Cursillo de Escalab JS Master

Clase 1 06-01-2021

Intro

Fetch usado para bd

Firebase.

Javascript

Lenguaje de programación usado por secuencia de comandos

Permite animaciones 2D y 3D

Js es el motor con el cual se realizan las acciones

Lo bueno es que es flexibe lo mismo, por lo mismo hace la desventaja

Tipado = define los tipos de datos y no se pueden cambiar

JS **No es tipado**

Alternativas de JS

Dart

Elm

Typescript (tipado)

Coffeescript (compila a JS)

Transpilar = código bite, de ninario a código (o al revés no recuerdo)

Valley of the boom = película de la historia de js

Los browser poseen un motor(engine) para poder compilar los js

Engine Motor V8 Google es el mas potente actualmente para js

Funcionamiento compila por partes a medida que se va consultando

Por ende carga mucho mas rápido.

Mozilla engine spidermonkey

Safari engine nitro (squirelFish)

Explicación lunes próximo sobre porque el motor abierto consume mucha ram

Se creo el ECMA -262 para standarizar (conocido también como ECMAScript)

2015 se creo un estándar para crear los ECMAScript.next siguientes años (cada año 6 2015, ecmascript 7 2016…)

\*\*\*Todo material será compartido\*\*\*

Tipos de datos

* Primitivos : es un tipo de variable que no tiene mas funciones de lo que es( no posee métodos)
  + Number \*-> es simplemente un numero nada mas
  + Sring \*
  + Bollean\*
  + Null
  + Underfines
  + symbol
* No primitivo : son todos los demás variables que tienen mas cosas que ofrecer
  + Se refiere a los Object

Var -> declara variables

Al definir variables, no permite comenzar con números

Una vez que se compila comienza a usar la memoria en las variables.

Var permite el joisting

Definir con var permite entregar mas cosas y declarar una variable sin var es mas global

En cuanto a float y number para js siempre son typeof number

Las variables de tipo null🡪 son typeof Object

String() -🡪 es un protitpo para transformar directamente pasandole una variable

Number() 🡪 si no se le pasa ningún argumento es por defecto 0

Number(‘hola’)Si pasamos un texto devuelve un NaN

Si se pasa un boleano Number(false) -🡪 devuelve un 0 true un 1

Boolean() vacio -> false

Si tiene algo es true

Boleano() si es un numero 0 es false y cualquier otro numero es true

Boolean(‘’) –false

Boolean(undefined) – false

Window donde tiene sus funciones es el \_proto\_

String viene del primer objeto **Object** todos los objetos son Object

Métodos de string

{strin},chartAt(num) 🡪 devuleve lposicion letra

Substring() , permite cortar texto por medio del índice

indexOf permite saber en que índice comienza la palabra

includes(), permite verificar si existe la palabra

toda acción que se declara sin return , devuelve un **undefined**

**próxima clase ver closure**

Clase 2 11-01-2021

Function en js

Debe seguir el mismo regla que la creación de variables normales

Dentro de los paréntesis van los datos que recibe

Toda función debe retornar algo

Funcionamientos de js

Js trabaja en un solo hilo de ejecución

Js esta formado por memory sheap

Es el espacio que tiene disponible el navegador para usarlo

Call stack …

Estructura de datos

Para la llamadas de funciones y

Cosas externas de js

Webs apis

Even loop

Error común es llamar a una función que esta dentro de la misma función

Function foo(){

Foo()

}

Realiza un loop error

El problema de spidermonkey no fue tan eficiente porque no cumple la función de en tiempo real, (compilar a medida que se llama)

Motor V8 2017 (Google) Función en tiempo real

El turbofán solo toma trozos del byte código, y lo va mejorando y lo agrega al binario original y lo va juntando cada vez que se vuelve a consultar va mejorando el código es por eso que consume demasiada ram el Chrome,

**Nota,** antes si se caía una pagina se debía cerrar todo el navegador actualmente no es necesario.

Ejmp de funciones con JS

Function name\_ (parámetros\_){

}

Nam\_(argumentos\_)

Función propia de js trae Math.PI

Math.floor() se usa para redondear un float

Math.random() arroja un valor float al azar

En lugar de void no se coloca nada , y al llamarlo siempre retorna algo en caso de ser una función sin return retornara un **underfined**

La función Date() debe ser con el new para declarar de que es una nueva instancia

