Java Script

**Clase 1 (info básica)**

Una vez vinculado con Java el html, crear en JAVA un “alert (“mensaje enviado al browser”);

De esa manera sabremos si el vinculo con el html quedó bien hecho.´

Una manera mas prolija de hacerlo es con console.log(“mensaje enviado a la consola del browser”);

De esa manera el mensaje aparecerá solamente en la consola del inspector de Google Chrome.

Es recomendable dejar en el código todas las variables juntas y todo el calculo abajo.

**Resumen**

. En JS podemos agregar comentarios de una sóla línea con la combinación “//”, o comentarios de *bloque*, de varias líneas, envolviendo el comentario con los signos “/\* \*/”:

Ejemplo:

*// comentario de una línea*

*/\*   comentario de  
     bloque de varias líneas \*/*

*Palabras reservadas de Java:*

Algunos ejemplos de estos términos son: break, case, catch, continue, default, let, delete, do, else, finally, for, function, if, in, instanceof, new, return, switch, this, throw, try, typeof, var, void, while, with, etcétera. En el curso aprenderemos a utilizar los más importantes para construir aplicaciones.

Variables:

Una variable es un espacio reservado en la memoria dónde almacenamos algún tipo de dato, y que puede variar con el tiempo y transcurso de nuestra aplicación

Hay dos procesos en la declaración de variables:

1. Declaración: Es la creación de la variable. Se escribe let o const seguido del nombre de la variable. Ejemplo “let colores” (sin las comillas)
2. Asignación

Dos tipos de Variables:

let: Puede recibir multiples asignaciones en el trascurso de la aplicación.

const: Puede recibir una única asignación al momento de su declaración.

**Tipos de valores:**

* Number: Es un valor numerico (“integer” si es un valor entero “float” si es un valor decimal)
* String: Compuesto por uno o mas caracteres y se define entre comillas.

Ambos tipos de valores se pueden convinar.

Para asignar la variable se hace por medio del = (igual).

let nombre = "Gabriel";

Entonces se escribiría asi: let nombre = "Gabriel"

**Operaciones con Variables numericas:**

Son operaciones comunes matemáticas, que pueden ser suma; resta; multiplicación; división.

Ejemplo:

let numeroa = 8;

let numerob = 2;

let numeroc = 1;

let suma = numeroa + numerob;

let resta = numeroa - numerob;

let division = numeroa / numerob;

let multiplicasion = numeroa \* numerob;

let calculo = ((numeroa + numerob) / numerob) + numeroc;

**Operaciones con Variables string:**

Estas variables pueden concatenarse y crear nuevas cadenas.

Por ejemplo:

let modelo ="Fiat UNO Fire "

let año = "2006 "

let cilindrada = "1.3"

let autodegabriel = modelo + año + cilindrada;

***IMPORTANTE:*** *Los espacios se ponen agregándole un espacio al finalizar el valor de la variable.*

Prompt, Consola y Alert

**Prompt():** Con ella se puede solicitar un dato al usuario para asignar el valor ingresado a una variable. Todo lo que ingrese por Prompt se lo conoce como “entradas”.

Ejemplo de uso:

let nombreingresado = prompt("Ingrese su nombre");

**Console():** Por medio de console.log() podremos mostrar el valor de una variable en la consola haciendolo de esta forma:

let segundonombre = "Alejandro"

console.log(segundonombre)

O mostrar un mensaje en particular escribiendo entre paréntesis de esta manera:

console.log("Mensaje de prueba")

**Alert():** Es conocido como “salida”, ya que salen de la aplicación como resultado de un procesamiento, siendo visibles por el usuario.

Por ejemplo:

**Algoritmo**

Es el conjunto de instrucciones ordenadas que programamos para realizar una tarea.

Basicamente es darle el paso a paso de como hacer algo a la computadora.

Ejemplo:

let entrada = prompt("Ingresa tu nombre")

let salida = "Hola" + " " + entrada + " " + "bienvenid@ a nuestra pagina!!"

alert(salida);

Definimos una variable y la asignamos como prompt, luego definimos otra variable como mensaje y por ultimo definimos un alert que use esa variable como salida.

**Clase 2 ( CONTROL DE FLUJO)**

**Condicionales:**

Vamos a encontrar los valores booleanos: Estos son TRUE or FALSE.

