

## 7

## Navegação

CINCO OBJETOS JAVASCRIPT ESTÃO RELACIONADOS COM A NAVEGAÇÃO em hipertexto. Com eles é possível ler e alterar as localidades representadas pelos links, redirecionar as janelas do browser para outras páginas e controlar as informações contidas no histórico de navegação de uma janela.

*Area*, *Link* e *Anchor* permitem manipular com as propriedades dos elementos HTML `<AREA>`, `<A HREF>` e `<A NAME>` contidos em uma página HTML. Os objetos *History* e *Location* permitem mudar o conteúdo das janelas dinamicamente.

## Objeto *History*

O objeto *History* é um vetor de strings somente-leitura usado por uma janela do browser para armazenar os lugares já visitados durante uma sessão. O conteúdo da lista é o equivalente ao encontrado nas opções “Histórico”, “History” ou “Go” dos browsers Microsoft Internet Explorer e Netscape Navigator. Os botões “Back” ou “Voltar” e “Forward” ou “Avançar” usam as informações no histórico para navegar através dele.

*History* é uma característica da janela. Todo objeto *Window* possui uma propriedade *history*. Na janela atual, pode ser usado também como uma referência global, usando simplesmente o nome *history*.

As propriedades de *History* são quatro mas apenas uma é utilizável na prática, que é *length*. As outras só são suportadas em browsers Netscape e com várias restrições:

Propriedade	Descrição
<i>length</i>	<i>Number</i> . Contém o número de itens do histórico do browser
<i>current</i>	<i>String</i> . Contém uma string com a URL da página atual.
<i>next</i>	<i>String</i> . Contém uma string com a URL da próxima página do histórico
<i>previous</i>	<i>String</i> . Contém uma string com a URL da página anterior do histórico.

Em JavaScript 1.1, o acesso às propriedades acima, exceto `length`, só é possível se o modelo de segurança `data-tainting` estiver ativado no browser do cliente<sup>1</sup>. Tendo o acesso às URLs do histórico, pode-se redirecionar a janela do browser até qualquer página já visitada, fazer buscas e imprimir o histórico. Tudo isso pode ser feito de forma simples, sem as restrições do `data-tainting`, usando os métodos de *History*.

Método	Ação
<code>go(<math>\pm n</math>)</code> ou <code>go("string")</code>	Avança ou volta $n$ páginas no histórico. A segunda forma procura no histórico até encontrar a primeira página que tenha a string especificada no título do documento ou nas palavras da sua URL.
<code>back()</code>	Volta uma página no histórico (simula o botão "Back" ou "Voltar" do browser).
<code>forward()</code>	Avança uma página no histórico (simula o botão "Forward" ou "Avançar" do browser).
<code>toString()</code>	Retorna <i>String</i> . Converte o histórico em uma tabela HTML de URLs, cada uma com seu link. Pode ser impressa usando <code>document.write()</code> .. Só funciona se o modelo de segurança <code>data-tainting</code> estiver ativado.

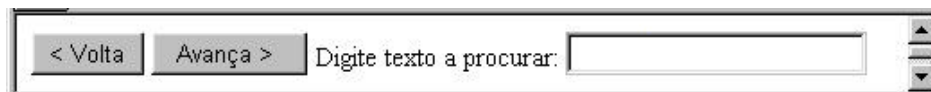
O trecho abaixo simula um painel de navegação em uma janela filha que controla o histórico da janela que a criou<sup>2</sup> usando métodos de *History*.



```
<form>
<input type=button value="&lt; Volta"
        onclick="opener.history.back()" >
<input type=button value="Avança &gt;"
        onclick="opener.history.forward()" >
</form>
```

## Exercícios

- 7.1 Utilize o painel de navegação apresentado como exemplo nesta seção dentro de um frame. Acrescente um mecanismo de busca no histórico usando o método `go("string")`. Utilize uma caixa de textos (`<input type=text>`) para que o usuário possa entrar com uma palavra para procurar.



<sup>1</sup> O que, na maioria dos casos, não ocorre. `Data-tainting` (veja capítulo 6) é suportado por browsers Netscape, apenas, e mesmo assim, exige que o usuário habilite a opção em seu browser (não é default).

<sup>2</sup> O painel de navegação funcionará enquanto o usuário se mantiver no mesmo site (a não ser que `data-tainting` esteja ativado).

## Objeto Location

*Location* é uma propriedade das janelas que representa a URL cujo documento está sendo exibido<sup>3</sup>. Todo objeto *Window* possui uma propriedade `location`. As propriedades de *Location* são strings com partes da URL atual. Se forem alteradas, a URL atual irá mudar e o browser tentará imediatamente carregar o recurso localizado pela nova URL na janela atual.

A propriedade mais usada de *Location* é `location.href`, que contém a URL completa. Mudar o valor de `location.href` é uma forma de causar o redirecionamento dinâmico:

```
location.href = "http://www.aeiouy.com/pag2.html"
```

carrega a página localizada por `http://www.aeiouy.com/pag2.html` no momento em que o browser interpretar a linha acima.

