

Recuerden poner a
grabar la clase



Clase 2

Condicionales

Diplomatura UNTREF



Temario

- Parsear Datos
- Condicionales en Js
 - Operadores de comparación
 - Estructura if
 - Declaración if simple
- El operador else
- Operadores lógicos
- If anidados: if -else if
- La estructura switch



Parsear Datos

Parsear datos

"Parsear" en JavaScript se refiere a convertir una cadena de texto en otro tipo de dato, como un número, un booleano o un objeto. En general, JavaScript considera todas las entradas de usuario como cadenas de texto, incluso si el usuario ingresa números o valores booleanos.

Cuando necesitas trabajar con datos en un formato diferente al de una cadena de texto, puedes utilizar métodos de parseo para convertir esos datos en el tipo de dato adecuado.

Parsear datos “parseInt{”

Este método se utiliza para convertir una cadena de texto en un número entero. Elimina los caracteres no numéricos al principio de la cadena y devuelve el número entero resultante.

```
let num1 = "10";  
let num2 = "20";  
  
let suma = parseInt(num1) + parseInt(num2);  
console.log(suma); // Resultado: 30
```

Parsear datos “parseFloat{”

Este método se utiliza para convertir una cadena de texto en un número decimal (de punto flotante). También elimina los caracteres no numéricos al principio de la cadena y devuelve el número decimal resultante.

```
let num1 = "3.14";  
let num2 = "2.5";  
  
let suma = parseFloat(num1) + parseFloat(num2);  
console.log(suma); // Resultado: 5.64
```

Parsear datos “boolean{”

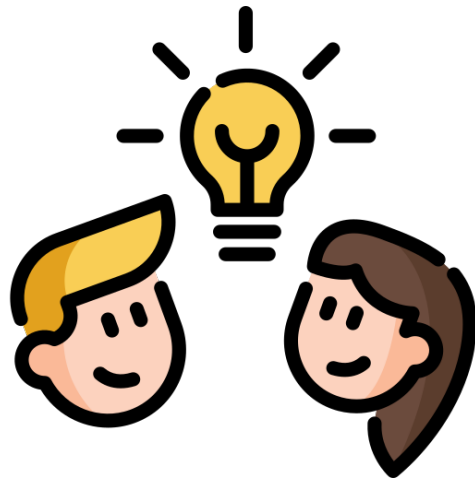
Este constructor de objetos se utiliza para convertir un valor en su equivalente booleano. Puede convertir valores como cadenas de texto, números, objetos y más.

```
var valor1 = "Hola";  
var valor2 = "";  
var valor3 = 0;  
  
console.log(Boolean(valor1)); // Resultado: true  
console.log(Boolean(valor2)); // Resultado: false  
console.log(Boolean(valor3)); // Resultado: false
```


Condicionales

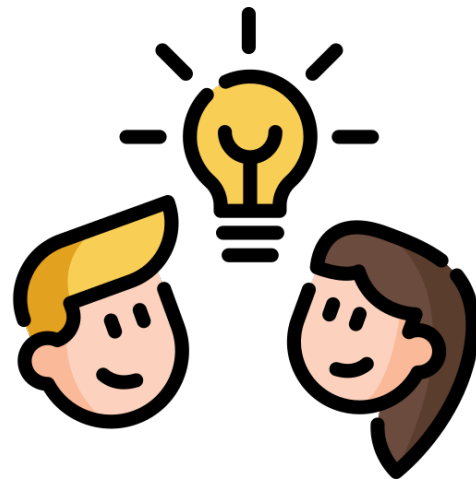
Condicionales

Los condicionales en JavaScript son estructuras de control que te permiten tomar decisiones basadas en el resultado de una evaluación lógica. Se utilizan para ejecutar diferentes bloques de código según se cumplan o no ciertas condiciones. Los condicionales más comunes son "if", "else", y "else if", que ya mencionamos anteriormente.



Condicionales

Cuando utilizas condicionales, se evalúa una expresión o condición que produce un valor booleano (verdadero o falso). Dependiendo de si el valor de la condición es verdadero o falso, se ejecuta el bloque de código correspondiente.



Operadores de comparación

Operadores de comparación

En JavaScript, existen varios operadores de comparación que se utilizan para comparar valores y producir un resultado booleano (verdadero o falso) según la evaluación de la comparación.



