pulei o artigo de loops
FONTE

Funções — blocos reutilizáveis de código

Funções são encontradas praticamente sempre que se faz uso de um uma estrutura JavaScript em que tem um par de parênteses e **não** é usado uma estrutura embutida da linguagem como um loop, ou if...else.
Ex: .replace('substitui a primeira strings' 'pela segunda ').
.join()concatena todos os itens de uma lista
Math.random()gera um número aleatório entre 0 e 1

Funções vs métodos

Tecnicamente falando, funções embutidas de navegador não são funções, são **métodos**. A distinção é que métodos são funções definidas dentro de objetos. Funções embutidas de navegador (métodos) e variáveis (que são chamadas propriedades) são armazenadas dentro de objetos estruturados, para tornar o código mais estruturado e fácil de manipular.

Funções anônimas

São funções que não tem um nome (function (){}).
Como essas funções não têm nomes é impossível chamá-las mais de uma vez código, a não ser que a função esteja ligada a uma variável.
Ex:
```
var myButton = document.querySelector('button');
myButton.onclick = function() {
alert('hello');
}
```
A função está definida como a valor da variável myButton, que está ligada ao Button no HTML, ou seja, todas vez que o botão for clicado a função será executada.
Também é possível definir uma função como sendo o valor de uma ou mais variáveis e chamá-la usando os nomes das variáveis seguidos por parênteses. Porém esse método é confuso e não é recomendado.
De forma geral, funções anônimas são usadas com manipuladores de eventos (ex acima).

Parâmetro de funções

Parâmetros (argumentos, propriedades, ou atributos) são valores que vão dentro das parênteses, algumas funções precisam desses valores especificados para fazer seu trabalho corretamente. Como por exemplo a função .replace(), que precisa de dois parâmetros, a substring encontrada na string principal, e a substring para substituí-la.

Função escopo e conflitos

Quando uma função é criada, todas as coisas definidas dentro dela ficam dentro de seu próprio **escopo**, significando que elas estão trancadas em seu próprio compartimento, inacessíveis de dentro de outras funções ou de código fora das funções.
Já os valores definidos no **escopo global** são acessíveis de todo o código.

Manter parte de seus código trancada em funções evita tais problemas, e é considerado boa prática.

Funções dentro de funções

É possível chamar uma função de qualquer lugar, até mesmo dentro de outra função. Isso é frequentemente usado para manter o código organizado, caso uma função grande e complexa, é mais fácil de entendê-la se quebra-lá em várias subfunções.