



Introducción a Javascript

Clase 1



Variables



Variables

¿Qué es una variable?

- Es un espacio reservado en la memoria RAM de la CPU.
- Es una estructura que puede cambiar a lo largo de la ejecución de un programa.
- Está asociada a algún tipo de dato.

Declaración:

- Declarar una variable es reservar el espacio de memoria.
- La declaración se hace con la palabra reservada `var` seguida del nombre de la variable. Se debe declarar solamente una vez. (Preferentemente, utilizamos `let`)



Variables

Asignación e inicialización

- Para almacenar un valor en una variable utilizaremos el operador de asignación
- La primera vez que se le asigna un valor a la variable se la **inicializa**.
- La inicialización puede suceder al declarar la variable o durante el desarrollo del programa.

Ejemplos

var unaVariable;	->	Declaración
var otraVariable = 2	->	Declaración e inicialización (le asignamos el valor 2)
unaVariable = 8	->	inicialización (le asignamos el valor 8)



Variables

Nombre de variable

- En Javascript el nombre de la variable debe cumplir con los siguientes requisitos
 - Utilizar caracteres del **alfabeto inglés** (a-z, sin ñ, ni tildes, ni caracteres especiales)
 - Puede tener **números**, **PERO NO empezar** con uno
 - Puede empezar y/o contener => Guión bajo : **_** || Signo peso: **\$**
 - **No** puede tener **espacios en blanco**
 - JS es sensible a caracteres (**case sensitive**), diferencia minúsculas de mayúsculas.

Por ejemplo

`var unaVariable;`

`var unaVariableE;`

`var unaVARiable;`



Variables

Palabras reservadas

- Existe una serie de **palabras reservadas** que no pueden utilizarse como nombre de variables: *break, case, catch, continue, default, delete, do, else, finally, for, function, if, let, in, instanceof, new, return, switch, this, throw, try, typeof, var, void, while, with, abstract, boolean, byte, char, class, const, debugger, double, enum, export, extends, final, float, goto, implements, import, int, interface, long, native, package, private, protected, public, short, static, super, synchronized, throws, transient, volatile.*

Algunas recomendaciones

- Conviene que el **nombre de la variable tenga una relación con su contenido**, de esta manera al leer el código será más entendible el uso de dicha variable.
- Se puede utilizar una forma de escritura llamada **notación camello (Camel Case)**



Tipos de datos



Tipos de datos

¿Qué es un tipo de dato?

- Es una restricción impuesta por el sistema informático que determina los valores que va a manejar la variable.
- Es una estructura que puede cambiar a lo largo de la ejecución de un programa.
- Está asociada a algún tipo de dato.

Conozcamos los 7 tipos de datos que existen en Javascript

number - string - logical - object - function - undefined - null



Tipos de datos

Número (number)

- Punto flotante (**float**)
 - Son los que tenían "coma"
 - Se utiliza el punto para separar la parte entera de la "decimal"
 - De 0 a 9
- Infinito (**Infinity**)
 - Cuando el número es muy grande, se obtiene un **Infinity**
- No un número (**NaN**)
 - Al realizar una operación aritmética entre dos valores, si uno o ambos no son números, el resultado será un NaN (Not a Number)



Tipos de datos

Cadenas (**string**)

- Formada por cero o más caracteres (letras, dígitos y signos de puntuación)
- Se identifican porque su valor está encerrado entre
 - Par de comillas simples (')
 - Par de comillas dobles (")

Lógicos (**boolean**)

- Existen solamente dos valores lógicos
 - Verdadero (**true**)
 - Falso (**false**)



Tipos de datos

Objeto (**object**)

- Los objetos de Javascript son colecciones de propiedades y métodos
- En Javascript el object se divide en
 - Number
 - String
 - Boolean
 - Object
 - Date
 - Array



Tipos de datos

Funciones (function)

- Bloques de código que pueden ser llamados para realizar cierta tarea
- Se pueden reutilizar, permiten ahorrar escritura de código

Indefinido (undefined)

- Se obtiene este estado cuando
 - Se usa una propiedad de objeto que no existe
 - A una variable que se ha declarado aún no se le asignó un valor

Nulo (null)

- Indica que una variable está vacía



Operadores



Operadores

Conozcamos los 5 principales grupos de operadores que existen en Javascript

- Aritméticos (8)
- Concatenación (1)
- Asignación (6)
- Comparación (8)
- Lógicos (3)



Operadores

Aritméticos

- | | | |
|----------------------------------|---|----|
| ● Suma | → | + |
| ● Resta | → | - |
| ● Multiplicación | → | * |
| ● División | → | / |
| ● Resto | → | % |
| ○ El resto de la división entera | | |
| ● Incremento | → | ++ |
| ○ Suma 1 al valor numérico | | |
| ● Decremento | → | -- |
| ○ Resta 1 al valor numérico | | |
| ● Potencia | → | ** |
| ○ Potencia de un número | | |



Operadores

Concatenación

- Operador concatenador → +
- Si está entre cadenas, concatena
 - "hola" + " " + "mundo" → obtendremos string "hola mundo"
- Si está entre números, suma
 - 2 + 3 + 4 + 5 + 6 → obtendremos number 20
- Si tenemos números y "números"
 - 2 + 3 + "4" + 5 + 6 → obtendremos string "5456"
 - "2" + 3 + 4 + 5 + 6 → obtendremos string "23456"
 - 2 + 3 + 4 + "5" + 6 → obtendremos string "956"