Operadores de igualdad ('==')

El operador de igualdad compara si dos valores son iguales, sin tener en cuenta el tipo de dato. Si los valores son iguales, devuelve true; de lo contrario, devuelve false.

```
console.log(5 == 5); // true  
console.log("5" == 5); // true  
console.log(5 == 10); // false
```

Operadores de desigualdad ('!=')

El operador de desigualdad compara si dos valores no son iguales, sin tener en cuenta el tipo de dato. Si los valores son diferentes, devuelve true; de lo contrario, devuelve false.

```
console.log(5 == 5); // true  
console.log("5" == 5); // true  
console.log(5 == 10); // false
```

Operadores de igualdad estricta ('===')

El operador de igualdad estricta compara si dos valores son iguales y del mismo tipo de dato. Si los valores son iguales y del mismo tipo, devuelve true; de lo contrario, devuelve false.

```
console.log(5 === 5); // true  
console.log("5" === 5); // false
```


Operadores de desigualdad estricta ('!==')

El operador de desigualdad estricta compara si dos valores no son iguales o si son de diferente tipo de dato. Si los valores son diferentes o de diferente tipo, devuelve true; de lo contrario, devuelve false.

```
console.log(5 !== 5); // false  
console.log("5" !== 5); // true
```

“Operador de mayor que (>)”

El operador de mayor que compara si el valor de la izquierda es mayor que el de la derecha. Si la condición es verdadera, devuelve true; de lo contrario, devuelve false.

```
console.log(5 > 2); // true  
console.log(2 > 5); // false
```

“Operador de mayor o igual que (\geq)”

El operador de mayor o igual que compara si el valor de la izquierda es mayor o igual que el de la derecha. Si la condición es verdadera, devuelve true; de lo contrario, devuelve false.

```
console.log(5 >= 5); // true  
console.log(5 >= 2); // true  
console.log(2 >= 5); // false
```

“Operador de menor o igual que (<=)”

El operador de menor o igual que compara si el valor de la izquierda es menor o igual que el de la derecha. Si la condición es verdadera, devuelve true; de lo contrario, devuelve false.

```
console.log(5 <= 5); // true  
console.log(2 <= 5); // true  
console.log(5 <= 2); // false
```

Estructura “if”

“estructura if”

La estructura if en JavaScript permite tomar decisiones en el flujo de ejecución de un programa basadas en una condición. Se utiliza para ejecutar un bloque de código si la condición especificada es verdadera y, opcionalmente, se puede proporcionar un bloque else para ejecutar otro conjunto de instrucciones en caso de que la condición sea falsa.

```
if (condición) {  
    // Bloque de código a ejecutar si la  
    // condición es verdadera  
} else {  
    // Bloque de código a ejecutar si la  
    // condición es falsa (opcional)  
}
```

“Declaración de if simple”

En una declaración de if simple, solo se proporciona el bloque de código a ejecutar cuando la condición es verdadera. El bloque de código está delimitado por llaves {} y puede contener una o varias instrucciones.

```
var edad = 20;

if (edad >= 18) {
  console.log("Eres mayor de edad.");
  console.log("Puedes ingresar al sitio.");
}
```

Operador “else”

“Operador Else”

El bloque else se utiliza en conjunto con la estructura if en JavaScript para proporcionar una alternativa de código que se ejecuta cuando la condición del if es falsa. Proporciona una rama de ejecución alternativa cuando la condición especificada no se cumple.

```
if (condición) {  
    // Bloque de código a ejecutar si la  
    condición es verdadera  
} else {  
    // Bloque de código a ejecutar si la  
    condición es falsa  
}
```

“Operador Else”

En el bloque else, se especifica el código que se ejecutará cuando la condición del if sea falsa. El bloque else es opcional, pero puede ser útil para manejar casos alternativos o establecer un comportamiento predeterminado cuando la condición del if no se cumple.

```
var hora = 14;

if (hora < 12) {
  console.log("Buenos días");
} else {
  console.log("Buenas tardes");
}
```

“Operador Else”

El bloque else permite definir una rama alternativa de ejecución, lo que significa que se pueden realizar diferentes acciones según el resultado de la condición. Puede haber múltiples bloques else if después del if para evaluar y manejar diferentes condiciones.

```
var hora = 20;

if (hora < 12) {
  console.log("Buenos días");
} else if (hora < 18) {
  console.log("Buenas tardes");
} else {
  console.log("Buenas noches");
}
```

