DHTML - MÓDULO 1: VISÃO GERAL E INTRODUÇÃO

Por Marcelo Camargo

Visão geral

A tecnologia DHTML (Dynamic HTML) foi apresentada inicialmente como um conjunto de inovações ao Internet Explorer 4.0. DHTML disponibiliza para o criador de uma página web a habilidade de criar documentos que interagem com o usuário, sem a necessidade de um processamento do lado do servidor. A tecnologia DHTML nos disponibiliza, entre outras coisas: Animação: através da movimentação de elementos em uma página.

Formatação e posicionamento: através da flexibilidade para formatar elementos HTML, como a alteração dinâmica de cor, tamanho, fonte, posicionamento e outras características. Interação: através de uma linguagem de programação, como o JavaScript, para gerar interação com o usuário.

Iremos através deste tutorial entender como essa tecnologia funciona e explorar as suas mais importantes facetas.

Primeiramente iremos entender o que é DHTML, através de uma introdução a essa tecnologia. Essa introdução é importante para dividir nosso estudo em partes, de forma a explorar cada uma dessas partes em tempos diferentes. Neste módulo iremos responder à pergunta "O que é DHTML?", além de conhecer as várias partes que compõe essa tecnologia.

O que é DHTML?

Muitos pensam que a tecnologia DHTML (Dinamic HTML) é uma linguagem. Errado. DHTML não é uma linguagem, mas sim um conjunto de tecnologias que juntas disponibilizam as ferramentas necessárias para tornar dinâmica a nossa conhecida linguagem HTML. Essas tecnologias são:

- **HTML (Hyper Text Markup Language):** A conhecida linguagem baseada em tags para a construção de páginas web estáticas.
- **CSS (Cascading Style Sheets):** Conhecida como folhas de estilo, essa tecnologia permite controlar a formatação dos diversos elementos que compõem uma página web.
- **CSS Positioning:** Permite a alteração do posicionamento de um elemento da página como textos e imagens dinamicamente, através de uma linguagem client scripting.
- **Client scripting:** Trata-se de um pequeno programa, que será interpretado pelo browser do cliente e não no servidor. Algumas linguagens de scripting que podem ser utilizadas são JavaScript e VBScript.

- DOM (Document Object Model): Trata-se do modelo de objetos (com suas propriedades e métodos) que são expostos ao programador DHTML. Através do envio de mensagens a estes objetos, o programador pode explorar a interatividade com o usuário.

Mas, qual o papel de cada uma dessas tecnologias? Bem, vamos analisar um exemplo simples para entendermos melhor. Mais adiante, nestes tutoriais, iremos entrar em mais detalhes sobre a integração entre essas tecnologias.

Suponha uma página que movimente, de um lado a outro horizontalmente, o texto "Olá Mundo DHTML!", alterando de tempos em tempos sua formatação (cor e tamanho). Para fazer isso, primeiramente precisamos da linguagem HTML para montar a página. Para a formatação do texto, utilizamos CSS que nos possibilita a alteração da cor e do tamanho da fonte. Mas, e para fazer o movimento do texto? Bem, aqui é que entra a parte mais interessante de DHTML, o DOM. Através do DOM poderemos acessar o objeto que controla o texto e alterar as suas propriedades de posicionamento (CSS Positioning). Então podemos mudar as coordenadas X e Y do posicionamento, e com isso movimentar o texto. E como acessamos os objetos do DOM? Isso será possível através da utilização de uma linguagem de scripting como JavaScript ou VBScript.

Características de DHTML
Antes de prosseguirmos, iremos conhecer algumas características da tecnologia DHTML, para não sermos pegos de surpresa no futuro.

- Performance: Qual a performance que pode ser obtida com DHTML? Como vimos, o processamento é realizado localmente, ou seja, no browser do usuário, o que garante boa performance já que não exige o tráfego de informações pela rede durante a interação.

- Compatibilidade: O DHTML não apresenta boa compatibilidade entre os browsers. Na verdade não existe um padrão para o DOM, que é o centro dessa tecnologia. Tanto a Microsoft como a Netscape já suportam esse padrão a partir das versões 4.0 de seus browsers, mas cada uma com seu modelo de objetos. Logo, o código *client scripting* deve ser escrito de acordo com o browser destino, a menos que se faça uso das propriedades protegidas, que são um subconjunto das funcionalidades comuns a ambos os browsers.

