Javascript: Leng. Programación para web dinámicas (efectos)

Variables: “Caja para guardar valores. Tienen identificador (nombre para guardar valores), con ellas creamos lenguajes genéricos.

Ej.
Var algo; // declaramos variables
Var algo = “hola”; // inicializamos variables, en este caso “hola” sería el identificador

Nombre de Variable: formado por letras, Nº y símbolos ($ _. Y el 1º carácter no puede ser Nº, ni tampoco se una Keyword, ni incluir espacios.

Se sugiere el CamelCase.

Al final del código siempre se termina con ; o { }

Tipos de Variables:

1.Numéricas o Numbers=
- 2.enteros o *integer (Positivos y Negativos)*
- 3.decimales o *float (Positivos y Negativos)*
- Valores simbólicos = + Infinity, - Infinity , Nan (que sig. no es un numero)

**Para los decimales se usa punto y no coma.*

2.Cadena o String = Cadenas de textos, que se guardan en comillas dobles “ ” o simples.

carácteres especiales dentro de una string

“\n” ——— nueva línea dentro del string
“\t” ——— nueva tabulación dentro del string
“\”” ——— nueva comillas dentro del string
“\\” ——— devuelve \ dentro del string

**Las cadenas de texto no pueden usar operadores aritméticos (multiplicar, dividir o restar) pero sí sumar (+) para concadenar strings*

**Tampoco se debe mezclar operaciones entre number o string*

.

3. Boleamos o Booleans = Variables lógicas, son V (true) o F (false).

4. Undefined (Indefinido)

5. Null (nulo)

* Undefinded y Null = ausencia de valor significativo, son valores en si mismo pero no poseen info.

su ≠ es un accidente de JS y no importa casi siempre

Operadores:

Permiten manipular las var, hacer operaciones matemáticas o comparar var.

Incremento = ++ / Decremento = --

Ej. var age = 33;
Age ++ ;
//valor final en consola = 34
Age -- ;
//valor final en consola = 33

1. Operador lógico = negación var visible == true;
Var oculto != visible;

Si contiene/ var string = “” // false
var string = “ ” // true

&& (y) = se cumple si son dos && es true
ll (o) = se cumple si uno es true

Módulo % = es el residuo de una división

Ej 10 % 5 = 0

2. Operadores relacionales = Booleanos

< , > =, < =, === (comparar exactamente) , !== (diferente exactamente)
En el caso de los strings las Mayúscula son siempre menos que las minúsculas
"Zeyla" > "ana";
// return false
//No entiendo: Funciona lo contrario en consola ?¿?¿

* Sólo un valor no es igual en si mismo —— NaN === NaN // returns false porque si una operación es indefinida no es igual a otra operación indefinida

== (mellizos) / === (Idénticos)

3. Operadores aritméticos (de asignación)

+= , *=, -= , /=, %= (suma, multiplicación, resta, división, módulo)

ej. 100/0 // returns Infinity
100/-0 // returns -Infinity

Infinity - Infinity = NaN

Modulo
n%2 == 0 (par)
n%2 == 1 (impar)

“ “ === 0 // no son idénticos

4. Operadores unarios = Son valores escritos **NO** signos.

- **typeof**

```
var forma="redonda"
typeof forma
// devuelve “string”
```

* El operador - es unario y binario.

4. Operador ternario o condicional= Son valores en si mismos pero no poseen información

Ej.
True ? 1 : 2 // escoge el valor central, osea 1.
false ? 1 : 2 // escoge el valor de la derecha osea 2.

console.log ("") // imprime algo en la consola
document.write (“”) // Imprime algo en la web
prompt (“Que edad tienes?”) // guarda valor input del usuario
alert (“estas acá”); // aparece un pop up con lo que está dentro del paréntesis.

Propiedades Javascript

DICEN algo sobre el valor asociado

- **String length** =
  var example
  example.length // Largo de un string

Métodos Javascript

Hacen algo sobre el valor asociado

- **Replace ();**=
  var example=“This is a post”;
  example = example.replace(“hola”);

```
console.log(example);
//No entiendo, no me funciona en consola
```

- **Math.round ()** = redondea Nº

```
var roundUp = 1.5;
var rounded = Math.round(roundUp );
console.log(roundUp);
//No entiendo que hace la función
```

- **.toString ()**; = devuelve una cadena

```
var n = 256;
 n=n.toString ();
 console.log(n);
```

- **str.charAt ()**= devuelve un carácter especificado de una cadena indicando el índice.

```
var cualquierCadena="Brave new world";
console.log("El carácter en el índice 0 es '" +
cualquierCadena.charAt(0) +      "'")
```

*METER NUMERO

-
- **str.indexOf (“ “)**= Al contrario de CharAt obtiene el índice de un carácter de un string. Nos entrega si un string está o no dentro de un Array.

```
var cualquierCadena="Brave new world"
document.write("<P>The index of the first w from the beginning is
" +
  cualquierCadena.indexOf("w"))         // Muestra 8
```

METER CARACTER

En caso de dos parámetros `cadena.indexOf(valorBusqueda[, indiceDesde])`

```
"Blue Whale".indexOf("Whale",5) // returns 5
```

//No entiendo preguntar en clases

- **str.lastIndexOf () =** La búsqueda se realiza empezando por el final de la cadena que realiza la llamada, empezando en **`indiceDesde`**.

```
"canal".lastIndexOf("a",2);
 // devuelve 1
```

//No entiendo bien preguntar en clases

- **str.toUpperCase ()** = Convierte el string a mayúsculas

- **str.toLowerCase ()** = Convierte el string a minusculas

- **string.slice ();** (Inicio Trozo, Fintrozo (sin incluirlo, sino el anterior, excepto cuando es negativo)

```
      var cadena1 = "La mañana se nos echa encima.";
      var cadena2 = cadena1.slice(3, -2);
      console.log(cadena2);
// returns mañana se nos echa encim

 var cadena2 = cadena1.slice(3, 8);
      console.log(cadena2);
// returns mañan

//Incluye el primer parámetro y no incluye segundo parámetro o índice
```

- **parseInt();** = Convierte string a números enteros.También lo ocupamos para sacar decimales a un Nº.
  Ej. parseInt(*string*, *base*);
  ej.parseInt(10);
  //10

* Me cuesta entender cuando hay más parámetros