9

Imagens

EM JAVASCRIPT, É POSSÍVEL MANIPULAR COM AS IMAGENS DE UMA PÁGINA, alterando a URL que localiza o arquivo de imagem. Assim, pode-se trocar a imagem que está sendo exibida por outra durante a exibição da página. Também é possível criar novos objetos representando imagens que inicialmente não aparecem na página e transferir seus arquivos previamente em *background*, para que estejam disponíveis na memória na hora da substituição. Com esses recursos, pode-se incrementar a página com recursos dinâmicos, como ícones que mudam de aparência quando ocorre um evento, animações e *banners*.

As imagens utilizadas em JavaScript podem ser carregadas de duas formas: através do HTML e através de instruções JavaScript. As *imagens estáticas*, fornecidas pela página HTML através do descritor `<IMG>`, são representadas como objetos da página (`document`), acessíveis através da sua propriedade `images`: um vetor que contém referências para todas as imagens do documento. As *imagens dinâmicas*, que não são fornecidas pelo HTML, podem ser criadas como objetos JavaScript dentro de qualquer bloco `<SCRIPT>` ou atributo HTML de eventos usando o operador '`new`' e o construtor `Image()`.

Neste capítulo, conheceremos as duas formas de manipular imagens em JavaScript, e como utilizá-las para criar páginas dinâmicas eficientes.

Image

Tanto uma imagem visível em uma página HTML como uma imagem carregada na memória, porém invisível, podem ser representadas em JavaScript por um objeto do tipo *Image.*. Para criar uma referência para uma imagem que não existe na página, é preciso usar `new`:

```
figura5 = new Image(50, 100);
```

onde os números passsados como parâmetros (opcionais) correspondem respectivamente à largura e altura da imagem na página em pixels. Pode-se também usar:

```
figura6 = new Image()
```

que calculará o tamanho da imagem quando ela for carregada.

Depois que um objeto do tipo *Image* é criado, e suas dimensões definidas, seu tamanho não pode mais ser alterado. Todas as imagens passadas para a referência `figura5` abaixo serão redimensionadas para 50x100 pixels:

```
figura5.src = "square.gif";    // square.gif agora tem 50x100 pixels
```

A propriedade `src` tem a mesma função do atributo `SRC` do descritor HTML `<IMG>`: indicar a URL do arquivo-fonte da imagem.

Toda página que possui o descritor HTML `<IMG>` já possui um objeto *Image* que pode ser manipulado através da sua propriedade `document.images` (do tipo *Array*). Para criar uma nova imagem no documento, é preciso usar HTML e o descritor `<IMG>`, cuja sintaxe geral está mostrada abaixo:

```
<IMG SRC="URL do arquivo-fonte da imagem"
   NAME="nome_do_objeto"
   ALT="texto alternativo (descrição da imagem)"
   LOWSRC="URL de arquivo-fonte de baixa-resolução"
   HEIGHT="altura em pixels"
   WIDTH="largura em pixels"
   HSPACE="margens externas laterais em pixels"
   VSPACE="margens externas verticais em pixels"
   BORDER="largura da borda de contorno em pixels "
   ALIGN="left" ou "right" ou "top" ou "middle" ou "bottom" ou
         "texttop" ou "absmiddle" ou "absbottom" ou "baseline"
   ISMAP           <!-- é imagem mapeada do lado do servidor -->
   USEMAP="#mapa"  <!-- é imagem mapeada por 'mapa' no cliente -->
   ONABORT="Código JavaScript"
   ONERROR="Código JavaScript"
   ONLOAD="Código JavaScript" >
```

Todos os atributos, com exceção de `SRC`, são optionais. Para manipular uma imagem do HTML em JavaScript, é preciso usar o vetor `images` que contém referências para cada uma das imagens do documento, na ordem em que elas aparecem no código HTML:

```
prima = document.images[0];    // primeira imagem da página atual
nona = document.images[8];     // nona imagem da página atual
```

Assim como formulários e *frames*, que são acessíveis através de vetores ou nomes, as imagens podem receber um nome, para tornar o seu acesso mais fácil. O atributo HTML opcional `NAME`, se presente, pode ser usado pelo JavaScript para fazer referência à imagem, em vez de usar o vetor `images`. É uma boa idéia, pois torna o código mais legível e independente da ordem e número de imagens na página. Por exemplo, a imagem:

```
<img src="tapir.gif" name="anta" width=380 height=200>
```

pode ser referenciada do JavaScript da forma:

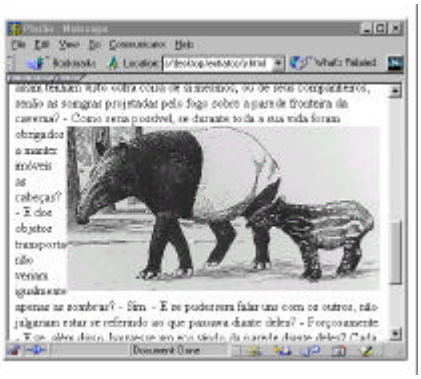
```
prima = document.anta;
```

Para manipular com a fonte da imagem, ou acompanhar o status de seu carregamento na página, é preciso recorrer às propriedades definidas para o tipo *Image*, listadas abaixo. Com exceção de `complete` têm o mesmo nome que os atributos HTML do descritor `<IMG>`. Com exceção de `src` e `lowsrc`, todas são *somente-leitura*.

