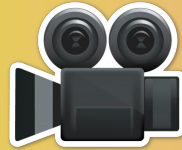


***RECUERDA PONER A GRABAR LA
CLASE***





¿DUDAS DEL ON-BOARDING?

MIRALO AQUI



Clase 02. JAVASCRIPT

CONTROL DE FLUJOS

GLOSARIO:

Clase 1

JavaScript: es un lenguaje de programación que se utiliza principalmente para aportar dinamismo a los sitios web.

Variable: es un espacio reservado en la memoria que, como su nombre indica, puede cambiar de contenido a lo largo de la ejecución de un programa. Podemos almacenar un número, un texto, un listado de números, etcétera.

Algoritmo: en programación, es un conjunto de procedimientos o funciones que se necesitan para realizar cierta operación o acción.

Operadores lógicos: permiten agrupar expresiones lógicas. Las expresiones lógicas son todas aquellas expresiones que obtienen como resultado verdadero o falso. Los operadores lógicos son aquellos que hacen de nexo de este tipo de expresiones.

Anidar: en programación, se refiere a escribir una sentencia junto a una subsiguiente dentro de la misma estructura sintáctica. Es decir, que no hay un salto de línea en el medio.

Parsear: es una palabra devengada del inglés "parse". Refiere en programación, a una actividad que consiste en el análisis de texto para determinar si cumple o no reglas o patrones y en base a esto tomar alguna determinación.

Script: un script es una secuencia de instrucciones que realizan una o más tareas.



