Java Script

Nace para generar dinamismo con las páginas web, para poder interactuar en dichas páginas, orientado 100% para trabajar en web.

Características:

* Lenguaje interpretado
* Orientado a Objectos
* Debilmente Tipado
* Dinamico

**Orientado a Objetos:** Genera objetos los cuales contendrán la informacion contenida de algo por ejemplo usuarios.

**Debilmente Tipado:**  Significa que java-script puede interpretar que tipo de variable se requiere para realizar una operación o ejecutar una línea de código.

4 + “7”; //47

4 \* “7”; //28

**Dinámico:**

* **Etapa de Compilacion:** traduce de lengua humano a lenguaje de máquina
* **Etapa Runtime:** Ejecuta en tiempo real

**Lenguajes estaticos:** deben compilarse primero para ejecutarlos

**Lenguajes dinámicos:** corren directamente en runetime no necesita compilarse.

**Lenguaje Interpretado:** Se puede ejecutar linea a linea a runtime y en el proceso con la interacción el navegador lo ejecuta y si existe un error se detiene.

**Motor de Javascript V8**

Etapa de compilaciòn ***just in time compiler*** permite traducir el lenguaje de alto nivel a lenguaje de maquina. Interpreta el código y lo traduce es decir la compilacion para ponerlo en bitecode.

**Versiones de Lenguajes de Programación**

**Forwards:** Ser compatible con versions futuras significa que incluir una adición al lenguaje en un programa no causaría que se rompa si se ejecuta en un motor de JS anterior. JS no es compatible con versiones futuras

**JS es Backwards:** La compatibilidad con versions anteriores signficia que cuando se acepta como JS váido, no habrá un cambio future que haga que el código deje de funcionar.

Por qué JavaScript?

Comunidad enorme de desarrollo

Utilidad

**Desarrollo Web:** Angular, React, View productos Web Frameworks

**Desarrollo de Apps:** React Native (IOS y Android)

**Desarrollo Desktop:** Electron

**Back-end / IOT :** node JS

Variables

Datos que guardamos en memoria

Tareas: Funciones que haremos con esta data

Valores Primitvos

Numericos

Strings

Boleanos

null espacio reservado para colocar un valor

undefined valor reservado

Valores No primitivos o tipo Objetos

Array [1,2,3]

Valor Tipo Objeto {nombre :”Diego”}

Estados de una variable

var edad; -> **declarar** la variable reserve espacio en memoria

edad = 35; -> **inicializar**

**Funciones**

//Declarativas

function miFuncion () {

    return 3;

}

//Expresión -> Se les conoce como anonimas

var miFuncion = function(){

    return a + b;

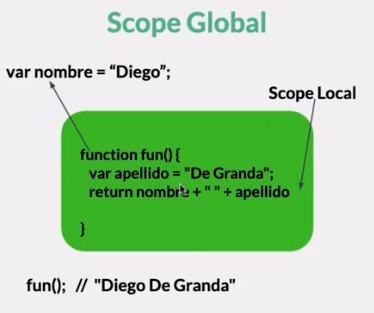
}

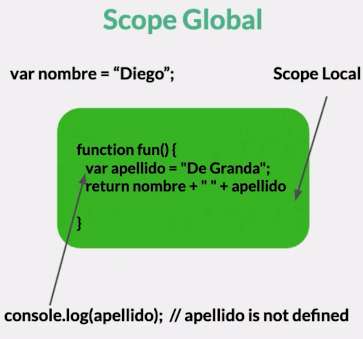
**Scope**

Alcance de las variables

Scope Global: Todas las variables y funciones

Scope Local: Variables dentro de la function





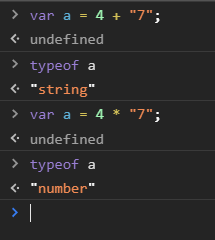
Hoisting

Para versiones bajas MS 5 para abajo

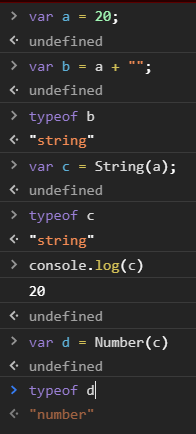
Variables y funciones se procesan antes de ejecutar cualquier tipo de codigo

**Coercion**

**Implicita:** Lenguaje nos ayuda de un tipo de valor a otro

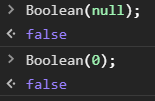


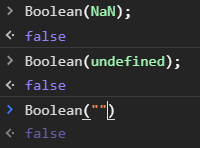
**Explicita:** Obligamos a cambiar las variables de un tipo a otro tipo



**Truthy y Falsy**

**Valores Falsos**





**Valores Verdaderos**

Valores(strings, numeros, arrays)

Funciones

**Operadores**

**Asignacion:** =

**Comparacion: ==, >=, <=, ===** (sean estrictamente iguales), && and, || or

**Aritmeticos:** + - \* / **->** operadores binaries**,** ++, +=

**Link para ver todos los operadores**

<https://developer.mozilla.org/en-US/docs/Web/JavaScript/Guide/Expressions_and_Operators>

**Condicionales**

Reglas para verificar si una condicion es verdadera o falsa

if (condicion 1){

accion;

