



INFINITY SCHOOL

VISUAL ART CREATIVE CENTER

AULA 02 - CONDICIONAIS EM JAVASCRIPT

O QUE IREMOS APRENDER

- 01** REVISÃO DA AULA ANTERIOR
- 02** OPERADORES LÓGICOS
- 03** OPERADOR DE COMPARAÇÃO
- 04** CONDICIONAIS
- 05** TERNÁRIOS CONDICIONAIS
- 06** SWITCH

Revisão da Aula Anterior

- INTRODUÇÃO AO JAVASCRIPT
- ALERT() & CONSOLE.LOG()
- VARIÁVEIS
- TIPOS DE DADOS
- ENTRADA DE DADOS
- OPERADORES ARITMÉTICOS
- OPERADORES DE ATRIBUIÇÃO
- CONCATENAÇÃO E TEMPLATE STRINGS



OPERADORES LÓGICOS

Operadores lógicos combinam expressões ou valores booleanos para retornar **true** ou **false**. Eles são essenciais para decisões baseadas em múltiplas condições.

Os três principais operadores são: **&& (AND)**, **|| (OR)** e **! (NOT)**.

Operador	Significado
&&	and (y)
 	or (o)
!	not (no)

OPERADOR AND (&&)

O operador lógico **AND** retorna **true** se ambas as expressões operandos forem verdadeiras; caso contrário, retorna **false**.

E lógico (&&):

- Retorna verdadeiro se ambas as expressões forem verdadeiras.
- Exemplo: true && true retorna true, true && false retorna false.

```
let a = true;  
let b = false;  
console.log(a && b); // Output: false
```

A	B	A e B
TRUE	TRUE	TRUE
TRUE	FALSE	FALSE
FALSE	TRUE	FALSE
FALSE	FALSE	FALSE

TABELA VERDADE
DO OPERADOR AND

OPERADOR OR (||)

O operador lógico **OR** retorna **true** se pelo menos uma das expressões operandos for verdadeira; caso contrário, retorna **false**.

OU lógico (| |):

- Retorna true se pelo menos uma das expressões for true.
- Exemplo: true || false retorna true, false || false retorna false.

```
let a = true;  
let b = false;  
console.log(a || b); // Output: true
```

A	B	A e B
TRUE	TRUE	TRUE
TRUE	FALSE	TRUE
FALSE	TRUE	TRUE
FALSE	FALSE	FALSE

TABELA VERDADE
DO OPERADOR OR

IN

OPERADOR NOT (!)

O operador lógico **NOT** inverte o valor booleano de uma expressão. Se a expressão é **TRUE**, **!** retorna **FALSE**; se a expressão é **FALSE**, **!** retorna **TRUE**.

NÃO lógico (!):

- Retorna o oposto do valor booleano da expressão.
- Exemplo: !true retorna false, !false retorna true.

```
let a = true;  
console.log(!a); // Output: false
```

A	Não A
TRUE	FALSE
FALSE	TRUE

TABELA VERDADE
DO OPERADOR NOT

IN

ATIVIDADE PRÁTICA

Atividade 01

Solicite dois números ao usuário e verifique se ambos são positivos, se pelo menos um é positivo, e se nenhum é positivo, exibindo os resultados no console.

Use prompt para solicitar os números, armazene-os em numero1 e numero2, e utilize operadores lógicos para as verificações.

Objetivo:

Aprender a utilizar operadores lógicos (`&&`, `||`, `!`) para avaliar expressões e controlar o fluxo do código.



OPERADOR DE COMPARAÇÃO

Comparam dois valores e retornam um valor booleano (true ou false), essenciais para a tomada de decisões e controle de fluxo. Principais operadores de comparação:

Igualdade (==):

Compara se dois valores são iguais, convertendo os tipos se necessário.

Estritamente Igualdade (==):

Compara se dois valores são iguais sem converter os tipos.

Desigualdade (!=):

Compara se dois valores são diferentes, convertendo os tipos se necessário.

Estritamente Desigualdade (!==):

Compara se dois valores são diferentes sem converter os tipos.

Maior Que (>):

Compara se o valor à esquerda é maior que o valor à direita.

Maior ou Igual a (>=):

Compara se o valor à esquerda é maior ou igual ao valor à direita.

Menor Que (<):

Compara se o valor do lado esquerdo é menor que o valor do lado direito.

Menor ou Igual a (<=):

Compara se o valor do lado esquerdo é menor ou igual ao valor do lado direito.