Para el armado de un condicional nos encontramos con las siguientes estructuras:

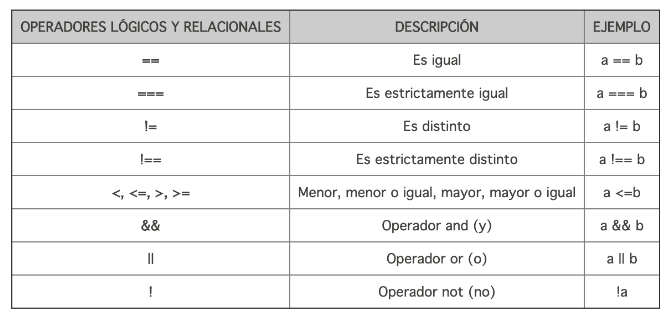
* **Estructura if:**  Se utiliza para ejecutar un bloque si la condición es verdadera.   
    
  Ejemplo de aplicación: if (dentro del paréntesis va lo que queremos analizar, por ejemplo (es la variable nombre==”Gabriel”); Luego, en las llaves se va a indicar donde queremos que nos muestre la respuesta o que queremos que haga cuando la respuesta sea verdadera.   
    
  Operadores: “=” Define la variable “==” Compara la variable con un valor
* **Estructura if…else:** Se utiliza para sumar un bloque de código que se ejecutara únicamente si la primer comparación (el if) es falso.   
  Siempre se escribe luego de cerrar el primer bloque con ;}
* **Estructura if…else if:** Se utiliza cuando se quiere comparar mas de un valor. Lo que va a hacer esta estructura es definir primero que pasa si la comparación es verdadera; después con un else if va agregar mas comparaciones que se ejecutaran de arriba hacia abajo y de izquierda a derecha; por ultimo va a agregarse la estructura else, la cual arrojara un valor en caso que ninguno de los anteriores se haya verificado.

**Variable booleana:**

Este tipo de variable nos va a permitir utilizar una variable que sea igual a una función y luego utilizar el resultado de esa función en conjunto a las estructuras if; else; else if; antes vistas.

**OPERADORES LÓGICOS Y RELACIONALES**

Son elementos que nos permiten evaluar valores y/o variables.



**Operadores de igualdad: (==) (===):**

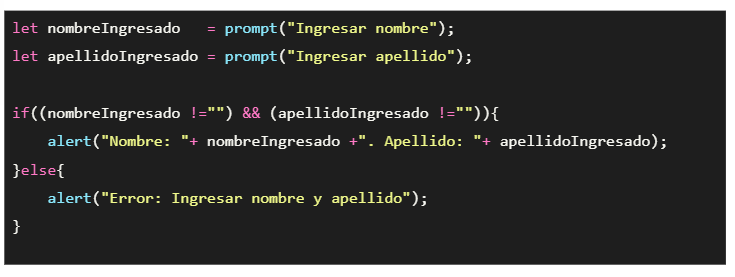
Sirve para preguntarnos si dos valores son iguales. La diferencia entre (==) y (===), es que == compara dos valores sin importar el tipo de dato, en este caso nos referimos que no importa si el valor ingresado es un numero o un string, si ambos son iguales, entonces la comparación será verdadera.   
En el caso de ===, la comparación va a ser estricta y compara valores string con string y valores numéricos con valores numéricos.

**Operadores de desigualdad: (!=) (!==):**Sirve para saber si dos variables o valores son iguales. Al igual que el caso anterior, con un igual compara sin importar el tipo de dato, con dos iguales compara estrictamente.

**Operadores lógicos (&&); ||; (!):**

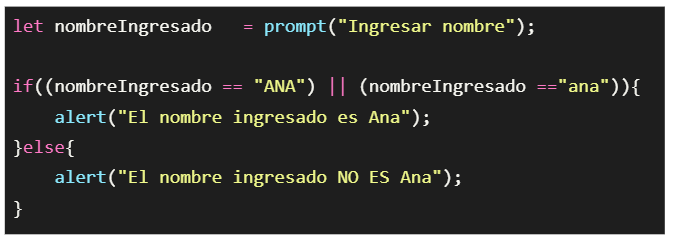
1. **&& (and):** Sirve para comparar dos valores booleanos. Es verdadera solamente si ambos valores son verdaderos.
2. **|| (or):** Sirve para comparar dos valores booleanos, la comparación es falsa si ambos valores son falsos.
3. **! (not):** Sirve para transformar e invertir el valor de la verdad o falso, ejemplo !true (es falso); !false (es verdadero)

