Clase 13

JavaScript

¿Qué es?

JavaScript es el lenguaje de programación del lado del cliente, esto significa que el navegador es quien lo interpreta. A diferencia de HTML y CSS, JavaScript no es estático, si no que es dinámico y le aporta funcionalidad a nuestra página.
JavaScript es un lenguaje de scripting multiparadigma y débilmente tipado.

¿Qué es?

Los programas escritos en este lenguaje se llaman scripts, se ejecutan en un navegador como texto sin formato y se van interpretando a medida que carga la página, el navegador es quien se encarga de compilar el código a medida que lo interpreta línea a línea.

¿Para qué es?

JavaScript en el navegador puede hacer todo lo relacionado con la **manipulación de la página web**, la interacción con el usuario y el servidor.

- ✔ Cambiar todo el contenido de una página web (tipo de letra, colores, animaciones, etc.)
- ✔ Enviar información a través de la red a servidores remotos, descargar archivos.
- ✔ Almacenamiento local en el navegador (recuperar, almacenar información durante la ejecución y visualización de la página web).

¿Para qué NO es?

JavaScript no puede acceder a las circuitos integrados de una computadora tales como:

- ✔ Disco Duro (Acceso a eliminar información, modificar o leer).
- ✔ Acceso a la memoria RAM, ROM.
- ✔ Acceso a la tarjeta de RED o Procesadores.

El objetivo de **JavaScript** en el navegador solo se limita al uso exclusivo a lo que una **página web** te puede brindar.

Consola

Ahora, abrimos la consola en el navegador y ejecutamos nuestro primer script:

alert('Hola, mundo!’)

Consola

Lo que hicimos con anterioridad (ejecutar JavaScript desde la consola del desarrollador) no es la forma correcta para hacer proyectos de desarrollo de software.
La consola del desarrollador, sirve para los casos que necesitamos ir tras esos errores de programación difícil de encontrar.

Vincular JS con HTML

Vamos Buenos Aires

Hay dos formas de incluir nuestro código JavaScript en nuestro documento HTML.

- ✔ Insertar JavaScript en HTML utilizando la etiqueta:

```
<script> Aquí va tu código</script>
```

- ✔ Agregar código JavaScript en un archivo externo:

```
<script src="archivo.js"></script>
```

Variables

Una Variable es un valor que puede cambiar con el tiempo (mutable), al contrario de una Constante que no cambia una vez definida (inmutable).
Tanto las variables como las constantes en JavaScript son almacenamiento con nombre y sirven para guardar información (valores, datos).

Variables

Las variables se usan como nombres simbólicos para valores en nuestra aplicación. Los nombres de las variables se rigen por ciertas reglas: tienen que comenzar por una letra, un guion bajo o el símbolo de $, los valores subsiguientes pueden ser números, JavaScript diferencia entre mayúsculas y minúsculas, por lo tanto las letras incluyen desde la "A a la "Z" y desde la "a" a la "z".

Variables

Para asignarle un nombre a las variables o constantes (llamados también identificadores), deben cumplir las siguientes reglas:

- ✔ El nombre debe contener solo letras, dígitos o los símbolos $y _.
- ✔ El primer carácter no debe ser un número.

Nombres reservados:

- ✔ ver acá

Variables

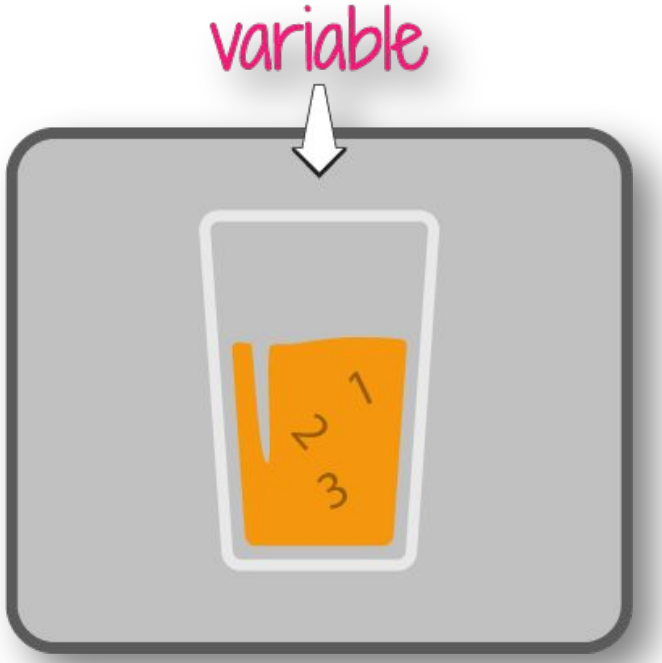
Hay 3 tipos de variables en JavaScript:

- ✔ **var:** declara una variable, iniciándola opcionalmente a un valor. Podrá cambiar el mismo y su scope es global o de función.
- ✔ **let:** declara una variable en un bloque de ámbito, iniciándola opcionalmente a un valor. Podrá cambiar su valor.
- ✔ **const:** declara una constante de sólo lectura en un bloque de ámbito. No será posible cambiar su valor mediante la asignación.

Var

var: las variables se hacen visibles en el ámbito global, es decir, que sin importar donde se definan, puede ser accedida desde cualquier parte del documento y permite que su valor pueda ser reasignado. El uso de ésta, puede dar a resultados inesperados, por eso, hay que tener cuidado de cómo se usa.

