BOOTCAMP

Generación T

streambe

Clase 25

Valores en JS

Valores En JavaScript

JavaScript manipula distintos tipos de datos o valores, con una sintaxis particular para cada uno de ellos. Es decir, deberás prestar atención a cómo escribís tu código para no tener malos entendidos con tu programa.

Valores En JavaScript

⚠️Importante: JavaScript es case sensitive, es decir, las mayúsculas y las minúsculas tienen codificaciones distintas.

Esto se debe al código ASCII, creado en 1963 por el Comité Estadounidense de Estándares. Hoy en día, casi todos los sistemas informáticos usan el código ASCII para representar caracteres, símbolos, signos y textos. Por eso, la A ocupa la celda 65, mientras que la a (minúscula) ocupa la 97.

Valores En JavaScript

⚠️Importante: JavaScript es case sensitive, es decir, las mayúsculas y las minúsculas tienen codificaciones distintas.

Esto se debe al código ASCII, creado en 1963 por el Comité Estadounidense de Estándares. Hoy en día, casi todos los sistemas informáticos usan el código ASCII para representar caracteres, símbolos, signos y textos. Por eso, la A ocupa la celda 65, mientras que la a (minúscula) ocupa la 97.

Valores En JavaScript

En síntesis, los datos pueden ser primitivos o pueden ser más complejos, como son las Funciones y los Objetos. Ahora, nos enfocaremos en los valores primitivos:

- Números: Funcionan igual que en la matemática. Pueden ser números enteros o racionales. Se pueden sumar, restar, multiplicar o dividir y siguen la lógica de resolución que privilegia las operaciones dentro de los paréntesis para luego resolver el resto.
- Strings: Son cadenas de caracteres que incluyen letras, números y espacios. Debemos encerrarlos entre comillas simples (‘’) o dobles ("") para que JS entienda que es texto y no la confunda con una Variable.
- Booleanos: Son datos de tipo true (verdadero) o false (falso), es decir, que activan o desactivan cierta parte del programa según el input recibido.
- Undefined: Es un valor que posee una Variable que está sin definir en ese momento.
- Null: Es un valor que posee una Variable que está explícitamente vacía (y hay una razón para ello).

¿Cuál Es La Diferencia Entre Undefined Y Null?

Si pensamos en una Variable como una caja, el valor undefined indica que la caja está vacía (hasta que le demos contenido).

En cambio, en el caso de null, el programador le asignó explícitamente el valor para que la caja permanezca vacía.

Función typeof

typeof es una Función nativa de JavaScript a la que le podés pasar un Párametro y retornará qué tipo de dato es: si se trata de un String, null, undefined, etc.

```
typeof(5)
"number"
typeof("5")
"string"
```

Números

Los Números son uno de los valores primitivos de JavaScript.

Características Principales

Funcionan igual que en las matemáticas.

Se usan los operadores matemáticos para sumar, restar, multiplicar y dividir (+, -, *, /).

Se resuelve primero todo lo que esté dentro de los paréntesis ( ). Si hay más de un par de paréntesis, se resuelve la operación desde adentro hacia afuera, por ejemplo: ((3+2)*3)+1=16 vs. 3+2*3+1=10.

El operador de Módulo (%) no se considera como el valor absoluto de un número sino como el resto de una división. Por ejemplo, 3%2=1.

Practica

```
3 + 3
// 6
7 - 4
// 3
8 * 3
// 24
9 / 3
// 3
((3 + 2) * 2) + 6
// 16
15 % 2
//1
```

Strings

Los Strings son cadenas de caracteres. Para que JS los reconozca como tal, el texto debe escribirse entre comillas dobles ("") o simples ('').

Características Principales

Los Strings son cadenas de caracteres cuya información puede ser representada como un texto.

Un caracter puede ser tanto un texto, como un espacio o un número, siempre y cuando estén entre comillas:

```
"Tengo"+" "+"18"
```

Strings

⚠️ Importante: Los caracteres son case sensitive. Esto significa que escribir una A y a será interpretado de forma distinta.

- Pueden concatenarse usando el operador de suma (+).
- Los datos pueden coercionarse. Por ejemplo, cuando se concatena un String con un número, este último será forzado a comportarse como un texto:

```
"Tengo " + 18 + " años"
```

Si bien, el 18 es un valor numérico, JavaScript lo interpretará como un texto: Tengo 18 años.

¿Por Qué Es Fundamental El Entrecomillado?

Números: Si un número no está entrecomillado, JS lo interpretará como un valor numérico o lo coercionará para que funcione como un texto.

Textos: Si un texto no está entrecomillado, JS lo interpretará como si fuera una Variable o instrucción que puede, o no, estar definida.

Propiedades Y Métodos De Los Strings

JavaScript tiene muchas Propiedades y Métodos.

