

DEV.F.:

Clase 1

**Introducción y
Fundamentos de JS**



| Temas de la clase (180 min)

Re-conociendo al grupo (10-15 min)

Presentación del sensei

¿Cómo llegan a este módulo?

¿Qué es JS? (40-45 min)

Qué es un lenguaje de programación

Porqué aprender JS

Cómo usar la consola del navegador

Tipos de datos (35-40 min)

Primitivos: String, number, boolean

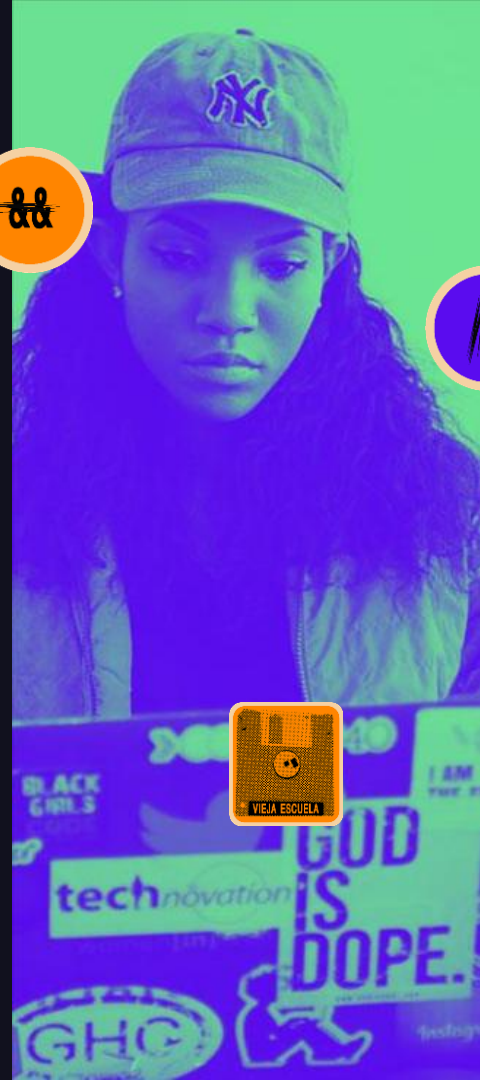
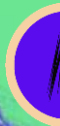
null y undefined

Practicar con “console.log (typeof)”

Buenas prácticas y comentarios (40-45 min)

Comentarios inline y en bloque

Cuándo y porqué usarlos



| Re-conociendo al grupo

Este es el módulo 3 de 8 en este master

**En este módulo veremos JS
Aprenderemos lo básico y
practicaremos cómo abordar
problemas de lógica de
programación**

DEV.F:

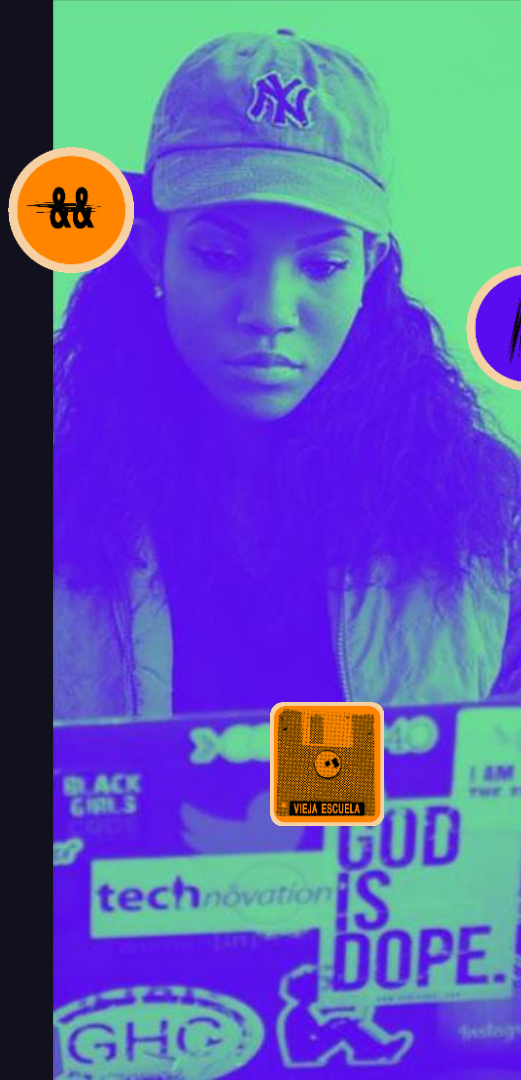


| Objetivos de la clase (180 min)

Al finalizar esta clase deberías ser capaz de identificar los **diferentes tipos de datos en JS**.

El ejercicio de esta clase consiste en crear un archivo que “imprima” los diferentes tipos de datos que vimos en clase. (chechar Campus)

(¿ todos los tipos de datos que existen en JS están en esta presentación?, ¿hay más?)

