**JAVASCRIPT**

1. **INTRODUCCION**

* **Enlace del html al script**

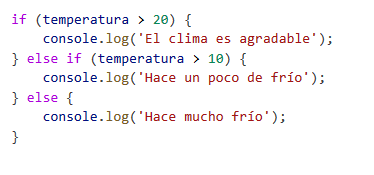


* **Tipos de datos**
  + String
  + Boolean
  + Number
  + BigInt
  + Symbol
  + Null -> falsy
  + Undefined -> falsy
  + Object
  + NaN
* **Mostrar Datos**
  + **console.log('mensaje').** Muestra un mensaje genérico en la consola
  + **console.error('mensaje').** Muestra un mensaje de error en la consola.
  + **console.warn('mensaje').** Muestra un mensaje de advertencia en la consola.
  + **console.debug('mensaje').** Muestra un mensaje de depuración en la consola
  + **console.info('mensaje').** Muestra un mensaje de información en la consola
  + **alert(‘ ‘ ):** Lo muestra en una ventana
* **Recoger Datos**

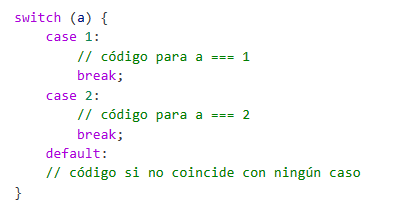
****

1. **CONDICIONALES Y BUCLES**

* **Condicionales**
  + **If**

****

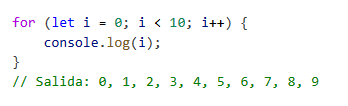
* + **switch**

****

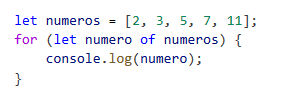
* + **ternario**

****

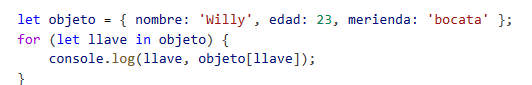
* **Bucles**
  + **For**

****

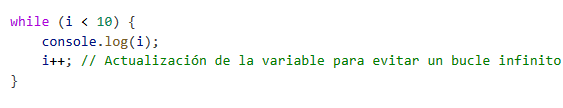
* + **For of**

****

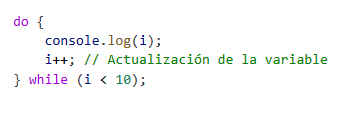
* + **For in**

****

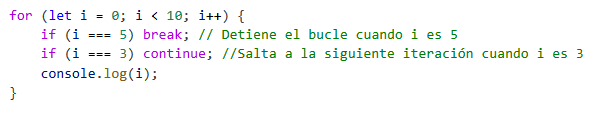
* + **While**

****

* + **Do while**

****

* + **Instrucciones continue y break**
    - **Break**: rompe el bucle, manda la ejecución a la primera instrucción fuera del bucle
    - **Continue:** termina la iteración actual del bloque, salta a la siguiente iteración del bucle

****

1. **ARRAYS Y JSON**

* **Arrays**
  + **Declaraciones**



****

****

* + **Acceso a elementos**

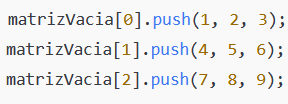
****

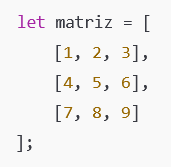
* + **Tamaño**

****

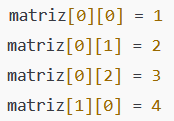
* **Array Multidimensional**
  + **Declaraciones**