**CONDICIONES COMPUESTAS CON && (AND)**



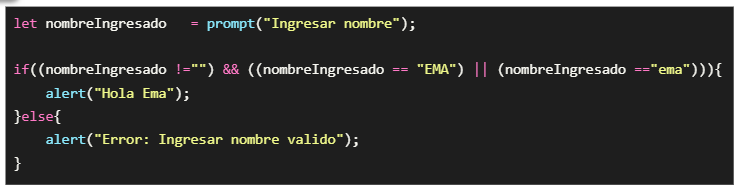
En este caso se esta comparando que ambas variables tengan algún valor cargado, ya que si alguna de ellas esta vacia, el navegador tirara el alert.

**CONDICIONES COMPUESTAS CON || (OR)**



Aca lo que se haces es decir, si el valor ingresado es alguno de estos dos, entonces, la variable es verdadera, sino, es falsa y tenes que mostrarme el mensaje de alerta.

**COMBINACIONES DE AND Y OR**

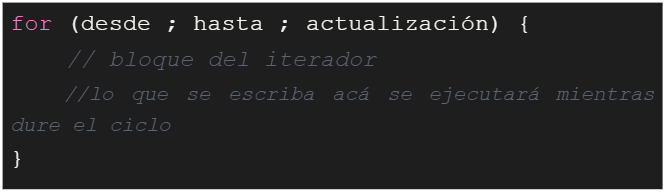
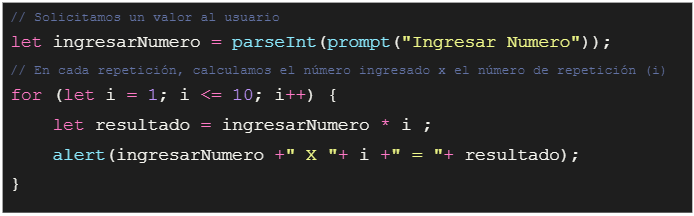
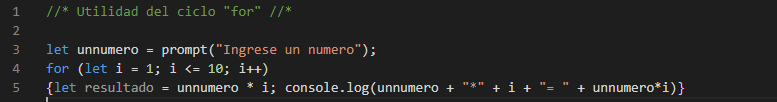
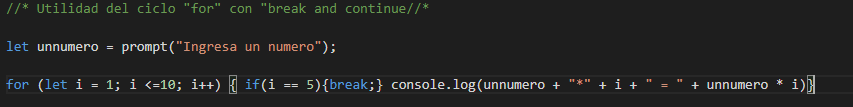
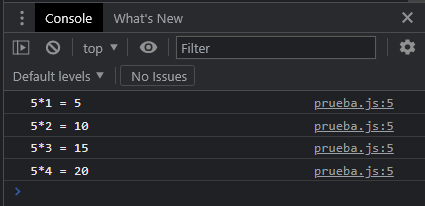
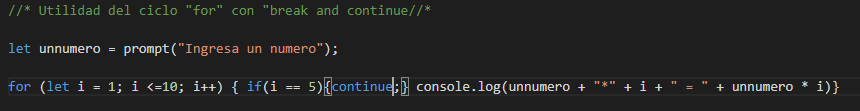
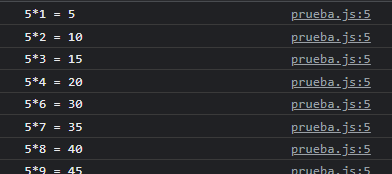
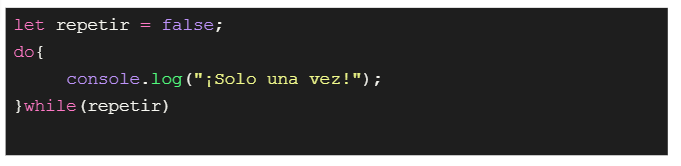


Lo que hacemos aca es primero asegurarnos que se haya ingresado un valor y que no hayamos dejado un campo vacio. El paso siguiente es fijarse si el valor ingresado es alguno de los dos admitidos como verdaderos, en caso contrario mostrara una aleta.