Operadores

Asignación

- Asignación → =
 - **v1 = v2**, asigna a v1 el contenido de v2.
- Asignación con suma → +=
 - **v1 += v2**, asigna a v1 el contenido de v1 + v2 (v1 = v1 + v2)
- Asignación con resta → -=
 - **v1 -= v2**, asigna a v1 el contenido de v1 - v2 (v1 = v1 - v2)
- Asignación con multiplicación → +=
 - **v1 *= v2**, asigna a v1 el contenido de v1 * v2 (v1 = v1 * v2)
- Asignación con división → +=
 - **v1 /= v2**, asigna a v1 el contenido de v1 / v2 (v1 = v1 / v2)
- Asignación con resto → +=
 - **v1 %= v2**, asigna a v1 el contenido de v1 % v2 (v1 = v1 % v2)



Operadores

Comparación

- Igualdad → ==
 - Devolverá boolean true si los valores comparados son iguales.
- Igualdad estricta → ===
 - Devolverá boolean true si los valores y tipos de datos comparados son iguales.
- Desigualdad → !=
 - Devolverá boolean true si los valores comparados no son iguales.
- Desigualdad estricta → !==
 - Devolverá boolean true si los valores comparados o tipos de datos no son iguales.
- Mayor que → >
- Mayor o igual que → >=
- Menor que → <
- Menor o igual que → <=



Operadores

Lógicos



- Operador Y (AND) → **&&**
 - Este operador tiene la siguiente **tabla de la verdad**
 - Si todas las expresiones son verdaderas, el resultado será verdadero

Expresión 1	Operador	Expresión 2	Resultado
true	&&	true	true
true	&&	false	false
false	&&	true	false
false	&&	false	false



Operadores

Lógicos

- Operador O (OR)  
 - Este operador tiene la siguiente **tabla de la verdad**
 - Si todas las expresiones son falsas, el resultado será falso

Expresión 1	Operador	Expresión 2	Resultado
true		true	true
true		false	true
false		true	true
false		false	false



Operadores

Lógicos

- Operador NEGADOR (NOT) → !
 - Este operador tiene la siguiente **tabla de la verdad**
 - Invierte el valor lógico de la expresión

Expresión 1	Operador	Resultado
true	!true	false
false	!false	true



La sintaxis de Javascript



Sintaxis

Veamos algunas reglas de escritura particulares de este lenguaje

- No se tienen en cuenta espacios en blanco o saltos de líneas por fuera de las cadenas
- Distingue entre minúsculas y mayúsculas (es **case sensitive**)
- No se define el tipo de dato de las variables (**no es un lenguaje estricto**)
- Las variables se declaran con la palabra reservada **var** (nosotros usamos **let**)
- Cada **sentencia** debería terminar con **punto y coma (;)**
- Vamos a utilizar el modo estricto, el mismo se logra colocando el string "use strict" al comienzo del bloque de código Javascript, de tal forma que nos va a mostrar todos los errores de sintaxis.



Javascript y HTML, comentarios

Etiqueta `<script></script>`

- Es de apertura y cierre, lo único que contienen es código Javascript
- Se pueden colocar tanto en el head como en el body, tantas como sea necesario
- Tiene el atributo `src` que permite pasarle como valor la ruta de un archivo de extensión `.js`, que es la forma prolija y correcta de trabajar con Javascript; y la que utilizaremos en la materia

Comentarios

- En Javascript se puede comentar de dos maneras
 - Una solo línea `=> // Comentario`
 - Una o más líneas `=> /* abre-cierra */`
- Los comentarios sirven para ordenar el código y hacer no ejecutables las líneas de código; y los aprovecharemos para todo aquello que quieran que deje anotado en los archivos.



Consola



Herramientas de desarrolladores

Veamos algunas reglas de escritura particulares de este lenguaje

- En casi todos los navegadores existe una serie de herramientas para desarrolladores, entre ellas la consola (**console**)

¿Cómo se accede?

- En PC suele abrirse con la tecla F12.
- En Mac Cmd + Option + J (Chrome) o Cmd + Option + K (Firefox).
- Haciendo clic derecho en el "sitio" y seleccionar "Inspeccionar elemento".



Herramientas de desarrolladores

- Además, desde el código podemos escribir en la consola, para mostrar nuestros mensajes como desarrolladores, principalmente de las siguientes maneras
 - `console.log()` genera un mensaje de registro en la consola.
 - `console.info()` genera un mensaje como si fuera una información
 - `console.warn()` genera un mensaje como si fuera una advertencia.
 - `console.error()` genera un mensaje como si fuera un error



Manejo de información



Ingreso y egreso

Solicitud y muestra

- **prompt()**
 - Muestra una ventana emergente que solicita un valor con el cual trabajar
 - Devuelve un valor de tipo string o null
- **alert()**
 - Muestra una ventana emergente que informa sobre algo

Conversión numérica

- **parseInt()**
 - Convierte el valor entre paréntesis a un número entero, es decir, suprime la parte flotante
- **parseFloat()**
 - Convierte el valor entre paréntesis a un número flotante, sirve para enteros y flotantes.



Ingreso y egreso

Confirmación

- Cuando se necesita que el usuario ingrese un valor del tipo de dato boolean, podemos utilizar el método
- **confirm()**
 - Mediante una ventana emergente, solicita un valor lógico con el cual trabajar
 - Podrán ser los valores lógicos **true** y **false** (únicamente)



Generación de HTML desde Javascript

Escribiendo el documento

- Si bien los métodos `alert` y `console.algo` permiten mostrar información, existe un método del documento que es mucho más "sofisticado"
- **`document.write()`**
 - Es un método que permite generar código HTML desde Javascript para el egreso de información
 - A diferencia del `alert` y el `console.algo`, analiza el contenido de la cadena entre sus paréntesis y si detecta una seguidilla de código HTML, lo renderiza como tal, logrando generar la etiqueta identificada, en lugar del texto plano



Fin de la clase

Este es el espacio para dudas y/o preguntas existenciales