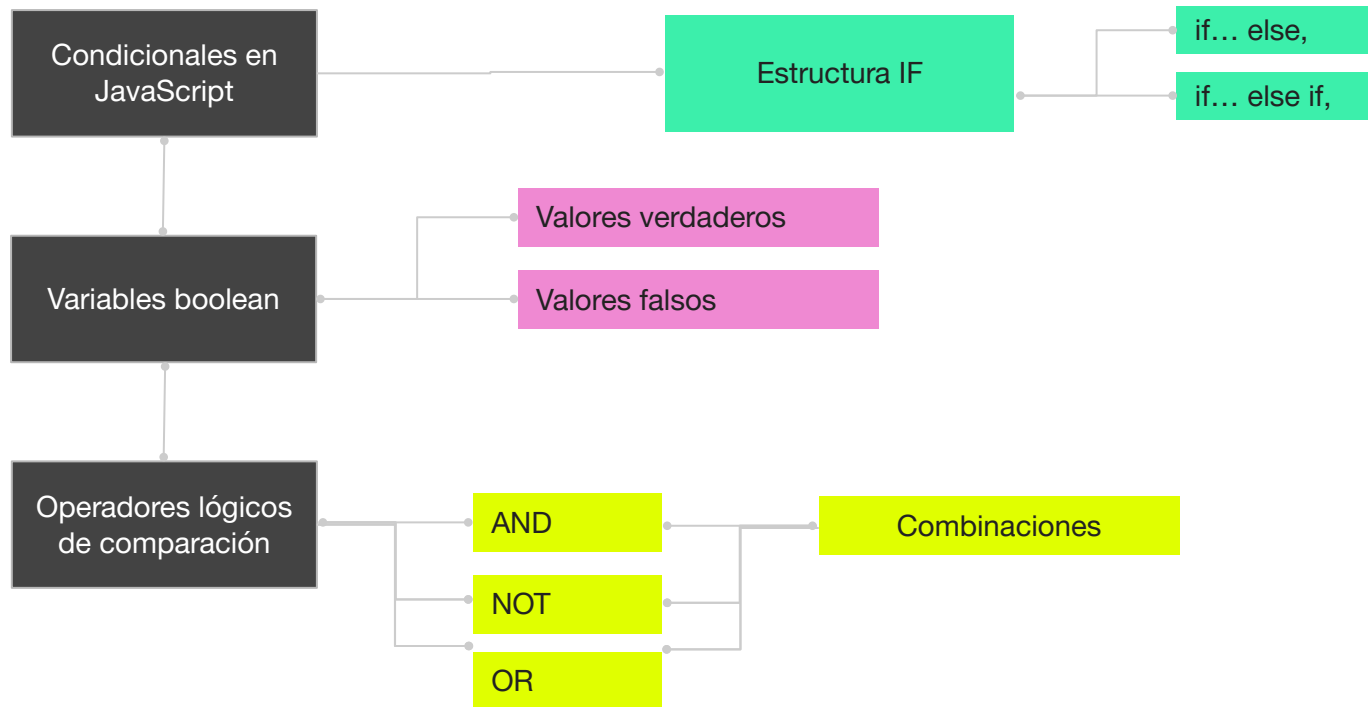
OBJETIVOS DE LA CLASE

- Entender qué es un **condicional** y cómo nos permite tomar decisiones.
- Comprender cómo JavaScript evalúa un valor como **verdadero** o **falso**.
- Identificar **operadores** lógicos de comparación y comprender su aplicación en condicionales.

MAPA DE CONCEPTOS

MAPA DE CONCEPTOS CLASE 2

¡Para
recordar!



MÓDULOS DE TRABAJO

MÓDULO 0 NIVELACIÓN

CLASE 0 -
INTRODUCCIÓN A JAVASCRIPT

MÓDULO 1 CONCEPTOS BÁSICOS

CLASE 1 -
CONCEPTOS GENERALES:
SINTAXIS Y VARIABLES

CLASE 2 -
CONTROL DE FLUJOS

CLASE 3 -
CICLOS E ITERACIONES

CLASE 4 -
FUNCIONES

- **Desafío entregable**

MÓDULO 2 OBJETOS & ARRAYS

CLASE 5 -
OBJETOS

CLASE 6 -
ARRAYS

CLASE 7 -
FUNCIONES DE ORDEN SUPERIOR

- **1ra pre-entrega**



HERRAMIENTAS DE LA CLASE

Les compartimos algunos recursos para acompañar la clase

- Guión de clase N° 2 [aquí](#).
- Quizz de clase N° 2 [aquí](#)
- Booklet de Javascript [aquí](#)
- FAQs de Javascript [aquí](#)

EMPEZAMOS...



CODER HOUSE

CONDICIONALES EN JS

SI



CONDICIONALES: DEFINICIÓN

Cuando en programación hablamos de condicionales, hablamos de una **estructura sintáctica** que sirve para tomar una **decisión** a partir de una **condición**.

Si *<condición>* *entonces*
<operación>

CONTROL DE FLUJOS

Hasta ahora todas las instrucciones que escribimos se ejecutan en *línea recta*, una sentencia después de la otra.



La idea del control de flujos es marcar puntos en nuestra aplicación donde, a partir de alguna evaluación, nuestro programa pueda tomar varios caminos posibles de acción.



VALORES BOOLEANOS

Para generar estos controles empezamos a trabajar con los valores ***booleanos***. Estos pueden tomar dos valores posibles, **true** o **false**, verdadero o falso.

Así, para que nuestro programa tome un camino u otro le vamos a decir que resuelva alguna evaluación/comparación. Si es verdadera (true) tomará un camino, y si es falsa (false) tomará otro.

ESTRUCTURA IF

La estructura más utilizada en la mayoría de los lenguajes, y por ende también en JS, es la estructura *if*

```
// si - condicion
if (true) {
    // bloque de código a ejecutar
    console.log("vas a ver este mensaje");
}
```

ESTRUCTURA IF

Si la condición se cumple (es decir, si su valor es **true**) se ejecutan todas las instrucciones que se encuentran dentro de {...}. Si la condición no se cumple (es decir, si su valor es **false**) no se ejecuta ninguna instrucción contenida en {...} y el programa continúa ejecutando el resto de instrucciones del script.

```
if (false){  
    console.log("no vas a ver este mensaje");  
}
```


COMPARACIÓN

Cuando utilizamos operadores *matemáticos* entre dos valores numéricos, éstos resuelven un nuevo tipo de valor numérico que es el resultado de la operación. Cuando ***comparamos*** dos valores a través de un operador de comparación, ésta operación siempre se resuelve en ***true*** o ***false***, es decir la comparación es verdadera o falsa.

