

A woman with long dark hair and glasses is shown from the chest up, looking down at a laptop screen. The background is dark and slightly blurred.

IN

INFINITY SCHOOL

VISUAL ART CREATIVE CENTER

AULA 02 - CONDICIONAIS EM JAVASCRIPT

O QUE IREMOS APRENDER

- 01** OPERADORES LÓGICOS
- 02** AND, OR, NOT
- 03** OPERADORES DE COMPARAÇÃO
- 04** CONDICIONAIS
- 05** IF, ELSE IF, ELSE
- 06** TERNÁRIOS CONDICIONAIS
- 07** SWITCH
- 08** MÃOS NO CÓDIGO

OPERADORES LÓGICOS

São usados para realizar operações lógicas. Elas podem ser do tipo AND, OR e NOT. Os operandos devem ser lógicos, verdadeiro ou falso. Também podem operar sobre expressões lógicas, ou seja, que retornem valores verdadeiro ou falso.

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

OPERADOR AND (&&)

O operador AND (&&) recebe dois operandos e retorna verdadeiro se, e somente se ambos os operandos sejam verdadeiros. Retorna falso, caso contrário.

```
let num1 = 1
let num2 = 2
console.log(num1 == 1 && num2 == 10)
```

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

TABELA VERDADE DO OPERADOR AND

OPERADOR OR (||)

Para o operador OR (||) retornar verdadeiro, basta que um dos operandos seja verdadeiro. Ele também retorna verdadeiro caso os dois operandos sejam verdadeiros. Retorna falso, se os dois forem falsos.

```
let num1 = 1
let num2 = 2
console.log(num1 == 1 || num2 == 10)
```

A	B	A ou B
TRUE	TRUE	TRUE
TRUE	FALSE	TRUE
FALSE	TRUE	TRUE
FALSE	FALSE	FALSE

TABELA VERDADE DO OPERADOR OR

OPERADOR NOT (!)

O operador de negação (!) é um operando unário, isto é, opera sobre apenas um operando. Ele nega, inverte o valor lógico do operando.

```
let num1 = 1
let num2 = 2
console.log(!num1 == 1)
```

A	não A
TRUE	FALSE
FALSE	TRUE

TABELA VERDADE DO OPERADOR NOT

OPERADOR DE COMPARAÇÃO

Você tem dois valores numéricos e quer comparar ambos para ver qual deles é maior. Para isso, o JavaScript nos traz os **operadores de comparação**, eles são meios que a linguagem nos traz para comparar valores. São eles:

`==`: Igualdade abstrata – Verifica se um valor é igual ao outro.

`==`: Igualdade estrita – Verifica se o valor e tipo são iguais.

`!=`: Diferença abstrata – Verifica se um valor é diferente de outro.

`!=`: Diferença estrita – Verifica se o valor ou tipo são diferentes.

`>`: Maior que – Verifica se um valor é maior que outro.

`<`: Menor que – Verifica se um valor é menor que outro.

`>=`: Maior igual – Verifica se um valor é maior ou igual ao outro.

`<=`: Menor igual – Verifica se um valor é menor ou igual ao outro.



OPERADOR DE COMPARAÇÃO - IGUALDADE

Igualdade Abstrata: Verifica se um valor é igual ao outro.

```
let num1 = 1
let num_um = '1'

console.log(num1 == num_um)
/* Irá retornar true pois a igualdade
   abstrata verifica apenas os valores. */
```

Igualdade Estrita: Verifica se um valor e o tipo são iguais.

```
let num1 = 1
let num_um = '1'

console.log(num1 === num_um)
/* Irá retornar false pois a igualdade
   estrita verifica os valores e o tipo. */
```

OPERADOR DE COMPARAÇÃO - DIFERENÇA

Diferença Abstrata: Verifica se um valor é diferente ao outro.

```
let int_10 = 10
let str_10 = '10'

console.log(int_10 != str_10)
/* Irá retornar 'false', pois a
igualdade abstrata verifica os valores. */
```

Diferença Estrita: Verifica se um valor e o tipo são diferentes.

```
let int_10 = 10
let str_10 = '10'

console.log(int_10 !== str_10)
/* Irá retornar 'true', pois a igualdade
estrita verifica os valores e o tipo. */
```

OPERADOR DE COMPARAÇÃO - MAIOR

Maior que: Verifica se um valor é maior que outro.

```
let num1 = 10
let num2 = 10

console.log(num1 > num2)
/* Irá retornar 'false', pois 10 não é maior que 10 */
```

Maior igual: Verifica se um valor é maior ou igual ao outro.

```
let num1 = 10
let num2 = 10

console.log(num1 >= num2)
/* Irá retornar 'true', pois 10 não é maior que 10, mas eles são iguais. */
```

OPERADOR DE COMPARAÇÃO - MENOR

Menor que: Verifica se um valor é menor que outro.

```
let num1 = 10
let num2 = 10

console.log(num1 < num2)
/* Irá retornar 'false', pois 10 não é menor que 10. */
```

Menor igual: Verifica se um valor é menor ou igual ao outro.

```
let num1 = 10
let num2 = 10

console.log(num1 <= num2)
/* Irá retornar 'true', pois 10 não é menor que 10, mas eles são iguais. */
```

OPERADOR DE COMPARAÇÃO - IGUALDADE

Esses operadores sempre vão nos retornar um valor booleano, ou seja, verdadeiro ou falso, a depender do resultado da comparação.