**Clase 3 ( CICLOS / ITERACIONES)**

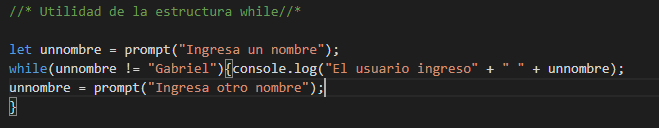
**CICLO:** Son estructuras que nos permiten repetir un conjunto de instrucciones (bloques) una cantidad determinada de veces.

Tipos de ciclos

* **Ciclos x conteo:** La estructura es **for**, la cual garantiza un numero de repeticiones hasta que una variable numérica tenga un valor especificoy se ecribe de la siguiente manera  
    
    
    
    
  (la “i” significa índice)  
    
    
    
  **Utilidad del Break y Continue**  
    
  Estas sentencias lo que nos permiten es cortar una iteración o saltear una iteración.  
    
    
  Ejemplo Break:  
    
    
    
    
  Ejemplo Continue  
    
    
  
* **Ciclos condicionales:** Implican comparaciones de todo tipo. De la misma forma que construimos los condicionales ***if***, las estructuras de este tipo interpretarán el bloque si la comparación es verdadera, sólo que a diferencia de ***if***, las instrucciones se repiten hasta que el condicional se evalúa como falso. Las estructuras ***while*** y ***do...while*** son de este tipo.  
    
    
  

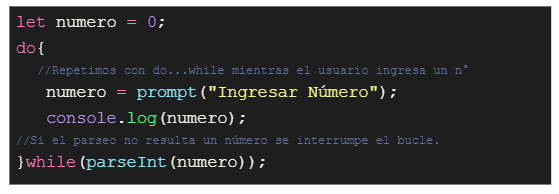
**While:**

Cuando existe la necesidad de utilizar un bucle, pero su repetición está condicionada por el valor de una o más variables no numéricas, la estructura ***while*** es más apropiada.  
El bucle continuara hasta que se ingrese un valor verdadero.

  
  
Es de suma importancia que luego de indicar que hacer si el valor es falso, indicar una acción siguiente, ya que sino se generara un bucle infinito.

Si el valor que se ingresa primero ya es correcto, entonces no se generará ninguna repetición.

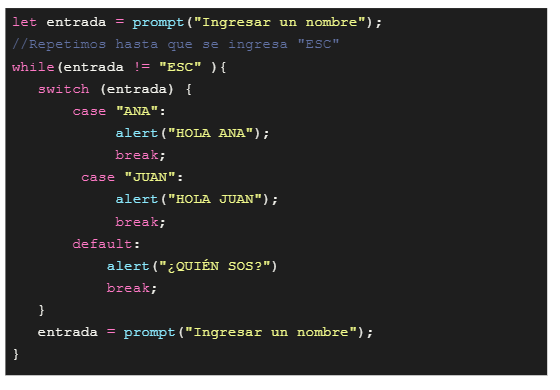
**Do while:** A diferencia del While, este permite evaluar el condicional al final de la iteración, por lo que garantiza al menos una iteración en caso de ser el valor falso.  
Un caso de ejemplo de uso es para el logeo de un usuario.



Parseint(string, base), Lo que hace esta función es convertir (parsear) un argumento de tipo cadena y devuelve un entero de la base especificada.  
Si encuentra un carácter que no es un numeral de la base especificada, lo ignora a el y a todos los caracteres correctos siguientes.

**Estructura Swhitch:** Lo que permite esta estructura es poder evaluar valores específicos y ejecutar un bloque de código puntual.   
Por medio de Shitch (variable) {case (escribimos el valor); break (para saltar a la siguiente comprobación; default (En caso que no sea ninguno de los case que hacer).

Ejemplo:



En conclusión es importante distinguir que en programación una cosa es la resolución del problema y otra es la resolución del problema de la manera mas apropiada.

Hay diferentes formas de afrontar el mismo problema. Requiere practica saber como combinar todo lo visto.

**Clase 4 ( PROGRAMACION CON FUNCIONES)**

Que son las funciones:   
  
Las funciones son elementos que creamos con la intención de agrupar instrucciones a ser utilizadas una o más veces, a lo largo del script. En programación, el uso de funciones permite separar un problema en partes más pequeñas, independientes y reutilizables.

Como declarar una función:  
  