****

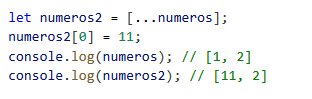
****

****

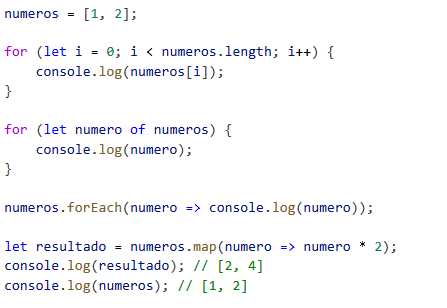
* + **Acceso a elementos**

****

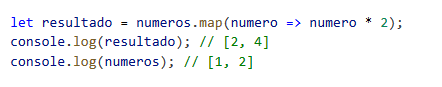
* + **Copia de arrays**

****

* + **Recorrer un array**

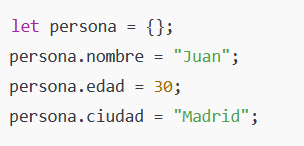
****

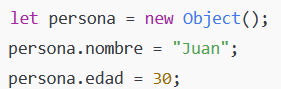
* + **Map**

****

* **JSON**
  + **Declaración**

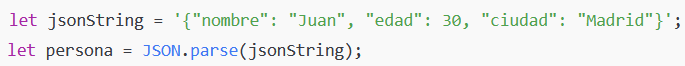
****

****

****

****

* + **Convertir un string a JSON**

****

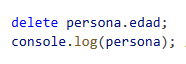
* + **Convertir JSON a string**

****

* + **Acceso a elementos**

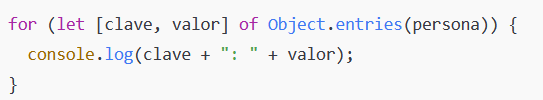
****

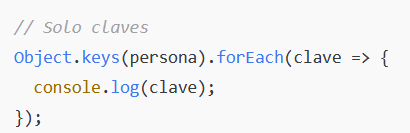
* + **Eliminar una propiedad**

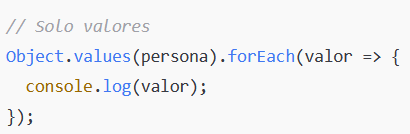
****

* + **Recorrer un JSON**

****

****

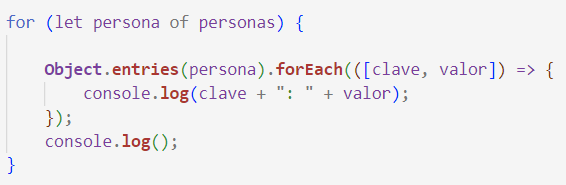
****

****

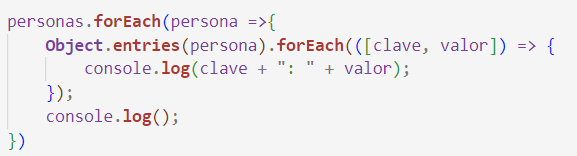
**Recorrer JSON array de objetos**

****

**For of itera sobre cada entrada {} del array y entries sobre cada propiedad**

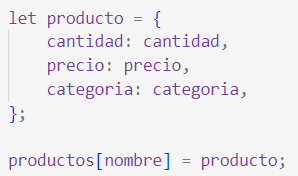
****

**Otro forma de for of**

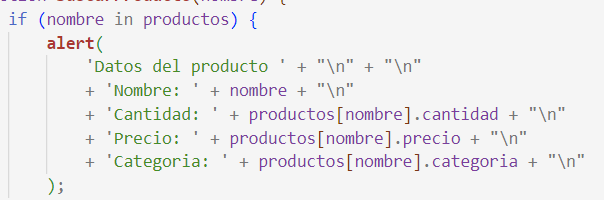
****

**Ejemplos:**

* **Agregar a un JSON con una key**

****

* **Buscar por la key si existe un producto**

****

**METODOS DE ARRAYS**

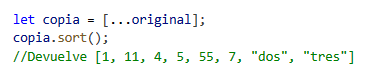
* **.indexOf(valor):** Devuelve el índice del elemento, -1 si no lo encuentra. Admite un segundo parámetro para indicar la posición de comienzo de la búsqueda.



* **filter(función):** Devuelve un nuevo array con los elementos que cumplen la condición.