OPERADOR DE COMPARAÇÃO - IGUALDADE

Igualdade (==):

Verifica se os valores dos operandos são iguais;
Realiza conversão de tipo implícita, se necessário.

```
1 console.log(5 == 5); // true
2 console.log(5 == '5'); // true
3 console.log(5 == 10); // false
```

Igualdade Estrita (===):

Verifica se os valores e os tipos dos operandos são iguais.
Não realiza conversão de tipo implícita.

```
1 console.log(5 === 5); // true
2 console.log(5 === '5'); // false
3 console.log(5 === 10); // false
```

OPERADOR DE COMPARAÇÃO - DIFERENÇA

Desigualdade (!=):

Verifica se os valores dos operandos não são iguais.
Realiza conversão de tipo implícita, se necessário.

```
1 console.log(5 != 5); // false
2 console.log(5 != '5'); // false
3 console.log(5 != 10); // true
```

Desigualdade Estrita (!==):

Verifica se os valores e os tipos dos operandos não são iguais.
Não realiza conversão de tipo implícita.

```
1 console.log(5 !== 5); // false
2 console.log(5 !== '5'); // true
3 console.log(5 !== 10); // true
```

OPERADOR DE COMPARAÇÃO - MAIOR

Maior Que (>):

Verifica se o valor do operando da esquerda é maior que o valor do operando da direita.

```
1 console.log(10 > 5); // true
2 console.log(5 > 10); // false
3 console.log(5 > 5); // false
```

Maior ou Igual Que (>=):

Verifica se o valor do operando da esquerda é maior ou igual ao valor do operando da direita.

```
1 console.log(10 >= 5); // true
2 console.log(5 >= 10); // false
3 console.log(5 >= 5); // true
```

OPERADOR DE COMPARAÇÃO - MENOR

Menor Que (<):

Verifica se o valor do operando da esquerda é menor que o valor do operando da direita.

```
1 console.log(5 < 10); // true  
2 console.log(10 < 5); // false  
3 console.log(5 < 5); // false
```

Menor ou Igual Que (<=):

Verifica se o valor do operando da esquerda é menor ou igual ao valor do operando da direita.

```
1 console.log(5 <= 10); // true  
2 console.log(10 <= 5); // false  
3 console.log(5 <= 5); // true
```

ATIVIDADE PRÁTICA

Atividade 02

Crie um código que solicite ao usuário dois números, compare-os com operadores de comparação.

Use prompt para obter os números e armazene-os em numero1 e numero2. Utilize operadores de comparação e mostre os resultados com console.log.

Objetivo:

Aprender a utilizar operadores de comparação (==, ===, !=, !==, <, >, <=, >=) para comparar valores e controlar o fluxo do código.

CONDICIONAIS

Condicionais são estruturas de controle de fluxo que permitem executar diferentes blocos de código com base em certas condições. Elas são fundamentais para a tomada de decisões dentro de um programa.

As principais estruturas condicionais são:

- if (Se):
- else if (Senão se):
- else (Senão):
- switch (Troca):

CONDICIONAIS - IF (SE)

if (Se):

- Executa um bloco de código se a condição especificada for verdadeira.

Sintaxe:

```
1 if (condicao) {  
2     // código a ser executado se  
3     a condição for verdadeira  
4 }
```

Exemplo:

```
1 let idade = 18;  
2  
3 if (idade >= 18) {  
4     console.log('Você é maior de idade.');// Output: Você é maior de idade.
```



CONDICIONAIS - ELSE IF (SENÃO SE)

else if (Senão se):

- Adiciona uma nova condição para testar se a condição anterior foi falsa.

Sintaxe:

```
1 if (condicao1) {  
2     // código a ser executado se a condição1  
3     for verdadeira  
4 } else if (condicao2) {  
5     // código a ser executado se a condição2  
6     for verdadeira  
7 }
```

Exemplo:

```
1 let idade = 16;  
2  
3 if (idade >= 18) {  
4     console.log('Você é maior de idade.');// Output: Você é maior de idade.  
5 } else {  
6     console.log('Você é menor de idade.');// Output: Você é menor de idade.  
7 }
```



CONDICIONAIS - ELSE (SENÃO)

else (Senão):

- Executa um bloco de código se todas as condições anteriores forem falsas.

Sintaxe:

```
1 if (condicao1) {  
2     // código a ser executado se a condição1 for verdadeira  
3 } else if (condicao2) {  
4     // código a ser executado se a condição2 for verdadeira  
5 } else {  
6     // código a ser executado se todas as condições anteriores  
    // forem falsas  
7 }
```

Exemplo:

```
1 let nota = 85;  
2  
3 if (nota >= 90) {  
4     console.log('Nota A');  
5 } else if (nota >= 80) {  
6     console.log('Nota B');  
7 } else if (nota >= 70) {  
8     console.log('Nota C');  
9 } else {  
10    console.log('Nota F');  
11 }  
12 // Output: Nota B
```



ATIVIDADE PRÁTICA

Atividade 03

Crie um código que solicite ao usuário um número, verifique se é positivo, negativo ou zero, e exiba uma mensagem no console.

Use prompt para obter o número, armazene-o em numero, e utilize if, else if, e else para a verificação.
Exiba a mensagem com console.log.

Objetivo:

Aprender a utilizar estruturas condicionais (if, else if, else) para controlar o fluxo do código com base em diferentes condições.

TERNÁRIOS CONDICIONAIS

O operador ternário, ou operador condicional, é uma forma concisa de escrever uma instrução if-else em uma única linha. Ele avalia uma expressão e retorna um dos dois valores com base no resultado.

