Operadores em JavaScript: Um Guia Prático

No mundo da programação, especialmente com JavaScript, os **operadores** são a espinha dorsal da lógica. Eles nos permitem realizar cálculos, comparar valores e tomar decisões. Neste tutorial, vamos explorar os três tipos mais fundamentais: **aritméticos**, **relacionais** e **condicionais**, usando **exemplos práticos** do dia a dia de uma loja de eletrônicos.

---

1. Operadores Aritméticos: Os Calculadores do JavaScript

Os operadores aritméticos são como a sua calculadora embutida no código. Eles realizam operações matemáticas básicas.

Operador	Descrição	Exemplo	Resultado
`+`	Adição	`10 + 5`	`15`
`-`	Subtração	`10 - 5`	`5`
`*`	Multiplicação	`10 * 5`	`50`
`/`	Divisão	`10 / 5`	`2`
`%`	Módulo (Resto)	`10 % 3`	`1`
`**`	Exponenciação	`2 ** 3`	`8`
`++`	Incremento	`let x = 5; x++`	`6`
`--`	Decremento	`let y = 5; y--`	`4`

Exemplo Prático: Cálculo de Impostos de Importação

Um novo lote de 50 smartwatches ($150.00 cada) chegou, e o imposto de importação é de 20% sobre o valor total dos produtos.

```
const precoUnitarioSmartwatch = 150.00;
const quantidadeSmartwatches = 50;
const taxaImposto = 0.20; // 20%

// Calcula o valor total dos produtos e depois o imposto
const valorTotalProdutos = precoUnitarioSmartwatch * quantidadeSmartwatches;
const valorImposto = valorTotalProdutos * taxaImposto;
const custoTotalComImposto = valorTotalProdutos + valorImposto;

console.log(`Custo total dos smartwatches (com imposto): $${custoTotalComImposto.toFixed(2)}`); // Saída: Custo total dos smartwatches (com imposto): $9000.00
```

---

2. Operadores Relacionais: Os Comparadores do JavaScript

Operadores relacionais são usados para comparar dois valores e sempre retornam um valor **booleano**: `true` (verdadeiro) ou `false` (falso).

Operador	Descrição	Exemplo	Resultado
==	Igual a (compara valor)	`5 == '5'`	`true`
===	Estritamente igual a (compara valor e tipo)	`5 === '5'`	`false`
!=	Diferente de (compara valor)	`5 != 10`	`true`
!==	Estritamente diferente de (compara valor e tipo)	`5 !== '5'`	`true`
>	Maior que	`10 > 5`	`true`
<	Menor que	`10 < 5`	`false`
>=	Maior ou igual a	`10 >= 10`	`true`
<=	Menor ou igual a	`10 <= 5`	`false`

Exemplo Prático: Verificação de Meta de Vendas

Um vendedor tem uma meta de vendas de $5000 para o mês. Ele já vendeu $4800. Ele atingiu ou superou a meta?

```
const vendasAtuaisVendedor = 4800;
const metaVendas = 5000;

// Usamos >= para verificar se as vendas atuais são maiores ou iguais à meta
const metaAtingida = vendasAtuaisVendedor >= metaVendas;

console.log(`Vendedor atingiu a meta: ${metaAtingida}`); // Saída: Vendedor atingiu a meta: false
```

---

3. Operadores Condicionais: Tomando Decisões no Código

Operadores condicionais, muitas vezes combinados com operadores relacionais, nos permitem executar diferentes blocos de código com base em certas condições.

3.1. `if`, `else if` e `else`: A Estrutura Básica de Decisão

Essa é a forma mais comum de controlar o fluxo do seu programa.

```
if (condicao1) {
    // Executa se a condicao1 for verdadeira
} else if (condicao2) {
    // Executa se a condicao1 for falsa E a condicao2 for verdadeira
} else {
    // Executa se todas as condições anteriores forem falsas
}
```

Exemplo Prático: Status de Devolução de Produto

Um cliente deseja devolver um produto. Ele pode devolver se o produto estiver com a caixa original E em até 30 dias após a compra.

```
const diasDesdeCompra = 25; // Cliente comprou há 25 dias
const temCaixaOriginal = true; // Cliente tem a caixa original
let statusDevolucao;

if (diasDesdeCompra <= 30 && temCaixaOriginal) {
    statusDevolucao = "Devolução Aceita";
} else {
    statusDevolucao = "Devolução Negada";
}

console.log(`Status da Devolução: ${statusDevolucao}`); // Saída: Status
da Devolução: Devolução Aceita
```

