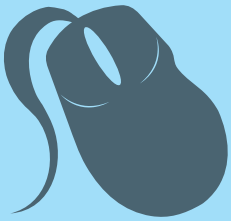


Computação

Desenvolvimento para Web

Joaquim Celestino Júnior
Robério Gomes Patricio

Ed UECE

Geografia

História

Educação Física

Química

Ciências Biológicas

Artes Plásticas

Computação

Física

Matemática

Pedagogia

Computação

Desenvolvimento para Web

Joaquim Celestino Júnior
Robério Gomes Patricio

2ª edição
Fortaleza-Ceará

Ed UECE

2015

Geografia

História

Educação Física

Química

Ciências Biológicas

Artes Plásticas

Computação

Física

Matemática

Pedagogia

Editora Filiada à
ABEU
ASSOCIAÇÃO BRASILEIRA DAS EDITORAS UNIVERSITÁRIAS

Dados Internacionais de Catalogação na Publicação
Sistema de Bibliotecas
Biblioteca Central Prof. Antônio Martins Filho
Thelma Marylanda Silva de Melo – CRB-3 / 623
Bibliotecária

C392d Celestino Júnior, Joaquim.
Desenvolvimento para web / Joaquim Celestino Júnior, Robério Gomes Patrício. – 2. ed. – Fortaleza, CE : EdUECE, 2015.

68 p. : il; 20,0cm x 25,5cm. (Computação)

Inclui referências.

ISBN: 978-85-7826-418-5

1. World wide web (sistema de recuperação da informação). I. Patrício, Robério Gomes. II. Título.

CDD : 005.75

Editora da Universidade Estadual do Ceará – EdUECE
Av. Dr. Silas Munguba, 1700 – Campus do Itaperi – Reitoria – Fortaleza – Ceará
CEP: 60714-903 – Fone: (85) 3101-9893
Internet: www.uece.br – E-mail: eduece@uece.br

Secretaria de Apoio às Tecnologias Educacionais
Fone: (85) 3101-9962

Sumário

Apresentação

São inúmeros os avanços tecnologicos promovidos pelo novo modelo de distribuição de conteudos e serviços pela Internet. A Web tornou-se uma grande "teia de conhecimentos", provida de largo alcançe e altamente democrática quanto ao conteúdos e seu publico alvo.

A partir do momento que percebeu-se o seu grande potencial, a Web trouxe uma nova demanda pela capacidade de gerar conteúdos adequados e adaptados aos seus protocolos, restrições e características. Um grande conjunto de tecnologias e tentativas de padronização surgiram, destancando-se dentre elas o HTML, CSS, Javascript e as linguagens de programação adaptadas ao modelo da Web. Partindo de cenários bastante elementares, com páginas contendo apenas texto simples com algumas referências e links para outros documentos, os web sites evoluiram em seus conteúdos e na experiência promovida ao usuário. Hoje, os usuários são capazes de ter páginas de alta riqueza de conteúdos e interações, com funcionalidades nunca antes previstas para um ambiente inicialmente tão elementar e pobre em interatividade.

Esse livro fala sobre a Internet a algumas de suas tecnologias, explorando conceitos e ferramentas amplamente utilizadas na construção de sites e aplicativos que executam na Internet. Aqui a linguagem HTML é abordada como a estrutura básica para a criação de páginas para a Web. Com a adição da linguagem de estilo CSS é possível conferir maior riqueza visual e interativa a essas páginas e, aliada a tecnologias como Javascript e linguagens de script que executam no lado servidor, é possível gerar conteúdos dinâmicos, os quais podem ser aplicados nos mais diferentes contextos, desde o comercio eletrônico ao mercado do entretenimento.

Os Autores

Capítulo 1

Histórico e Fundamentos

Objetivo

- Contextualizar o aluno quanto aos conceitos e termos relacionados à Internet, em particular com a Web, dando a este uma oportunidade de torna-se familiar com algumas das tecnologias aqui abordadas.

Introdução

Esse capítulo foi desenvolvido para contextualizar o aluno quanto aos conceitos e termos relacionados à Internet, em particular com a Web. Caso você não se sinta familiar com alguns dos termos e tecnologias abordados nesse texto, ou classifica a si mesmo como apenas mais um consumidor de conteúdos e serviços da Internet, os tópicos a seguir serão cruciais para o entendimento do restante livro.

Os principais tópicos cobertos nesse capítulo são:

- A Internet
- World Wide Web
- Scripting no lado do Cliente
- Programação no lado Servidor

1. A Internet

A Internet é basicamente uma coleção de computadores, todos conectados em conjunto a uma grande rede. Sua topologia provê um mecanismo de acesso e entrega da informação de modo que fragmentos desta possam tomar múltiplas rotas de um ponto a outro. Essas múltiplas rotas tornam a rede incrivelmente robusta – se uma parte falha, a informação pode ainda viajar por outras rotas alternativas.

Essa informação tipicamente toma a forma de:

- Pedidos por informações ou serviços
- Respostas contendo a informação ou resultado de um serviço
- Respostas informativas, isto é, as razões por que o serviço ou informação não pode ser entregue

A informação é entregue em um modelo Cliente/Servidor; em outras palavras, clientes(usuários) conectam-se a servidores na Internet e usam seus recursos.

Em um cenário real, muitos Clientes podem conectar-se a um dado Servidor ao mesmo tempo, concorrentemente – e causar uma sobrecarga em sua capacidade caso este logo alcance seus limites de poder de processamento. Essa capacidade é medida de duas maneiras – a potência do Servidor e a quantidade de recursos de rede disponíveis para ele. Esses fatores irão afetar o número de usuários que podem conectar-se ao mesmo tempo sendo ainda assim possível a entrega dos serviços requisitados.

Em seus primeiros dias a Internet foi impulsionada pelo poder de processamento de servidores menos capacitados que os disponíveis atualmente. Nessa época a maioria dos conteúdos disponíveis eram somente em formato de texto. Os servidores não eram capazes de entregar nada mais complexo que isso, e as redes interconectadas não eram rápidas nem confiáveis suficientemente para permitir a transferência de algo mais complexo que texto puro.

Um dos primeiros tipos de aplicação para a Internet disponível para o público em geral foi o E-mail. Isso trouxe uma revolução nas possibilidades de comunicação: era uma solução barata que apresentava o imediatismo de uma chamada telefônica, mas se dava de forma assíncrona, tal como uma carta.

Outros serviços de interface de texto incluíam o Gopher [1], que oferecia menus hierárquicos de informação. Estes foram acessados através de uma interface somente texto, chamado de cliente Telnet [2], sendo considerado como o precursor do browser. Telnet, neste caso, é o protocolo usado para a comunicação entre o cliente e o servidor.

Isso permitiu um certo nível de interatividade, mas com tempos de resposta lentos. Isso ficou conhecido como Lag entre os primeiros usuários da Internet. Lag é o tempo decorrente entre a solicitação e a resposta.

Os gargalos de desempenho de rede também se refletiram no tempo que levava para realizar o download de arquivos de imagens, documentos e software.

Como o desempenho da rede aumentou, juntamente com o poder de computação em geral, a primeira onda de inovações trouxe o conceito de hipertexto. Hipertexto(do termo em inglês hipertext), é uma forma de vincular documentos em conjunto, utilizando referências que contenham a localização de outros documentos e recursos binários.

Logo veio a idéia de incorporar recursos em documentos. O programa que foi usado para exibir estas páginas tornou-se conhecido como um browser; nessa primeira onda de aplicações surgiu o NCSA Mosaic [3], que mais tarde se tornou o Netscape Navigator.

O conceito de hipertexto transformou-se em HTML (HyperText Markup Language, também conhecida na época como HyperText Meta Language), um padrão aberto, usada para troca de informações. Como uma linguagem

de marcação(ver Figura 1), HTML fornecia os detalhes reais de como a página deve parecer para o browser(Figura 2).

```
<html>
    <head>
        <title>Meu primeiro HTML</title>
    </head>

    <body>
        <h1>Iniciando com HTML!</h1>
    </body>
</html>
```

Figura 1 - Exemplo de código em HTML

Figura 2 – Página HTML exibida no browser

Os documentos originais são codificados em texto simples. Quaisquer elementos gráficos tem que ser baixados em tempo de execução; esses recursos são referenciados no documento e exibidos.

