02

Funções

Transcrição

Repare no código que mostra as idades de cada pessoa:

```
var ano = 2012;
document.write("Eu nasci em : " + (ano - 25) + "<br>");
document.write("Adriano nasceu em : " + (ano - 26) + "<br>");
document.write("Paulo nasceu em : " + (ano - 32) + "<br>");
```

Outra maneira de organizá-lo seria remover o "br" para outra chamada do `document.write`, por exemplo:

```
var ano = 2012;
document.write("Eu nasci em : " + (ano - 25));
document.write("<br>");
document.write("Adriano nasceu em : " + (ano - 26));
document.write("<br>");
document.write("Paulo nasceu em : " + (ano - 32));
document.write("<br>");
```

Ao fazer essa alteração até parece que o código fica mais organizado! O problema disso é se quisermos pular mais de uma linha entre cada resposta, passar um traço ou fazer qualquer outra inovação. Lembrando que para fazer qualquer uma dessas ações é preciso reproduzir a mudança em **todos** os lugares do código onde há o `document.write("<br>");`. Para evitar todo esse trabalho, podemos criar uma **funcionalidade** nova - uma que não exista no sistema - e chamamos a esse tipo de recurso de **função**. Vamos criar uma função que execute a tarefa de pular linhas, ela será nomeada de `pulaLinha`. Observe:

```
function pulaLinha() {
    document.write("<br>");
}
```

Até aqui já apareceram algumas novidades! A primeira é que acabamos de descobrir o recurso de criar uma `function`, a segunda é o uso de chaves `{` e `}` e a terceira é o deslocamento do `document.write` para a direita. Nossa preocupação não deve ser entender todos estes detalhes agora, pois vamos explicá-los melhor ao longo das aulas!

Assim:

- O uso da `function` indica a criação de um novo procedimento, inexistente até então, é útil para não precisar copiar e colar o código diversas vezes.
- As **chaves** são utilizadas para indicarem o começo e o fim do procedimento, isto é, tudo que está dentro delas faz parte da função.
- O `document.write` está mais a direita por uma questão fundamental de legibilidade. Em programação é uma convenção escrever o código dessa forma, para deixar claro que o trecho recuado encontra-se dentro da função declarada. Para criar recuos deve-se utilizar o `TAB` do teclado e essa ação chama-se **indentar** (neologismo do inglês *indent*) e o procedimento é conhecido como **indentação** do código.

Vamos voltar um pouco e pensar: Como utilizar essa nova função chamada `pulaLinha` ? Diferente de uma variável que guarda número ou string, queremos **chamar** a função para que o código que está dentro dela seja executado. Por isso escrevemos o `pulaLinha();` acompanhado de parênteses, pois isso indica que o código "pula linha" deve ser executado. O código ficará da seguinte maneira:

```
var ano = 2012;
document.write("Eu nasci em : " + (ano - 25));
pulaLinha();
document.write("Adriano nasceu em : " + (ano - 26));
pulaLinha();
document.write("Paulo nasceu em : " + (ano - 32));
pulaLinha();
```

Lembrando que a **declaração** do `pulaLinha` - o local onde escrevemos `function pulaLinha() ...` - deve ficar em cima do restante do código.

Mas, o que de fato ganhamos com essa abordagem? E no que ainda podemos melhorar?

Mas, no lugar de pular uma linha, podemos usar um efeito visual mais interessante. Que tal colocar uma linha que cruza o navegador de lado a lado? Para fazer isso vamos utilizar a tag `hr` no HTML. Para que a function `pulaLinha` utilize essa tag basta alterar sua declaração:

```
function pulaLinha() {
    document.write( " <hr> " );
}
```

O mais legal de usar esse recurso é que não precisamos mudar o código em diversos lugares para que ele continue a funcionar como desejamos. Essa é exatamente a grande vantagem de utilizar funções!

Agora, vamos escrever o código! Primeiro, criamos um novo arquivo, o `mostra_idades.html` , e nele começamos o código usando a tag `script` e depois dela declaramos a função `pulaLinha` e dentro desta última colocamos o `document.write("<br>")` . A forma de escrever a sintaxe pode parecer estranha no começo, mas observe o resultado:

```
<script>

function pulaLinha() {
    document.write("<br>");
}
```

Ao escrever isso, o `pulaLinha` passa a referir-se a uma função que deve ser chamada. Para chamá-la repetimos o que já fizemos com o `alert` , isto é, usar parênteses após seu nome:

```
var ano = 2012;
document.write("Eu nasci em : " + (ano - 25));
pulaLinha();
document.write("Adriano nasceu em : " + (ano - 26));
pulaLinha();
document.write("Paulo nasceu em : " + (ano - 32));
pulaLinha();
```

Ao fazer isso, toda vez que aparecer o `pulaLinha();` , o navegador irá executar o código da função `pulaLinha` . Dessa forma evitamos os códigos duplicados! Na sequência das aulas vamos utilizar cada vez mais as funções para facilitar a escrita do código.

Durante as aulas é importante nos concentrarmos em aprender o texto e na revisão partimos para a prática, isto é, faremos o código que está sendo descrito aqui. Nessa aula aprendemos que podemos mudar o comportamento do programa fazendo alterações em um único lugar. Em nosso caso, esse lugar é a função que criamos, a `pulaLinha` .