05 Tipos de datos

JavaScript es un lenguaje dinámico (no hay que especificar qué tipo de dato tendrá la variable una variable puede cambiar su tipo de dato) y débilmente tipado (el intérprete no detecta el tipo de dato que va a emplear hasta que no se produce su ejecución).

El último estándar ECMAScript define ocho tipos de datos:

Siete tipos de datos que son primitivos:

- Boolean. true y false.
- null. Una palabra clave especial que denota un valor nulo. Como JavaScript es case-sensitive, null no es lo mismo que Null, NULL, o cualquier otra variante.
- undefined. Una propiedad de alto nivel cuyo valor no es definido.
- Number. Un número entero o un número con coma flotante. Por ejemplo: 42 o 3.14159.
- BigInt. Un número entero con precisión arbitraria. Por ejemplo: 9007199254740992n
- String. Una secuencia de caracteres que representan un valor "Hola"
- Symbol (nuevo en ECMAScript 6). Un tipo de dato cuyas casos son únicos e inmutables

y Object.

Aunque estos tipos de datos son una cantidad relativamente pequeña, permiten realizar funciones útiles con tus aplicaciones. Los otros elementos fundamentales en el lenguaje son los Objetos y las funciones.

Tipo de datos numérico

En este lenguaje sólo existe un tipo de datos numérico, al contrario que ocurre en la mayoría de los lenguajes más conocidos. Todos los números son por tanto del tipo numérico, independientemente de la precisión que tengan o si son números reales o enteros. Los números enteros son números que no tienen coma, como 3 o 339. Los números reales son números fraccionarios, como 2.69 o 0.25, que también se pueden escribir en notación científica, por ejemplo 2.482e12.

Con Javascript también podemos escribir números en otras bases, como la hexadecimal. Las bases son sistemas de numeración que utilizan más o menos dígitos para escribir los números. Existen tres bases con las que podemos trabajar

- Base 10, es el sistema que utilizamos habitualmente, el sistema decimal. Cualquier número, por defecto, se entiende que está escrito en base 10.

- Base 8, también llamado sistema octal, que utiliza dígitos del 0 al 7. Para escribir un número en octal basta con escribir ese número precedido de un 0, por ejemplo 045.
- Base 16 o sistema hexadecimal, es el sistema de numeración que utiliza 16 dígitos, los comprendidos entre el 0 y el 9 y las letras de la A a la F, para los dígitos que faltan. Para escribir un número en hexadecimal debemos escribirlo precedido de un cero y una equis, por ejemplo 0x3EF.

Tipo booleano

El tipo booleano, boolean en inglés, sirve para guardar, un verdadero o un falso. Se utiliza para realizar operaciones lógicas, generalmente para realizar acciones si el contenido de una variable es verdadero o falso.

```
let aceptaCdnes = true;
let mayorEdad = false;
```

Tipo de datos cadena de caracteres (string)

JavaScript sólo tiene un tipo de datos para guardar textos alfanuméricos y en él se pueden introducir cualquier número de caracteres. Un texto puede estar compuesto de números, letras y cualquier otro tipo de caracteres y signos. Los textos se escriben entre comillas, dobles o simples.

```
miTexto = "Acepto las condiciones";
miTexto = '23%%$ Letras & *--*';
```

Todo lo que se coloca entre comillas, como en los ejemplos anteriores es tratado como una cadena de caracteres independientemente de lo que coloquemos en el interior de las comillas.

Caracteres de escape en cadenas de texto

Hay una serie de caracteres especiales que sirven para expresar en una cadena de texto determinados controles como puede ser un salto de línea o un tabulador. Estos son los caracteres de escape y se escriben con una notación especial que comienza el caracter de barra invertida (también llamada contrabarra) y luego se coloca el código del carácter a mostrar.

Un carácter muy común es el salto de línea, que se consigue escribiendo n. Otro carácter muy habitual es colocar unas comillas, pues si colocamos unas comillas sin su carácter especial nos cerrarían las comillas que colocamos para iniciar la cadena de caracteres. Las comillas las tenemos que introducir entonces con " o ' (comillas dobles o simples). Existen otros caracteres de escape, que veremos en la tabla de abajo más resumidos, aunque también hay que destacar como carácter habitual el que se utiliza para escribir una contrabarra, para no confundirla con el inicio de un carácter de escape, que es la doble contrabarra .

Las principales secuencias de caracteres de escape

Salto de línea: \n
Comilla simple: \'
Comilla doble: \"
Tabulador: \t
Retorno de carro: \r
Avance de página: \f
Retroceder espacio: \b
Contrabarra: \\

Null

Null es empleado para inicializar una variable pero que no contenga ningún valor, por ejemplo:

```
edad = null;
```

Undefined

Cuando la variable no haya sido inicializada su valor será indefinido, por ejemplo:

```
var mayordeEdad;
mayordeEdad; //devuelve undefined
```

JavaScript es un lenguaje de tipo dinámico. Esto significa que no es necesario especificar el tipo de dato de una variable cuando se declara, y los tipos de datos son convertidos automáticamente de acuerdo a lo que se necesite en la ejecución del código (Type Coercion). Así, por ejemplo, se puede definir una variable de la siguiente manera:

```
var respuesta = 42;
```

Y luego, se le puede asignar una cadena a esa misma variable, por ejemplo:

respuesta = "Gracias por todo...";

Debido a que es un lenguaje de tipos dinámicos, esta asignación no causa un mensaje de error.

Para averiguar el tipo de dato de un valor o variable se emplea el operador typeof

```
> a = "Hello";
<- "Hello"
> typeof a
<- "string"
> typeof true
<- "boolean"
> typeof 3
<- "number"
> typeof {nombre:"Pedro",apellidos:"López"}
<- "object"
```

Como particularidad de JavaScript las variables de tipos primitivos son también objetos, de hecho disponen de propiedades como por ejemplo su longitud:

var a = “pepe”;

a.length

4