O HTML também criou a possibilidade de publicar informações a um público muito grande nas mãos dos usuários. Esta nova liberdade deu origem a uma "web", uma teia em inglês, de documentos e outros recursos, todos conectados uns aos outros, daí o termo World Wide Web, ou WWW.

2. World Wide Web

A World Wide Web (WWW ou simplesmente Web) é uma coleção de recursos vinculados. Documentos (chamados de páginas), imagens, vídeos, applets interativos (mini aplicações) e música (MP3, por exemplo) são apenas alguns dos recursos disponíveis para os surfistas ou navegantes. No entanto, ela não funciona por mágica, e é útil ter um pouco de conhecimento sobre as tecnologias subjacentes para entender melhor como programação na Web funciona na prática.

3. Servidores e Protocolos

A Web ainda é um ambiente cliente/servidor, com base em pedidos e respostas. Essas conversas entre o navegador(o cliente) e o servidor precisavam de uma estrutura padrão para a comunicação, e alguns protocolos foram estabelecidos para fornecer uma forma padronizada de troca de informações.

Os dois protocolos mais frequentemente usados na internet são:

- HTTP (HyperText Transfer Protocol) - Protocolo de Transferência de Hipertexto: é um protocolo de comunicação utilizado para sistemas de informação de hipermedia distribuídos e colaborativos. É amplamente utilizado para recuperação e envio de dados na comunicação com sites da web.
- FTP(File Transfer Protocol) - Protocolo de Transferência de Arquivos): é um protocolo padrão de rede usado como forma bastante rápida e versátil de transferir arquivos, sendo uma das mais usadas na Internet. Há também um padrão, MIME (Multipurpose Internet Mail Extensions), para a troca de e-mail, o qual baseia-se em uma especificação para o formato de anexos de e-mail que não são textos, permitindo que o anexo seja enviado pela Internet. Por meio do MIME é possível a um aplicativo cliente de e-mail ou Browser enviar e receber anexos como planilhas e arquivos de áudio, vídeo e gráficos através de e-mails na Internet.

Todos esses protocolos prevêem também um conjunto de respostas padronizadas que permitem que o navegador exiba as informações adequadas nos casos que as requisições não podem ser atendidas.

4. Motores de busca

Dado o grande volume de documentos e artefatos na Web, tornou-se imperativo que os usuários tivessem uma maneira de realizar buscas diante do que estava disponível. Surgiu daí a ne cessidade de algum tipo de serviço capaz de indexação e recuperação de informações com base em seu conteúdo. Estes ficaram conhecidos como motores de busca, também conhecidos como sites de busca.

Um motor de busca é um serviço baseado na Web que permite aos consumidores localizar artefatos ou recursos na Internet como um todo. O usuário de hoje estará familiarizado com nomes como Google [4] , Yahoo [5], Bing [6], Altavista [7], e assim por diante, os quais são exemplos de motores de busca.

Dado o volume de informações disponíveis na Web, os mecanismos de busca tornaram-se uma parte vital da relação produtor-consumidor. Motores de busca são o único modo pelo qual os surfistas (os consumidores) podem muitas vezes localizar a informação que lhes interessa.

Idealmente, você pode desejar que essas pesquisas aconteçam em tempo real, mas desde que isso não é realmente viável, os resultados são baseados em pesquisas sobre cópias pré-indexadas. Cada sessão de indexação cria um banco de dados que pode ser pesquisado sob demanda, com páginas retornadas que se encaixam à consulta feita pelo surfista.

5. Scripting no lado do Cliente

A programação do lado cliente sofreu e promoveu também uma grande revolução na Internet. Desde os Applets Java[8], passando por animações em Adobe Flash[9] até chegar ao tão falado atualmente HTML5[10] foi bastante longo o caminho. A idéia era trazer para os browser parte do processamento necessário para a execução de algumas ações, livrando o lado servidor de tarefas que envolviam exibição, formatação ou animações em cima dos dados enviados aos browsers.

A seguir são listadas algumas das tecnologias utilizadas para criação de páginas dinâmicas e animações na Web usando programação no lado do cliente:

- Applets Java
- DHTML
- Adobe Flash
- Adobe Flex
- SVG
- HTML 5

6. Programação no lado Servidor

A programação do lado servidor é quem impulsiona a maior parte dos sites na internet atualmente. Desde um servidor HTTP (como Apache ou IIS), que lida com as solicitações e entrega de dados, até as várias aplicações interativas (como Facebook [11], MySpace [12], YouTube [13]), sem a existência de alguma lógica sendo executada no lado servidor nenhum destes serviços seria possível.

A fim de criar Web sites mais interativos, dinâmicos e personalizados foi necessário disponibilizar formas de programação no lado do servidor. Assim, as primeiras páginas web que eram baseadas apenas em texto estático deram lugar a folhas de estilo e scripts do lado do cliente, os quais ajudaram a tornar tais páginas mais sofisticadas visualmente e ricas em funcionalidades, usabilidade e interatividade.

Com o avanço dessas tecnologias, os programadores de sites da Web sentiram a necessidade de dar um passo adiante, criando mecanismos que ajudassem a gerar páginas dinamicamente. Logo os programadores começaram a criar pequenos programas que foram projetados para rodar em segundo plano e capazes de gerar conteúdo em formato HTML. Esses programas executavam nativamente nos servidores, utilizando-se de recursos dos sistemas operacionais subjacentes, como aplicações capazes de realizar a interface com o servidor Web, interagindo com os bancos de dados e provendo as funcionalidades necessárias para a apresentação de conteúdo.

Esta acabou não sendo uma boa solução uma vez que como todos os Sistemas Operacionais são diferentes, havia a necessidade de encontrar uma forma de executar scripts de maneira padronizada, funcionando igualmente em todos os servidores Web e gerando conteúdo para todos os tipos de navegadores. As razões a adoção de linguagens de script eram muitas, mas as motivações principais eram:

As linguagens de script podem ser misturadas com HTML.

Os scripts executados no lado servidor podem ter acesso aos serviços oferecidos pelos próprios servidores Web, como autenticação, acesso a Bancos de Dados, dentre outros.

Para reforçar a viabilidade, as linguagens de script são interpretadas. Isto significa que o provedor de uma linguagem pode distribuir um programa interpretador, construído em várias versões de plataforma, como Windows, Linux, MacOS, e assim por diante, e assim todos os usuários da linguagem podem trocar scripts sem ter que escrevê-los especificamente para cada plataforma. Isso confere às linguagens de script a vantagem da portabilidade do código produzido a ser executado no lado servidor.

Hoje, para facilitar a vida dos desenvolvedores e publicadores de conteúdos, existem os CMS (ou Content Management Systems), que permitem criar sistemas de entrega de documentos complexos de uma forma bem amigável. Blogger [14] e WordPress[15] é um exemplo de uma implementação bastante simples de CMS. Mais CMS mais complexos permitem a categorização de documentos, a publicação, controle de versão, criação de menus dinâmicos, e outras características.

Com essa capacidade de criar e compartilhar informações de forma mais dinâmica e fácil, o conceito da Web 2.0 surgiu e cresceu e assim, graças à integração de tecnologias de lado cliente e servidor através da Internet os usuários têm acesso a conteúdos cada vez mais dinâmicos e alinhados aos seus interesses e expectativas.

Atividades de avaliação

1. Pesquise sobre Applets Java, Adoble Flash, JavaScript e HTML 5, criando um quadro comparativo sobre essas tecnologias, destacando a data de criação, pontos fortes e pontos fracos de cada um destes.
2. Pesquise sobre CGI, ASP, APS .NET, linguagem PHP e Java, criando um quadro comparativo sobre essas tecnologias, destacando a data de criação, pontos fortes e pontos fracos de cada um destes.

3. Pesquise sobre 5 diferentes motores de busca atualmente encontrados na Internet, especificando a data de criação, fundadores e que empresa atualmente é responsável por cada um destes.