} else if (condicion 2) {

accion;

} else {

accion;

}

codition ? true : false

var numero = 1;

var resultado = numero === 1 ? ‘Si soy un uno’ : ‘No, no soy uno’;

**Switch**

Sirve para validar casos

var numero = 1;

switch(numero){

    case 1:

        console.log('Soy un 1');

        break;

    case 10:

        console.log('Soy un 10')

        break;

    case 100:

        console.log('Soy un 100')

        break;

    default:

        console.log('No soy Nadie')

}

**ARRAY**

array.push -> anade un element al final del array

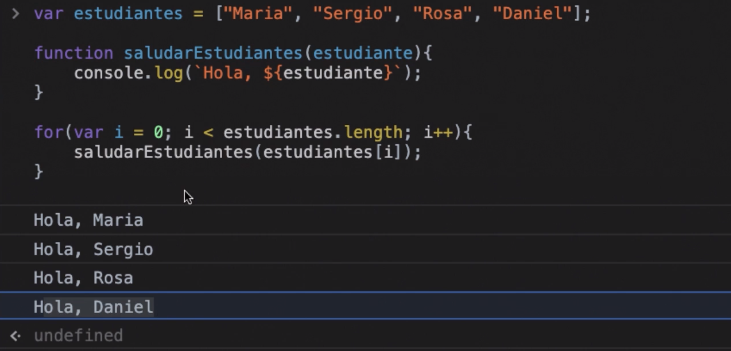
array.pop -> elimina el ultimo element del array

array.unshift(‘elemento’) -> mueve el elemento seleccionado al principio del array array.shift(‘elemento’) -> elimina el primer element del array

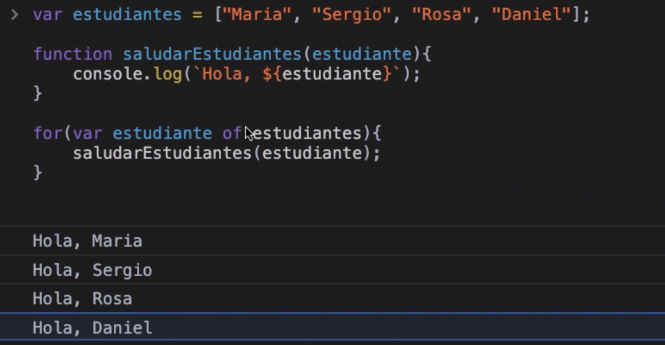
array.indexOf(“Cereza”) -> indica el index del element

**Loops: For y For…of**

**For normal iterando con i**



**For para cada uno de los elementos del array**

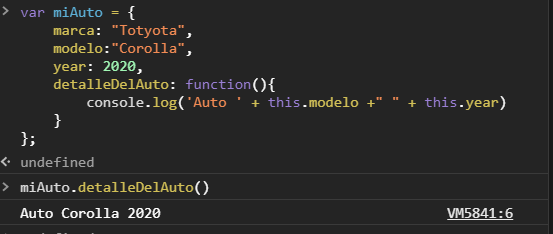


**Loops: While**

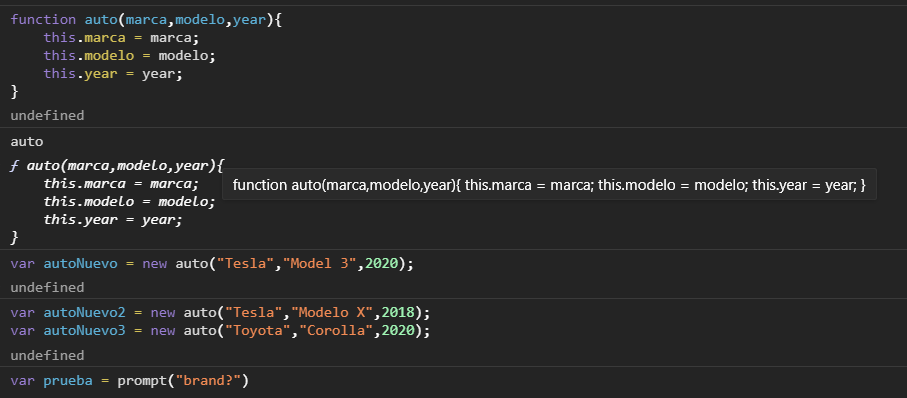


**Objetos**

**this ->** hace referencia a este objeto



**Funcion Constructura**



**promt->** sirve para poder ingresar variables

**Metodos de recorridos de Arrays**

**Filtra**



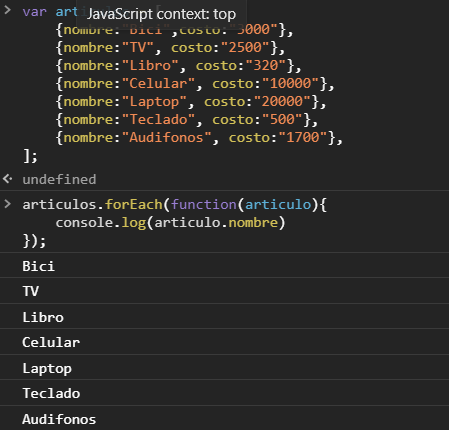
Mapea



Find



Foreach



Some