****

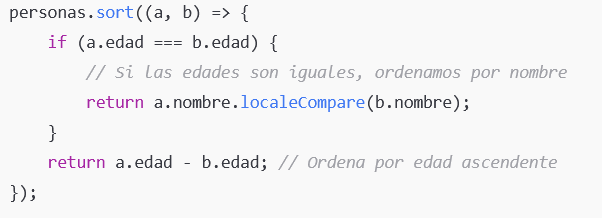
* **.sort():** Mutable, ordena sobre el propio array modificándolo

****

* **toSorted():** No mutable, devuelve una copia ordenada del array

****

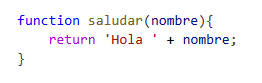
**Ordenacion personalizada**

****

* **push(…elementos):** Añade uno o más elementos al final del array.
* **pop():** Elimina el último elemento del array y lo devuelve
* **shift():** Elimina el primer elemento del array y lo devuelve.
* **unshift(…elementos):** Añade uno o más elementos al principio del array.
* **join(separador):** Devuelve una cadena con todos los elementos del array unidos
* mediante el valor del separador.
* **split(separador):** Divide una cadena según el separador, devuelve un array con las
* Subcadenas
* **concat(…arrays):** Combina dos o más arrays en un nuevo array
* **flat(profundidad=1):** Aplana arrays anidados en un nuevo array hasta la profundidad indicada.
* **fill(valor, inicio?, fin?):** Llena el array con el valor indicado desde los índices de inicio hasta fin
* **reverse():** Invierte el array
* **toReversed():** Devuelve un copia del array invertido
* **slice(inicio, fin?):** Devuelve una copia de una porción del array con los valores indicados en el rango

1. **FUNCIONES**

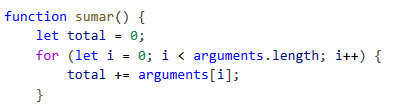
* **Declaracion**



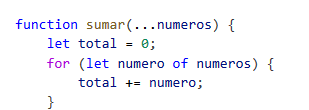
Valor por defecto



* **Arguments:** todos los argumentos que se pasaron a la función

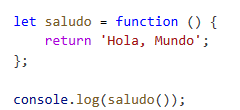
****

* **Operador … :** agrupar un numero indefinido de argumentos en un array

****

****

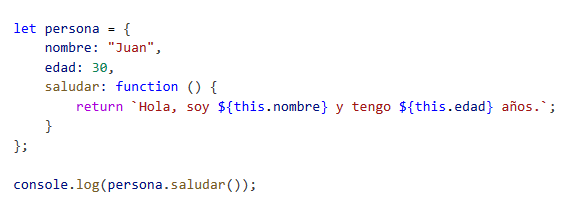
* **Funciones anónimas:** funciones sin nombre que se pueden asignar a una variable

****

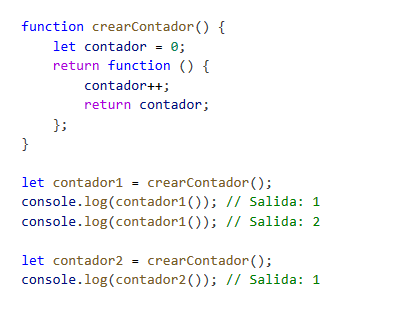
* **Arrow functions:** forma abreviada de escribir funcionas

