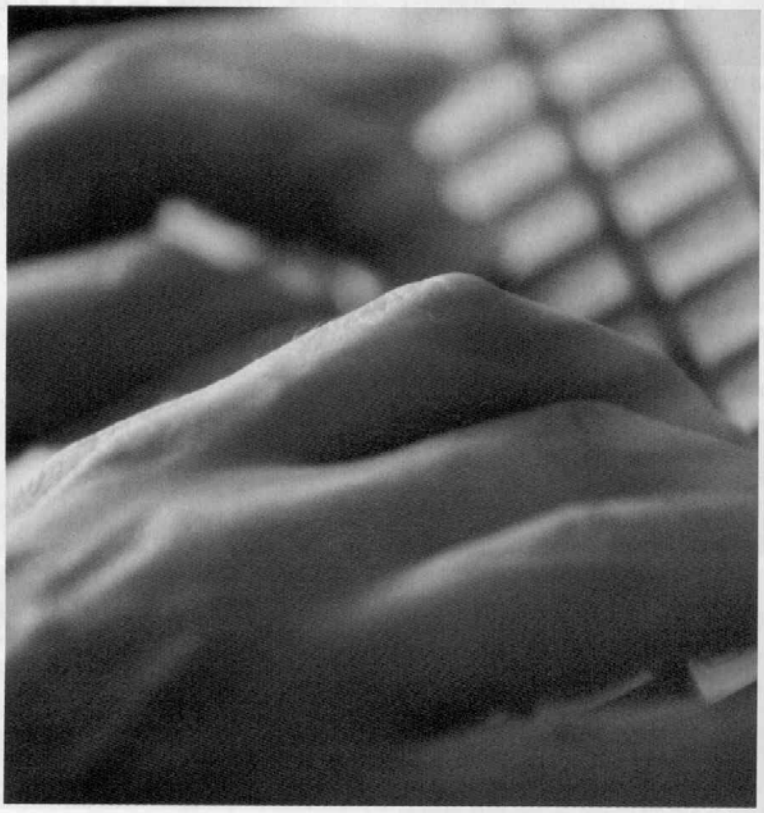
Crie páginas web inteligentes II
Exame Informática, Julho 2001, pág. 104-108

SOLUÇÕES

Crie páginas Web inteligentes com a Javascript

PARTE II

Continuamos a guiá-lo no universo da linguagem que o ajuda a tornar o seu *site* único



Regressamos a este tutorial exactamente no ponto onde tínhamos terminado o mês passado. Na altura, falávamos sobre a capacidade que esta linguagem tem de de lhe permitir conceber distinções entre várias possibilidades. Ainda em relação ao IF e ELSE, existem as seguintes opções:

AND, OR, NOT

Para melhorar as suas declarações "if" poderá ainda usar os operadores lógicos. O **AND** será escrito como **&&** e é usado para verificar se mais do que uma condição é verdadeira.

A sintaxe será:
if (condição && condição) {acção}

Exemplo:
if(hora==12&& minutos==0) {alert("é meio-dia")};
O **OR** será escrito como ||.

A sintaxe será:
if (condição || condição) {acção}

Exemplo:
if (hora==11 || hora==10) {alert("faltam menos de duas horas para o meio-dia")};
O NOT é escrito como ! e é usado para inverter o resultado.

A sintaxe será:
if (!(condição)) {acção}

Exemplo:
if (!(hora==11)) {alert("falta mais de uma hora para o meio-dia")};

Funções

Em vez de adicionar o seu código em javascript nas suas páginas, e ter o *browser* a executar esse mesmo código assim que o lê, o leitor poderá querer só executá-lo se determinado evento ocorrer.
Por exemplo, se o leitor escrever um código em Javascript que mude a cor da página em *background* quando se pressiona determinado botão, então será necessário "dizer ao *browser*" para não executar o *script* quando o ler.
Para impedir que o *browser* execute o *script* assim que o ler, o leitor necessitará de escrevê-lo como uma função.

Veja o seguinte exemplo de código em *script* escrito como função:

```
<html>
<head>
<script>
function minhafuncao()
{
alert("Bem-vindo ao meu mundo!!!);
}
</script>
</head>

<body>
<form name="minhaform">
<input type="button" value="Pressione-me" onClick="minhafuncao()">
</form>
</body>
</html>
```

Se a linha: alert ("Bem-vindo ao meu mundo!!!); não tivesse sido escrita como

uma função, teria sido executada quando o *browser* a carregasse. Mas como foi escrita como uma função, só será executada quando o utilizador premir o botão.