Exemplo Avançado: Avaliação de Preço de um Produto

Classifique o preço de um fone de ouvido: "Muito Caro" (se >300), "Preço Justo" (se entre 100 e 300, inclusive), ou "Barato" (se <100).

```
const precoFoneOuvido = 180; // Pode testar com 350, 50, etc.
let classificacaoPreco;

if (precoFoneOuvido > 300) {
    classificacaoPreco = "Muito Caro";
} else if (precoFoneOuvido >= 100 && precoFoneOuvido <= 300) {
    classificacaoPreco = "Preço Justo";
} else {
    classificacaoPreco = "Barato";
}

console.log(`Classificação do Preço: ${classificacaoPreco}`); // Saída:
Classificação do Preço: Preço Justo
```

3.2. Operadores Lógicos (`&&`, `||`, `!`)

Esses operadores combinam ou negam expressões booleanas.

Operador	Descrição	Exemplo	Resultado
`&&`	**AND lógico**: Verdadeiro se ambos forem `true`	`(5 > 3) && (10 < 20)`	`true`
`	`	`	**OR lógico**: Verdadeiro se um OU ambos forem `true`	`(5 > 10)
`!`	**NOT lógico**: Inverte o valor booleano	`!(5 > 3)`	`false`

Exemplo Prático: Critério para Frete Grátis

O frete é grátis se a `compraOnline` for `true` E o `valorCompra` for maior ou igual a $200.

```
const compraOnline = true;
const valorCompra = 210;

if (compraOnline && valorCompra >= 200) {
    console.log("Parabéns! Você ganhou frete grátis!");
} else {
    console.log("Para ter frete grátis, compre online e seu pedido deve
ser de pelo menos $200.");
} // Saída: Parabéns! Você ganhou frete grátis!
```

Exemplo Prático: Requisitos para Vaga de Emprego

Um candidato a um cargo de suporte técnico precisa ter `experienciaEmTI` OU `certificacaoTecnica`.

```
const experienciaEmTI = false;
const certificacaoTecnica = true;

if (experienciaEmTI || certificacaoTecnica) {
    console.log("Candidato apto para entrevista.");
} else {
    console.log("Candidato não atende aos requisitos mínimos.");
} // Saída: Candidato apto para entrevista.
```

3.3. `switch`: Uma Alternativa para Múltiplas Condições

O `switch` é útil quando você tem uma variável que pode ter vários valores específicos e você quer executar um código diferente para cada valor.

JavaScript

```
switch (expressao) {
    case valor1:
        // Código a ser executado se expressao for igual a valor1
        break;
    case valor2:
        // Código a ser executado se expressao for igual a valor2
        break;
    default:
        // Código a ser executado se expressao não corresponder a nenhum
caso
}
```

Exemplo Prático: Mensagens de Boas-Vindas por Tipo de Cliente

Exiba uma mensagem personalizada com base no `tipoCliente`: "Gold", "Silver", "Bronze" ou "Padrão".

```
const tipoCliente = "Gold"; // Pode testar com "Silver", "Bronze", "Novo"

switch (tipoCliente) {
    case "Gold":
        console.log("Bem-vindo(a), cliente Gold! Aproveite seus
benefícios exclusivos.");
        break;
    case "Silver":
        console.log("Olá, cliente Silver! Temos ofertas especiais para
você.");
        break;
    case "Bronze":
        console.log("Seja bem-vindo(a), cliente Bronze!");
        break;
    default:
        console.log("Bem-vindo(a) à nossa loja!");
}

// Saída: Bem-vindo(a), cliente Gold! Aproveite seus benefícios
exclusivos.
```

3.4. Operador Condicional Ternário (`? :`)

Este é um atalho conciso para uma instrução `if/else` simples.

```
condicao ? expressaoSeVerdadeiro : expressaoSeFalso;
```

Novo Exemplo Prático: Mensagem de Entrega Agendada

Se o `pagamentoAprovado` for `true`, a mensagem deve ser "Entrega agendada". Caso contrário, "Aguardando pagamento".

```
const pagamentoAprovado = false; // Pode testar com true ou false

const mensagemEntrega = pagamentoAprovado ? "Entrega agendada" :
"Aguardando pagamento";

console.log(`Status: ${mensagemEntrega}`); // Saída: Status: Aguardando
pagamento
```