



Condicionais no Javascript

Condicionais no Javascript

Condicionais são instruções em um programa que permitem executar diferentes ações com base em certas condições. Em JavaScript, usamos principalmente a estrutura `if...else` para criar condicionais.

Condicionais no Javascript



```
1      if (condição) {  
2      // código a ser executado se a condição for verdadeira  
3  } else {  
4      // código a ser executado se a condição for falsa  
5  }
```

Se a condição for verdadeira, o bloco de código dentro do if será executado. Se a condição for falsa, o bloco de código dentro do else será executado!

Condicionais no Javascript

Mas e se tivermos mais de 2 condições? Nesse caso, usamos o else if.

O primeiro é sempre o if, o segundo sempre else if (senão se) e o último sempre else. Você pode adicionar quantos else if's quiser!

Condicionais no Javascript



```
1  if (condição) {  
2    // código a ser executado se a condição for verdadeira  
3  } else if (condição_nova) {  
4    // código a ser executado se a primeira condição for  
5    //falsa e a condição nova, verdadeira  
6  }else{  
7    //código a ser executado caso todas as outras condições  
8    //sejam falsas  
9  }
```


Operadores de Comparação

Mas como o computador sabe quando é verdadeiro ou falso? Através das condições. As condições são perguntas que fazemos ao computador, e ele nos retorna verdadeiro (True) ou falso (False). Para isso, precisamos dos operadores de comparação, para escrever nossas condições!

Operadores de Comparação

Os operadores de comparação são usados para comparar dois valores e retornar um resultado verdadeiro ou falso (booleano). Aqui estão alguns dos operadores de comparação mais comuns:

- `==`: Igual a



```
1 let num1 = 5
2 let num2 = 6
3 console.log(num1==num2)
4 //resultado: False. 5 não é igual a 6
```

Operadores de Comparação

- **!=: Diferente de**

```
1 let num1 = 5
2 let num2 = 6
3 console.log(num1!=num2)
4 //resultado: True. 5 realmente é diferente
5 // de 6
```

- **>: Maior que**

```
1 let num1 = 5
2 let num2 = 6
3 console.log(num1>num2)
4 //resultado: False. 5 não é maior
5 // do que 6
```


Operadores de Comparação

- **<: Menor que**

```
1 let num1 = 5
2 let num2 = 6
3 console.log(num1<num2)
4 //resultado: True. 5 é menor
5 // do que 6
```

- **>=: Maior ou igual a. No maior ou igual, são duas perguntas, vai dar True se for maior, ou se for igual!**

Operadores de Comparação


- `>=`: Maior ou igual a. No maior ou igual, são duas perguntas, vai dar `True` se for maior, ou se for igual!



```
1 let num1 = 5
2 let num2 = 6
3 console.log(num1>=num2)
4 //resultado: False. 5 não é
5 // nem igual a 6
```

Operadores de Comparação

- **<=: Menor ou igual a**



```
1  let num1 = 5
2  let num2 = 6
3  console.log(num1<=num2)
4  //resultado: True. 5 não é igual,
5  // mas é menor do que 6
```

- **Temos um operador de comparação um pouco diferente, ele compara o valor e o tipo da informação!**

Operadores de Comparação

- **===: Estritamente igual a (verifica igualdade de valor e tipo)**



```
1 let num1 = 5    //valor 5, tipo de dado: number
2 let num2 = "5"  //valor 5, tipo de dado: string
3
4 console.log(num1==num2)
5 //resultado: True. Dois iguais compara apenas o valor
6
7 console.log(num1===num2)
8 //resultado: False. Três iguais compara o valor e o tipo
9 //do dado. O valor é igual, mas o tipo de dado é diferente!
```

Operadores de Comparação

Enquanto dois iguais (==) compara apenas o valor, três iguais (===) compara o valor e o tipo do dado. Isso é muito interessante na hora de criar condicionais, e principalmente fazer cálculos.

Mas como assim? Vamos descobrir!

Operadores de Comparação

E se quisermos somar `num1 + num2`? Qual seria o resultado?



```
1 let num1 = 5    //valor 5, tipo de dado: number
2 let num2 = "5"  //valor 5, tipo de dado: string
3
4 console.log(num1+num2)
5 //resultado: 55
```

Mas por que deu 55? é como se o "5" fosse entendido como "cinco" por debaixo dos panos, e a conta real feita foi 5 + cinco. Para não quebrar o código, o Javascript concatena (cola um no outro) e devolve a resposta errada.

Operadores de Comparação



```
1 let num1 = 5    //valor 5, tipo de dado: number
2 let num2 = "5"  //valor 5, tipo de dado: string
3
4 console.log(num1+num2)
5 //resultado: 55
```

O erro era justamente no tipo de informação que foi passada, então os três iguais ajudam muito a evitar esse tipo de erro!

Temos mais um tipo de operador, que é o da diferença estrita!

Operadores de Comparação

- `!==`: Estritamente diferente de: Testa se tanto o valor quanto o tipo são diferentes!



```
1 let num1 = 5    //valor 5, tipo de dado: number
2 let num2 = "5"  //valor 5, tipo de dado: string
3
4 console.log(num1!==num2)
5 //resultado: true
```

Isso retorna `true`, indicando que 5 não é estritamente igual a `'5'`. Isso ocorre porque, apesar de ambos os valores serem 5, o primeiro é um número e o segundo é uma string.

Testando conhecimentos

1. Escreva um programa que verifique se um número é positivo, negativo ou zero e exiba uma mensagem correspondente.
2. Escreva um programa que determine se um aluno foi aprovado ou reprovado com base em sua nota (suponha que a nota de corte seja 7). Peça a nota ao usuário!