Métodos son los funciones dependientes de un objeto

Ejmp string\_.upper() (upper viene siendo el método)

La función RegExp(‘valor a buscar’, ‘g’) 2parametro es para indicar que sea global es mas ordenado

Funciones declarativa son las que posen nombre

Las que no se llaman anónimas

Una función anónima se usa para no tener que crear un nombre

Ejmp var funcionDeclarativa = function(){}

Las funciones anónimas se usa mucho en Ecmascript 6

En las fucioens al usar el **return** las líneas de códigos que se agreguen debajo no se ejecutaran debido a que el return es escapar de la función , termina la ejecucionde la función

Curry funcition , closure o nested (funciones anidadas)

Curry acepta múltiples argumentos que acepta mas de argumento a la vez

Funciones nuevas pasando nuestro función base

Curry en lugar de un return base (variables )

Se return una función

Function suma(a){

Return function(b){

Return a + b

};

}  
para llamar (2) en ese caso retorna una funcition por lo que se puede hacer la otra llamada de la segunda function

Suma(1)(3)

Una vez llamada una función padre

Incrementos

Var\_++ 🡪 print luego suma

++ var -> suma luego print

Los métodos mueren dentro y vovoen dentro de la function a menos que se returne

--- Hoisting ---

Proceso de js que mueve las declaraciones al inicio del scope

No se ve visualmente porque pasa entre la cmplilacion y el bitecode

Scope = contexto = léxico

No se puede llamar una variable que se encuentra dentro de una función y la consulta esta fuera de la declaración del alfuncion

Var involucra en muchas cosas , hace hoistin cambios en Windows

**ojo al usar var cuando se usan en nombre de variables**

todo nombre de variables deben ser con minúsculas los prototipos con mayúsculas o las clases también con mayuscilas

--- condicionales ---

* If
* Ternario

Ternario

Se usan los if y los else

Var nombre = ‘alex’ ? nombre : ‘sin nombre’

? 🡪 if

: 🡪 else

Y 🡪 &&

O 🡪||

Switch case

Switch(valor){

Case 0: conole.log(‘es 0’): break;

Case 1:console.log(‘es 1’): breake

Default : console.log(‘es nada’)

}

Si no tuviera brake se puede hacer con return

**Próxima clase var globales**

Clase 3 13-01-2021

Repaso de Prototiupos

Pincelada

Prototipos es una característica de js

Es un amanera de incorporar OO (orientacio a objetos=)

Es similar a una clase el cual se heredan y no es necesario volver a escribirlo nuevamente

Los protipos siempre se defien con mayúsculas

Los prototipos se definen de la misma forma que una función normal pero se diferencian porque se hacen con mayúsculas

Y para confirmale al navegador que es un prototipo se debe de ingresar unos parámetros y se utiliza la palabra **this.algo** el cual hace referencia de llamar a la misma función(prototipo)

Cada vez que se llame a un prototipo se debe de hacer con crear un nuevo objeto var variable\_ = new prototipo\_(parámetros que requiere el prototipo de entrada)

También se pueden crear funciones dentro de los prototipos

Prototipo\_.prototype.nombre\_funcion\_ = function(){

// bloque Aquí función que se desea q retorne

}

**Las funciones de los prototipos se deben declarar fuera del prototipo**

Anti patron = son las malas practicas

**Iteraciones (loop)**

.for

.while

Los ciclos si afectan a la rapidez del navegador

Mas iteraciones mas lento

* Como todos los lenguajes

**Para ver la memoria del browser**

Se puede ver en la variable **performance**

Do while

Hace primero la ejecución luego evalua

While primero evalua luego hace la ejecución

condicional && 🡪 AND || 🡪 OR

== en la condiciones al usar == realiza un cambio de tipo de dato es como var\_numero.toString == 1..toString()

1**..**toString

El primer punto indica que sea flotante el segundo lo transforma a string

Variables globales

Tiene un scope externo y permite que la variable este disponible en todas partes

-BOM vs DOM

BOM permite la comunicación con los componentes como history, location etc

DOM árbol del html

-windowa

Location

-document

Web storage api

Al crear una variable con var fuera de una function esta se vee en wndows.variable\_

Lo que hace riesgoso que si se declara en otro archivo se puede pisar una varibales anteriormente creada.