El primer operador de comparación es el operador de equivalencia ==

EJEMPLO DE CONDICIONALES

```
let unNumero = 5

// Con (unNumero == 5) comparamos si unNumero es igual a 5
if (unNumero == 5){
  console.log("vas a ver este mensaje");
}

// Con (unNumero == 6) comparamos si unNumero es igual a 6
if (unNumero == 6){
  console.log("no vas a ver este mensaje");
}
```

En este ejemplo, las comparaciones se realizan entre el valor de la variable **unNumero** y un valor numérico.

En el primer condicional, como los dos valores coinciden, la igualdad se cumple, y por lo tanto la condición es cierta; su valor es **true**, y se ejecutan las instrucciones contenidas en el bloque del if.

EJEMPLO DE CONDICIONALES

```
let unNumero = 5

// Con (unNumero == 5) comparamos si unNumero es igual a 5
if (unNumero == 5){
    console.log("vas a ver este mensaje");
}

// Con (unNumero == 6) comparamos si unNumero es igual a 6
if (unNumero == 6){
    console.log("no vas a ver este mensaje");
}
```

En el segundo caso unNumero no es igual a 6; su valor es **false**, y **no** se ejecutan las instrucciones contenidas en el bloque del if.

EJEMPLO DE CONDICIONALES

```
let unNumero = 5

// Con (unNumero == 5) comparamos si unNumero es igual a 5
if (unNumero == 5){
    console.log("vas a ver este mensaje");
}

// Con (unNumero == 6) comparamos si unNumero es igual a 6
if (unNumero == 6){
    console.log("no vas a ver este mensaje");
}
```

La comparación del ejemplo suele ser el origen de muchos errores de programación, al confundir los operadores `==` y `=`. Las comparaciones siempre se realizan con el operador `==`, ya que el operador `=` sirve para asignar valores.

IF...ELSE

En ocasiones, las decisiones que se deben realizar no son del tipo «*si se cumple la condición, hazlo; si no se cumple, no hagas nada*». Normalmente las condiciones suelen ser del tipo **«*si se cumple esta condición, hazlo; si no se cumple, haz esto otro*»**.

```
let unColor = "Rojo"

// Con (unColor == "Rojo") comparamos si unColor es igual "Rojo"
if (unColor == "Rojo"){
    console.log("el color es Rojo");
}else{
    //La instrucción se interpreta cuando unColor NO es "Rojo"
    console.log("el color NO es Rojo");
}
```

EJEMPLO DE IF...ELSE

```
let nombreUsuario = prompt("Ingresar nombre de usuario");

if (nombreUsuario == "") {
    alert("No ingresaste el nombre de usuario");
}
else {
    alert("Nombre de usuario ingresado " + nombreUsuario);
}
```

CONDICIONES ANIDADAS IF..ELSE IF

```
let precio = 100.5;

if (precio < 20) {
    alert("El precio es menor que 20");
}
else if (precio < 50) {
    alert("El precio es menor que 50");
}
else if (precio < 100) {
    alert("El precio es menor que 100");
}
else {
    alert("El precio es mayor que 100");
}
```

VARIABLES BOOLEAN

TRUE o FALSE

Las variables booleanas son las que sólo tienen dos valores, true or false. Pueden recibir el valor a partir de una evaluación booleana sobre otras variables:

```
let numero    = 10;  
let esMayor5 = (numero > 5); // su valor sera true  
  
if (esValida) {  
    alert("Es boolean true");  
}
```

Ejemplo
en vivo



¡VAMOS A PRACTICAR LO VISTO!

CODER HOUSE



BREAK

¡5/10 MINUTOS Y VOLVEMOS!

OPERADORES LÓGICOS



OPERADORES EN JS

En JavaScript, disponemos de los **operadores lógicos** habituales en lenguajes de programación como son: **es igual, es distinto, menor, menor o igual, mayor, mayor o igual, and (y), or (o) y not (no).**

La sintaxis se basa en símbolos, como veremos a continuación.

