Uma expressão de exclusão avalia para verdadeira se a exclusão for bem -sucedida ou se
O delete não teve efeito (como excluir uma propriedade inexistente).
excluir também avalia como verdadeiro quando usado (sem sentido) com um
expressão que não é uma expressão de acesso à propriedade:
Seja o = {x: 1};// O tem propriedade própria x e herda
Propriedade ToString
Exclua O.x // => true: exclui a propriedade x
Exclua o.x // => true: não faz nada (x não existe)
Mas é verdade de qualquer maneira
excluir o.toString // => true: não faz nada (a tostragem não é
uma propriedade própria)
exclua 1 // => true: bobagem, mas verdade de qualquer maneira
Delete não remove propriedades que têm um atributo configurável
de falso.Certas propriedades de objetos embutidos não são confundíveis,
assim como as propriedades do objeto global criado pela Declaração Variável
e declaração de função.No modo rigoroso, tentando excluir um não
A propriedade configurável causa um TypeError.No modo não estrito, exclua
Simplesmente avalia o FALSE neste caso:
// No modo rigoroso, todas essas deleções jogam TypeError
Em vez de retornar falso
excluir object.prototype // => false: propriedade não é
configurável
var x = 1;// declarar uma variável global
exclua globalthis.x // => false: não é possível excluir isso
propriedade
função f () {} // declarar uma função global
Exclua globalthis.f // => false: Não é possível excluir isso
propriedade também
Ao excluir propriedades configuráveis ??do objeto global no não rito
modo, você pode omitir a referência ao objeto global e simplesmente
Siga o operador Excluir com o nome da propriedade: