**Módulo 1**

* ECMA: Hacen referencia a European Computer Manufacturers Association. Se trata de una organización internacional dedicada a la creación de estándares para la comunicación y la información fundada en 1960 por algunas de las compañías de tecnología más grandes a nivel mundial como, por ejemplo, IBM.
* LET y CONST: Se introdujeron a partir de ECMAscript 6(2015). Algo muy importante que no se puede hacer con el uso de ellas es volver a repetir la variable de su nomenclatura, tanto en let como en const pj:

let nombre = “Fernando”

let nombre = “Ariel” -> Va a aparecer una línea roja debajo indicando que ya está en uso.

* El Scope de una variable es dependiendo donde la declaremos, si lo hacemos al inicio de nuestro programa, fuera de una función, arreglo, etc. Es declarada como:

1. Scope Global:

var variableGlobal= “Soy Globlar”;

function ejemplo() {

console.log(variableGlobal);

}

console.log(variableGlobal + “2”); // devuelve “Soy Global 2”

ejemplo(); // “Soy Global”

1. Socope Local:

function ejemploLocal(){

var variableLocal = “Soy Local”;

console.log(variableLocal);

}

ejemploLocal(); // devuelve=”Soy Local”.

* El Hoisting: Este término se refiere a un comportamiento en JavaScript donde las declaraciones de las variables y funciones son "movidas" **al inicio de su contexto de ejecución** durante la fase de compilación, antes de que el código se ejecute. Esto significa que, conceptualmente, las variables y funciones son "elevadas" (hoisted) a la parte superior de su ámbito (scope), lo que puede afectar la forma en que el código se ejecuta.

PJ:

1. Si usamos Var

console.log(x); //Undefined.

var x = 5;

console.log(x); //5

1. Si usamos let o const

console.log(y); //ERROR cannot acces “y” before initialization.

const y = 5;

* **Spread & Rest:** Ahora vamos a conocer 2 operadores introducidos en ES6 que llegaron a cambiar la forma de escribir código. En realidad, podríamos **considerarlo**como un solo operador ya que en los 2 casos se representa con tres puntos suspensivos (**...**), pero dependiendo de su posición y contexto cambia su función. Veamos a que nos referimos en el siguiente video.