Funções

definição

Uma função é um grupo de linha(s) de código de programação destinado a realizar uma tarefa bem específica e que podemos, se necessário, utilizar vária vezes. A utilização de funções melhora bastante a leitura do script.

Em Javascript, existem dois tipos de funções:

- As funções próprias do Javascript. Que chamamos de "métodos". Elas são associadas a um objeto bem particular como o caso do método Alert() com o objeto window.
- As funções escritas por você para executar o seu script. Vamos estudar estas agora.

declaração de funções

Para declarar ou definir uma função, utiliza-se a palavra **function**.
A sintaxe de uma declaração de função é a seguinte:

function nome_da_função(argumentos) {
... código de instruções ...
}

O nome da função segue as mesmas regras que as variáveis (número de caracteres indefinido, começado por uma letra pode incluir números...). Volto a lembrar que o Javascript é **case sensitive**. Assim função() não será igual a Função(). Todos os nomes de funções num script devem ser únicos.

A menção dos argumentos é facultativo mas no caso dos parêntese devem ficar. É, aliás, graças aos parênteses que o interpretador Javascript distingüe as variáveis das funções. Voltaremos a falar mais sobre os argumentos e outros parâmetros logo a frente.

O fato de definir uma função não vai executar os comandos que ela contém. Só é executada quando chamamos a função.

chamada de uma função

da função com parêntese).

Seja por exemplo **nome_da_função();**

Convém verificar (porque o browser lê o script de cima a baixo) que a sua função deve estar bem definida antes de a chamar.

As funções dentro <HEAD>...</HEAD>

É assim aconselhável inserir todas as declarações de funções no cabeçalho da página , isto é entre as tags <HEAD>...</HEAD>. assim terão a certeza que as funções já estarão carregadas antes de serem chamadas no <BODY>.

exemplos

Neste exemplo, definimos dentro das tags HEAD, uma função chamada mensagem() que insere um texto "Bem vindo a minha página". Esta função será chamada no carregamento da página ver **onLoad=....** na tag <BODY>.

```
<HTML>
<HEAD>
<SCRIPT LANGUAGE="Javascript">
<--
function mensagem() {
document.write("Bem vindo a minha página");
}
//-->
</SCRIPT>
</HEAD>
<BODY onLoad="mensagem()">
</BODY>
</HTML
```

passar um valor a uma função

Pode-se passar valores ou parâmetros as funções Javascript. Assim as funções podem utilizar valores.
Para passar um parâmetro a uma função, fornece-se um nome de uma variável dentro da declaração da função.

Um exemplo simples para compreender. Esta é uma função que insere uma caixa de aviso em que o texto pode ser alterado.

Na declaração da função, escreve-se:

```
function Exemplo(Texto) {
alert(texto);
}
```

O nome da variável é Texto e é definida como um parâmetro da função.

Na invocação da função, fornece-se o texto:

```
Exemplo("Bom dia a todos");
```

passar vários valores a uma função

Pode-se passar vários parâmetros a uma função. Como é frequente o caso em Javascript, separa-se os parâmetros por vírgulas.

```
function nome_da_função(arg1, arg2, arg3) {
... código de instrução ...
}
```

Declaração de função:

```
function Exemplobis(Texto1, Texto2){...}
```

Invocação da função:

Exemplobis("Bem vindo a minha página", "Bom dia a todos"

voltar a um valor

O princípio é simples (a prática por vez não é tão simples). Para reenviar um resultado, basta escrever a palavra chave **return** seguido da expressão a reenviar. Note que devemos cercar a expressão de parênteses. Por exemplo:

```
function cubo(numero) {
var cubo = numero*numero*numero
return cubo;
}
```

A instrução return é facultativa e podemos encontrar vários return na mesma função.

Para explorar este valor da variável reenviada pela função, utiliza-se uma formulação do tipo document.write(**cubo(5)**).

variáveis locais e variáveis globais

Com as funções, o bom uso de variáveis locais e globais tem toda a sua importância.

Uma variável declarada dentro uma função pela palavra chave **var** terá uma invocação limitada a esta própria função. Não se pode assim chama-la fora do script. Chamamos assim variável local.

```
function cubo(numero) {
var cubo = numero*numero*numero
}
```

Assim a variável cubo neste exemplo é uma variável local. Se fizer-mos referência a ela fora do script, esta variável será desconhecido pelo interpretador Javascript (mensagem de erro).

Se a variável é declarada contextualmente (sem utilizar a palavra var), a sua invocação será global.

```
function cubo(numero) {
cubo = numero*numero*numero
}
```

A variável cubo declarada será aqui uma variável global.

As variáveis declaradas logo no início do script, fora e antes de todas as funções, serão sempre globais, seja ela declarada com var ou de maneira normal.

```
<SCRIPT LANGUAGE="javascript">
var cubo=1
function cubo(numero) {
var cubo = numero*numero*numero
}
</SCRIPT>
```

A variável cubo será global.

Para facilitar o controle das variáveis, posso aconselhar que as declare logo no início do script (como a maior parte das linguagens de programação). Este hábito pode prevenir algumas complicações.