****

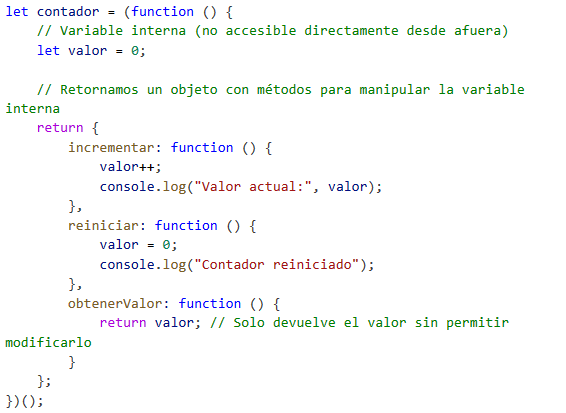
**Ejemplo completo:**

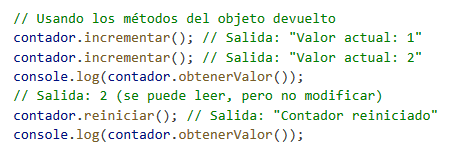
****

* **Clousures:** Una función dentro de una función, la función externa inicializa el entorno y la función interna es la que utilizamos

****

* **Funciones autoinvocadas:** se define y se ejecuta en el momento en que se crea

****

****

1. **OBJETOS PREDEFINIDOS**

* **Propiedades**
  + **globalThis**

****

* **Funciones**
  + **isFinite(valor):** Devuelve true si el valor es finito, false en caso contrario o NAN.
  + **parseFloat(cadena):** Convierte una cadena en un número flotante.
  + **parseInt(cadena, base=10):** Convierte una cadena en un número entero. Opcionalmente se puede indicar la base para la cadena de entrada, por defecto 10.
* **Funciones temporizadores**
  + **setTimeout(función, tiempo):** Ejecuta la función indicada después de un tiempo específico. El tiempo se indica en milisegundos. Devuelve el ID del temporizador creado.
  + **clearTimeout(id):** Cancela un temporizador creado mediante “setTimeout”
  + **setInterval(función, tiempo):** Ejecuta la función indicada de forma repetitiva según el tiempo indicado. El tiempo se indica en milisegundos. Devuelve el ID del temporizador creado.
  + **clearInterval(id):** Detiene la ejecución de una función repetitiva establecida mediante “setInterval”.

1. **STRINGS**

* **Funciones**
  + **charAt(posición):** Devuelve el carácter en una posición específica dentro de la cadena
  + **.concat(...cadenas):** Combina una o más cadenas y las une a la cadena original
  + **.indexOf(subcadena, posiciónInicio=0):** Devuelve el índice de la primera aparición de una subcadena, -1 si no se encuentra.
  + **.slice(inicio, fin):** Extrae una porción de la cadena entre los índices inicio y fin
  + **replace(subcadena, nuevaSubcadena):** Reemplaza la primera aparición de una subcadena con otra
  + **split(separador):** Divide una cadena en un array de subcadenas. Devuelve un array de subcadenas
  + **toLowerCase():** Convierte la cadena a minúsculas
  + **toUpperCase()**: Convierte la cadena a mayúsculas.
  + **trim():** Elimina los espacios en blanco al inicio y al final de la cadena.
  + **localeCompare(cadena, locales?):** Compara dos cadenas según las convenciones del idioma especificado. Los locales son opcionales, por defecto recupera los de la página. Devuelve -1 si la cadena original es menor, 0 si son iguales y 1 si la cadena original es mayor.

1. **Math**
   * **Math.PI:** Constante PI
   * **Math.ceil(x):** Redondea un número hacia arriba al entero más cercano.
   * **Math.floor(x):** Redondea un número hacia abajo al entero más cercano
   * **Math.round(x):** Redondea un número al entero más cercano.
   * **Math.abs(x):** Devuelve el valor absoluto de un número
   * **Math.max(...valores):** Devuelve el mayor número de una lista de valores.4
   * **Math.min(...valores):** Devuelve el menor número de una lista de valores
   * **Math.pow(base, exponente):** Eleva un número a una potencia
   * **Math.sqrt(x):** Devuelve la raíz cuadrada de un número**.**
   * **Math.trunc(x):** Elimina los decimales de un número, devolviendo solo la parte entera.
   * **Math.random():** Devuelve un número pseudoaleatorio entre 0 (inclusive) y 1 (exclusivo)
2. **DATE**
   * **new Date():** Crea un nuevo objeto Date con la fecha y hora actuales.
   * **new Date(valor):** Crea un objeto Date usando un valor de milisegundos desde el 1 de enero de 1970 (Epoch Unix).
   * **new Date(cadenaFecha):** Crea un objeto Date usando una cadena de texto que representa una fecha (formato “YYYY-MM-DD HH:mm:ss.sss” o compatible, lo mínimo YYYY).
   * **new Date(año, mes=1, día=1, horas=0, minutos=0, segundos=0, milisegundos=0):** Crea un objeto Date especificando cada parte de la fecha. Los meses se cuentan desde 0 (enero = 0, diciembre = 11).