Todas as outras propriedades de `location` são substrings do string que contém a URL completa contida na propriedade `href`. Todas podem ser lidas e alteradas:

Propriedade	Descrição
<code>href</code>	A URL completa. Exemplo: <code>http://www.abc.com:80/sub/dir/index.html?name=Cookie1#parte2</code>
<code>protocol</code>	O protocolo da URL. Ex: <code>http:</code>
<code>host</code>	O nome da máquina. Ex: <code>//www.abc.com</code>
<code>port</code>	A porta do servidor. Ex: <code>80</code>
<code>hostname</code>	O nome do servidor. Ex: <code>//www.abc.com:80</code>
<code>pathname</code>	O caminho. Ex: <code>/sub/dir/index.html</code>
<code>hash</code>	O fragmento. Ex: <code>#parte2</code>
<code>search</code>	O string de busca. Ex: <code>?name=Cookie1</code>

Os métodos de `location` são dois: `reload()` é usado para fazer uma página ser recarregada e `replace()` apaga a página anterior do histórico, substituindo-a com uma nova:

Método	Ação
<code>reload()</code> ou <code>reload(true)</code>	Sem argumentos, recarrega a página atual caso não tenha sido modificada. Com o argumento <code>true</code> , carrega a página incondicionalmente.
<code>replace("URL")</code>	Carrega a página especificada pela URL e substitui o registro anterior do histórico com o registro atual.

<sup>3</sup> Tem o mesmo significado que o cabeçalho HTTP "Location"

## Exercícios

- 7.2 Crie uma “roleta” que jogue o usuário em um site escolhido aleatoriamente a partir de uma lista armazenada em um vetor.
- 7.3 Crie uma janela pequena, sem barra de menus, status, ou scrollbars, para servir de barra de navegação flutuante para um site. Ela deve abrir quando o usuário clicar em um link “SiteMap” e ficar sempre na frente das outras janelas. Todas as URLs das páginas do site devem estar em um componente `<select>` que, ao ter uma opção escolhida pelo usuário, deve fazer com que a janela que o abriu (se ela ainda existir) passe a exibir a nova URL. Se a janela não mais existir, uma nova deverá ser criada.
- 7.4 Usando `setTimeout()` (método de *Window*), escreva uma rotina que faça com que uma página carregue outra que a substitua na janela do browser em 30 segundos.

## Objetos Area e Link

Os objetos *Area* e *Link* são a mesma coisa. Ambos representam *vínculos* (referências de hipertexto). O objeto *Area* representa o vínculo acionado a partir de uma imagem mapeada do lado do cliente (client-side *imagemap*) pelo do descritor `<AREA>` e *Link* representa o vínculo criado a partir de um `<A HREF>`.

O acesso a todos os vínculos de um documento é obtido a partir da propriedade `links` de `document`. *Link* e *Area* são objetos do tipo *Location*, mas não são a mesma coisa que a *propriedade* `location` de `window`. A alteração da propriedade `href` ou qualquer outra, muda a URL à qual o vínculo se refere e não interfere na URL do documento exibido na janela. O efeito da mudança só será notado quando o usuário clicar sobre o vínculo.

Todos os objetos *Link* e *Area* de uma página estão disponíveis através do vetor `document.links`. Eles podem ser acessados pelo índice do vetor correspondente à ordem em que aparecem no código HTML da página ou pelo nome (se houver). Por exemplo, suponha que uma página tenha o seguinte código HTML:

```
<BODY>
<h1><a name="top"></a>Mapa Interativo</h1>

<map name="brasil">
  <area href="norte.html" shape="poly" coords="..." name="n">
  <area href="nordeste.html" shape="poly" coords="..." name="ne">
  <area href="centroeste.html" shape="poly" coords="..." name="co">
  <area href="sudeste.html" shape="poly" coords="..." name="se">
  <area href="sul.html" shape="poly" coords="..." name="s">
</map>
<p align=center>
<a href="index.html">Volta para Revendedores</a>
```

```
<a href="../index.html" name="hp">Home Page</a>
</BODY>
```

Todos os elementos HTML marcados em negrito acima fazem parte do vetor `document.links`. Para saber quantos vínculos *Area* e *Link* existem em uma página, pode-se usar a propriedade `length` do vetor:

```
numLinks = document.links.length; // numLinks contém 7
```

O elemento `<A NAME="top"></a>`, no código acima, não é um objeto *Link*, pois não contém o atributo `HREF`. É apenas um objeto *Anchor*. *Links* que têm o atributo `NAME` e serão tratados ao mesmo tempo como objetos *Link* e *Anchor*.

Há duas maneiras para fazer com que o objeto *Area*

```
<area href="sudeste.html" shape="poly" coords="..." name="se">
```

do exemplo acima tenha sua URL destino alterada para "sudeste/index.html": através do índice do vetor `links` ou através do nome:

```
document.links[3].href = "sudeste/index.html";
document.se.href = "sudeste/index.html";
```

As propriedades de *Link* e *Area* são praticamente as mesmas de *Location*. A única diferença é a propriedade `target`, que não existe em *Location*.