Propriedade	**Descrição**
`complete`	*Boolean.* Contém `true` se a imagem foi carregada completamente.
`border`	*String.* Reflete o valor do atributo HTML `BORDER`
`height`	*String.* Reflete o valor do atributo HTML `HEIGHT`
`name`	*String.* Reflete o valor do atributo HTML `NAME`
`src`	*String.* Reflete o valor do atributo HTML `SRC` se for uma imagem da página HTML, e permite redefini-lo. Sempre indica o arquivo-fonte ou URL da imagem.
`lowsrc`	*String.* Reflete o valor do atributo HTML `LOWSRC` e permite redefini-lo. Indica o arquivo-fonte de baixa-resolução temporário da imagem, que é carregado antes do arquivo em `SRC`.
`hspace`	*String.* Reflete o valor do atributo HTML `HSPACE`
`vspace`	*String.* Reflete o valor do atributo HTML `VSPACE`
`width`	Retorna o valor do atributo HTML `WIDTH`

Das propriedades acima, `src` é a mais importante. É ela quem indica o arquivo-fonte da imagem através de uma URL (pode ser uma URL relativa, contendo apenas o nome de um arquivo localizado no mesmo diretório que a página). Um string contendo essa URL, atribuído à propriedade `src` de um objeto *Image*, fará com que o browser tente (imediatamente) carregar o arquivo-fonte da imagem:

`bat.jpg (200x200)`



`<img src="tapir.gif" height=200 width=380>`

```
animal = new Image();
animal.src = "../figuras/bat.jpg";
```

As imagens carregadas através das instruções JavaScript acima não são exibidas automaticamente quando a página é carregada. Precisam um contexto gráfico na página onde possam ser exibidas. Esse contexto é fornecido pelo descritor HTML `<IMG>`. Não é possível exibir imagens em páginas que não tenham pelo menos um descritor `<IMG>`, mesmo que essas

imagens tenham sido carregadas[1]. O descritor `<IMG>` funciona como uma janela onde a imagem pode ser exibida. Havendo um descritor `<IMG>` na página, sua imagem original poderá ser substituída por qualquer imagem, cujo arquivo tenha sido carregado dinamicamente, através da propriedade `src`. Por exemplo:

```
document.images[0].src = animal.src;
```

fará com que a imagem `bat.jpg` ocupe o lugar da primeira imagem da página (`tapir.gif`, na figura acima). O contexto gráfico não pode ser redimensionado[2], então a imagem `bat.jpg`, que tem dimensões 200x200 será redimensionada e ocupará o espaço antes ocupado por `tapir.gif` (380x200). O resultado será uma imagem "esticada". Veja a figura ao lado.

A imagem também poderia ter sido substituída diretamente, sem precisar criar um novo objeto do tipo *Image*:

```
document.images[0].src = "../figuras/bat.jpg";
```

mas isto faria com que o browser tentasse carregar a imagem no momento em que a instrução acima fosse interpretada. Se uma página já foi completamente carregada e um evento dispara a instrução acima, o usuário teria que esperar que a imagem fosse carregada através da rede. No outro exemplo, a carga da imagem poderia ter sido feita antes. Quando o evento causasse a troca das imagens, ela estaria disponível no cache e seria substituiria a antiga imediatamente.

Quando se utiliza várias imagens, é útil carregá-las todas antes do uso. Isto pode ser feito colocando instruções em um bloco `<SCRIPT>` dentro do `<HEAD>` de uma página, o que garante que será executado antes de qualquer outra instrução no `<BODY>`. O código abaixo carrega 10 imagens chamadas `tela1.gif`, ..., `tela10.gif` e as armazena em um vetor `telas`:

```
<head>
<script>
  telas = new Array(5);
  for (i = 0; i < imagens.length ; i++) {
    telas[i] = new Image();
    telas[i].src = "tela" + (i+1) + ".gif";
  }
</script>
(...)
<head>
```

[1] É possível, porém, inserir dinamicamente um descritor `<IMG>` na página, usando `document.write()`.

[2] JavaScript 1.1

As 10 imagens podem ser usadas para substituir outras imagens da página ou para fazer uma animação. Usando vetores, fica fácil manipular toda a coleção através de seus índices. Quando uma substituição ocorrer:

```
document.images[2].src = telas[5];
```

o arquivo será trocado imadiatamente, pois está disponível localmente.

