Resumen clase #1

prompt

toLowerCase

window.confirm=

importante ver sitio w3schools best practices

revisar que es eval y hacer un resumen

buena práctica declarar variables en una sola línea
tener un balance de tener el código entendible y un poco lento y entra rápido minimizando el código en una línea

investigar de todas las palabras reservadas de javascript y las posibles palabras reservadas y las que fueron antes

Resumen clase #2

Web browser engine = por lo que entendi es un software que toma contenido renderizado se usan típicamente en navegadores web.clientes de correo electrónico y otras aplicaciones.

Todos los navegadores utilizan un motor de renderizado , algunos de los motores de renderizado mas conocidos son:

Trident: el motor de internet Explorer
Webkit: el motor de Safari
Presto: el antiguo motor de Opera
Blink: el nuevo motor de Chrome y Opera
Gecko: el motor de Mozilla Firefox

V8: es un motor de código abierto Javascript creado por Google y esta creado en C++ y es usado por Google Chrome este compila y ejecuta el Javascript de origen, se encarga de la asignación de memoria para los objetos y elimina la basura de los objetos que ya no necesita

Server-side and client-side programming

la comunicación del desarrollo WEB esta entre 2 partes sobre el protocolo HTTP, el Servidor y el Cliente
el servidor se encarga de recibir las páginas y el cliente solicita las páginas desde el servidor, el cliente es un navegador en internet o ver videos online , etc…

Server-side programming: es el nombre general para los tipos de programas que se ejecutan en el servidor como procesos de entrada del

usuario o interactuar con el almacenamiento permanente (archivos, sql , etc..)

Ejemplo los lenguajes de programación son : PHP, ASP, Net , etc…

Client-side programming: al igual que al lado del servidor , es el nombre para los tipos de programas que se ejecutan de lado del cliente, como la creación de WEBs alternativas formas dinámicas en la WEB, interactuar con el almacenamiento temporal y almacenamiento local (cookies, localStorage).

Ejemplos de lenguajes de programación utilizados con Javascript(principalmente) cualquier lenguaje que se ejecuta en el dispositivo cliente que interactúa con un servicio remoto es un lenguague del lado del cliente

Resumen clase #3

```
typeof undefine      // undefine
typeof null     // object
null === undefined    //false
```

```
null == undefined    // true
```

delete = para dejar una variable totalmente sin valor

ejemplo = delete objectName

hacer un repositorio en github que diga js-2015-3c

Inline scripting:

onchange

onclick

onmouseover

onmouseout

onkeydown

onload

investigar diferencia entre onload y document.ready

el onload si espera a q carguen las imagenes para ejecutar una funcion despues de eso

```
var test = function(){
    console.log('hola');
} == constructor
console.log(typeof test);
```

constructor esta dentro de una clase que me permite la instancia del resultado

getTime para saber cuanto dura una funcion antes poner una variable con Date();

Resumen tarea para el sábado 31
Leer hasta el capitulo 1,2,3 del libro the good parts
Lectura ”Type conversion” y Mathematical Precedence
Objeto Date
POO Investigar → Constructor

The Good Parts

Resumen capítulo 1

Al principio comenta que el lenguaje de JavaScript es un lenguaje con buenas y malas prácticas que al utilizarse las buenas practicas se trabaja mejor y es una muy buena programación comenta también de que

JavaScript se hizo muy popular y que x eso es el lenguaje de programación por defecto de la Web.

Su intención es que veamos la bondad de JavaScript como lenguaje porque el piensa y explica lo robusto y poderoso que JavaScript que mucha gente lo desprecia pero no se toman el tiempo de aprenderlo.

El analiza por su parte JavaScript y da una opinión de sus cosas malas y buenas, por un lado las cosas buenas son las muy buenas ideas incluyen funciones, mecanografía suelta, objetos dinámicos, y una expresiva notación literal de objetos, por otro lado las malas como lo son las variables globales xq depende de ellas para la vinculacion

Una de sus mayores propiedades es de tener la posibilidad de heredar prototipos

Resumen capítulo 2

En este capítulo se se dan a conocer de las buenas partes de JavaScript

Whitespace: De vez en cuando es necesario utilizar espacios en blanco para separar secuencias de caracteres para que no se unifiquen en un único token.
Numbers: No solo se utiliza un solo tipo de números
String: es una cadena de caracteres envuelto entre comillas simples o dobles puede tener números dentro
Statements: Contiene un conjunto de instrucciones ejecutables en el navegador dentro de las etiquetas <script> las cuales el navegador las intrepreta eh ejecuta de inmediato.
Expressions: son tanto de strings como de numbers o de boleanos
Literals: Se especifican los objetos nuevos con los objebct literals se pueden especificar como names o como strings y no como nombres de variables, las propiedades deben ser conocidas como en el tiempo de copilacion
Function: puede llamarse de forma recursiva, puede mantener una lista de parámetros que actúan como variables eh incluyen defi

Resumen capítulo

Hay varios tipos de objetos entre ellos los array , las funciones y los objetos son objetos.
Los objetos son un contenedor de propiedades, donde las propiedades son el nombre y el valor.
niciones y declaraciones

Resumen clase #4

investigar indexOf =

si creamos una función y tiene 3 parámetros ella va estar esperando los 3 parámetros al mencionarlo como o llamarlo como new libro y si se ponen menos parámetros o ponemos 0 parámetros saldrian en undefined

vistazo a la página 25 the good parts y la 101 a 108

tarea metodo hasownproperty =su función es para averiguar si un atributo o instancia de una función ya declarada y nos dice si la instancia tiene o no el atributo que le preguntemos.

Resumen clase #5

throw = le devuelve el control a la función que lo llama
sintaxis simple

```
try{
 // código a intentar ejecutar
 }catch(e){
 //lo q se ejecuta si falla
}finally{ //siempre pase o no x el catch
}
```

Tarea

modificar el ejecicio que nos dio seria que lance un excepción para cuando es un boleano
hacer que el catch de la funcion de contexto se ejecute si no es un string igual el console.log pero si es un boleano nada mas tire el error
y leer el capitulo 4 completo no leer el curry
leer recursibidad

```
var saludo = function(nombre){

    if(typeof nombre !== 'string'){
        throw{
            name: "No String",
            message:"El valor ingrsado es '"+typeof nombre+"', esperado String"
        };
```

```
    }
    console.log("Bienvenido" + nombre);

}
var contexto = function(){
    //pasa algo
    try{
        saludo(23423432);
    }catch(exception){
        console.log("ERROR ["+exception.name+"], "+exception.message);
    }
    //otra cosa
}
contexto();
```

Hay 4 tipos de funcion

Funcion Literal: es la clasica funcion cuando la funcion no es parte de ningin objeto

sintaxis
```
function suma(a,b){ // Patron invocacion de una funcion literal
        return a+b;
}
```

Funcion tipo metodo

sintaxis
```
app.usuario = {

     incrementarOrdenes: function(){ //Patron invocacion de una funcion literal
        this.ordenes++
     }
```

Funcion tipo Constructor toda funcion va con la inicial en mayuscula

sintaxis var Libro function(nombre){ // Patron de invocacion constructor

```
    this.nombre = nombre;
}
```

var Persepolis = new Libro (“persepolis”);

Tolas las funciones van a tener un return undefine si no lo llamamos con return undefine