

▼ C# Fundamentals


Operadores lógicos

Operadores lógicos:

Operador de negação lógica !

O operador de prefixo unário `!` computa a negação lógica de seu operando. Ou seja, ele produz `true`, se o operando for avaliado como `false`, e `false`, se o operando for avaliado como `true`:

C#

 Copiar

 Executar

```
bool passed = false;  
Console.WriteLine(!passed); // output: True  
Console.WriteLine(!true);  // output: False
```

A partir do C# 8,0, o operador de sufixo unário `!` é o operador NULL-tolerante.

Operadores lógicos:

Operador AND lógico&

O operador `&` computa o AND lógico de seus operandos. O resultado de `x & y` será `true` se ambos `x` e `y` forem avaliados como `true`. Caso contrário, o resultado será `false`.

O `&` operador avalia os dois operandos mesmo se o operando esquerdo for avaliado `false`, de modo que o resultado da operação seja `false` independente do valor do operando à direita.

No exemplo a seguir, o operando à direita do operador `&` é uma chamada de método, que é executada independentemente do valor do operando à esquerda:

```
C# Copiar Executar

bool SecondOperand()
{
    Console.WriteLine("Second operand is evaluated.");
    return true;
}

bool a = false & SecondOperand();
Console.WriteLine(a);
// Output:
// Second operand is evaluated.
// False

bool b = true & SecondOperand();
Console.WriteLine(b);
// Output:
// Second operand is evaluated.
// True
```

Operadores lógicos:

Operador OR exclusivo lógico ^

O operador `^` computa o OR exclusivo lógico, também conhecido como o XOR lógico, de seus operandos. O resultado de `x ^ y` é `true` se `x` é avaliado como `true` e `y` avaliado como `false`, ou `x` avaliado como `false` e `y` avaliado como `true`. Caso contrário, o resultado será `false`. Ou seja, para os operandos `bool`, o operador `^` computa o mesmo resultado que o [operador de desigualdade](#) `!=`.

C#

 Copiar Executar

```
Console.WriteLine(true ^ true);    // output: False
Console.WriteLine(true ^ false);   // output: True
Console.WriteLine(false ^ true);   // output: True
Console.WriteLine(false ^ false);  // output: False
```


Operadores lógicos:

Operador OR lógico |

O operador `|` computa o OR lógico de seus operandos. O resultado de `x | y` será `true` se `x` ou `y` for avaliado como `true`. Caso contrário, o resultado será `false`.

O `|` operador avalia os dois operandos mesmo se o operando esquerdo for avaliado `true`, de modo que o resultado da operação seja `true` independente do valor do operando à direita.

No exemplo a seguir, o operando à direita do operador `|` é uma chamada de método, que é executada independentemente do valor do operando à esquerda:

C#

Copiar

Executar

```
bool SecondOperand()
{
    Console.WriteLine("Second operand is evaluated.");
    return true;
}

bool a = true | SecondOperand();
Console.WriteLine(a);
// Output:
// Second operand is evaluated.
// True

bool b = false | SecondOperand();
Console.WriteLine(b);
// Output:
// Second operand is evaluated.
// True
```

Operadores lógicos:

Operador AND lógico condicional&&

O operador AND lógico condicional `&&`, também conhecido como operador AND lógico de "curto-circuito", computa o AND lógico de seus operandos. O resultado de `x && y` será `true` se ambos `x` e `y` forem avaliados como `true`. Caso contrário, o resultado será `false`. Se `x` for avaliado como `false`, `y` não será avaliado.

No exemplo a seguir, o operando à direita do operador `&&` é uma chamada de método, que não é executada se o operando à esquerda for avaliado como `false`:

```
C# Copiar Executar

bool SecondOperand()
{
    Console.WriteLine("Second operand is evaluated.");
    return true;
}

bool a = false && SecondOperand();
Console.WriteLine(a);
// Output:
// False

bool b = true && SecondOperand();
Console.WriteLine(b);
// Output:
// Second operand is evaluated.
// True
```

Operadores lógicos:

Operador OR lógico condicional ||

O operador OR lógico condicional `||`, também conhecido como operador OR lógico de "curto-circuito", computa o OR lógico de seus operandos. O resultado de `x || y` será `true` se `x` ou `y` for avaliado como `true`. Caso contrário, o resultado será `false`. Se `x` for avaliado como `true`, `y` não será avaliado.

No exemplo a seguir, o operando à direita do operador `||` é uma chamada de método, que não é executada se o operando à esquerda for avaliado como `true`:

C#

Copiar

Executar

```
bool SecondOperand()
{
    Console.WriteLine("Second operand is evaluated.");
    return true;
}

bool a = true || SecondOperand();
Console.WriteLine(a);
// Output:
// True

bool b = false || SecondOperand();
Console.WriteLine(b);
// Output:
// Second operand is evaluated.
// True
```

Operadores lógicos:

- Links de documentação oficial:
- **Documentação:**
- <https://docs.microsoft.com/pt-br/dotnet/csharp/language-reference/operators/boolean-logical-operators>



Dúvidas?