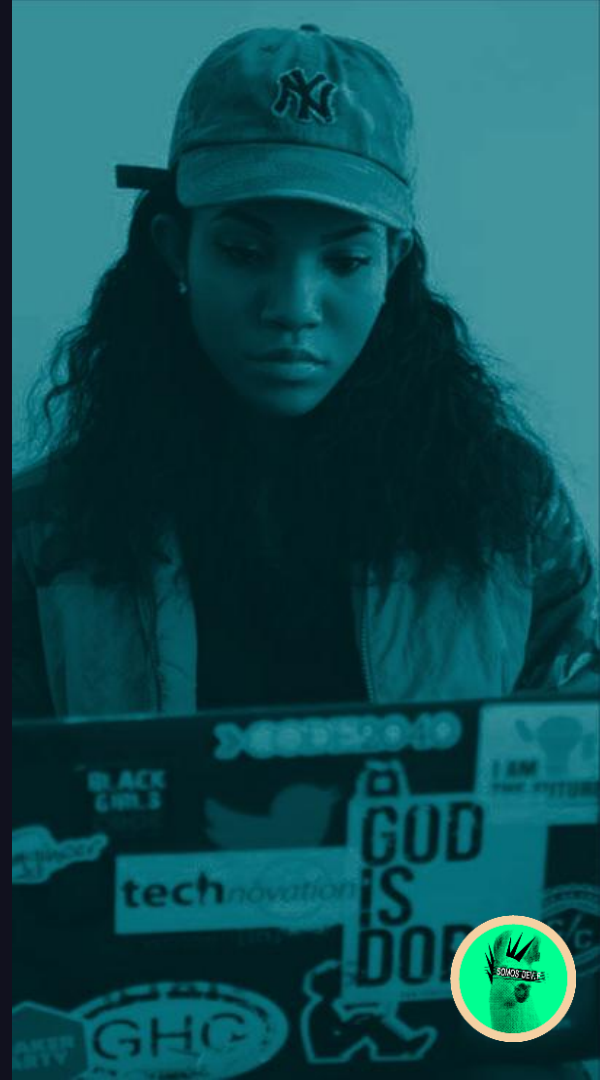


| Lenguaje de programación

- **Herramienta para comunicarse con una computadora**
- **Sistema de reglas y sintaxis**
- **Traduce lógica humana a código ejecutable**
- **Automatizar tareas**
- **Desarrollo de aplicaciones y sistemas**
- **Lenguajes de alto y bajo nivel:**

Los lenguajes de programación pueden ser de alto nivel (fáciles de entender para los humanos) o de bajo nivel (más cercanos al lenguaje que entiende la computadora).

DEV.F:



| JavaScript

- **JavaScript consiste en un lenguaje de programación interpretado, que habitualmente se utiliza en sitios web para ejecutar acciones en el lado del cliente, estando embebido en el código fuente de la página web.**





| “backward compatibility”

**Actualizar JavaScript
podría **romper** algunas
características de la web,
por lo que su evolución
debe de ser cuidadosa,
añadiendo nuevas
funcionalidades sin
eliminar antiguas.**

| Mejorar la experiencia de usuario

Usar JS nos permite en nuestros sitios:

- Interactividad en tiempo real
- Validación de formularios
- Actualización dinámica de contenido
- Animaciones y transiciones
- Navegación mejorada
- Interacción basada en eventos
- Cargas asíncronas



Consola del navegador. Cómo y para qué se usa

- Consola del navegador



DEV.F.



Consola del navegador. Cómo y para qué se usa

- Consola del navegador



DEV.F.



| Tipos de datos

Los datos en JavaScript existen porque son fundamentales para cualquier programa, ya que permiten al código manejar información y realizar acciones con ella.

Pueden ser palabras, números, estados verdaderos o falsos, y mucho más.

DEV.F:



I Tipos de datos



Existen datos primitivos y compuestos.

En este módulo nos concentraremos en los primitivos y algunos compuestos.

Primitivos

string → "Hola"

number → 42

boolean → true / false

null → Representa "ausencia de valor"

undefined → Representa una variable sin inicializar

bigint → Para números muy grandes

symbol → Para valores únicos e inmutables

Compuestos

Object

Array

Function

Date, RegExp, Map, Set,
etc.

| “Strings”

El tipo **String** de JavaScript se utiliza para representar datos textuales.

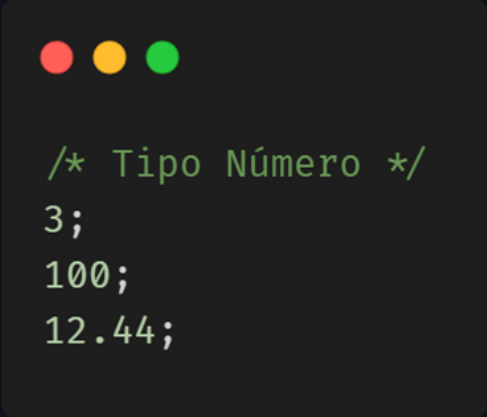
Normalmente se representan encerrando el valor textual entre comillas simples o dobles.



```
"Soy un String";  
'También soy un String';
```

| Numbers

Son valores numéricos (entre $-(2^{53} - 1)$ y $2^{53} - 1$). También comprende números de punto flotante (decimales).



```
/* Tipo Número */  
3;  
100;  
12.44;
```

También pueden contener valores **NaN** ("Not a Number" o No es un número).

| Booleans

En términos de informática, un **boolean** es un dato lógico que solo puede tener los valores **true (verdadero)** o **false (falso)**.



```
/* Tipo Boolean */  
true; // Verdadero  
false; // Falso
```

Juegan un papel fundamental en la lógica de programación, ya que podemos establecer acciones dependiendo si se cumple o no cierta condición.

| Null

El tipo Null tiene el valor: **nulo**.

```
var miVariable = null  
  
console.log(miVariable) // null
```

Un valor **nulo** significa que es un valor desconocido, indefinido o no inicializado

| Undefined

Una variable a la cual no se le haya asignado valor tiene el valor **indefinido**.

```
var saludo;  
console.log(typeof saludo); // "undefined"
```