Chama-se a função nesta linha:
<input type="button" value="Pressione-me" **onClick="minhafuncao()"**>

Como pode ver, colocámos o botão dentro de uma *form* e adicionámos o evento onClick="minhafuncao" às propriedades do botão.

A sintaxe geral para uma função será:
function nomedafuncao(variavel1, variavel2, ...,variávelX)
{
// Aqui escreve-se as linhas em Javascript para a função
}
Os { e } marcam o início e fim da função.

NOTA

Um erro típico quando se escrevem funções em Javascript é esquecer a importância das palavras em maiúscula nesta linguagem. A palavra function deverá ser escrita exactamente como function. Function ou FUNCTION causará um erro.

Eventos

Os eventos são acções que podem ser detectadas pela javascript.
Um exemplo poderá ser o evento onMouseOver, que é detectado quando o utilizador move o rato por cima de um objecto.
Outro evento será o onLoad, que é detectado assim que uma página acaba de ser carregada.
Habitualmente, os eventos são usados em combinação com funções para que determinada função não comece antes que um evento ocorra.
Um exemplo seria uma função para animar um botão.
A função simplesmente trocaria duas imagens. Uma imagem que mostraria o botão na posição "para cima" e outra que mostraria o botão na posição de "para baixo".
Se esta função fosse chamada usando um evento onMouseOver, o resultado seria quando o rato estivesse sobre a imagem, o botão estaria na posição "para baixo".

Eventos

Evento	Detectado quando...	Tags em HTML
onFocus=""	quando o campo na *form* obtém o *focus*	select, text, textarea
onBlur=""	quando o campo na *form* perde o *focus*	select, text, textarea
onChange=""	o conteúdo de um campo muda	select, text, textarea
onSelect=""	o texto é seleccionado	text, textarea
onMouseOver=""	o rato move-se sobre um *link*	A
onMouseOut=""	o rato move-se para for a de um *link*	A
onClick=""	o rato clica um objecto	A, button, checkbox, radio, reset, submit
onLoad=""	a página acaba de carregar	body, frameset
onUnload=""	o *browser* abre um novo documento	body, frameset
onSubmit=""	o botão é pressionado	form

A tabela em cima mostra os eventos mais importantes reconhecidos pela Javascript:
Os eventos são usados para atingir dois objectivos:
Para executar uma função após a detecção de um evento
Para mostrar uma caixa *popup* após a detecção de um evento.

onFocus, **onBlur** e **onChange** são usualmente usados em combinação com validações nos campos da *form*.

Por exemplo, digamos que concebeu uma função chamada validaremail() que vai verificar se o *e-mail* que o utilizador introduz contém um @ e, ainda, se contém um final com algum significado, tais como, "com", "net", etc. Suponha ainda que o utilizador poderá introduzir o seu *e-mail* numa *form*.
Poderia usar, então, o evento **onChange** para chamar a função, sempre que o utilizador mudasse o valor do campo:
<input type="text" size="20" onChange="validaremail()">;

onLoad e UnLoad

Os eventos onLoad e onUnload são habitualmente usados para *popups* que aparecem quando o utilizador entra ou sai de uma página. Outra utilização importante será em combinação com *cookies* que são enviados quando o utilizador entra ou sai de uma página.

Exemplo:
O leitor poderá ter um *popup* dizendo ao utilizador para introduzir o seu nome aquando da sua primeira visita a uma página. O nome será então introduzido num *cookie*. Quando o visitante sair da página, um *cookie* armazenará a data. Da próxima vez que o mesmo utilizador visite a mesma página, haverá um outro *popup* dizendo qualquer coisa do tipo: "Bem-vindo Rosa Mota, esta página não foi actualizada desde a sua última visita há 8 dias".

OnSubmit

O evento **onSubmit** é usado para atingir um grande objectivo: validar todos os campos numa *form* antes de submetê-la.
No exemplo acima exposto vimos como validar um campo numa *form* com o evento **onChange**. Mas, por vezes, o visitante pode achar complicado ter tantas validações no meio dos campos nessa *form*. Então poderá usar-se um único botão para validar todos os campos ao mesmo tempo. Isto poderá fazer-se usando o evento **onSubmit**.
Assuma que tinha feito uma função chamada **verificaform**() que validaria as entradas nessa *form*.
Agora pretende-se que esta função seja chamada quando o utilizador carrega num botão. Se os valores não forem aceites pela função, a submissão nessa *form* deverá ser cancelada. Deste modo, os dados não seriam introduzidos a não ser que a função aceite esses valores.