Window.closed retorna si esta cerrada la pagina true o false

Window.alert (‘’)

Window.prompt(‘dime la hora’)

---Location

Host – si tiene un puerto va aquí

Hostname solo el nombre

Location.href obtiene dirección

Location.protocol obtien el protocolo http o https

Location.seacrh() obtiene todo lo que viene después del sigono pregunta

Location.reload recarga la pagina

Location.assign 🡪 permite enviar una url nueva para navegar dentro de la mismo navegados

----Document

De manera antigua para buscar un objeto se usaba el document.getElementById()

Pero ahora sale el dociment.querySelector(‘.nomb\_clase’) .clase #id

* Los eventos listener son usados para llevar un control de las acciones del usuario en la pagian , cuando hace click, en un botno, eventos

---Local storage Api

Almacenamiento propio del navegador para las app externas

--- sesiónStorage

Sesión mantiene los valores hasta que la pagina se cierra

En cambio el local stroage aunque se cierre siguen los valores

**Próxima clase se dejara lo mas simple y se verán objetos, arreglos, scopes**

Clase 4 18-01-2021

22 de febrero ultima clase

Temas para hoy

* Objeos
* Arreglos
* Explicación de scope (enviroment)

Objetos

Variables con sus propiedades

Definición de un objeto

Var objeto\_ = {}

nombreobjeto\_.property1 = ‘primera propiedad’;

nombreobjeto\_.property2 = ‘primera propiedad’;

nombreobjeto\_.property3 = ‘primera propiedad’;

no se puede tener dos propiedades iguales se pisan si se usa la misma d emismo nombre

ejemplo si se desea crear otra nombreobjeto.property1 se pisa

mutabilidad = cambiar el valor de algo tanto como valor en si o como tipo de variable

* Array (arreglos)

Los valores pueden ser de cualquier tipo y se pueden repetir tanto string , objetos , function , number etc

* Scope

Contexto actual de ejecución

Hay dos tipos

* Local
* Global

Global 🡪 objeto Windows

Fuera de un bloque

**Hizo una pregunta técnica no escuche algo de las function**

Local dentro de un bloque

---En código

**Objetos**

Todos los objetos padres heredan del padre

Con objeto.\_proto\_

Se pueden ver todos los métodos que son heredadoes del padre

Al crear un objeto sin nada definido

Var objeto = {}

Al verlo con persona.\_proto\_

Ya trae funciones heredadas

Para crear propiedades

Objeto\_.nombrepropiedad = ‘valor que se le de’

La otra forma es

Var objeto\_ = {

Propiedad1 : ‘valor’,

Propiedad : ‘valorn’

}

Al usar null o underfined , al usar APIRest

Si se usa null llega como null , al usar el underfined llega vacio

Para utilizar una propiedad que próximamente se utilizara se utiliza en this.propiedad = vari\_que se da en la function

\*\* la única forma de crear una scope o un contexto es creando una function\*\*\*

si se tiene un objeto padre y dentro otro objeto hijo y dentro del objeto hijo se usa el this va a llamar al padre

acceder a los valores de los objetos

* objeto.propiedad
* objeto[‘nom\_propiedad’]
* buscar una propiedad de manera dinámica
  + var variable\_ = ‘nom\_propiedad’
  + console.log(objeto\_[variable\_])
  + esta manera se conoce como propiedad c

Object es el padre de todo

Windows se refiere a toda las características de browser

Para saber si una propiedad es propia del objeto o es heradada

Objeto\_.hasOwnProperty(‘bar’)

Devuelve true o false

Object.keys(objeto\_) método propio de object que recorre las claves del objeto , devuelve un arreglo

Object.values(objeto\_)

Object.entries(objeto\_)trae un arreglo con indice ,llave y valor

Para mantener un objeto constante y que nadie lo cambie (peviene la modificación)

Exm

Var personaCongelada = Object.freeze(persona);

Objeto a json

JSON.stringify(objeto\_)

De string(json) a objeto

JSON.parse(strinJSON\_)

**Arreglos**

* el foreach no retorna nada
* el .map si

por lo general los métodos de iteración de array no se modifican los originales

push() y pop() realizan cambios en el arreglo original por lo que muta, cada vez que se ejecute una vez va cambiando el arreglo original.

.splice(índice\_, cantidad\_) se usa para eliminar entre medio

.splice(índice\_, cantidad = 0, ‘objeto a agregar’) (no borrará nada 2do argumento = 0) agrega un objeto en la posición indicada

.reverse() cambia el orden del arreglo

**Próxima clase scope, set**