O operador ternário possui três partes:

- **Condição:**
 - A expressão a ser avaliada como verdadeira ou falsa.
- **ExpressãoSeVerdadeira:**
 - O valor a ser retornado se a condição for verdadeira.
- **ExpressãoSeFalsa:**
 - O valor a ser retornado se a condição for falsa.

Vantagens do Operador Ternário:

Conciso e Legível:

Permite escrever de forma compacta, melhorando a legibilidade do código para condições simples.

Ideal para Atribuições Simples:

Útil para atribuir valores a variáveis com base em uma condição.

Facilita Expressões Inline:

Pode ser usado diretamente em retornos de funções ou dentro de templates.

COMO UTILIZAR TERNÁRIOS CONDICIONAIS

Sintaxe:

```
1 let resultado = condição ? valorSeVerdadeiro : valorSeFalso;
```

Exemplo:

```
1 let idade = 18;
2 let status = (idade >= 18) ? 'maior de idade' : 'menor de idade';
3 console.log(status); // Output: maior de idade
```

Explicação:

- A condição (`idade >= 18`) é avaliada.
- Se `idade` for maior ou igual a 18, `status` recebe o valor '`maior de idade`'.
- Se `idade` for menor que 18, `status` recebe o valor '`menor de idade`'.

Comparação com if-else:

```
1 let idade = 18;
2 let status;
3 if (idade >= 18) {
4     status = 'maior de idade';
5 } else {
6     status = 'menor de idade';
7 }
8 console.log(status); // Output: maior de idade
```



SWITCH

A estrutura **switch** avalia uma expressão e executa o bloco de código correspondente ao seu valor. É útil para verificar múltiplos valores de uma variável ou expressão de forma clara e organizada.

Componentes da Estrutura **switch**:

Expressão:

A expressão ou variável que será avaliada.

Casos (case):

Blocos de código são executados se a expressão corresponder ao valor especificado

Cada caso termina com **break** para evitar a execução dos casos seguintes.

Padrão (default):

O bloco de código default é executado se a expressão não corresponder a nenhum dos casos. É opcional, mas recomendado para tratar valores inesperados.

Vantagens da Estrutura **switch**:

Organização:

Organiza e torna o código mais legível ao verificar várias condições em uma única variável ou expressão.

Clareza:

Facilita a compreensão do fluxo lógico em comparação a uma longa cadeia de **if-else if**

Eficiência:

Pode ser mais eficiente, pois o **switch** pode ser otimizado pelo compilador ou interpretador.



QUANDO USAR SWITCH EM VEZ DE IF-ELSE

Múltiplos Valores de Comparação:

- Use switch quando precisar verificar uma variável ou expressão contra muitos valores possíveis.

Melhor Legibilidade:

- switch é preferível quando a estrutura if-else se torna extensa e difícil de ler.

A sintaxe básica do switch é:

```
1 switch (expressao) {  
2     case valor1:  
3         // código a ser executado se expressao for igual a valor1  
4         break;  
5     case valor2:  
6         // código a ser executado se expressao for igual a valor2  
7         break;  
8     // outros casos  
9     default:  
10        // código a ser executado se expressao não corresponder a nenhum caso  
11 }
```

Exemplo de Uso:

```
1 let diaDaSemana = 3;  
2 let nomeDoDia;  
3  
4 switch (diaDaSemana) {  
5     case 1:  
6         nomeDoDia = 'Domingo';  
7         break;  
8     case 2:  
9         nomeDoDia = 'Segunda-feira';  
10        break;  
11     case 3:  
12         nomeDoDia = 'Terça-feira';  
13         break;  
14     case 4:  
15         nomeDoDia = 'Quarta-feira';  
16         break;  
17     case 5:  
18         nomeDoDia = 'Quinta-feira';  
19         break;  
20     case 6:  
21         nomeDoDia = 'Sexta-feira';  
22         break;  
23     case 7:  
24         nomeDoDia = 'Sábado';  
25         break;  
26     default:  
27         nomeDoDia = 'Dia inválido';  
28 }  
29  
30 console.log(nomeDoDia); // Output: Terça-feira
```



ATIVIDADE PRÁTICA

Atividade 04

Crie um código JavaScript que solicite ao usuário dois números e uma operação matemática, utilize switch para realizar a operação e exibir o resultado.

Solicite os números com prompt e armazene em numero1 e numero2.

Solicite a operação e armazene em operacao.

Utilize switch para realizar a operação escolhida e exiba o resultado com console.log.

Objetivo:

Aprender a utilizar a estrutura switch para criar uma calculadora simples que realiza operações básicas (adição, subtração, multiplicação, divisão) com base na escolha do usuário

SE LIGA NO CONTEÚDO DA PRÓXIMA AULA!

AULA 03 DE JAVASCRIPT.
LAÇOS DE REPETIÇÃO I.



INFINITY SCHOOL
VISUAL ART CREATIVE CENTER

Aula 03 - Laço de repetição `for`

ESTRUTURAS DE REPETIÇÃO

FOR

BREAK E CONTINUE



IN

INFINITY SCHOOL

VISUAL ART CREATIVE CENTER



AULA 02 - CONDICIONAIS EM JAVASCRIPT