***Apuntes***

console.log(Date());

let fecha= NewDate();

console.log(fecha);

console.log(fecha.getDate()); // Lo que hace es devolver el dia en forma de numero. Ejemplo: 6.

console.log(fecha.getDay()); // Devuelve el numero del dia de la semana, por ejemplo si fuera jueves nos devolvería 4. Teniendo en cuenta que la semana se cuenta de D-S representado en un arreglo 0-6

console.log(fecha.getMonth());

console.log(fecha.getFullYear());

console.log(fecha.getHours());

console.log(fecha.getMinutes());

console.log(fecha.get());

moment.js

***Math***

Math.pi devuelve el numero pi

Math.abs devuelve el numero absoluto. Ejemplo -7.9 devolvera 7.9. Siempre un numero positivo, no negativo

Math.celi devuelve el numero entero mayor, 7.8 => 8

Math.floor devuelve el numero entero menor 7.8 => 7

Math.round devuelve el numero entero mas cercano 7.8 => 8, 7.2 => 7

Math.sqrt devuelve la raíz cuadrada

Math.pow devuelve potencia en base a dos números. 2, 5 (a, b) = 32

Math.sign devuelve si el numero es negativo, positivo o cero. Negativo=-1 , cero=0, positivo=1 . Ejemplo -7.8 devuelve -1 y 6.7 devuelve 1.

Math.random devuelve un numero aleatorio entre 0 y 1

Si yo lo que deseo es tener el resultado de 1 a 1000 para una rifa lo que puedo hacer es lo siguiente: math.random () \* 1000. Y esto me devolverá un numero de 1 a 1000

Operadores Cortocircuito

Funcionan poniendo 2 condiciones

Cortocircuito OR //cuando el valor de la izquierda en la expresión siempre pueda validar a true, es el valor que se cargara por defecto

Cortocicuito AND //cuando el valor de la derecha en la expresión siempre pueda validar a false, es el valor que se cargara por defecto

Function saludar (nombre){

Nombre = nombre || “Desconocido”;

Console.log(“Hola ${nombre}”);

}

Saludar(“Jon”);

Saludar();

Trabajando con OR = ||

Devuelve el valor de la izquierda:

Console.log(“cadena” || “Valor de la derecha”)

Console.log(19 || “Valor de la derecha”)

Console.log(true || “Valor de la derecha”)

Console.log( [] || “Valor de la derecha”)

Console.log( {} || “Valor de la derecha”)

Console.log( -2 || “Valor de la derecha”)

Devuelve el valor de la derecha:

Console.log( false|| “Valor de la derecha”)

Console.log( null || “Valor de la derecha”)

Console.log( undefined || “Valor de la derecha”)

Console.log(“” || “Valor de la derecha”)

Console.log(0 || “Valor de la derecha”)

Trabajando con AND o &&

Devolvera el valor de la izquierda si es falso

Console.log(false && “Valor de la derecha”)

Console.log(null && “Valor de la derecha”)

Console.log(undefined && “Valor de la derecha”)

***alert, confirm y prompt***

alert(“Hola esto es una alerta”); //arroja un cuadro de alerta con el botón aceptar

confirm(“Hola esto es una confirmacino”); //es igual a alert pero añade el botón cancelar

prompt(“Hola esto es un prompt el cual permite ingresar datos como si fuera un imput”); es como un confirm junto con un input.

Let alerta = alert(“Hola esto es una alerta”),

Confirm = confirm(“Hola esto es una confirmacino”),

Aviso = prompt(“Hola esto es un prompt el cual permite ingresar datos como si fuera un imput”);

Expresiones Regulares

Las expresiones regulares son usas “letras” que le dan un significado a nuestro código de que debe hacer como por ejemplo -m en Git que significa “Mensaje/comentario”

Se le llama banderas a estas letras clave

“G” significa global y que por lo tanto no se detendrá en la primer coincidencia

“I” significa que no hará diferencia entre minúsculas y mayusculas

“/d” busca seros

Ejemplo:

 let cadena = “Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate velit se cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id est laborum.”

 Let expReg = new RegExp(“lorem”,”ig”);

 let expReg2 = /lorem/ig;

 console.log(expReg.test(cadena));

 console.log(expReg.exec(cadena));

 console.log(expReg2.test(cadena));

 console.log(expReg2.exec(cadena));

Funciones Anonimas Autoejecutables

Es una función en la cual se engloba todo el código que se desea ejecutar

//Funcion anonima autoejecutable

 (function(){

    console.log("Mi primer IIFE");

 })();

 (function(d,w,c){

    console.log("Mi segunda IIFE");

    console.log(d);

    console.log(w);

    c.log("Este es un console.log");

 })(document,window,console);

 //Formas de escribir funciones Anónimas autoejecutables

 (function(){

     console.log("Version Clasica")

 })();

//La Crockford (JavaScript The Good of parts)

((function(){

    console.log("Version Crockford");

})());

//Unaria

+function(){

    console.log("Version Unaria");

}();

//Facebook

!Function(){

    console.log("Version Facebook");

}();

Módulos

<body>

    <h1>Modulos en JavaScript</h1>

<script scr="js/moudulos.js" type="module"></script> //importa los archivos js que veremos en la consola de nuestra pagina

<script scr="js/no-moudulos.js" nomodule></script>    //si el navegador no soporta modulos js se ejecutara este archivo. nomodule lo que hace es no ejecutar el archivo si el navegador tiene modulos

</body>



//Archivo modulos.js

import{PI} from "./constantes.js";

console.log("Archivos modulos.js");

console.log(PI);

//Archivo constantes.js

export const PI=Math.PI;

Que en el archivo HTML en el script src no este el type module seria un gran problema ya que no tendría como vemos en el código de arriba la programación en modulos js ya que no se podría lograr la importación.

//Importar funciones (modulos.js)

import{aritmetica}from "./aritmetica.js";

console.log(aritmetica.sumar(3,4));

//aritmetica.js

function sumar(a,b){

return a+b;

}

function restar(a,b){

return a-b;

}

export const aritmetica = {

    sumar,

    restar

}

“Export default” lo que hará es que cuando importamos de manera default sin que lo pidamos se ejecutara lo que este en export default.

Funciones literales son aquellas que simplemente declaramos

Funciones expresadas son aquellas funciones que almacenamos en una variable

Las funciones expresadas no pueden ser exportadas por default ya que estas no pueden ser ejecutadas previamente a su creación a diferencia de las funciones literales

Si yo quisiera exportar de manera default una función expresada o variable lo tendría que hacer de la siguiente manera:

//exportacion default de una variable o funcion expresada

let password="12345";

export default password;