- Orientado a objetos: Cada elemento de uma página HTML é visto como um objeto, que pode ser acessado e ter suas propriedades, como cor e posicionamento, alteradas dinamicamente.

Conclusão
A tecnologia DHTML nos abre caminho para uma nova geração de aplicações

web. Através de DHTML é possível ter o total controle sobre os elementos que compões uma página HTML, o que nos fornece grande flexibilidade para interagir com o usuário.

DHTML - MÓDULO 2: EXEMPLOS PRÁTICOS

Visão geral

No módulo anterior estudamos quais são e como se integram as diferentes tecnologias que compõe a DHTML. Neste módulo veremos um exemplo prático de sua aplicação.

Iremos analisar cada parte do código, entendendo os objetos, métodos e propriedades envolvidos.

Esse módulo fornece a base para o entendimento de como utilizar a tecnologia DHTML na prática. A partir disso o leitor será capaz de estudar novos objetos, propriedades e métodos, de forma a criar seus próprios efeitos e aplicações com DHTML.

Exemplo de Menu com DHTML

O exemplo consiste na criação de um efeito de menu para páginas web. Quando o usuário passa o mouse sobre o texto de um item do menu, a formatação do texto do item é alterada da seguinte forma:

adição de efeito "underline" (sublinhado) ao texto.
alteração de cor do texto.
Com a retirada do mouse sobre o texto do item, sua formatação normal é restaurada. Esse efeito é bastante utilizado por diversos sites na Internet. O código apresentado funcionará apenas com o Internet Explorer devido as diferenças existentes no DOM entre navegadores.

Para a implementação deste exemplo, utilizaremos as seguintes tecnologias de DHTML (além da conhecida linguagem HTML):

CSS (Cascading Style Sheets): Conhecida como folhas de estilo, essa tecnologia permite controlar a formatação dos diversos elementos que compõe uma página web.
Client scripting: Trata-se de um pequeno programa, que será interpretado pelo browser do cliente e não no servidor. Algumas linguagens de scripting que podem ser utilizadas são JavaScript e VBScript.
DOM (Document Object Model): Trata-se do modelo de objetos (com suas propriedades e métodos) que são expostos ao programador DHTML. Através do envio de mensagens a estes objetos, o programador pode explorar a interatividade com o usuário.

Nossa página DHTML inicia com a declaração das folhas de estilo. Através das folhas de estilo, criaremos dois estilos que serão utilizados em nosso efeito: o estilo "normallink" para a formatação do texto normal (sem o mouse sobre o texto) e o estilo "ativolink" para a formatação do texto ativo (com o mouse sobre o texto). A sintaxe a seguir deve ser utilizada para essa declaração:

```
<style>
<!-
.normallink {color: #336699; text-decoration: none}
.ativolink {color: #0099ff; text-decoration: underline}
->
</style>
```

Para a declaração de um estilo deve-se utilizar o ponto "." antes de seu nome. Em seguida declarar entre chaves "{}" as propriedades e seus valores.

A propriedade color indica a cor do texto no formato RGB em hexadecimal, ou seja, os dois primeiros dígitos são o valor em hexadecimal da cor vermelho, os dois seguintes da cor verde e os dois últimos da cor azul. A propriedade text-decoration controla a apresentação ou não da linha sublinhando o texto de um link (o padrão na web é apresentar um link sublinhado). As propriedades, métodos e objetos disponíveis para programação são definidos pelo DOM.

O próximo passo é adicionar estes estilos ao nosso menu. O código a seguir complementa nossa página de exemplo e apresenta os menus já com os estilos.

```
<html>
<head>
<style>
<!-
.normallink {color: #336699; text-decoration: none}
.ativolink {color: #0099ff; text-decoration: underline}
->
</style>
<title>Exemplo de menu com DHTML</title>
</head>
<body>
<p>
<a href="menu.htm" class="normallink"
    OnMouseOver="className = 'ativolink';"
    OnMouseOut="className = 'normallink';">Empresa</a>
<a href="menu.htm" class="normallink"
    OnMouseOver="className = 'ativolink';"
    OnMouseOut="className = 'normallink';">Clientes</a>
```

```
<a href="menu.htm" class="normallink"
    OnMouseOver="className = 'ativolink';"
    OnMouseOut="className = 'normallink';">Ajuda</a>
</p>
</body>
</html>
```

Primeiramente definimos a classe a qual o texto pertence através da propriedade class. Dessa forma o texto já aparece com o estilo normallink. Depois fazemos a ligação entre os eventos OnMouseOver (mouse sobre o texto) e OnMouseOut (mouse fora do texto) com a execução de códigos em JavaScript que alteram a propriedade classname dos elementos.

Dessa forma quando o usuário passa o mouse sobre o texto, ocorre a ativação do evento OnMouseOver que dispara a execução do código em JavaScript "classname = 'ativolink' " alterando o estilo do texto para a formatação definida. Quando da saída do mouse sobre o texto, o evento OnMouseOut é executado e o texto volta a seu estilo normal. Realizamos esse mesmo procedimento para os três textos de nosso menu: Empresa, Clientes e Ajuda.

Existem outras formas de obter o mesmo resultado, sem a necessidade de escrever para cada item de menu os eventos OnMouseOver e OnMouseOut, como a utilização do modificador hover na declaração das folhas de estilo.

Para visualizar essa página em execução, salve-a num diretório temporário com o nome menu.htm e abra-a a partir do Internet Explorer.

Conclusão

Agora que já criamos nosso primeiro efeito com DHTML e conhecemos na prática o relacionamento entre suas tecnologias, veremos no próximo módulo mais detalhes sobre o DOM, seus objetos, propriedades e métodos disponíveis para programação.

DHTML - Módulo 3: Document Object Model

Visão geral

No módulo anterior estudamos um exemplo prático de como fazer um efeito de menu para páginas Web. Neste módulo iremos conhecer em mais detalhes o Document Object Model (DOM - Modelo de Documento Objeto).

O DOM é o mecanismo que permite a programação DHTML. O DOM é modelo de objetos expostos pelo navegador e acessível a partir de uma linguagem de client-

scripting (linguagem cujo código é executado no navegador do visitante), como JavaScript ou VBScript.

O Object Model já existia em versões anteriores dos navegadores atuais, como o Microsoft Internet Explorer 3.x e Netscape 3.x, porém numa forma bem reduzida. A sua nova versão, que vem com o Internet Explorer versões 4 e 5, permite acesso a todos os elementos HTML de uma página. Isso significa que cada elemento HTML de uma página pode ter um script associado, o qual interage com o usuário ou muda o conteúdo da página dinamicamente, sem a necessidade de um novo acesso ao servidor Web.

O DOM possui um Event Model (Modelo de Evento) que permite associar eventos do usuário a client-scripts programados numa página Web. O Event Model permite que um documento reaja a uma ação (evento) do usuário sobre um elemento HTML, como movimentar o mouse sobre um determinado elemento, pressionar uma tecla ou entrar informação numa caixa de texto de um formulário. Cada evento pode disparar um client-script que pode utilizar o DOM para alterar as características de elementos HTML e até mesmo alterar o conteúdo da página, adicionando ou removendo partes do documento sem acesso ao servidor Web.

Devido aos client-scripts serem executados diretamente pelo navegador do usuário, isso implica num ganho de performance e velocidade.

Acessando elementos HTML com Script

Cada elemento HTML representa um objeto no Document Object Model, o qual tem seu próprio conjunto de propriedades, métodos e eventos. Entretanto para escrever scripts o programador precisa entender como interagir com esses objetos. O modelo de objetos é montado quando o navegador recebe as informações da página HTML e as apresenta ao usuário. Nesse momento os objetos são dispostos numa hierarquia, onde existem pais e filhos. Esses elementos são organizados em coleções de objetos relacionados. As coleções mais importantes são all e children. Vejamos o exemplo a seguir:

```
<HTML>
<BODY>
<DIV>
<P>Algum texto em um <B>paragraph</B>
<IMG id=imagem1 src="imagem.gif">
</DIV>
<IMG id=imagem2 src="imagem.gif">
</BODY>
</HTML>
```

Neste exemplo o objeto DIV contém (é pai) o objeto P e o objeto IMG de nome imagem1. O objeto IMG de nome imagem2 é filho do objeto BODY. E todos os objetos são filhos do objeto HTML. Cada objeto possui as coleções all e children.

No exemplo, a coleção all do objeto DIV contém os objetos P, B e IMG de nome imagem1, porém, a sua coleção children contém apenas os objetos P e IMG de nome imagem1.

Além das coleções para cada elemento HTML, este documento HTML possui também suas coleções. A coleção all do objeto document possui todos os elementos da página e é a forma primária de acesso a todos os objetos.

Para que seja mais fácil se acessar elementos HTML a partir de scripts, deve-se dar um nome a eles. Em nosso exemplo cada imagem tem um nome diferente. Especificamos o nome de um elemento HTML através do comando ID dentro da declaração desse elemento. Podemos então acessar um elemento nomeado da seguinte forma:

document.all.nome_do_elemento.propriedade_ou_método

Eventos

Clicar em um botão, mover o mouse sobre uma parte do documento, selecionar texto... todas essas ações disparam eventos do DOM, que podem ser programados através de client-scripting. Outros navegadores já possuíam eventos - como o Internet Explorer 3.x e Netscape 3.x. Porém esses eventos foram largamente expandidos no Internet Explorer 4. Na verdade, todos os elementos HTML disparam eventos nessa versão do navegador. Isso disponibiliza ao programador uma enorme flexibilidade para interagir com o usuário.

A lista seguinte representa o conjunto de eventos mais comuns que os elementos HTML possuem:

Eventos de mouse	**Gerados quando o usuário:**
onmouseover	Move o ponteiro do mouse para dentro da área de um elemento.
onmouseout	Move o ponteiro do mouse para fora da área de um elemento.
onmousedown	Pressiona qualquer um dos botões do mouse.
onmouseup	Libera qualquer um dos botões do mouse.
onmousemove	Move o mouse dentro da área de um elemento.
onclick	Clica o botão esquerdo do mouse sobre um elemento.
ondblclick	Efetua um duplo clique com o botão esquerdo do mouse sobre um elemento.

Eventos de teclado	Gerados quando o usuário:
onkeypress	Pressiona e solta uma tecla (o ciclo completo).
onkeydown	Pressiona uma tecla (ainda com a tecla abaixada).
onkeyup	Solta uma tecla.

Quando um evento é disparado para um elemento HTML, caso este possua ou não um manipulador para tratá-lo, ele subirá também para o seu objeto pai na hierarquia. Este evento vai subindo até chegar no objeto document que está no topo da hierarquia. Isso é chamado de Event Bubbling. Para interromper a subida de um evento para um objeto pai, o objeto originador do evento deve explicitamente cancelá-lo em seu manipulador de evento.

Conclusão

O Document Object Model representa o coração da tecnologia DHTML. Através dele interagimos com o navegador e com o usuário. Veremos no próximo módulo mais alguns exemplos da aplicação prática dessa tecnologia.

DHTML - MÓDULO 4: UM EXEMPLO DE TEXTO DINÂMICO

Visão geral

No módulo anterior estudamos o **Document Object Model (DOM)** do Internet Explorer. Foram apresentados vários eventos disponíveis para programação e também como enviar mensagens para os objetos do DOM. Neste módulo veremos um exemplo prático de como utilizar alguns dos eventos apresentados e também de chamadas de propriedades de alguns objetos.

Como funciona o texto dinâmico

O exemplo deste módulo apresenta um texto que terá suas propriedades alteradas dinamicamente através da interação do usuário. Iremos alterar as propriedaes de cor, visibilidade e movimentação do texto na página. Como sabemos, cada elemento que colocamos em uma página HTML pode ser acessado através do DOM.