Cabe destacar que hay que prestar atención a no confundir '==' con '=' porque implican distintas cosas.

OPERADORES EN JS



OPERADORES LÓGICOS Y RELACIONALES	DESCRIPCIÓN	EJEMPLO
==	Es igual	a == b
===	Es estrictamente igual	a === b
!=	Es distinto	a != b
!==	Es estrictamente distinto	a !== b
<, <=, >, >=	Menor, menor o igual, mayor, mayor o igual	a <= b
&&	Operador and (y)	a && b
	Operador or (o)	a b
!	Operador not (no)	!a

CONDICIONES COMPUESTAS CON &&

Ante una combinación de operadores && (AND) será requisito que todas las comparaciones sean verdaderas para que la condición compuesta sea verdadera.

```
let nombreIngresado = prompt("Ingresar nombre");  
let apellidoIngresado = prompt("Ingresar apellido");  
  
if((nombreIngresado != "") && (apellidoIngresado != "")){  
    alert("Nombre: "+nombreIngresado +"\nApellido: "+apellidoIngresado);  
}else{  
    alert("Error: Ingresar nombre y apellido");  
}
```

CONDICIONES COMPUESTA CON ||

En caso de utilizar || (OR), será requisito que al menos una de las comparaciones sea verdadera para que la condición compuesta sea verdadera.

```
let nombreIngresado = prompt("Ingresar nombre");

if((nombreIngresado == "ANA") || (nombreIngresado == "ana")){
    alert("El nombre ingresado es Ana");
}else{
    alert("El nombre ingresado NO ES Ana");
}
```


COMBINACIÓN DE OPERADORES && y ||

También **es posible combinar || (OR) y && (AND)** para hacer comparaciones cada vez más complejas.

```
let nombreIngresado = prompt("Ingresar nombre");

if((nombreIngresado != "") && ((nombreIngresado == "EMA") || (nombreIngresado == "ema"))){
    alert("Hola Ema");
}else{
    alert("Error: Ingresar nombre valido");
}
```

COMBINACIÓN DE OPERADORES && y ||

Ya que las expresiones lógicas son evaluadas de izquierda a derecha, es necesario agrupar las operaciones para asegurar que se cumplan como uno lo desea. El cambio de agrupación con los paréntesis produce resultados diferentes.

No es lo mismo:

```
if((nombreIngresado != "") && ((nombreIngresado == "EMA") || (nombreIngresado == "ema"))){
```

que:

```
if(((nombreIngresado != "") && (nombreIngresado == "EMA")) || (nombreIngresado == "ema")){
```

¡HANDS ON!



CODER HOUSE



ACTIVIDAD EN CLASE

¡Llevemos lo visto hasta el momento a la acción!
Les proponemos que en salas de zoom lideradas por su tutor/a
puedan realizar la siguiente actividad.

Tiempo estimado 25/30 minutos



CREAR UN ALGORITMO CON UN CONDICIONAL

Crea un algoritmo que solicite al usuario uno o más valores ingresados por `prompt()`, compare las entradas y, en función de ciertas condiciones, muestre un resultado.

Tiempo estimado 25/30 minutos.

CREAR UN ALGORITMO CON UN CONDICIONAL

Formato: Página HTML y código fuente en JavaScript en archivo .js vinculado al html.

Sugerencia: Tener en cuenta que los valores obtenidos por prompt() son string, si se busca operar con números hay que parsearlos antes y si van a usar cadenas recordar tener cuidado con mayúsculas y minúsculas en las comparaciones de igualdad. (Ej. "Hola" y "HOLA" no son iguales)



>> Consigna: Crea un algoritmo que solicite al usuario uno o más valores ingresados por prompt(), compare las entradas y, en función de ciertas condiciones, muestre por consola o alert() el resultado según los valores ingresados y las condiciones cumplidas.

>>Aspectos a incluir en el entregable:

Archivo HTML y Archivo JS, referenciado en el HTML por etiqueta `<script src="js/miarchivo.js"></script>`, que incluya la definición de un algoritmo en JavaScript que emplee instrucciones condicionales.