Como no exemplo:

```
<script>
    //Declaração de variáveis
    let numero1=10//variável tipo inteira
    let valor1= '10'//variável tipo string

    //saída será true pois numero1 e valor1 possuem o mesmo valor
    console.log(numero1==valor1)
    //saída será false pois numero1 e valor1 possuem o mesmo valor mas não o mesmo tipo
    console.log(numero1===valor1)
    //saída será false pois numero1 e valor1 possuem o mesmo valor
    console.log(numero1!=valor1)
    //saída será true pois numero1 e valor1 possuem o mesmo valor mas não o mesmo tipo
    console.log(numero1!==valor1)
    //saída será false pois numero1 e valor1 possuem o mesmo valor
    console.log(numero1>valor1)
    //saída será false pois numero1 e valor1 possuem o mesmo valor
    console.log(numero1<valor1)
    //saída será true pois numero1 e valor1 possuem o mesmo valor
    console.log(numero1>=valor1)
    //saída será true pois numero1 e valor1 possuem o mesmo valor
    console.log(numero1<=valor1)
</script>
```

CONDICIONAIS

As estruturas condicionais estão ligadas à tomada de decisão de um algoritmo. Ao utilizar expressões que retornam verdadeiro ou falso, o algoritmo executa o bloco de comandos relativos a este resultado.

Vamos aprender a utilizar as estruturas condicionais no JavaScript, começando pelo uso do if(se), else if e else(senão).



CONDICIONAIS

Exemplo: Tomar uma decisão sobre se deve levar ou não um guarda-chuva.

A decisão é tomada com base nessas condições. Se estiver chovendo, a recomendação é levar um guarda-chuva. Caso contrário, se não estiver chovendo, a decisão é que não é necessário levar um guarda chuva.

Esse é um exemplo simples que reflete uma decisão com base em condições específicas, algo que fazemos rotineiramente em situações do dia a dia.

CONDICIONAIS - IF (SE)

if: A estrutura if é a forma mais básica de uma estrutura condicional. Ela avalia uma condição e executa um bloco de código se a condição for verdadeira (ou seja, seu valor é true).

```
let a = 10
let b = 10

if (a == b){
  console.log('os valores são iguais')
}
```

```
let idade = 16

if ( idade >= 16 && idade <= 65 ){
  console.log('Obrigatório votar')
}
```

CONDICIONAIS - ELSE IF (SENÃO SE)

else if: A estrutura else if é usada quando você quer verificar mais de uma condição, além da condição inicial do if. Se a condição do if não for verdadeira, o código verificará a condição do else if subsequente e, se essa condição for verdadeira, executará o bloco de código associado.

```
let media_escolar = 5.9

if (media_escolar >= 6){
    console.log('Aprovado')
}
else if (media_escolar < 6 && media_escolar >=4 ){
    console.log('Recuperação')
}
```

CONDICIONAIS - ELSE (SENÃO)

else: A estrutura else é usada junto com o if e/ou else if. Se todas as condições anteriores forem falsas, o bloco de código associado ao else será executado.

```
let media_escolar = 3

if (media_escolar >= 6){
    console.log('Aprovado')
}
else if (media_escolar < 6 && media_escolar >=4 ){
    console.log('Recuperação')
}
else{
    console.log('Reprovado')
}
```

```
let idade = 9

if ( idade >= 16 && idade <= 65 ){
    console.log('Obrigatório votar')
}
else if ( idade > 65){
    console.log('Não é obrigatório votar')
}
else {
    console.log('Não vota')
}
```

TERNÁRIOS CONDICIONAIS

Os ternários condicionais facilitam o código, tornando o código mais curto, frequentemente usada como um atalho para a instrução if.

Uma diferença crucial em relação ao if é que o ternário é um operador e não uma declaração tipo função ou variável. Ele sempre resulta em um valor que é retornado conforme sua lógica.

TERNÁRIOS CONDICIONAIS

condicao ? valorSeVerdadeiro : valorSeFalso;

condicao: Uma expressão que será avaliada. Se o resultado for verdadeiro (ou seja, seu valor é true), o valorSeVerdadeiro será retornado; caso contrário, o valorSeFalso será retornado.

valorSeVerdadeiro: O valor retornado caso a condição seja verdadeira.

valorSeFalso: O valor retornado caso a condição seja falsa.

