



Fundamentos do JavaScript

JavaScript é uma linguagem de script leve, multiplataforma e interpretada. É bem conhecido pelo desenvolvimento de páginas web, porém muitos ambientes que não são de navegador também o usam. JavaScript pode ser usado para desenvolvimentos tanto do lado do cliente quanto do lado do servidor. JavaScript contém uma biblioteca padrão de objetos, como Array, Date e Math, e um conjunto básico de elementos de linguagem, como operadores, estruturas de controle e instruções.

JavaScript, como você deve saber, é onipresente no mundo atual de desenvolvimento de software. É a base do desenvolvimento da web de front-end e é o ingrediente principal em estruturas como ReactJS, Angular e VueJS. Ele também pode ajudar a criar back-ends sólidos com plataformas como Nodejs, executa aplicativos de desktop como Slack, Atom e Spotify e é executado em telefones celulares como Progressive Web Apps (PWAs).

Variáveis

Basicamente, existem dois tipos de variáveis: globais e locais. Uma variável global é válida em todo o script. Se for alterado em uma função, também será alterado em outra função. Variáveis globais são criadas sendo nomeadas fora de uma função.

Variáveis locais são variáveis que existem apenas dentro da função específica. Se uma variável local for alterada dentro de uma função, essa alteração não afetará outras funções. Uma variável é considerada local quando é descrita pela primeira vez em uma função. Então, ele só existe no momento da sequência funcional. Diferentes funções podem ter variáveis locais com o mesmo nome, sem que elas se sobreponham.

Se uma variável é declarada dentro de uma função que tem o mesmo nome de uma variável global, a variável recém-criada é considerada uma variável local independente. A variável global com o mesmo nome não é substituída. Exemplo:

`var`

A instrução `var` declara (nomeia) uma ou mais variáveis. Para fazer isso, o nome da nova variável é informada após `var`. Se várias variáveis forem criadas, elas também serão anexadas, separadas por vírgulas. Além disso, cada variável pode receber um valor ao mesmo tempo. Para fazer isso, informe um sinal de igual (`=`) e o valor após o nome da variável . Tudo deve terminar com um ponto e vírgula (`;`). Exemplo:

`var`

A instrução `var` declara (nomeia) uma ou mais variáveis. Para fazer isso, o nome da nova variável é informada após `var`. Se várias variáveis forem criadas, elas também serão anexadas, separadas por vírgulas. Além disso, cada variável pode receber um valor ao mesmo tempo. Para fazer isso, informe um sinal de igual (`=`) e o valor após o nome da variável . Tudo deve terminar com um ponto e vírgula (`;`). Exemplo:

```
var a = 'abc';
```

```
var b, c = 3, d, e, f = 4;
```

Variáveis que são criadas dentro de uma função só se aplicam a esta função e só podem ser usadas durante o tempo de execução desta função. Variáveis

que são declaradas ou criadas fora de uma função podem ser usadas em todo o script.



Observe que a declaração da variável pode realmente ser omitida, uma vez que as variáveis são inicializadas automaticamente.

const

A chamada const define uma ou mais constantes (variáveis com característica de somente leitura). Ela serve para poder usar um valor constante. Para isso, o nome da nova constante é informado depois que a chamada é feita. Se várias forem criadas, devem ser separadas por vírgulas. Além disso, cada constante deve receber um valor. Para fazer isso, informe um sinal de igual (=) e o valor após o nome da constante. Tudo deve terminar com um ponto e vírgula (;). A chamada é válida apenas para uma linha. Se outras constantes forem declaradas, devem ser feitas na próxima linha.


Exemplo:

```
const a = 'b';  
const temperatura = 19, v = 4, c = 98,21;
```

Funções

Funções são processos específicos e bem definidos. Elas têm vários objetivos, como reagir a eventos ou verificar e calcular um ou mais valores. Elas servem para simplificar os processos usados com frequência, tendo apenas que defini-las uma vez.

Uma função consiste em: uma declaração de função (também chamada de cabeçalho de função ou especificação de função) e uma parte principal (também chamada de corpo da função). A declaração de uma função começa com a palavra function, seguida do nome da função e, imediatamente após, um parêntese de abertura e fechamento ().

Todas as instruções que devem ser concluídas quando a função é realizada são anotadas no corpo da função. Essas instruções devem estar dentro  e um bloco de instruções, ou seja, devem ser delimitadas por uma chave de abertura e uma chave de fechamento ({ }). Exemplo:

```
function minhaFuncao()  
{  
  ... declarações ...  
}
```

parâmetro

Parâmetros são informações adicionais que são passadas para uma função. Vale ressaltar que uma função pode ou não ter parâmetros, como pode ser visto no exemplo abaixo. Dizemos que uma função recebe parâmetros quando observamos na sua declaração uma variável ou uma lista de variáveis entre parênteses. A transferência de valores para uma função ocorre então quando a função é chamada, em que os valores ou variáveis que devem ser transferidos para a função também devem ser informados entre colchetes. É importante que haja sempre o mesmo número de valores a serem transferidos. Isso significa que se uma função deve ser transferida, por exemplo, 3 valores, 3 valores de variáveis também devem ser anotados quando a função é chamada.

As variáveis declaradas podem então ser usadas dentro da função. Exemplo:

```
function funcao1(a,b,c)  
{  
  x = a+b+c;  
  alert(x);  
}
```

```
function funcao2()  
{
```

```
m = 1;  
n = 1+1;  
funcao1(m,n,3);  
}
```



return

O return é usado para encerrar uma função antecipadamente ou retornar um resultado. Se aparecer, a função é encerrada neste ponto. Além disso, pode ser informado entre parênteses qual valor retornar para a função pai (ou seja, a função que chamou a função). Este valor pode ser uma variável, um número ou qualquer objeto. A única coisa importante é que apenas um valor é retornado. Exemplo:

```
function minhafuncao(a,b)  
{  
  x = a+b*100;  
  return(x);  
}
```

Atividade Extra

Para aprofundar seus conhecimentos acerca dos temas abordados neste módulo, você pode consultar os materiais através dos links abaixo:

[Entenda um pouco mais sobre JavaScript. Veja o capítulo 1](#)

[JavaScript é tão poderoso que pode ser usado em iot](#)



Referência Bibliográfica

DUCKETT, J. JavaScript de alto impacto . Alta Books, 2015.

CASTRO, R. JavaScript: guia prático. Alta Books, 2019.

FREEMAN, E... Use a cabeça! JavaScript. Alta Books, 2015.

Ir para exercício