Acesso à propriedade condicional com?.e?. [] é um dos mais recentes
Recursos de JavaScript.No início de 2020, esta nova sintaxe é suportada
nas versões atuais ou beta da maioria dos principais navegadores.

4.5 Expressões de invocação

Uma expressão de invocação é a sintaxe de JavaScript para chamadas (ou
executando) uma função ou método.Começa com uma expressão de função
que identifica a função a ser chamada.A expressão da função é
seguido de um parêntese aberta, uma lista separada por vírgula de zero ou
Mais expressões de argumento e um parê de parênteses estreitos.Alguns exemplos:

```
f (0) // f é a expressão da função;0 é o
expressão de argumento.
Math.max (x, y, z) // math.max é a função;x, y e z são
os argumentos.
a.sort () // a.sort é a função;Não há
argumentos.
```

Quando uma expressão de invocação é avaliada, a expressão da função é
avaliado primeiro, e depois as expressões de argumento são avaliadas para
produzir uma lista de valores de argumento.Se o valor da função
A expressão não é uma função, um TypeError é jogado.Em seguida, o argumento
Os valores são atribuídos, para os nomes de parâmetros especificados quando
A função foi definida e, em seguida, o corpo da função é executado.
Se a função usar uma declaração de retorno para retornar um valor, então isso
O valor se torna o valor da expressão de invocação.Caso contrário, o
O valor da expressão de invocação é indefinido.Detalhes completos
na invocação de funções, incluindo uma explicação do que acontece quando
O número de expressões de argumento não corresponde ao número de
Os parâmetros na definição da função estão no capítulo 8.