TERNÁRIOS CONDICIONAIS

A vantagem do ternário para atribuição de valor é clara: não é preciso repetir o nome da variável. Por exemplo, nesta atribuição com if usamos 2 linhas (incluindo a declaração da variável)

```
// Primeira forma:  
let idade = 18  
idade >= 18 ? console.log('Maior de idade') : console.log('Menor de idade')  
  
//Segunda forma:  
let idade = 18  
let mensagem = idade >= 18 ? 'Você é maior de idade.' : 'Você é menor de idade.'  
console.log(mensagem)
```

SWITCH

O switch é uma outra estrutura condicional em JavaScript que permite avaliar o valor de uma expressão e executar um bloco de código correspondente ao valor dessa expressão. É especialmente útil quando você tem várias condições para verificar e deseja evitar várias declarações if, else if e else

```
let idade = 18

switch (idade){
  case 20:
    console.log('20')
    break
  case 18:
    console.log('18')
    break
  default:
    console.log('nenhuma das alternativas')
}
```

TERNÁRIOS CONDICIONAIS

```
let nome= 'raama'

switch(nome){//A expressão cujo valor será comparado com os casos.
    case 'joao':
        /*Case: O uso de condicionais permite que um bloco de código seja
         executado se a expressão for igual a um valor específico.*/
        console.log("o nome é João")
        break;
        /*O comando "break" é utilizado para sair do bloco switch após
         a execução do código correspondente a um caso.*/
    case 'raama':
        console.log("o nome é Raama")
        break;
    case 'jose':
        console.log("o nome é José")
        break;
    default:
        //É opcional e representa o caso em que nenhum dos valores dos casos corresponde à expressão.
        console.log("Nenhum dos nomes")
}
```



ATIVIDADE PRÁTICA

Atividade 01

Peça ao usuário para inserir um número e escreva um programa que determine se o número é positivo e par.

Atividade 02

Crie um programa que solicite ao usuário seu peso (em kg) e altura (em metros) e calcule o IMC. Em seguida, informe a categoria de acordo com a tabela de IMC (por exemplo, abaixo do peso, peso normal, sobrepeso, etc.).

ATIVIDADE PRÁTICA

Atividade 03

Peça ao usuário para inserir o preço de um produto e sua idade. E calcule o preço final com desconto, Se o usuário tiver menos de 18 anos, aplique um desconto de 10%. Se o usuário tiver 18 anos ou mais, não aplique desconto.

Atividade 04

Peça ao usuário para inserir os comprimentos dos lados de um triângulo e escreva um programa que determine se o triângulo é equilátero, isósceles ou escaleno.

ATIVIDADE PRÁTICA

Atividade 05

Crie um programa que solicite um número de 1 a 7, representando o dia da semana. Use uma estrutura switch (ou equivalente) para imprimir o nome do dia correspondente.

Atividade 06

Escreva um programa que solicite um número e determine se ele é um número primo.

DESAFIO PRÁTICO

Crie um programa que apresente um menu com várias opções de cálculos geométricos. O usuário deve ser capaz de escolher uma opção e inserir os valores necessários para realizar o cálculo correspondente. Utilize condicionais para direcionar o programa com base na escolha do usuário.

Opções do Menu:

1. Calcular Área do Triângulo
2. Calcular Área do Retângulo
3. Calcular Volume do Cubo
4. Calcular Área do Círculo
5. Sair

SE LIGA NO CONTEÚDO DA PRÓXIMA AULA!

AULA 03 DE JAVASCRIPT.
LAÇOS DE REPETIÇÃO I.



INFINITY SCHOOL
VISUAL ART CREATIVE CENTER

Laço de Repetição

Um Laço de Repetição, ou loop, é uma estrutura em programação que repete uma sequência de instruções até que uma condição específica seja atendida.

É um recurso que a linguagem de programação fornece para executar as mesmas instruções uma determinada quantidade de vezes.

O JavaScript nos fornece diferentes estruturas de repetição, nesta aula vamos abordar sobre a estrutura WHILE



Laço de Repetição

Exemplos de **laço de repetição While** em JavaScript:

```
let contador = 0 // Variável inicializada com o valor 0

while (contador < 10) {
    // O laço se repetirá enquanto o contador for menor do que 10
    console.log(contador)
    contador++ // A variável contador está sendo atualizada a cada laço
}
```

A woman with long, dark hair and glasses is shown from the chest up, looking down at a laptop screen. The background is dark and slightly blurred.

IN

INFINITY SCHOOL

VISUAL ART CREATIVE CENTER

AULA 02 - CONDICIONAIS