Operadores lógicos

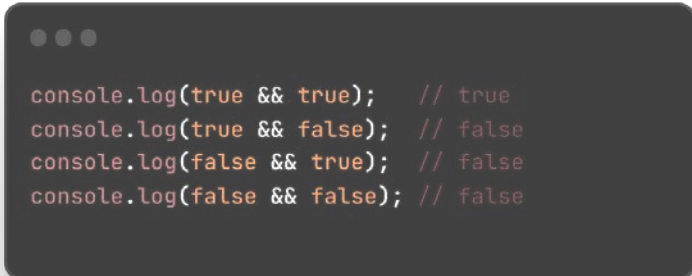
Operadores lógicos

Los operadores lógicos en JavaScript se utilizan para combinar y evaluar múltiples expresiones lógicas. Hay tres operadores lógicos principales: **&&** (AND lógico), **||** (OR lógico) y **!** (NOT lógico).



Operador AND Lógico (&&):

El operador && devuelve true si ambas expresiones que se evalúan a su izquierda y a su derecha son verdaderas. Si al menos una de las expresiones es falsa, devuelve false.



```
console.log(true && true);    // true
console.log(true && false);   // false
console.log(false && true);   // false
console.log(false && false);  // false
```

Operador OR Lógico (||)

El operador `||` devuelve `true` si al menos una de las expresiones que se evalúan a su izquierda o a su derecha es verdadera. Solo devuelve `false` si ambas expresiones son falsas.

```
console.log(true || true);    // true
console.log(true || false);   // true
console.log(false || true);   // true
console.log(false || false);  // false
```

Operador NOT Lógico (!)

El operador “!” Invierte el valor de una expresión lógica. Si la expresión es true, devuelve false. Si la expresión es false, devuelve true.

```
console.log(!true); // false  
console.log(!false); // true
```


Operadores lógicos

Estos operadores lógicos se utilizan para combinar y evaluar expresiones lógicas más complejas en JavaScript. Pueden ser útiles para tomar decisiones condicionales basadas en múltiples condiciones.

```
let edad = 25;
let tieneLicencia = true;

if (edad >= 18 && tieneLicencia) {
  console.log("Puede conducir un automóvil.");
} else {
  console.log("No puede conducir un
automóvil.");
}
```

If anidado

If anidado

El "if anidado" se refiere a la práctica de incluir una estructura if dentro de otra estructura if. Esto permite evaluar múltiples condiciones de forma jerárquica y tomar decisiones basadas en esos resultados.

```
if (condición1) {  
    // Bloque de código a ejecutar si la  
    condición1 es verdadera  
  
    if (condición2) {  
        // Bloque de código a ejecutar si la  
        condición2 es verdadera  
    } else {  
        // Bloque de código a ejecutar si la  
        condición2 es falsa  
    }  
  
} else {  
    // Bloque de código a ejecutar si la  
    condición1 es falsa  
}
```

If anidado ejemplo

En este ejemplo, primero se verifica si edad es mayor o igual a 18. Si es así, se muestra "Es mayor de edad." Luego, dentro de ese bloque, se evalúa si tieneLicencia es verdadero o falso. Si tiene licencia, se muestra "Puede conducir un automóvil." Si no tiene licencia, se muestra "No tiene licencia de conducir." Si la edad es menor de 18, se muestra "Es menor de edad."

```
var edad = 18;
var tieneLicencia = true;

if (edad >= 18) {
  console.log("Es mayor de edad.");

  if (tieneLicencia) {
    console.log("Puede conducir un
automóvil.");
  } else {
    console.log("No tiene licencia de
conducir.");
  }
} else {
  console.log("Es menor de edad.");
}
```

If anidado ejemplo

En este ejemplo, primero se verifica si edad es mayor o igual a 18. Si es así, se muestra "Es mayor de edad." Luego, dentro de ese bloque, se evalúa si tieneLicencia es verdadero o falso. Si tiene licencia, se muestra "Puede conducir un automóvil." Si no tiene licencia, se muestra "No tiene licencia de conducir." Si la edad es menor de 18, se muestra "Es menor de edad."

```
var edad = 18;
var tieneLicencia = true;

if (edad >= 18) {
  console.log("Es mayor de edad.");

  if (tieneLicencia) {
    console.log("Puede conducir un
automóvil.");
  } else {
    console.log("No tiene licencia de
conducir.");
  }
} else {
  console.log("Es menor de edad.");
}
```

Estructura else if

Estructura Else If

La estructura else if se utiliza como una extensión de la estructura if en JavaScript. Permite evaluar múltiples condiciones en secuencia y ejecutar diferentes bloques de código según el resultado de cada una de esas condiciones.

```
if (condición1) {  
    // Bloque de código a ejecutar si la  
    condición1 es verdadera  
} else if (condición2) {  
    // Bloque de código a ejecutar si la  
    condición2 es verdadera  
} else if (condición3) {  
    // Bloque de código a ejecutar si la  
    condición3 es verdadera  
} else {  
    // Bloque de código a ejecutar si ninguna de  
    las condiciones anteriores es verdadera  
}
```

Estructura switch

Estructura switch

La estructura switch en JavaScript es una declaración de control de flujo que permite evaluar una expresión y ejecutar diferentes bloques de código en función de los casos coincidentes. Proporciona una alternativa más legible y estructurada cuando se deben tomar múltiples decisiones basadas en el valor de una expresión.

```
switch (expresión) {  
  case valor1:  
    // Bloque de código a ejecutar cuando la  
    expresión coincide con valor1  
    break;  
  case valor2:  
    // Bloque de código a ejecutar cuando la  
    expresión coincide con valor2  
    break;  
  case valor3:  
    // Bloque de código a ejecutar cuando la  
    expresión coincide con valor3  
    break;  
  // Otros casos posibles  
  default:  
    // Bloque de código a ejecutar si no hay  
    coincidencias con ninguno de los casos  
    anteriores  
    break;  
}
```

Estructura switch a detalle

- **expresión:** Es la expresión que se va a evaluar y comparar con los diferentes casos. Puede ser cualquier valor o variable que se pueda evaluar.
- **valor1, valor2, valor3, etc.:** Son los posibles valores con los que se comparará la expresión. Si la expresión coincide con alguno de estos valores, se ejecutará el bloque de código correspondiente a ese caso.
- **case:** Cada caso representa una opción a evaluar en relación con la expresión. Si la expresión coincide con un valor determinado, se ejecutará el bloque de código asociado a ese caso.
- **break:** Es una declaración opcional que se utiliza para salir de la estructura switch después de que se haya ejecutado el bloque de código correspondiente al caso coincidente. Sin el break, el código continuaría ejecutándose en los casos siguientes, incluso si no hay coincidencias.
- **default:** Es un caso opcional que se utiliza cuando no hay coincidencias con ninguno de los casos anteriores. El bloque de código asociado a default se ejecutará en ese caso.

```
switch (expresión) {
    case valor1:
        // Bloque de código a ejecutar cuando la
        // expresión coincide con valor1
        break;
    case valor2:
        // Bloque de código a ejecutar cuando la
        // expresión coincide con valor2
        break;
    case valor3:
        // Bloque de código a ejecutar cuando la
        // expresión coincide con valor3
        break;
    // Otros casos posibles
    default:
        // Bloque de código a ejecutar si no hay
        // coincidencias con ninguno de los casos
        // anteriores
        break;
}
```

Estructura switch

La estructura switch es especialmente útil cuando se necesitan comparar múltiples casos posibles y se quiere evitar una serie de declaraciones if-else anidadas. Proporciona una forma más clara y concisa de estructurar el código.

```
var día = "Lunes";

switch (día) {
  case "Lunes":
    console.log("Hoy es lunes. Comienza la semana.");
    break;
  case "Martes":
    console.log("Hoy es martes.");
    break;
  case "Miércoles":
    console.log("Hoy es miércoles.");
    break;
  case "Jueves":
    console.log("Hoy es jueves.");
    break;
  case "Viernes":
    console.log("Hoy es viernes. ¡Fin de semana pronto!");
    break;
  default:
    console.log("Es un día diferente a lunes, martes, miércoles, jueves o viernes.");
    break;
}
```

¿Dudas o consultas?





¡Muchas Gracias!