Referências

[1] Gopher, disponível em: http://pt.wikipedia.org/wiki/Gopher

[2] Telnet, disponível em: http://pt.wikipedia.org/wiki/Telnet

[3] NCSA Mosaic, disponível em: http://pt.wikipedia.org/wiki/Mosaic

[4] Google, disponível em: http://www.google.com

[5] Yahoo, disponível em: http://www.yahoo.com

[6] Bing, disponível em: http://www.bing.com

[7] Altavista, disponível em: http://www.altavista.com.br

[8] Applets, disponível em: http://pt.wikipedia.org/wiki/Applet

[9] Adobe Flash, disponível em: http://www.adobe.com/br/products/flash.html

[10] HTML5, disponível em: http://pt.wikipedia.org/wiki/HTML5

[11] Facebook, disponível em: http://www.facebook.com

[12] MySpace, disponível em: http://myspace.com

[13] YouTube, disponível em: http://www.youtube.com

[14] Blogger, disponível em: http://www.blogger.com/

[15] WordPress, disponível em: http://wordpress.com/

Capítulo 2

Noções Básicas de HTML

Objetivo

- Dar ao aluno a oportunidade de encontrar exemplos e atividades que o ajudarão a compreender melhor o processo de criação de um web site usando a linguagem HTML.

Introdução

Esse capítulo apresenta os fundamentos da linguagem HTML[1], sua sintaxe e os elementos básicos com os quais é possível iniciar o processo de criação de páginas para a web. Aqui o aluno pode encontrar exemplos e atividades que o ajudarão a compreender melhor o processo de criação de um web site, possibilitando uma visão prática desse processo.

Os principais tópicos cobertos nesse capítulo são:

- O que é HTML
- Estrutura básica de um documento HTML
- Tags básicas para formatação de fonte e texto
- Imagens de fundo e cores de fundo
- Tabelas
- Elementos de formulário

1. O que é HTML

Criada como um linguagem de marcação, o HTML(HyperText Markup Language, que significa Linguagem de Marcação de Hipertexto)[1, 2, 3], tem como objetivo ajudar na definição da estrutura de documentos na Web. Trata-se de uma linguagem composta por TAGS (ou rótulos, elementos entre parênteses angulares < e >), que são usadas no inicio e no final do texto que vai ser envolvido e afetado pelo código.

Uma tag de inicio é por exemplo a tag que é usada para colocar uma porção de texto em negrito, dando-lhe destaque. A tag de fechamento correspondente será que deve ser usada ao final desse texto que se deseja formatar. Assim, quando se deseja colocar em negrito a palavra "Olá!" da frase abaixo, então o código HTML fica:

```
<b>Olá!</b> O meu nome é Carlos.
```

e o resultado visualmente será:

Olá! O meu nome é Carlos.

Uma tag é composta por comandos, atributos e valores. Os atributos modificam os comportamentos padrões dos comandos e os valores especificam com detalhes essa mudança. Veja o exemplo a seguir:

```
<hr color=”red” />
```

O código acima estabelece que seja criada uma linha horizontal de cor vermelha:

hr = elemento que desenha a linha horizontal

color = atributo que define a cor da linha

red = valor do atributo color, que especifica a cor da linha a ser desenhada

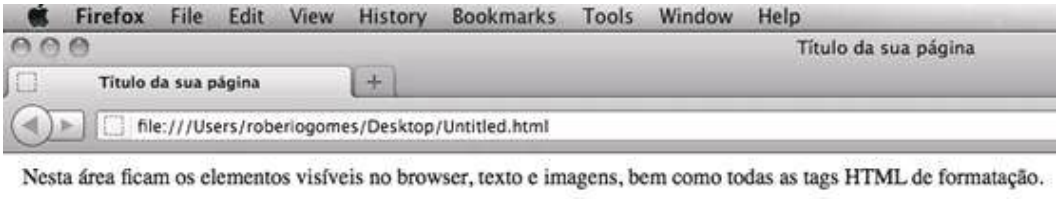
Estrutura Básica de um Documento HTML

A construção de um documento HTML inicia-se pela definição do layout básico deste. Observando o exemplo a seguir é fácil perceber a estrutura definida pelas tags HTML nesse documento:

```
<html>
<head>
<title>Título da sua página</title>
<meta name=”description” content=”Breve descrição da sua página”>
<meta name=”keywords” content=”Palavras chave da sua página”>
</head>
<body>

Nesta área ficam os elementos visíveis no browser, texto e imagens, bem
como todas as tags HTML de formatação.

</body>
</html>
```

Prática 01 – Meu Primeiro HTML

Antes de iniciar uma descrição mais aprofundada do comportamento de cada elemento no exemplo anterior, digite o código HTML acima usando um editor de textos simples(Bloco de Notas, por exemplo) e salve o arquivo com o nome “exemplo-01.html“ em uma pasta de sua preferência. Em seguida clique no arquivo para exibi-lo no seu navegador e veja o resultado final.



As tags <html> e </html> dizem ao browser onde começa e onde termina o código HTML.

As tags <title> e </title> define qual é o título da página. O título pode ser visto no canto superior esquerdo do seu browser. O texto que for definido entre estas tags é também o texto que é usado como título pelos motores de busca quando apresentam as páginas nos resultados de uma pesquisa.

A tag <meta name="description" ...> contem detalhes que podem ser úteis para os motores de busca. Eles podem usar o conteúdo dessa tag para fornecer uma descrição da página quando esta é apresentada nas pesquisas, podendo em alguns casos também ser usada uma parte aleatória do conteúdo do elemento <body> da página.

A tag <meta name="keywords" ...> pode também ser usada pelos motores de busca na indexação da página, o que ajuda nas pesquisas. É comum colocar as palavras-chave separadas por vírgulas no interior desta tag.

O conteúdo da página deve ser colocado entre as tags <body> e </body>, sendo este o que é visualizado no browser.

Uma vez inseridos os códigos HTML e o conteúdo pretendido, deve-se salvar o arquivo com a extensão «.html». Para visualizar a página no browser basta clicar duplamente sobre o arquivo.

2. Tags Básicas para formatação de Fonte e Texto

Prática 02 – Formatação de Fonte e Texto

Crie um novo documento HTML e execute os seguintes exemplos, fazendo experimentos com as seguintes tags que serão demonstradas.

Novo Parágrafo: Inicia um novo parágrafo inserindo uma linha em branco entre o novo parágrafo e o anterior.

texto

Exemplo: Esse é apenas um simples exemplo de como criar um parágrafo!

Alinhar texto à esquerda:

texto

Exemplo: Esse é apenas um simples exemplo de como criar um parágrafo alinhado à esquerda!

Alinhar texto à direita:

texto

Exemplo: Esse é apenas um simples exemplo de como criar um parágrafo alinhado à direita !

Itálico:

texto

Exemplo: Esse é apenas um simples exemplo de como criar um parágrafo!

Sublinhado:

<u>texto</u>

Exemplo: Esse é <u>apenas</u> um simples exemplo de como criar um parágrafo!

Negrito:

texto

Exemplo: Esse é <u>apenas</u> um simples exemplo de como criar um parágrafo!

Texto Riscado:

<s>texto</s>

Exemplo: Esse é exemplo mostra um texto <s>riscado</s>!

Quebra de linha: Esta tag faz com que o texto seguinte mude para a linha subsequente.

Centralizar:

texto

Alterar cor do texto: Pode-se também utilizar os códigos HEX para definir as mais variadas cores.

texto

Alterar o tamanho: (escolher entre 1 e 7)

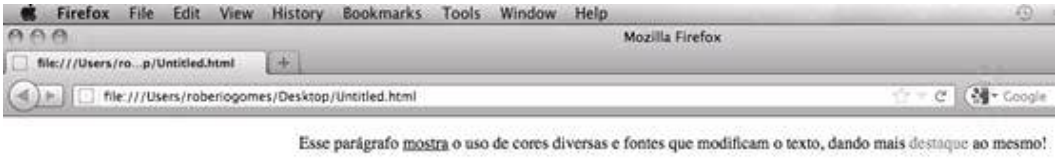
texto

Alterar a fonte:

texto

Prática 03 – Usando formatações de Fonte e Texto

Crie uma nova página HTML de acordo com o seguinte layout apresentado na figura abaixo. Use os elementos vistos acima para elaborar sua página o mais parecida possível.



3. Imagens de Fundo e Cores de Fundo

Cor de Fundo

Para usar cores de fundo deve-se procurar uma combinação entre a cor de fundo e a cor do texto que não dificulte a leitura. Em geral as cores claras funcionam bem com uma cor escura para o texto.

Para usar uma cor de fundo na sua página deve-se usar o seguinte código dentro da tag <body>:

<body bgcolor=”blue”>

Pode-se também utilizar os códigos HEX para definir as cores que combinem melhor.

3.1 Inserir uma Imagem de Fundo

Caso se deseje definir uma imagem de fundo deve-se usar o seguinte código:

<body background=”fotos/imagem.jpg”>

É importante lembrar-se que «fotos/imagem.jpg» é a localização do arquivo da imagem. Trata-se de uma URL relativa que aponta para a pasta «fotos» que deve estar no seu site. É possível também usar um URL absoluta, que pode apontar para o seu site ou então para outra imagem localizada em algum lugar na internet. Um exemplo do código da imagem de fundo com URL absoluta é:

<body background=»http://www.meudominio.com/fotos/imagem.jpg»>

3.2 Inserindo Imagens

A tag para inserir imagens é <img src=”imagem.jpg”>. O atributo “src” especifica a URL do arquivo da imagem. Além disso, o atributo “alt” deve ser usado para quando se deseja que um texto apareça todas as vezes que o mouse estiver sobre a imagem. Este mesmo texto também aparece como texto descritivo quando a exibição das imagens não está ativa ou não é possível carregar a imagem. A tag fica:

<img src=”imagem.jpg” alt=”Evento de Natal 2011 “>

3.3 Definir Altura e Largura

É possível definir a largura de uma imagem usando-se o atributo «width» e para altura o «height». Para isso, os valores desses atributos pode ser definidos em pixels ou em percentagem:

<img src=»imagem.jpg» width=»200» height=»100»>

3.4 Inserindo Imagens em Hiperlinks

Para transformar imagens em links clicáveis que possam redirecionar os usuários para outras páginas, basta usar uma combinação das duas tags HTML para imagem e para links:

<img src="fotos/imagem.jpg">

3.5 Inserir Hiperlinks

A tag para se inserir hiperlinks é Nome do Link. O resultado deste código é:

Nome do Link

3.6 Abrir links em Novas Janelas

Quando não se deseja que o usuário abandone completamente o site ao clicar nos links, é possível definir os links para abrirem numa nova janela com o atributo «target=›_blank›»:

Nome do Link

3.7 Endereços Absolutos e Relativos

Uma URL (de Uniform Resource Locator), é o endereço dos sites que todos estão habituados a colocar na barra de endereços do navegador de internet. Pode-se usar links para outras páginas usando URLs absolutas ou relativas.

URLs absolutas

Uma URL absoluta contém o caminho completo para a localização do arquivo/recurso do site que se pretende acessar.

Caso o usuário quisesse abrir uma página chamada alencar.html, que se encontrava no interior de uma pasta chamada "escritores", e esta pasta se encontrava na raiz do site, então a URL absoluta seria:

 José de Alencar

É uma boa prática o uso de URLs absolutas em páginas que possam mudar de posição na estrutura de pastas do site. Dessa forma se a página que contém os links mudar de pasta, os links vão continuar funcionando.

URLs relativas

Uma URL relativa indica apenas o caminho a percorrer desde a posição em que a página com o link está até chegar ao recurso desejado.

Se por exemplo a partir da página alencar.html (que se encontra dentro da pasta "escritores") e deseja-se linkar de volta para a página principal (para a página index.html da raiz do site), então a URL relativa seria:

Página Principal

Os dois pontos, seguidos de uma barra, dizem ao browser para subir um nível na hierarquia na estrutura de pastas do site.

A URL relativa permite que o site possa ser testado offline. Desta forma os links vão funcionar se abrirmos as páginas do site a partir do disco.

4. Criando Tabelas

Usando-se tabelas é possível ordenar os conteúdos do site de uma determinada forma, ordenando-se imagens lado a lado, ou em linha, criar colunas para o layout do site, criar uma barra lateral, etc.

4.1 Tags de tabela básicas

São 3 as tags básicas para inserir tabelas em páginas HTML:

- <table> é a tag que inicia e finaliza uma tabela
- <tr> que significa «**t**able **r**ow"(linha de tabela) é a tag que representa uma linha na tabela
- <td> que significa "**t**able **d**ata" (dados da tabela) é a tag que representa uma célula da tabela dentro da linha

Pelo que foi proposto no parágrafo anterior, é fácil perceber que em HTML então não existe o conceito de "linhas" e "colunas" como em uma matriz. Na verdade, o que existem são "linhas" que podem conter "células" contendo dados, mas na prática vai ser comum ouvir e dizer que tem-se "linhas" e "colunas".

O exemplo a seguir mostra como é possível criar uma tabela simples:

```
<table>
<tr>  <td>Célula A</td> <td>Célula B</td> <td>Célula 3</td>  </tr>
<tr>  <td>Célula D</td> <td>Célula E</td> <td>Célula F</td>  </tr>
</table>
```

E o resultado desta tabela é:

Célula A	Célula B	Célula C
Célula D	Célula E	Célula F

Observa-se que a tabela tem 3 linhas e 2 colunas. No código HTML acima, cada <tr> é uma linha. Como no interior de cada linha existem 2 tags <td> isso resulta em 2 células por cada linha, ou 2 colunas na tabela final.

4.2 Definindo contorno para a Tabela

A definição de um contorno para uma tabela é feita ao colocar no atributo «border» a espessura do contorno desejado:

```
<table border=»2»>
<tr>  <td>Célula A</td> <td>Célula B</td> <td>Célula C</td>  </tr>
<tr>  <td>Célula D</td> <td>Célula E</td> <td>Célula F</td>  </tr>
</table>
```

E o resultado é:

Célula A	Célula B	Célula C
Célula D	Célula E	Célula F

4.3 Definindo Espaço que Envolve as Células - CELLSPACING

Caso deseje-se aumentar ou diminuir o espaço no interior da tabela que envolve as células, é necessário usar o atributo “cellspacing”.

```
<table border=”2” cellspacing=”15”>
<tr>  <td>Célula 1</td> <td>Célula 2</td> <td>Célula 3</td>  </tr>
<tr>  <td>Célula 4</td> <td>Célula 5</td> <td>Célula 6</td>  </tr>
</table>
```

assim:

Célula A	Célula B	Célula C
Célula D	Célula E	Célula F

4.4 Definindo espaço no Interior das Células - CELLPADDING

Para aumentar ou diminuir o espaço no interior das células usa-se o atributo “cellspadding”.

```
<table border=”2” cellpadding=”10”>
<tr>  <td>Célula 1</td> <td>Célula 2</td> <td>Célula 3</td>  </tr>
<tr>  <td>Célula 4</td> <td>Célula 5</td> <td>Célula 6</td>  </tr>
</table>
```

ficando:

Célula 1	Célula 2	Célula 3
Célula 4	Célula 5	Célula 6

4.5 Definindo a largura de uma Tabela

Para definir a largura de uma tabela usa-se o atributo “width”, podendo o seu valor ser definido em pixels ou em percentagem. Exemplo:

```
<table border=”2” width=”100%”>
<tr>  <td>Célula 1</td> <td>Célula 2</td> <td>Célula 3</td>  </tr>
<tr>  <td>Célula 4</td> <td>Célula 5</td> <td>Célula 6</td>  </tr>
</table>
```

resultado:

Célula 1	Célula 2	Célula 3
Célula 4	Célula 5	Célula 6

Ao usar-se um valor em pixels, por exemplo 400px, a tabela vai ter uma largura fixa de 400px.

```
<table border=”2” width=”400”>
<tr>  <td>Célula 1</td> <td>Célula 2</td> <td>Célula 3</td>  </tr>
<tr>  <td>Célula 4</td> <td>Célula 5</td> <td>Célula 6</td>  </tr>
</table>
```

o resultado final desta tabela:

Célula 1	Célula 2	Célula 3
Célula 4	Célula 5	Célula 6

4.6 Definindo a largura das Colunas

Caso não se deseje que todas as colunas tenham a mesma largura, é possível definir um atributo de largura dentro da tag <td>. Ao usar percentagens tem-se por exemplo o seguinte código:

```
<table border=”2” width=”400”>
<tr>  <td width=”50%”>Célula 1</td> <td>Célula 2</td> <td>Célula 3</td>  </tr>
<tr>  <td width=”50%”>Célula 4</td> <td>Célula 5</td> <td>Célula 6</td>  </tr>
</table>
```

e o resultado da tabela:

Célula 1	Célula 2	Célula 3
Célula 4	Célula 5	Célula 6

4.7 Definindo a Altura de uma Tabela

É possível também definir uma altura para a tabela adicionando o atributo «height»:

```
<table border=»2» height=»200» width=»400»>
<tr>  <td width=»250»>C 1</td> <td width=»75»>C 2</td> <td width=»75»>C 3</td>  </tr>
<tr>  <td width=»250»>C 4</td> <td width=»75»>C 5</td> <td width=»75»>C 6</td>  </tr>
</table>
```

resultado:

C 1	C 2	C 3
C 4	C 5	C 6

4.8 Alinhando Horizontalmente o Conteúdo das Células

Por padrão o conteúdo das células encontra-se alinhado à esquerda, entretanto é possível também centralizar o conteúdo e alinhar à direita. Para isso usa-se o atributo "align" no interior da tag <td>:

```
<table border="2" width="400">
<tr>  <td width="250" align="center">C 1</td> <td width="75">C 2</td> <td width="75">C 3</td>  </tr>
<tr>  <td width="250" align="center">C 4</td> <td width="75">C 5</td> <td width="75">C 6</td>  </tr>
</table>
```

resultado:

C 1	C 2	C 3
C 4	C 5	C 6

Caso deseje-se alinhar à direita é necessário usar align=»right».

4.9 Alinhando Verticalmente o Conteúdo das Células

Por padrão os dados ficam alinhados verticalmente no meio (middle) da célula, entretanto é possível alinhar o conteúdo para o fundo (bottom) e para o topo (top) da célula. Para isso usa-se o atributo "valign" no interior da tag <td>:

```
<table border="2" height="200" width="400">
<tr>  <td width="250" valign="bottom">C 1</td> <td width="75">C 2</td> <td width="75">C 3</td>  </tr>
<tr>  <td width="250" valign="top">C 4</td> <td width="75">C 5</td> <td width="75">C 6</td>  </tr>
</table>
```

resultado:

C 1	C 2	C 3
C 4	C 5	C 6

Prática 04 – Criando tabelas

Crie um novo arquivo HTML, digite o seguinte código nele e em seguida salve o arquivo em uma pasta de sua preferência. Veja se o resultado final parece com a figura vista logo a seguir:

```
<html>
        <body>
                <table border=”1”>
                        <caption
                        <tr>
                                <th>Turno</th>
                                <th>Segunda</th>
                                <th>Terca</th>
                                <th>Quarta</th>
                                <th>Quinta</th>
                                <th>Sexta</th>
                        </tr>
                        <tr>
                                <td>AB</td>
                                <td>Algoritmos</td>
                                <td>PAA</td>
                                <td>Algoritmos</td>
                                <td>PAA</td>
                                <td>POO</td>
                        </tr>
                        <tr>
                                <td>CD</td>
                                <td>Automatos</td>
                                <td>Web 2.0</td>
                                <td>Automatos</td>
                                <td>Web 2.0</td>
                                <td>POO</td>
                        </tr>
                </table>
        </body>
</html>
```

Turno	Segunda	Terca	Quarta	Quinta	Sexta
AB	Algoritmos	PAA	Algoritmos	PAA	POO
CD	Automatos	Web 2.0	Automatos	Web 2.0	POO

Figure 1 - Exemplo de Tabela

4.10 Definindo Listas

Lista numerada

Para apresentar uma lista de itens numerados usa-se a tag <ol>, iniciais de "ordered list", para a lista, e a tag <li>, iniciais de "list item", para cada item:

```
<ol>
<li>Item 1</li>
<li>Item 2</li>
<li>Item 3</li>
</ol>
```

e o código resultante é:

1. Item 1
2. Item 2
3. Item 3

Lista não numerada

Para apresentar uma lista de itens não numerada usa-se a tag <ul>, iniciais de "unordered list", para a lista, e a tag <li>, iniciais de "list item", para cada item:

```
<ul>
<li>Item 1</li>
<li>Item 2</li>
<li>Item 3</li>
</ul>
```

e o código resultante é:

- Item 1
- Item 2
- Item 3

5. Definindo formulários HTML e seus elementos

Formulários podem ser usados para enviar dados pela web. Muitas vezes são usados como formulários de contato para converter informação inserida por um usuário em um e-mail, ou mesmo para capturar os dados de um usuário que deseja efetivar uma compra em um site de comércio eletrônico. A figura a seguir exibe um formulário usado para envio de e-mail:

Figura 2 - Exemplo de formulário

As tags básicas usadas num formulário HTML são form, input, textarea, select e option.

5.1 Form

A tag form é usada para definir o formulário propriamente dito. Essa tag possui um atributo obrigatório, o action, que especifica o endereço do destino onde os dados do formulário serão processados.

O atributo opcional method diz a forma a ser utilizada pelo formulário no enviado dos dados, podendo ter o valor get (que é o valor padrão) ou post. Esses valores correspondem aos métodos suportados pelo protocolo HTTP

e comportam-se exatamente como estes. É uma boa prática usa-se post que esconde a informação (get envia a informação através da URL, deixando-a totalmente exposta). Exemplo:

```
<form action="enviarEmail.do" method="post"></form>
```

5.2 Input

A tag input é a mais versátil de todas quando o assunto é construção de formulários, podendo apresentar-se de dez formas diferentes, como é possível ver abaixo:

- Text: é a caixa de texto padrão. Ela também pode ter uma atributo value, que define o valor inicial da caixa de texto.
- Password: é similar à **caixa de texto**, mas o que for digitado pelo usuário não vai ser visível.
- Checkbox: é uma **checkbox**, que pode ser marcada e desmarcada pelo usuário.
- Radio: é similar a um checkbox, mas o usuário só pode selecionar um **radio button** em um grupo.
- File: é uma campo que permite procurar e escolher um arquivo no computador, e é usado para permitir ao usuário fazer upload de arquivos.
- Submit: é um botão que quando clicado **envia** o formulário.
- Image: é uma **imagem** que vai enviar as coordenadas do ponto em que o usuário clicou.
- Button: é um **botão** que não faz nada sem código extra adicionado.
- Reset: é um botão que quando clicado **limpa** os campos do formulários para seus valores default.
- Hidden: é um campo que não aparece na tela e é usado para passar informação não exibível ao servidor que processa o formulário.

6. Textarea

A tag textarea define uma caixa de texto grande que requer os atributos **rows** (linhas) e **cols** (colunas) e funciona da seguinte forma:

```
<textarea rows="5" cols="20">O texto fica aqui.</textarea>
```

6.1 Select

A tag select trabalha junto com a tag option pra criar caixas de seleção drop-down. Exemplo:

```
<select>
  <option value="primeira opção">Opção 1</option>
  <option value="segunda opção">Opção 2</option>
  <option value="terceita opção">Opção 3</option>
</select>
```

Todos os campos de formulários precisam de **nomes** pra que possam ser identificados pelo programa responsável por eles. Isso é feito por meio do atributo name. Exemplo de formulário:

```
<form action="algumscript.php" method="post">
<p>Nome:</p>
<p><input type="text" name="nome" value="Seu nome" /></p>
<p>Comentários: </p>
<p><textarea rows="10" cols="20" name="comentarios">Seus comentários</textarea></p>
<p>Você é:</p>
<p><input type="radio" name="vocee" value="homem" />Homem</p>
<p><input type="radio" name="vocee" value="mulher" />Mulher</p>
<p><input type="submit" value="Enviar"/></p>
<p><input type="reset" value="Limpar"/></p>
</form>
```

O formulário acima deve ser exibido como pode ser visto na figura a seguir:

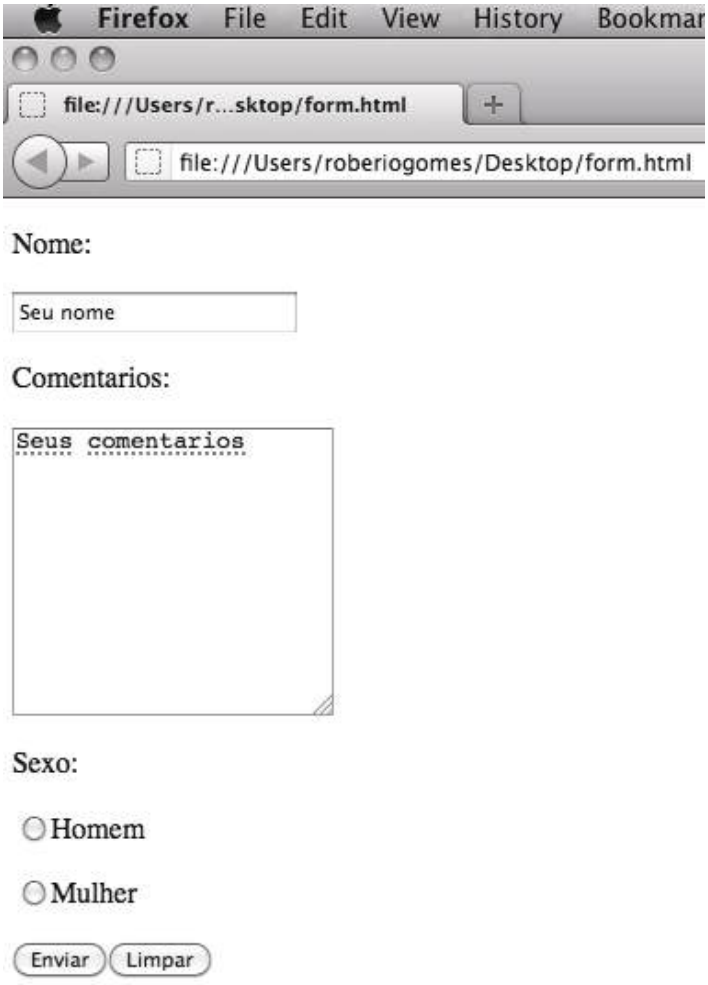


Figura 3 – Formulário de Opinião

Exibindo caracteres Especiais

Alguns caracteres especiais não são exibidos corretamente durante a execução da página no navegador. A tabela a seguir apresenta os códigos que devem ser usados para exibir alguns desses caracteres:

Código	Carácter	Descrição
&lt;	<	Menor que
&gt;	>	Maior que
&trade;	™	Trademark
&amp;	&	Ampersand
&reg;	®	Marca Registada
&copy;	©	Copyright
&dagger;	†	Cruz
&quot;	“	Aspas

»	»	Seta de citação
«	«	Seta de citação
—	–––	Tracejado
°	20°	Grau
Código	**Carácter**	**Descrição**
¼	¼	Um quarto
½	½	Um meio
¾	¾	Três quartos
·	·	Ponto no meio
¡	¡	Ponto exclamação invertido

Para que apareçam um dos símbolos da coluna amarela em destaque, basta adicionar à página HTML o respectivo código.

Atividades de avaliação

1. Pesquise sobre os seguintes tipos de links e forneça exemplos de como cria-los:

   Hiperlinks Âncora ou Marcadores

   Links para E-mails

2. Hiperlinks por definição têm a cor azul. Pesquise e forneça exemplos de como é possível alterar suas cores.
3. Pesquise no site da Adobe Kuller[4] combinações de cores que você posteriormente usaria na construção de sua própria Home Page.
4. Construa usando os elementos HTML visto nesse capítulo sua Home Page, apresentando os seguintes itens:

   a. Dados Cadastrais com foto

   b. Formação escolar

   c. Experiências profissionais

   d. Cursos e Treinamento

5. Usando tabelas, listas enumeradas e outros elementos HTML, crie uma agenda semanal contendo suas tarefas a serem executadas com suas devidas descrições, horários e datas.

Referências

[1] HTML 4.01 Specification - W3C, disponível em: **www.w3.org/TR/html401/**

[2] HTML Tutorial, disponível em: **www.w3schools.com/html/**

[3] HTML5 - W3C Brasil, disponível em : **http://www.w3c.br/Cursos/Curso-HTML5**

[4] Adobe Kuller, disponível em: **http://kuler.adobe.com/**

Capítulo 3

CSS

Objetivo

- Apresentar os elementos básicos usados em CSS que irão ajudar o aluno a construir páginas mais leves, interativas e fáceis de manter.

Introdução

Esse capítulo trata do uso de folhas de estilo CSS para construção e melhoria no design de páginas na Web. Aqui são apresentados os elementos básicos usados em CSS que irão ajudar o aluno a construir páginas mais leves e fáceis de manter, bem como mais atrativas e que estejam de acordo com as novas tendências da Web.

1. O que é CSS?

CSS[1] é um acrônimo para Cascading Style Sheet(termo em inglês que significa Folhas de Estilo Encadeadas), uma linguagem de estilo, simples e bastante útil, usada para definir a apresentação de documentos escritos em uma linguagem de marcação. Uma de suas principais características é sua capacidade de prover a separação entre o formato e o conteúdo de um documento, ajudando na confecção de documentos mais bem estruturados e fáceis de manter.

Quando se faz uso de CSS na construção de um Web site, por exemplo, é possível definir toda a aparência dos elementos visuais que compõem o mesmo a partir do CSS, deixando o código responsável pela formatação completamente livre desse tipo de tarefa. Assim, por apenas alterar o CSS, é possível obter-se uma aparência nova, com fontes, cores e texturas diferenciadas sem alterar a estrutura dos documentos de marcação em si.

O uso de CSS pode ser dar internamente nas páginas de conteúdo como também por meio de arquivos externos. O uso de arquivos externos ganhou muita popularidade na Web, sendo uma prática bastante comum uma vez que representa a possibilidade de modularizar e reaproveitar de forma mais racional os mesmos.

Em sua idéia inicial, o CSS foi criado para resolver um problema inerente ao HTML 4[2] com relação à presença de tags de estilo como <font> e que foram embutidas na grande maioria dos documentos HTML da época, tornando difícil a manutenção destes. Para resolver esse problema, o World Wide Web Consortium (W3C)[3] criou o CSS. A partir do HTML 4.0, toda a formatação pôde ser removida dos documentos HTML e guardada a salvo em arquivos separados.

Por ser capaz de definir como os elementos HTML devem ser exibidos, de uma forma declarativa, descrita em arquivos externos específicos .css, o uso de CSS facilitou em muito a modularização e o reuso de elementos visuais, ajudando também na consistência de estilo entre esses, podendo-se ainda contar com o fato de todos os browsers atualmente suportarem essa linguagem.

2. Sintaxe

Como toda linguagem, CSS apresenta uma sintaxe que deve ser seguida para se obter os resultados desejados quanto a estilo. Uma regra CSS é dividida em duas partes principais, um seletor e uma ou mais declarações[5]:

Figura 1 – Exemplo da Sintaxe CSS

Cada seletor é normalmente um elemento HTML sobre o qual deseja-se aplicar o estilo. Cada declaração consiste de uma um par propriedade e valor, sendo a propriedade o atributo estilo que deseja-se modificar de acordo com um dado valor.

Uma declaração CSS sempre termina com o símbolo de ; e um conjunto de declarações que precisam estar agrupadas devem ser envolvidas por um bloco definido pelos símbolos { e }, conforme pode ser visto no exemplo a seguir:

```
p {color:red;text-align:center;}
```

Figura 2 – CSS modificando conteúdo de um parágrafo

É muito comum colocar uma declaração em cada linha por questões de legibilidade e facilidade de manutenção como pode ser visto no exemplo a seguir:

```
p {
   color:red;
   text-align:center;
}
```

Figura 3 – CSS mais legível.

O exemplo a seguir demonstra como pode ser aplicado o estilo CSS definido acima em uma página HTML simples:

```
1  <html>
2      <head>
3          <style type="text/css">
4              p {
5                  color: red;
6                  text-align: center;
7              }
8          </style>
9      </head>
10
11     <body>
12         <p>Olá Mundo Web!</p>
13         <p>Esse parágrafo está estilizado com CSS.</p>
14     </body>
15 </html>
```

Figura 4 – CSS embutido na página

A seguir é possível visualizar o resultado:

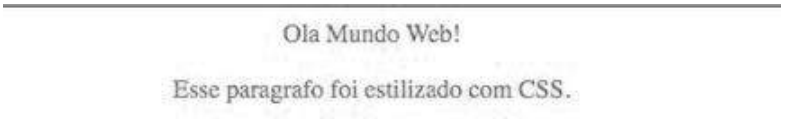


Figura 5 – Resultado do CSS embutido na página

Ao observar o exemplo acima, percebe-se o uso de CSS internamente ao documento HTML (linhas 3 a 8). O estilo definido estabelece que todo elemento do tipo parágrafo, isto é <p>, deve ter seu conteúdo exibido em vermelho e alinhado ao centro.

Ao definir estilos CSS é possível fazer uso de comentários para documentar e esclarecer os detalhes do uso do CSS. Os comentários CSS serão sempre ignorados pelo browser no momento de carregamento da página. Um comentário CSS começa com "/*" e termina como "*/" , como se pode ver a seguir:

```
/* Isso é um comentário. */
p {
   text-align:center;

   /* Esse é outro comentário. */
   color:black;
   font-family:arial;
}
```

Figura 6 – Usando comentário em CSS

3. Usando seletores

3.1 Seletor Universal

Um seletor universal é aquele que está associado a todo e qualquer elemento encontrado no documento HTML. Sua definição se dá pelo uso do caracter * (asterisco), como pode ser visto a seguir:

```
* {color : red;}
```

A regra acima quando executada irá exibir todo o texto da página na cor vermelha.

3.2 Seletor de Elemento

É aquele se está associado a todas as instâncias de um dado elemento na página. Exemplo:

```
p {color : blue;}
```

A regra acima quando executada vai exibir todos os parágrafos da página HTML em cor azul.

3.3 Seletor de Classe

É aquele que está associado a qualquer elemento ao qual tenha sido associado o nome da classe definida em CSS. Com isso, um mesmo estilo pode ser aplicado a vários elementos distintos, desde que nestes exista uma referência à classe CSS. A seguir é demostrado um exemplo de como é definido um seletor de classe, ficando para um momento posterior a demonstração de como aplica-lo:

```
.total {color : green;}
```

Vale observar que para um seletor de classe, é obrigatório o uso do caracter . (ponto) iniciando sua definição.

3.4 Seletor com ID

É aquele que está associado especificamente a um único elemento que contem um dado ID associado a ele como atributo. Para a definição esse seletor é necessário usar o caráter # como é visto a seguir:

```
#score {color: blue;}
```

O browser irá aplicar o estilo definido somente no elemento que apresentar o atributo ID com o valor score.

3.5 Seletores mais Complexos

A seguir são exemplificados cenários onde seletores podem ser combinados de uma maneira mais complexa visando obter resultados mais sofisticados:

```
h1.main {color: blue;}
```

O seletor acima foi definido para associar-se a todos os elemento h1 que estejam em primeiro nível e que usem a classe denominada .main.

```
h1, h2, h3 {color: green;}
```

O seletor acima foi definido para englobar todos os elemento h1, h2 e h3 exibindo-os em cor verde.

4. Integrando nas páginas HTML

Existem três maneiras diferentes de fazer uso de CSS em documentos HTML, a citar:

- Estilos em Linha(Inline Styles): os estilos são embutidos dentro dos elementos da página.
- Estilo embarcado(Embedded Style Sheet): os estilos são definidos na sessão head do documento HTML.
- Estilo Externo(External Style Sheet): os estilos são definidos em um arquivo externo o qual é referenciado a partir do documento HTML.

4.1 Estilos em Linha

Para fazer uso dessa estratégia adicione um atributo style a cada elemento que deseja acrescentar estilo. Exemplo:

```
<p style=''color:green;''> Olá mundo CSS! </p>
<p style=''font-size:9; color:red;text-align:center''> Bye CSS! </p>
```

No exemplo acima os dois parágrafos serão exibidos com estilos diferentes, o primeiro com a fonte na cor verde, o segundo em cor vermelha, com tamanho de fonte 9 e alinhamento de texto ao centro.

4.2 Estilo embarcado

Apesar de ser possível usar CSS em linha, aconselha-se evitar a não em casos muito simples de uso. Uma outra opção seria definir o CSS dentro da própria página mas em um só lugar facilitando o processo de manutenção e reuso de código.

Para usar CSS embarcado use a tag style dentro do elemento head, adicionando a ele as regras desejadas de estilo. Exemplo:

```
<html>
    <head>
      <title>Usando Estilo Embarcado </title>

      <style type=”text/css”>

            h1 {
               color : purple
                  text-decoration : underline;
                text-align : center;
          }

          h2 {
          color: orange;
            background-color: black;
            text-align: center;
          }

          p {
           font-weight: bold;
              font-style: italic;
              color: blue;
          }

    </style>

</head>

<body>
  <h1>Esse H1 tem um estilo e cor “purple”, sublinhado e com alinhamento de texto ao centro. </h1>
    <h2>Esse H2 tem um estilo em cor “orange” com fundo “black” e alinhamento de texto ao centro.</h2>
    <p>
      Esse paragrafo tem estilo em cor “blue” com texto em destaque em negrito e italico.
    </p>
      </body>
</html>
```

A figura a seguir exibe o resultado desse estilo:

Figura 7 – Resultado do uso do CSS

4.3 Estilo Externo

Os mesmos estilos CSS podem ser usados seguindo uma outra estratégia: os estilos devem ser escritos em um arquivo externo ao documento HTML e referenciado por meio da tag style. Com isso também é possível diminuir o tamanho do documento HTML, evitando perda de tempo no carregamento do arquivo. A seguir exemplo de como deve ser referenciado um css externo a partir de um HTML:

<link rel=''stylesheet'' type=''text/css'' href=''ENDEREÇO DO CSS'' />

O mesmo resultado obtido na figura anterior pode se dar usando essa abordagem como pode ser visto no exemplo a seguir com a página HTML e o estilo css declarados em um arquivos separados:

```
<html>
    <head>
        <title>Usando Estilo Embarcado </title>
        <link rel=”stylesheet” type=”text/css” href=”estilo.css” />
    </head>

    <body>
        <h1>Esse H1 tem um estilo e cor “purple”, sublinhado e com alinhamento de texto ao centro. </h1>
        <h2>Esse H2 tem um estilo em cor “orange” com fundo “black” e alinhamento de texto ao centro.</h2>
        <p>
          Esse paragrafo tem estilo em cor “blue” com texto em destaque em negrito e italico.
        </p>
    </body>
</html>
```

```
h1 {
    color : purple
        text-decoration : underline;
    text-align : center;
}

        h2 {
    color: orange;
        background-color: black;
        text-align: center;
        }

        p {
    font-weight: bold;
        font-style: italic;
        color: blue;
        }
```

Práticas

Prática 1: Navegação[7] com CSS

Nessa sessão o aluno vai criar através de exemplos práticos uma menu de navegação baseado em elementos HTML e CSS.

Passo 01

Crie um arquivo HTML com o seguinte conteúdo:

```
<html>
  <head>
    <link rel="stylesheet" type="text/css" href="menu.css " />
  </head>
  <body>
    <ul id="nav">
      <li id="nav_hom"><a href="/">Home</a></li>
<li id="nav_quem"><a href="/quem-somos.html">Quem Somos</a></li>
<li id="nav_neg"><a href="/nosso-negocio.html">Nosso Negócio</a></li>
<li id="nav_con"><a href="/contato.html">Contato</a></li>
    </ul>
  </body>
</html>
```

Passo 2

Em seguida crie o arquivo de estilo, chamado menu.css, e insira as seguintes linhas nele:

Passo 3

Clique no arquivo HTML e visualize-o no seu browser. O resultado esperado deve ser semelhante ao da figura seguinte:

```
/* Seletor 1 */
html {
  font: small/1.4 "Lucida Grande", Tahoma, sans-serif;
}
```

```
/* Seletor 2 */
body {
  font-size: 92%
}
/* Seletor 3 */
#nav-bar {
  margin: 0;
  padding: 0;
  background: #6F6146;
  list-style-type: none;
  width: 180px;
}

/* Seletor 4 */
#nav-bar li {
  margin: 0;
  padding: 0;
}

/* Seletor 5 */
#nav-bar a {
  display: block; /* aumenta a área clicável do link. */
  color: #FFF;
  text-decoration: none; /* link exibido sem sublinhado */
  padding: 0 15px;
  line-height: 2.5;
  border-bottom: 1px solid #FFF;
}

/* Seletor 6 */
#nav-bar #nav_con a {
  border: none;
}
```

```
/* Seletor 7 */
#nav a:hover {
  background: #4F4532;
}
```