Quando se têm muitas imagens, o browser poderá demorar para mostrar a página, enquanto executa as instruções no `<HEAD>`. Uma forma eficiente de evitar esse problema, é colocar as instruções dentro de uma função global, e chamá-la quando a página tiver terminado de carregar, usando `<BODY ONLOAD="nomeFuncao()">`. Por exemplo:

```
<head>
<script>
   var telas;      // variável global, para que possa ser usada em
                   // outras partes da página

   function carregaImagens() {
      telas = new Array(10);
      for (i = 0; i < imagens.length ; i++) {
         telas[i] = new Image();
         telas[i].src = "tela" + (i+1) + ".gif";
      }
   }
   (...)
</script>
</head>
```

poderá ser chamada depois que a página tiver sido carregada com:

```
<BODY ONLOAD="carregaImagens()">
 ...
</BODY>
```

Na transferência de imagens através da rede é comum acontecerem erros, provocando a interrupção da transferência ou a carga incorreta da imagem. A propriedade `complete` pode ser usada para verificar se uma imagem já foi carregada totalmente, antes de utilizá-la. `complete` contém o valor `true` somente se a imagem já foi carregada totalmente. Se ocorrer um erro ou a carga da imagem for interrompida, `complete` irá conter o valor `false`:

```
if (telas[9].complete) {
   iniciarAnimacao();
}
```

Eventos

Os atributos HTML de `<IMG>` que respondem a eventos associados com imagens são:

- `ONERROR` – executa quando acontece um erro (arquivo não encontrado, conexão perdida). Uma imagem com o atributo `ONERROR` usado da forma:

  ```
  <img src="coisa.gif" ONERROR="this.src = 'coisa.gif'">
  ```

  fará com que a transferência da imagem seja reiniciada se houver erro.
- `ONABORT` – executa quando o usuário solicita a interrupção da transferência da imagem. Uma imagem com o atributo `ONABORT` usado da forma:

  ```
  <img src="coisa.gif" ONABORT="this.src = 'coisa.gif'">
  ```

  fará com que a transferência da imagem seja reiniciada cada vez que o usuário tentar interrompê-la.
- `ONLOAD` – executa quando a imagem termina de ser carregada.

Freqüentemente, imagens são colocadas dentro de links, e os eventos aplicáveis a links podem ser usados para realizar a substituição de imagens. Isto é usado, por exemplo, em ícones dinâmicos, assunto do exercício a seguir.

Exercício Resolvido

Crie um icone ativo (link em torno de uma imagem) em uma página HTML que muda de cor quando o mouse passa sobre ele. Utilize duas imagens disponíveis no diretório `cap9/`. A primeira, `dullbart.gif`, mais apagada, deve estar presente na página quando ela for carregada; a outra `brightbart.gif`, deve substituir a primeira quando o mouse estiver sobre ela (evento `ONMOUSEOVER` do link `<A>`). A primeira imagem deverá voltar a ocupar a sua posição quando o mouse deixar a imagem (evento `ONMOUSEOUT` do link `<A>`). Há um esqueleto disponível em `bart.html`.

A substituição deve ser imediata. As duas imagens devem estar carregadas na memória antes de haver qualquer substituição.

Solução

A listagem a seguir apresenta uma possível solução ao problema proposto (está no arquivo `bartsol.html`).

```
<html>
<head>
<title>Imagens</title>
<script>
```

```
apagado = new Image();
aceso   = new Image();
apagado.src = "dullbart.gif";
aceso.src   = "brightbart.gif";

function apaga() {
      document.images[0].src = apagado.src;
}

function acende() {
      document.images[0].src = aceso.src;
}

</script>
</head><body>
<a href="" onmouseover="acende()" onmouseout="apaga()">
   <img src="dullbart.gif" width=43 height=58 border=0>
</a>
</body>
</html>
```

Exercícios

9.1 Crie um banner animado usando JavaScript que mostre várias imagens em seqüência (use as imagens `im-01.jpg` a `im-05.jpg` (figura ao lado) disponíveis no diretório `cap9/`. As imagens devem ser carregadas previamente, logo após a carga da página (use `ONLOAD`). O intervalo de tempo entre cada imagem deve ser de um segundo (solução em `cap9/anima.html`).

9.2 Na página `uma.html` (figura abaixo) há 5 imagens preto-e-branco. Faça as seguintes alterações para que sua aparência seja mudada de acordo com o movimento do mouse do usuário sobre as imagens (use os arquivos disponíveis no diretório `cap9/`):

a) Faça com que cada imagem `impb-nn.jpg` (onde `nn` é `01`, `02`, `03`, `04` ou `05`) seja trocada por sua correspondente a cores `im-nn.jpg` quando o mouse passar sobre a ela.

b) Quando o mouse deixar a imagem e passar para outra área da página,

a imagem original (em preto e branco) deve ser restaurada.

c) Faça com que cada imagem seja um link ativo para as páginas `pag-01.html` a `pag-05.html`.

9.3 Inclua as páginas `uma.html` e `menu.html` em uma estrutura de frames como ilustrado abaixo (use o arquivo `frame.html`). Clicar em uma das imagens deve fazer com que a janela principal seja ocupada pela página referenciada pelo link (`pag-01.html` a `pag-05.html`) e que a *imagem* (não a página) da janela secundária seja trocada por uma imagem correspondente (`menu01.gif` a `menu02.gif`).