* **Metodos**
  + **getFullYear():** Devuelve el año completo.
  + **getMonth():** Devuelve el mes (0 = enero, 11 = diciembre).
  + **getDate():** Devuelve el día del mes (1-31).
  + **getDay():** Devuelve el día de la semana (0 = domingo, 6 = sábado)
  + **getHours():** Devuelve la hora (0-23)
  + **getMinutes():** Devuelve los minutos (0-59).
  + **getSeconds():** Devuelve los segundos (0-59)
  + **getTime():** Devuelve el número de milisegundos transcurridos desde el 1 de enero de 1970.
* **Modificar una fecha**
  + **setFullYear(año):** Establece el año de la fecha.
  + **setMonth(mes):** Establece el mes (0-11).
  + **setDate(día):** Establece el día del mes (1-31).
  + **setHours(horas):** Establece la hora (0-23)
  + **setMinutes(minutos):** Establece los minutos (0-59).
* **Metodos estaticos**
  + **Date.now():** Devuelve el número de milisegundos transcurridos desde el 1 de enero de 1970
  + **Date.parse(cadenaFecha):** Analiza una cadena de texto que representa una fecha y devuelve el número de milisegundos desde el 1 de enero de 1970 hasta la fecha especificada.
  + **Date.UTC(año, mes, día, horas, minutos, segundos, milisegundos):** Devuelve el número de milisegundos desde el 1 de enero de 1970 para una fecha y hora especificada en el tiempo universal coordinado (UTC). Los meses empiezan desde 0 (enero).

1. **COLECCIONES**

* **Map**
  + **Declaracion**

****

* + **Añadir pares clave valor**

****

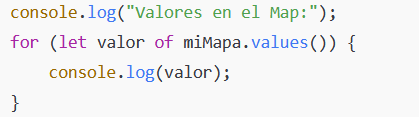
* + **Obtener un valor con la clave**

****

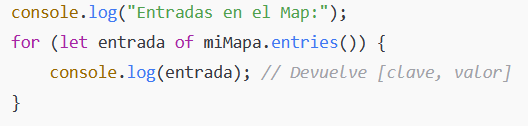
* + **Obtener todas las claves del map**

****

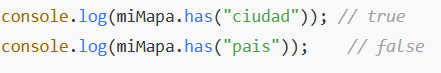
* + **Obtener todos los valores del map**

****

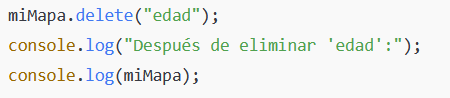
* + **Obtener pares clave valor del map**

****

* + **Verficiar si existe una clave en el map**

****

* + **Eliminar un elemento por su clave**

****

* + **Eliminar todos los elementos**

****

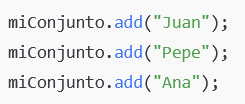
* + **Tamaño del map**

****

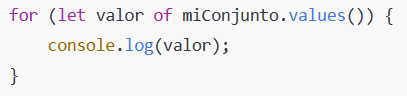
* **Set**
  + **Declaracion**

****

* + **Añadir valores**

****

* + **Obtener todos los valores del set**

****

* + **Verficiar si un valor esta presente**

****

* + **Eliminar un valor del set**

****

* + **Eliminar todos los valores del set**

****

* + **Tamaño del set**

****