argumentos inteiros.Um pouco está definido no resultado se o bit correspondente
está definido em um ou ambos os operandos.Por exemplo, 0x1234 |
0x00FF Avalia para 0x12ff.
Bitwise xor (^)
O operador ^ realiza um exclusivo booleano ou operação em cada
um pouco de seus argumentos inteiros.Exclusivo ou significa que também
O operando um é verdadeiro ou o operando dois é verdadeiro, mas não ambos.Um pouco é
definido no resultado desta operação se um bit correspondente for definido em um (mas
não ambos) dos dois operandos.Por exemplo, 0xff00 ^ 0xf0f0
Avalia para 0x0FF0.
Bitwise não (~)
O operador ~ é um operador unário que aparece antes de seu único
operando inteiro.Opera revertendo todos os bits no operando.
Por causa da maneira como os números inteiros assinados estão representados em JavaScript,
Aplicar o operador ~ a um valor é equivalente a mudar seu sinal
e subtrair 1. Por exemplo, ~ 0x0f avalia para
0xffffff0, ou -16.
Mudança para a esquerda (<<)
O operador << move todos os bits em seu primeiro operando para a esquerda pelo
número de lugares especificados no segundo operando, que deve ser
um número inteiro entre 0 e 31. Por exemplo, na operação A <<
1, o primeiro bit (o pouco) de A se torna o segundo bit (os dois
bit), o segundo bit de a se torna o terceiro, etc. Um zero é usado para
O novo primeiro bit e o valor do 32º bit são perdidos.Mudando a
O valor deixado por uma posição é equivalente à multiplicação por 2, mudando
Duas posições são equivalentes a multiplicar por 4 e assim por diante.Para
Exemplo, 7 << 2 avalia para 28.
Mudar bem com o sinal (>>)