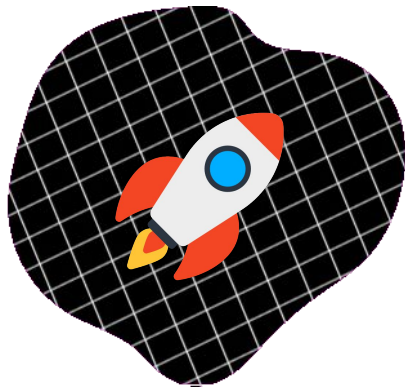
>>Ejemplo:

- Pedir número mediante prompt y si es mayor a 1000 mostrar un alert.
- Pedir un texto mediante prompt, y si es igual a "Hola" mostrar un alerta por consola.
- Pedir un número por prompt y evaluar si está entre 10 y 50. En caso positivo mostrar un alert.

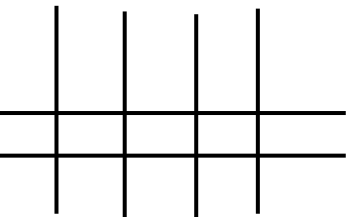
PREPARÁNDONOS PARA EL DESAFÍO ENTREGABLE N° 1

*En la clase 4 finaliza el primer módulo y se entregarán las consignas del **primer desafío entregable** del curso. El mismo, incluirá temas vistos en las clases 1, 2, 3 y 4.*





- El entregable se compone de temas vistos hasta el momento, más otros que verán durante el **módulo completo** 💪.
- Te recomendamos ir avanzando con los "Hands On" y "Desafíos Complementarios" ✨
- Recuerden que recién la consigna del desafío se entrega ¡en la clase N° 4! 🙌 **Y tendrán hasta 7 días para resolver el desafío y subirlo.**



DESAFÍO ENTREGABLE N° 1

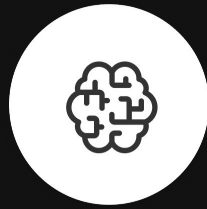
Compuesto por...



- a) **Crear un algoritmo con un condicional** ✓
- b) Crear un algoritmo utilizando un ciclo 🕒
- c) Armar un simulador interactivo, la estructura final de tu proyecto integrador 🕒

¿PREGUNTAS?





¡PARA PENSAR!

¿Te gustaría comprobar tus conocimientos de la clase?

Te compartimos a través del chat de zoom
el enlace a un breve quiz de tarea.

Para el profesor:

- *Acceder a la carpeta "Quizzes" de la camada*
 - *Ingresar al formulario de la clase*
 - *Pulsar el botón "Invitar"*
 - *Copiar el enlace*
- *Compartir el enlace a los alumnos a través del chat*



RECURSOS:

Material
ampliado



- Conversión de tipos de datos, operadores y sentencias condicionales |
Los apuntes de Majo (Página 9 a 16).
- Operadores y condicionales |
Te lo explico con gatitos Operadores.
Te lo explico con gatitos Operadores Lógicos.
Te lo explico con gatitos Condicionales.
- Práctica interactiva sobre operaciones con JavaScript |
Silent teacher.
- Documentación |
Documentación IF ELSE.
Documentación SWITCH.

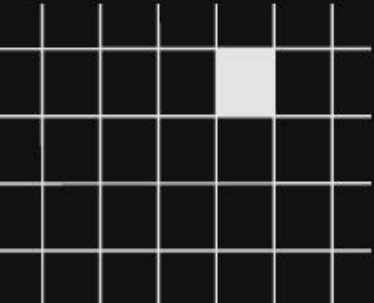
Disponible en [nuestro repositorio](#).

CODER HOUSE



¡MUCHAS GRACIAS!

Resumen de lo visto en clase hoy:

- Operador if, else y sus variantes.
 - Variables boolean.
 - Operaciones lógicas: AND, OR, NOT y combinaciones.
- 



OPINA Y VALORA ESTA CLASE

#DEMOCRATIZANDO LA EDUCACIÓN

CODER HOUSE