Propriedade	Descrição
<code>href</code>	A URL completa. Exemplo: <code>http://www.abc.com:80/sub/dir/index.html?name=Cookie1#parte2</code>
<code>protocol</code>	O protocolo da URL. Ex: <code>http:</code>
<code>host</code>	O nome da máquina. Ex: <code>//www.abc.com</code>
<code>port</code>	A porta do servidor. Ex: <code>80</code>
<code>hostname</code>	O nome do servidor. Ex: <code>//www.abc.com:80</code>
<code>pathname</code>	O caminho. Ex: <code>/sub/dir/index.html</code>
<code>hash</code>	O fragmento. Ex: <code>#parte2</code>
<code>search</code>	O string de busca. Ex: <code>?name=Cookie1</code>
<code>target</code>	O nome da janela ou frame onde a URL será exibida.

## Eventos

Não há métodos definidos para os objetos *Link* e *Area*. Existem, porém, três eventos manuseados por atributos dos elementos HTML que representam esses objetos. Os dois primeiros atributos aplicam-se tanto a elementos `<A HREF>` como a elementos `<AREA>`:

- `ONMOUSEOVER` – quando o usuário move o mouse sobre o vínculo ou imagem.
- `ONMOUSEOUT` – quando o usuário afasta o mouse que antes estava sobre o vínculo ou imagem.

O terceiro, só produz efeito em elementos `<A>`. É ignorado em elementos `<AREA>`:

- `ONCLICK` – quando o usuário clica o mouse sobre o vínculo.

Todos os eventos são tratados *antes* que o browser siga o vínculo do atributo `HREF`, por exemplo, no código abaixo, a URL no atributo `HREF` do vínculo abaixo nunca será carregada pois a janela será redirecionada para outra localidade assim que o usuário passar o mouse sobre o link:

```
<a href="http://www.sao.nunca.org" onmouseover="http://www.eh.aqui.com">
  Não chegue perto deste link! </a>
```

## Objeto Anchor

O objeto *Anchor* representa uma âncora fixa. Âncoras podem ser referenciadas como URLs destino localizando partes de um documento. Em HTML, qualquer elemento `<A>` que tiver um atributo `NAME` pode ser usado como âncora:

```
<a name="aqui"></a>
```

A âncora não precisa conter texto. Marca uma posição que pode ser localizada a partir de um vínculo local à página ou não. Dentro da página, pode-se criar um link para a âncora usando:

```
<a href="#aqui">Rolar a página até chegar lá</a>
```

Em páginas externas, o fragmento “#aqui” deve ser acrescentado ao link, logo após o nome do arquivo. No trecho de código abaixo, há dois objetos *Anchor*, destacados em negrito:

```
<BODY>
<h1><b><a name="top"></a></b> Mapa Interativo</h1>
(...)
<p align=center>
<a href="index.html">Volta para Revendedores</a>
<b><a href=" ../index.html" name="hp">Home Page</a></b>
</BODY>
```

Todas as âncoras de uma página estão na propriedade `anchors`, de `document`. Para saber quantos objetos *Anchor* existem em uma página, pode-se usar sua propriedade `length`:

```
numAncoras = document.anchors.length; // numAncoras contem 2
```

O primeiro `<A>` do código é *Anchor* e não é *Link* porque não tem o atributo `HREF`. O segundo é *Link* e não é *Anchor*, porque não têm o atributo `NAME`. O terceiro é ao mesmo tempo um *Link* e um *Anchor* e aparece tanto no vetor `links` como no vetor `anchors`.

Objetos *Anchor* podem ser referenciados pelo nome ou pelo índice do vetor correspondente à ordem em que aparecem no código. As formas abaixo são equivalentes:

```
location.href = "#" + document.anchors[1].name;
location.href = "#" + document.hp.name;
```

Objetos *Anchor* não possuem métodos ou eventos associados. Têm apenas uma propriedade *read-only*.

Propriedade	Descrição
name	Nome da âncora (texto que está no seu atributo <code>NAME</code> do HTML)

## Exercícios

- 7.5 Uma grande página HTML contém um glossário, organizado em ordem alfabética. No início da lista de palavras de cada letra há uma âncora do tipo:

```
<h2><a name="M"></a> M </h2>
```

Escreva um programa JavaScript que construa, no final da página, uma tabela HTML com links para todas as âncoras contidas na página.

- 7.6 Uma página possui 5 vínculos para páginas de um site. São páginas dependentes de browser. Se o browser for Netscape, os vínculos devem apontar para páginas localizadas em um subdiretório `./netscape/`. Se for Microsoft, devem apontar para um subdiretório `./microsoft/`. Se for outro browser, as páginas devem ser encontradas no diretório atual (`./`). Use JavaScript para identificar o browser e alterar as propriedades de *todos os links existentes* para que carreguem as páginas corretas quando o usuário as requisitar.