Para simplificar a movimentação do texto, utilizaremos uma técnica bastante usual: colocar o texto dentro de um container **DIV**. O DIV, assim como o **SPAN**,

são elementos HTML que possuem a propriedade de serem **containers** (ou seja, eles mantém vários elementos agrupados). Colocaremos nosso texto dentro de um elemento DIV e então iremos alterar as suas propriedades para realizar as tarefas desejadas.

A diferença entre o DIV e o SPAN é que o DIV força uma quebra de linha ao seu final enquanto o SPAN permite vários elementos na mesma linha. Ao alterarmos as propriedades de um elemento DIV ou SPAN, estas alterações se refletem em todos os elementos HTML que estes contém.

Alterando as propriedades do elemento DIV

O elemento DIV, assim como todos os outros elementos de uma página HTML, estão expostos através da coleção all do objeto **document**. As definições de estilo (**CSS**) estão disponíveis através da propriedade **style** do elemento HTML desejado. Dessa forma, para acessar as propriedades de estilo de um elemento DIV de identificação 'masters', deve-se utilizar:

```
document.all.masters.style.propriedade = valor
```

O exemplo de texto dinâmico

O código HTML a seguir apresenta o exemplo de texto dinâmico, com alteração de propriedades de estilo como cor, visibilidade e movimentação. Para executá-lo, copie o código para um arquivo com extensão .HTM (por exemplo, dhtml.htm) e abra-o em seu navegador. Não é preciso ter instalado um servidor web para poder executar a página pois o código é executado pelo navegador (client-side script):

```
<html>

<head>
<title>Exemplo de Texto Dinâmico</title>

<style>
<!
.ibestmasters {
position:ABSOLUTE;
TOP:0px;
LEFT:140px;
VISIBILITY:visible;
z-index:2;
FONT-SIZE:50px;
COLOR:#00FF00;
}
!>
</style>
</head>
```

```
<script language="javascript">
function ParaDireita() {
document.all.masters.style.left=250;
}
function ParaEsquerda() {
document.all.masters.style.left=140;
}
function ParaBaixo() {
document.all.masters.style.top=50;
}
function ParaCima() {
document.all.masters.style.top=0;
}
</script>

<body>
<table>
<tr>
<td height="100"><div ID="masters"
CLASS="ibestmasters"><p>iBest.Masters</p>
</div></td>
</tr>
</table>

<form>
<p><input type="button" value="Azul"
onclick="document.all.masters.style.color='0000FF'"> <input
type="button" value="Verde"
onclick="document.all.masters.style.color='00FF00'"><br>
<input type="button" value="Ocultar"
onclick="document.all.masters.style.visibility='hidden'"> <input type="button"
value="Exibir" onclick="document.all.masters.style.visibility='visible'"><br>
<input type="button" value="Esquerda" onclick="ParaEsquerda();"><input
type="button"
value="Direita" onclick="ParaDireita();"><br>
<input type="button" value="Para Cima" onclick="ParaCima();"><input
type="button" value="Para Baixo" onclick="ParaBaixo();"></p>
</form>
</body>
</html>
```

O evento OnClick

Neste exemplo utilizamos o evento **OnClick**. Como vimos no módulo anterior, esse evento é disparado quando o usuário pressiona o botão esquerdo do mouse sobre um elemento HTML. Mas para que isso funcione precisamos ligar esse

evento à execução de um código em **JavaScript**.

Para que isso ococrra, em nosso exemplo colocamos o evento **OnClick()** associado a cada botão da página. Dessa forma quando o usuário pressiona um dos botões da página, o evento associado ao botão pressionado é disparado e o código **JavaScript** adequado executado.

O seguinte código exemplifica essa ligação:

```
<input type="button" value="Para Cima" onclick="ParaCima();">
```

Conclusão

Através da tecnologia DHTML é possível realizar a alteração das propriedades de qualquer elemento de uma página. A utilização de containers como os elementos DIV e SPAN fornece mais flexibilidade e facilidade de implementação de códigos que devem agir sobre vários elementos HTML simultaneamente.

Para mais informações...

Para informações adicionais, consulte os links a seguir.

Portal do Web Developer no Brasil

Recomendação do W3C para Cascading Style Sheets (CSS)

Tutorial JScript, a implementação do JavaScript pela Microsoft