Operadores aritméticos

Os operadores aritméticos são usados para realizar operações matemáticas. Alguns dos operadores aritméticos em JavaScript incluem:

- **+** : Adição
- **-** : Subtração
- ***** : Multiplicação
- **/** : Divisão
- **%** : Módulo (retorna o resto da divisão)


Operadores aritméticos

Adição:



```
1 let num1 = 5
2 let num2 = 5
3
4 console.log(num1+num2)
5 //resultado: 10
```

Subtração:



```
1 let num1 = 15
2 let num2 = 5
3
4 console.log(num1-num2)
5 //resultado: 10
```

Operadores aritméticos

Multiplicação:



```
1 let num1 = 5
2 let num2 = 5
3
4 console.log(num1*num2)
5 //resultado: 25
```

Divisão:



```
1 let num1 = 15
2 let num2 = 5
3
4 console.log(num1/num2)
5 //resultado: 3
```

Operadores aritméticos

Módulo:



```
1  let num1 = 10
2  let num2 = 2
3
4  console.log(num1%num2)
5  //resultado: 0
```

Testando conhecimentos

Escreva um programa que calcule a média de três notas de um aluno e exiba se ele foi aprovado (média maior ou igual a 7) ou reprovado.

Escreva um programa que determine se um número é par ou ímpar e exiba uma mensagem correspondente.

Operadores Lógicos

Os operadores lógicos em JavaScript são usados para combinar ou manipular valores booleanos (verdadeiro ou falso). Existem três operadores lógicos principais em JavaScript: `&&` (E lógico), `||` (OU lógico) e `!` (NÃO lógico).

Operador `&&` (E lógico)

O operador `&&` (E lógico) retorna verdadeiro se todas as expressões combinadas forem verdadeiras, caso contrário, retorna falso.

Operadores Lógicos



```
1  let idade = 25;
2  let possuiCarteiraDeMotorista = true;
3
4  if (idade >= 18 && possuiCarteiraDeMotorista) {
5      console.log("Pode dirigir.");
6  } else {
7      console.log("Não pode dirigir.");
8  }
```


Operadores Lógicos

Neste exemplo, a expressão `idade >= 18 && possuiCarteiraDeMotorista` só será verdadeira (`true`) se a idade for maior ou igual a 18 E a variável `possuiCarteiraDeMotorista` for verdadeira (`true`). Se ambas as condições forem atendidas, a mensagem "Pode dirigir." será exibida.

Operadores Lógicos

Operador `||` (OU lógico)

O operador `||` (OU lógico) retorna verdadeiro se pelo menos uma das expressões combinadas for verdadeira.



```
1 let diaDaSemana = "sábado";  
2  
3 if (diaDaSemana === "sábado" || diaDaSemana === "domingo") {  
4     console.log("É fim de semana!");  
5 } else {  
6     console.log("É dia de semana.");  
7 }
```

Operadores Lógicos

Neste exemplo, a expressão `diaDaSemana === "sábado" || diaDaSemana === "domingo"` será verdadeira (`true`) se `diaDaSemana` for igual a `"sábado"` OU igual a `"domingo"`. Se uma dessas condições for atendida, a mensagem `"É fim de semana!"` será exibida.

Operadores Lógicos

Operador ! (NÃO lógico)

O operador ! (NÃO lógico) inverte o valor de uma expressão booleana. Ou seja, se a expressão for verdadeira, o operador ! a tornará falsa, e se a expressão for falsa, o operador ! a tornará verdadeira.

Operadores Lógicos



```
1  let chovendo = false;
2
3  if (!chovendo) {
4      console.log("Não esqueça o guarda-chuva!");
5  } else {
6      console.log("Pode sair sem guarda-chuva.");
7  }
```

Operadores Lógicos

Neste exemplo, a expressão `!chovendo` inverte o valor da variável `chovendo`, ou seja, se `chovendo` for falso (`false`), `!chovendo` será verdadeiro (`true`), e vice-versa. Assim, a mensagem "Não esqueça o guarda-chuva!" será exibida se `chovendo` for falso.

Testando conhecimentos

Escreva um programa que pergunte a idade do usuário e verifique se ele pode entrar em um clube noturno. Se o usuário tiver pelo menos 18 anos, exiba a mensagem "Pode entrar"; caso contrário, exiba a mensagem "Não pode entrar".

Escreva um programa que verifique se um usuário pode entrar em uma festa com base em sua idade e se está acompanhado por um adulto. Se o usuário tiver 18 anos ou mais, ou se estiver acompanhado por um adulto, exiba a mensagem "Pode entrar"; caso contrário, exiba a mensagem "Não pode entrar".

Testando conhecimentos

Escreva um programa que determine se um cliente tem direito a um desconto em uma compra com base no valor total da compra e se é um cliente premium. Se o valor da compra for maior que \$100 e o cliente for premium, exiba a mensagem "Desconto de 20% aplicado"; caso contrário, exiba a mensagem "Sem desconto aplicado".

Escreva um programa que determine se uma loja está aberta com base no dia da semana e no horário. A loja abre de segunda a sexta-feira das 9h às 18h. Se for segunda a sexta-feira e estiver dentro do horário de funcionamento, exiba a mensagem "Loja aberta"; caso contrário, exiba a mensagem "Loja fechada".

Segue lá nas redes sociais

No instagram, tem um post novo todo dia, premiações e desafios, além de lives que você não vai encontrar em mais nenhum lugar!

Já no LinkedIn, tem mais conteúdo voltado para a prática, experiências incríveis e muito mais!

