**¿Qué es JavaScript?**

*Clase 1*

¿Qué es JavaScript?  
Nace para dar dinamismo a la web para hacer las paginas estáticas mas dinámicas orientado a trabajar 100% web

Es un lenguaje interpretado, orientado a objetos, débilmente tirado y dinámico.

Orientado a objetos: permite generar objetos globales que son conjuntos de características para agrupar distintos elementos de nuestro sistema que a la vez forman parte de una base de datos y todos estos elementos pueden ser ocupados para mostrar información individual.

Débilmente tipado: los tipos de datos no están bien definidos en el leguaje y permite por ejemplo operaciones entre numerosos y letras porque el lenguaje asume tipos de datos que no necesariamente son los que se quiere representar.

++Dinámico: ++ el lenguaje corre directamente en el runtime(tiempo de ejecución).

Realmente es un lenguaje interpretado?  
Interpretado quiere decir que el leguaje va a ir leyendo el código linea por linea en el tiempo de ejecución y probando que todo funcione en el proceso con la interacción con el programa y si encuentra un error en ese momento lo marca.

Una vez que el navegador obtiene el código js el motor de js lo procesa convirtiéndolo a binario para que sea entendido por la maquina por lo que javascript es un leguaje compilado.

Backwards & Forwards

Forwards: Ser compatible con versiones futuras quiere decir que al implementar funcionalidades nuevas del leguaje no daña nuestro código y se puede usan en versiones anteriores del motor de compilación.

Backwards: Todas las funciones nuevas del lenguaje no rompen el entorno de trabajo actual pero no las puedes usar inmediatamente hasta que no se estandaricen .

Para hacer este cross versión del lenguaje se utilizan compiladores como BABEL que permite utilizar nuevas funcionalidades del lenguaje javascript dentro de nuestro entorno actual de trabajo traduciendo las nuevas versiones de javascript a versiones anteriores standard que el navegador entiende para que puedan ser utilizadas.

**Porqué JavaScript?**

3 Lenguajes estándares: **HTML5 - CSS3 - JS**

**HTML:** Para maquetar la información. Texto, imagenes, videos, etc.  
**CSS:** Para darle estilo a la información.  
**JS:** Programación para que la página sea dinámica o generar una app web.

En el 2019 la W3C decidió y subió como nuevo estandar y lenguaje **WA (WebAssembly)**. Es un lenguaje nuevo a utilizar para construir productos web. Con este no será necesario utilizar HTML, CSS y JS para hacer productos web.

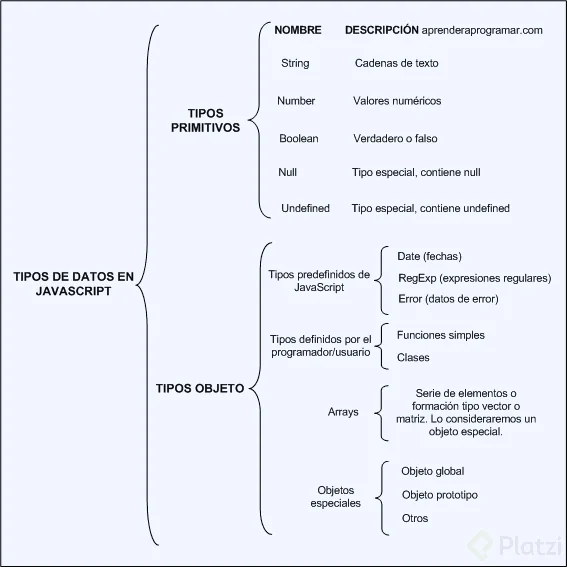
**JS** tiene una comunidad enorme de devs que ayudan a generar diferentes cosas. Para apps web hay muchos frameworks y librerías construidas en **JS**, que ayudan a desarrollar proyectos de una manera mucho más rápida, eficiente y robusta.  
**Angular, Vue, React** son algunos de los frameworks que podemos utilizar para hacer productos web.

Si no sólo quieres hacer productos web podemos utilizar un framework llamado **React Native**, para poder construir aplicaciones nativas para Android y IOS.  
**Electron:** framework JS que nos permite desarrollar aplicaciones para escritorio, tanto para Mac como para Windows.

En toda web y app tenemos 2 partes. El Front-end y el Back-end.

**Front-end:** Es todo lo que se ve en nuestra web/app y con lo que podemos interactuar.  
**Back-end:** Va manejando las bases de datos, las interacciones y peticiones que el Front-end le va a pedir.

**Node.JS** es un entorno de ejecución de JS que corre en el Back-end. Permite trabajar aplicaciones IOT (Internet de las cosas), hace inteligente ciertos dispositivos conectados a internet.



**VARIABLES**

Es la representación de algún lugar en memoria que vamos a reservar para poder guardar un valor, el cual puede tener diferentes tipos.

Javascript tiene una palabra reservada que utiliza para representar un valor. Ejem:

var nombre = “Oscar”;

El var le indica al programa que debe guardar un espacio en la memoria, el cual estará definido según el valor que le coloquemos, en el caso del ejemplo será “Oscar” (valor String).

**FUNCIONES**  
Son un conjunto de sentencias que nosotros podemos utilizar para generar ciertas acciones con los valores que ya guardamos antes en las variables.  
Tenemos dos tipos de funciones  
• Declarativas: va a inicializar un valor y lo guardamos en memoria.  
• Expresión (también conocidas como funciones anónimas): vamos a crear una variable donde guardamos la función en memoria.  
Las funciones también necesitan parámetros que estoy esperando recibir como valor para que la función pueda hacer algo.  
• Parámetros de la función: nos ayudan a generar cálculos o resultados de la función.

**Diferencias:**

A las funciones declarativas se les aplica hoisting, y a la expresión de función, no. Ya que el hoisting solo se aplica en las palabras reservadas var y function.

Lo que quiere decir que con las funciones declarativas, podemos mandar llamar la función antes de que ésta sea declarada, y con la expresión de función, no, tendríamos que declararla primero, y después mandarla llamar.

**Scope**

Es el alcance que tienen las variables en el código.

Existen 2 tipos de scopes.

* Global: Puede ser llamada a lo largo de nuestro programa.
* Local: Solo puede ser llamada dentro del bloque de código en el que se declaró.