O que o leitor deverá fazer é adicionar um evento **onSubmit** à tag **<form>** da seguinte maneira:

```
< form method="suaescolha"
action="suaescolha" onSubmit="return
verificaform()">
```

A função **verificaform()** devolve "verdadeiro" ou "falso".
Se retornar "verdadeiro" os dados serão validados.
Se retornar "falso" os dados serão cancelados.

Crie botões animados

Os eventos **onMouseOver** e **onMouseOut** são usados essencialmente com um objectivo: criar botões animados.
O leitor poderá ter notado que estes eventos só podem ser usados em combinação com a *tag* **<a>**. De qualquer modo, estes eventos são mais úteis em combinação com a *tag* **<img>**. O truque para fazer com que o evento funcione numa imagem será, simplesmente, fazer com que a imagem funcione como um *link*. (Se a imagem não é suposta funcionar como um *link*, o leitor poderá criar o *link* para uma âncora vazia, como mostramos no exemplo abaixo).

Exemplo:
Uma caixa *alert* aparece quando o evento onMouseOver é detectado numa imagem:

O código em HTML virá:

```
<a href="#"
onMouseOver="alert("Detectei um
evento onMouseOver"); return false"
onMouseOut="alert("Detectei um
evento onMouseOut"); return false">
<img src="teste.gif" width="60" hei-
ght="60">
</a>
```

NOTA

Escrevendo href="#" cria um *link* vazio. Se realmente pretende que o *link* aponte para uma página, deverá escrever o endereço dessa mesma página.

Ciclos

Imagine que quer escrever um *script* que permite realizar a mesma rotina umas cinquenta vezes seguidas. Um exemplo disso seria se o programador quisesse produzir uma tabela comparando temperaturas em Fahrenheit e Celsius. O *script* deveria escrever cinquenta linhas numa tabela, mostrando as diferentes temperaturas de acordo com as duas escalas. Em vez de se adicionar cinquenta linhas quase idênticas no seu código, o programador poderá usar ciclos.
Existem dois tipos diferentes de ciclos: o ciclo **for** e o ciclo **while**.
O ciclo **for** é usado quando o programador já sabe quantas vezes o *script* deverá correr. Por exemplo, se o programador quiser criar exactamente cinquenta linhas.
O ciclo **while** é usado quando se se pretende que o ciclo decorra até que uma determinada condição seja verdadeira.
Por exemplo, se o programador quiser criar uma tabela comparando valores em Celcius e Fahrenheit, saltando 15 graus em cada linha, e se pretender que a tabela mostre valores até 1200 graus em Celcius.
Em baixo mostramos uma descrição destes dois ciclos:

Ciclo For

Sintaxe:

```
for (variável=valorinicial;
variável<=valorfinal;
variável=variável+factorincremento)
{
// aqui vem o código do "script" a ser
repetido
}
```

Onde diz "variável" escreva o nome da variável a usar.
Onde diz "valorinicial" escreva o valor inicial onde começa o ciclo.
Onde diz "valorfinal" escreva o valor final onde acabará o ciclo.
Onde diz "factorincremento" escreva o factor a ser incrementado em cada ciclo.

NOTA

O factor a incrementar poderá ser negativo.

Exemplo:

```
<html>
<head>
<title>Conversor Celcius-Fahrenheit</
title>
</head>

<body>
<table border=3>
<tr><td>CELCIUS</
td><td>FAHRENHEIT</td></tr>
<script language="javascript">
for (celcius=0; celcius<=50;
celcius=celcius+1)
{document.write("<tr><td>"+celcius+"</
td><td>"
+((celcius*9/5)+32)+"</td></tr>");
}
</script>
</table>
</body>
</html>
```

Ciclo While

Sinaxe:

```
while (variável<=valorfinal)
{
// aqui vem o código do "script" a ser
repetido
}
```

Escreva o nome da variável onde diz "variável".
Escreva o valor final do ciclo onde diz "valorfinal".

Exemplo:

```
<html>
<head>
<title>Conversor Celcius-Fahrenheit</
title>
</head>

<body>
<table border=3>
<tr><td>CELCIUS</
td><td>FAHRENHEIT</td></tr>
<script language="javascript">
celcius=0;
while (celcius<=50)
{
document.write("<tr><td>"+celcius+
"</td><td>"+((celcius*9/5)+32)+"</
td></tr>");
celcius=celcius+1;
}
</script>
</table>
</body>
</html>
```

Para o mês que vem...

Na próxima edição continuaremos a explicar os restantes aspectos mais básicos desta linguagem *script* que pode mudar radicalmente o seu *site*.

SÉRGIO AZEREDO