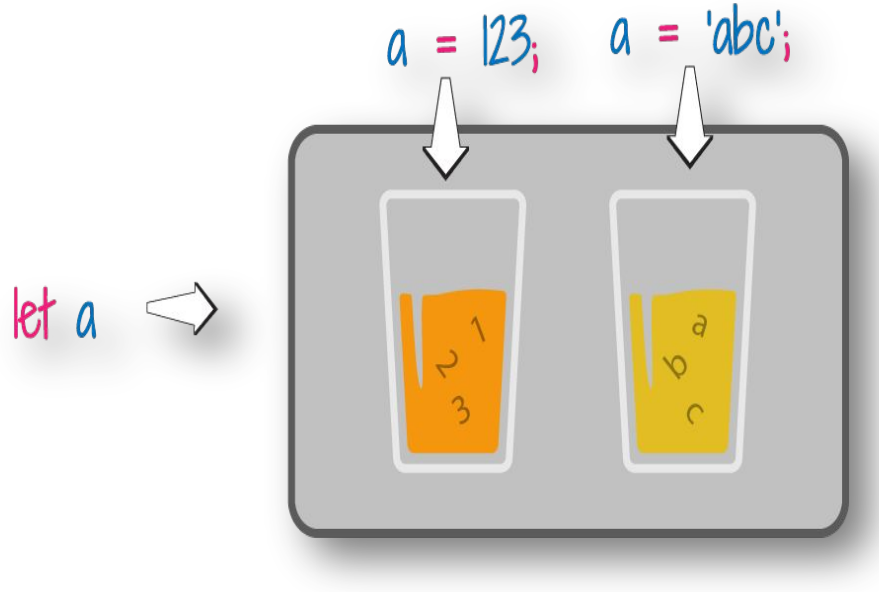
```
var nombreVariable;
var a;
var b;
```



Let

let: el alcance de estas variables, es que solo pueden ser accedidas dentro del bloque donde se definen. También, permiten que su valor pueda ser reasignado.

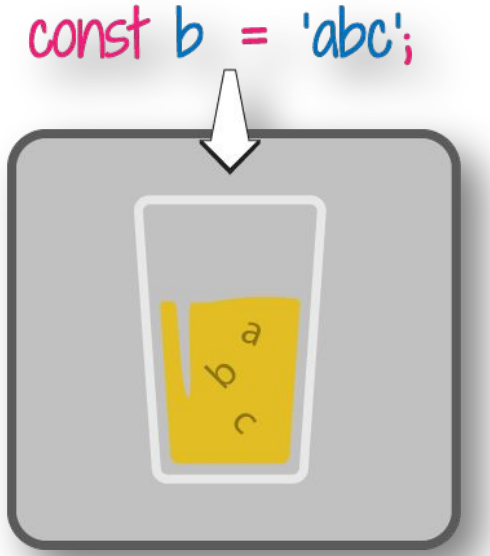
```
let nombreVariable = 'texto';
let a = 'abc';
a = 123;
let b = 1;
b = 5;
```



Const

const: estas variables (al igual que “let”) solo pueden ser accedidas dentro del bloque donde están definidas, pero no permite que su valor sea reasignado, es decir, la variable se vuelve inmutable.

```
const nombreVariable = 'texto';
const a = 'Hola Mundo';
const b = 'abc';
const c = 123;
```



Ejemplos de variables

```
// constantes
//siempre hay que inicializar
// una constante
const variableConstante = "";
const valorDeReferencia = 9.8;
// variable en desuso
var variableGlobal;
var definicionAntigua;
```

```
// Variables
// Camel Case
let nombreVariable;
// Pascal Case
let NombreVariable;
// Snake Case
let nombre_variable;
```

Ámbito de una variable

Cuando declaramos una variable fuera de una función se la denomina variable global. Cuando declaramos una variable dentro de una función se la denomina variable local, porque está disponible solo dentro de esa función donde fue creada. Las variables en JavaScript pueden hacer referencia a una variable declarada más tarde. Este concepto se lo conoce como **hoisting**. Las variables son "elevadas" a la parte superior de la función, las variables que no se han inicializado todavía devolverán un valor **undefined**.

Tipos de Datos

- ✔ **String:** Secuencia de caracteres que representan un valor.
- ✔ **Number:** Valor numérico, entero, decimal, etc.
- ✔ **Boolean:** Valores true o false.
- ✔ **Null:** Valor nulo.
- ✔ **Undefined:** Valor sin definir.
- ✔ **Symbol:** Tipo de dato cuyos casos son únicos e inmutables.
- ✔ **Object:** Objeto. {} Puede contener más variables en su interior.

Funciones String

Funciones / Propiedad	Descripción
string.toUpperCase()	Retorna el mismo texto (string) con las letras en mayúsculas
string.toLowerCase()	Retorna el mismo texto (string) con las letras en minúsculas
string.length	Retorna la cantidad de letras del texto (string)
string.repeat(n)	Retorna un texto repetido n veces
string.replace(str1,str2)	Retorna un texto reemplazando el texto str1 con str2

parseInt() y parseFloat()

parseInt() y parseFloat() son funciones creadas para analizar un string y devolver un número si es posible. JavaScript analiza la cadena para extraer las cifras que encuentre al principio, estas cifras al principio del string son las que se transforman a tipo numérico. Cuando se encuentra el primer carácter no numérico se ignora el resto de la cadena. Si el primer carácter encontrado no es convertible a número, el resultado será NaN (Not a Number).

Number

Ignora los espacios al principio y al final, pero, diferencia de los métodos anteriores, cuando un string contiene caracteres no convertibles a números el resultado siempre es NaN, no trata de 'extraer' la parte numérica. Con Number() podemos convertir booleans en números, false siempre se convierte en 0 y true en 1.

Conversion implicita '+'

La conversión implícita es una forma de conversión rápida a número. Este tipo de conversión, igual que Number(), devuelve NaN si el string contiene caracteres no numéricos.
Number() y '+' convierten el string vacio en 0.
Number() y '+' de un boolean devuelven 0 para false y 1 para true.