valores negativos empezará desde atrás.
                         Si no se incluye el final extrae hasta el final
                         (2,-4) --> Empieza a contar en el tercer caracter y los 4 últimos no los considera
trim()--> Elimina los espacios al inicio y al final de la cadena
```

String [MÉTODOS]

```
startsWith(valor [,inicio]) --> Sirve para saber si la cadena empieza con ese valor. Devuleve true o false
endsWith(valor [,longitud]) --> Sirve para saber si la cadena termina con ese valor. Devuleve true o false
includes(valor[,inicio]) --> Igual que indexOf pero devuelve true o false
repeat(valor) --> Repite el string el número de veces que le indiquemos.
Template Strings
```

PROPIEDADES

length --> Devuelve la longitud de la cadena

Es un objeto primitivo envolvente que permite representar y manipular valores numéricos de cualquier tipo: naturales, enteros y reales.

En JavaScript, los números se implementan en <u>Formato en coma flotante de doble precisión</u> de 64 bits IEEE 754 (es decir, un número entre $\pm 2^{-1022}$ y $\pm 2^{+1023}$, o aproximadamente $\pm 10^{-308}$ a $\pm 10^{+308}$, con una precisión numérica de 53 bits). Los valores enteros hasta $\pm 2^{53}$ - 1 se pueden representar con exactitud.

Además de poder representar números de punto flotante, el tipo number tiene tres valores simbólicos:
+ Infinity, - Infinity y NaN (Not-a-Number, no es un número).

Una adición más reciente a JavaScript es el <u>BigInt</u> que te permite representar números enteros que pueden ser muy grandes. Sin embargo, existen advertencias para usar <u>BigInt</u>.

Propiedad	Descripción
Number.MAX_VALUE	El número representable más grande (±1.7976931348623157e+308)
Number.MIN_VALUE	El número representable más pequeño (±5e-324)
<u>Number.NaN</u>	Valor especial not a number ("no es un número")
Number.NEGATIVE_INFINITY	Valor infinito negativo especial; devuelto por desbordamiento
Number.POSITIVE_INFINITY	Valor infinito positivo especial; devuelto por desbordamiento
Number.MIN_SAFE_INTEGER	Número entero seguro mínimo en JavaScript (-2 ⁵³ + 1 o -9007199254740991)
Number.MAX_SAFE_INTEGER	Máximo número entero seguro en JavaScript (+2 ⁵³ - 1 o +9007199254740991)

Método	Descripción
Number.parseFloat()	Analiza un argumento de cadena y devuelve un número de punto flotante. Igual que la función parseFloat() global.
Number.parseInt()	Analiza un argumento de cadena y devuelve un número entero de la base o raíz especificada. Igual que la función parseInt() global.
Number.isFinite()	Determina si el valor pasado es un número finito.
Number.isInteger()	Determina si el valor pasado es un número entero.
Number.isNaN()	Determina si el valor pasado es NaN . Versión más robusta del isNaN() global
Number.isSafeInteger()	Determina si el valor proporcionado es un número que es un entero seguro.

Método	Descripción
toExponential()	Devuelve una cadena que representa el número en notación exponencial.
toFixed()	Devuelve una cadena que representa el número en notación de punto fijo.
toPrecision()	Devuelve una cadena que representa el número con una precisión especificada en notación de punto fijo.

parseFloat(): convierte una cadena y devuelve un número de punto flotante.

parseInt(): Convierte una cadena y devuelve un entero.

Math

El objeto integrado Math tiene propiedades y métodos para constantes y funciones matemáticas.

Método	Descripción
abs()	Valor absoluto
<u>sin()</u> , <u>cos()</u> , <u>tan()</u>	Funciones trigonométricas estándar; con el argumento en radianes.
<pre>pow(), exp(), expm1(), log10(), log2()</pre>	Funciones exponenciales y logarítmicas.
<pre>floor(), ceil()</pre>	Devuelve el entero más grande/más pequeño menor/mayor o igual que un argumento.
<u>min()</u> , <u>max()</u>	Devuelven el valor mínimo o máximo (respectivamente) de una lista de números
random()	Devuelve un número aleatorio entre 0 y 1.
<pre>round(), fround(), trunc(),</pre>	Funciones de redondeo y truncamiento.
<pre>sqrt(), cbrt()</pre>	Raíz cuadrada, raíz cúbica.
<pre>sign()</pre>	El signo de un número, que indica si el número es positivo, negativo o cero.

Especiales

null

El valor null es un literal de Javascript que representa intencionalmente un valor nulo o "vacío".

Es uno de los valores primitivos de Javascript.

undefined

Un valor **primitivo** automáticamente asignado a las **variables** que solo han sido declarados o a los **argumentos** formales para los cuales no existe argumentos reales.

NaN

La propiedad global NaN es un valor que representa Not-A-Number.

Symbol

En <u>JavaScript</u>, Symbol es un <u>valor primitivo</u>. Un valor de **símbolo** representa un identificador único.

Boolean (booleano)

Tipo de dato lógico

El tipo de dato **lógico** o **booleano** es en computación aquel que puede representar valores de lógica binaria, esto es 2 valores, que normalmente representan *falso* o *verdadero*. Se utiliza normalmente en la programación,

El valor pasado como primer parámetro se convierte en un valor booleano, si es necesario. Si el valor se omite o es 0, -0, null, false, NaN, undefined, o la cadena vacía (""), el objeto tiene un valor inicial de false. Todos los demás valores, incluido cualquier objeto, un arreglo vacío ([]) o la cadena "false", crean un objeto con un valor inicial de true.

valores del Boolean <u>primitivo</u>, true y false

Cualquier objeto cuyo valor no sea <u>undefined</u> o <u>null</u>, incluido un objeto Boolean cuyo valor es false, se evalúa como true cuando se pasa a una declaración condicional. Por ejemplo, la condición en la siguiente declaración <u>if</u> se evalúa como true.

Fundamentos de Programación con JavaScript