⚠️ Importante: Las Propiedades también se conocen como Funciones. Aprenderemos sobre Funciones mas adelante.

¿Como se ve un metodo?

valor.metodo()

¿Como se ve una propiedad?

valor.propiedad

¿Como se ve una funcion?

funcion(valor)

Propiedades Más Frecuentes

Las Propiedades y Métodos que usarás con más frecuencia son:

Length: Permite contar la cantidad de caracteres que hay en un String.
[ ]: Permite acceder a un caracter puntual de un String para saber qué hay en esa posición.

⚠️ Importante: Las posiciones se cuentan del 0 en adelante.

Metodos Más Frecuentes

- toLowerCase(): Convierte todo el String en minúsculas.
- toUpperCase(): Convierte todo el String en mayúsculas.
- concat(): Concatena (es decir, une) dos o más Strings. Este método no cambia los Strings existentes, sino que devuelve uno nuevo que contiene el texto de los Strings unidos.
- includes(): Determina si un String contiene los caracteres especificados en el paréntesis. Este método devuelve true si los contiene y false si no.

⚠️ Importante: Este ultimo método distingue entre mayúsculas y minúsculas.

Variables

Una Variable es un contenedor que guarda información para, luego, usarla.

Sintaxis

- Para declarar la Variable se usa la palabra reservada let.

¿Qué Significa Declarar Y Definir?

- Cuando se habla de declarar una Variable, se la está creando. Cuando se habla de definir una Variable, se le está asignando contenido.

Variables

para definir una Variable, seguí este paso a paso:

1. Usá la keyword let para declarar la nueva Variable.
2. Luego, nombrala (usá camelCase y no le pongas comillas).
3. Separá la Variable de su valor usando el operador =.
4. Finalmente, completá (o no) su valor.

```
let nombreDeLaVariable = suValor;
```

La Variable En Acción

Cuando se declara una Variable se reserva ese contenedor para, luego, llenarlo con un valor.

👩‍👨‍Veamos un ejemplo. Supongamos que quisiéramos crear un programa que saludara a cada usuario por su nombre. Para hacerlo, deberíamos:

1. Declarar la Variable nombre `let nombre`
2. Definir la Variable con un contenido `nombre = 'Mery'`

La Variable En Acción

Sin embargo, una forma más simple de alcanzar este resultado sería realizando los dos pasos anteriores de una sola vez. En este sentido, estaríamos refactorizando nuestro código: re-escribiéndolo para que haga lo mismo de una manera más eficiente.

Nos quedaría así:

```
let nombre = 'Mery'
```

Modificar una variable

Ahora, si quisiéramos modificar el valor de la Variable (de 'Mery' a 'Tessie'), no hace falta volver a usar la palabra reservada let. Simplemente tomamos el nombre de la Variable y la completamos con el nuevo valor:

```
let nombre = 'Mery' // Declaración de la Variable
        nombre = 'Tessie' // Modificación del valor
```

De esta manera, el programa que creamos podría saludar a cada usuario por su nombre:

```
'Hola ' + nombre + ', ¿cómo estas?'
```

Recordá

la Variable puede modificarse (de ahí su nombre). Es decir, su valor puede cambiar con el uso o el tiempo. Podemos declararla asignándole un valor. Si no se lo damos, se completará más adelante en el programa.

Variables Con Datos Numéricos

Si el valor de la Variable es un dato numérico podemos llamarla directamente.

Por ejemplo, un banco podría calcular su comisión de esta manera:

```
let dineroEnLaCuenta = 100;
let nuevoDeposito = 1200;
let comisionBancaria = 0.05;
(dineroEnLaCuenta+nuevoDeposito)* comisionBancaria
```

(copiar y pegar en la consola)

Concatenación De Variables Y Strings

Las Variables pueden combinarse entre sí, y con otros datos, para manipular la información y obtener un resultado determinado. Para unirlos, se usa el operador matemático de suma (+).

Veamos un ejemplo de concatenación de Variables y Strings:

```
"Mi nombre es "+ nombre + " , tengo" + edad + " y estoy cursando en el" + stage + "."
```

El usuario1 vería esto al completar las Variables:

Mi nombre es Mery, tengo 19 y estoy cursando en el stage 1.

Mientras que el usuario2, vería esto otro:

Mi nombre es Lucas, tengo 17 y estoy cursando en el stage 2.

Keywords para definir variables

Si bien a los fines de este curso, siempre usaremos la palabra reservada let para definir las Variables, hay otras formas de hacerlo con algunas diferencias entre sí.

- var: Es una forma más flexible para crear una Variable ya que nos permite volver a crearla y reemplazar la anterior.
- let: Permite actualizar una Variable pero no volver a crearla.
- const: Permite crear una Variable que se mantendrá constante durante todo el programa. Es decir, no se podrá actualizar ni cambiar.

GENERACIÓN T

streambe