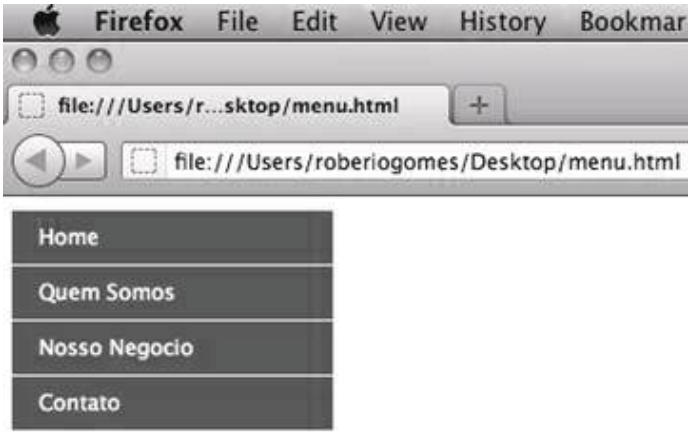


Figura 8 – Menus com CSS

4.4 Detalhes de Implementação

- Os seletores 1 e 2 definem detalhes sobre tipo de fonte a ser utilizado no documento.
- O seletor 3 especifica como deve ser exibida a lista não-enumarada de links:
    - Define a margem e a distancia interna entre a lista e seus elementos internos.
    - Define a cor de fundo do menu.
    - Define a largura do menu.
    - A linha list-style-type remove o símbolo de marcação geralmente associado a um item de lista não-enumarada.
- O seletor 4 define que os elementos do menu serão exibidos sem margem e sem espaçamento interno.
- O seletor 5 define que todo link que será usado como item de menu deve:
    - Ser maior que o habitual, com uma área clicável expandida;
    - Apresentar letras em cor branca;
    - Apresentar conteúdo dos links sem sublinhado;
    - Apresentar uma borda sólida em volta do item de menu.

- O seletor 6, define que o último item do menu deve ser exibido sem borda inferior, evitando que esta apareça caso o fundo da página não seja branco.
- O seletor 7 muda a cor do item de menu quando o mouse é passado por cima do item.

Atividades de avaliação

1. Construa um mini-site para uma empresa fictícia baseado na prática acima sobre menus com CSS e os conteúdos vistos nos demais capítulos. Seu site deve ter as seguintes sessões(páginas):
   a. Home Page
   b. Quem somos
   c. Nosso Negócio
   d. Contato
2. Pesquise na Internet sobre Web Standards e Tableless. O que significam essas palavras? De que tratam esses conceitos?
3. Pesquisa na Internet se existe diferença na construção de um site para ser acessado em dispositivos móveis. Os códigos usados em geral são os mesmos para celulares, tablets e computadores pessoais?

Referências

[1] CSS – W3C, disponível em: **www.w3.org/Style/CSS/**

[2] HTML 4.01 Specification - W3C: disponível em: **www.w3.org/TR/html401/**

[3] W3C, disponível em: http://www.w3c.br/

[4] Curso de CSS3, disponível em: **http://www.w3c.br/Cursos/CursoCSS3**

[5] Guia de Referência CSS 2.1, disponível em: **http://www.w3c.br/divulgacao/guiasreferencia/css2/**

[6] CSS Tutorial, disponível em: **http://www.w3schools.com/css/**

[7] Adams & Cols. A arte e a Ciência do CSS, Bookman, 2009

Capítulo 4

Noções básicas de JavaScript

Objetivo

- Apresentar os fundamentos da linguagem *JavaScript* e descrever por meio de cenários e exemplos simples como ela pode ser utilizada para construir aplicações mais dinâmicas e usáveis na Web.

Introdução

Esse capítulo apresenta os fundamentos da linguagem JavaScript e descreve por meio de cenários e exemplos simples como ela pode ser utilizada para construir aplicações mais dinâmicas e usáveis na Web.

1. O que é JavaScript?

Segundo [1], JavaScript, ou JS como muitos referenciam, pode ser definida como "A linguagem da Web". Ela é usada em bilhões de páginas Web para adicionar funcionalidades, validar formulários, comunicar com o lado servidor e muito mais.

Esse capítulo aborda somente sobre o JavaScript em client-side, ou seja, somente códigos que são inseridos nas páginas HTML sendo executadas pelo browser do cliente.

JS permite ao programador o acesso a todos os elementos de uma página Web, como imagens, elementos de um formulário, links entre outros. Estes objetos podem ser manipulados ou modificados de maneira programática, uma vez que JavaScript permite ao programador capturar eventos, como um click do mouse ou uma tecla pressionada de seu teclado. Com isso é possível criar ações em resposta ou mesmo baseadas nas ações do usuário ao integragir com as páginas Web.

2. Iniciando

Para inserir códigos JS em um documento HTML é necessário o uso de uma tag específica para isso, a tag <script>. O conteúdo JS deve ficar envolvido por essa tag como pode ser visto a seguir:

```
<script>
    comandos...
</script>
```

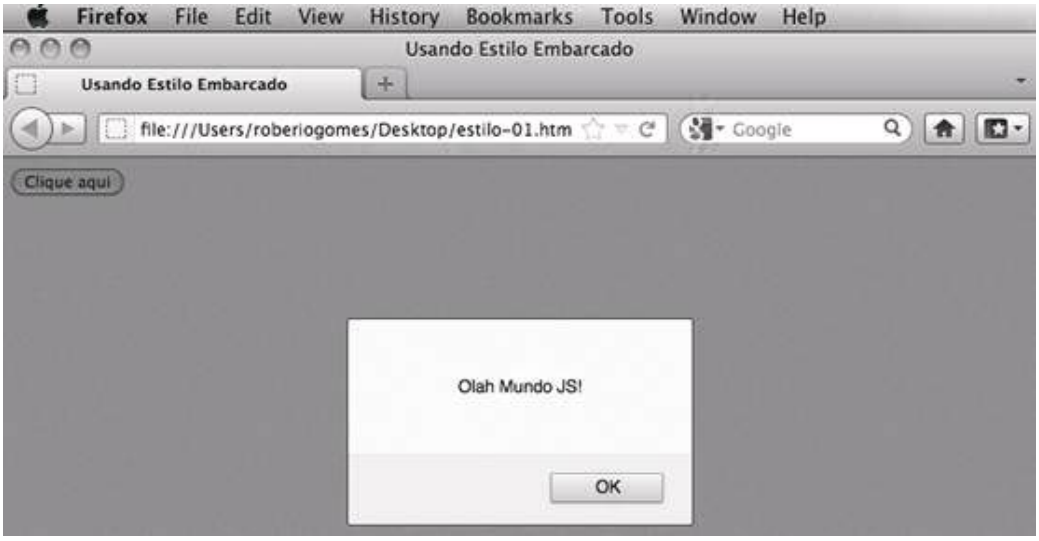
É possível também inserir os códigos JS em um arquivo externo e em seguida fazer referência a esse arquivo a partir da página HTML. Para isso a mesma tag deve ser utilizada com a adição do atributo SRC que deve apontar para a localização do arquivo JS(veja a seguir).

```
<script src="meucodigo.js"></script>
```

É recomendado que a tag script seja inserida logo no topo do documento HTML, mas na verdade ela pode estar inserida em qualquer posição do mesmo. Como uma boa prática, deve-se colocar essa tag dentro do cabeçalho da página, isto é, dentro da tag <head> como pode ser visto a seguir:

```
<html>
    <head>

  <title>Usando Estilo Embarcado </title>

  <link rel="stylesheet" type="text/css" href="estilo.css" />
  <script>
        function mostrarAlerta(){
            alert("Olah Mundo JS!");
        }
  </script>
    </head>

    <body>
  <p>
        <input type="button" onclick="javascript:mostrarAlerta();"
value="Clique aqui" />
  </p>
    </body>
</html>
```

A seguir veja o resultado da visualização da página HTML acima e a execução do código JS. Observe no código acima como código JS é usado na página, embutido na tag script.



2.1 A sintaxe

A sintaxe do JavaScript é similar a sintaxe do C/C++ e do Java, sendo assim como estas também uma liguagem case sensitive(faz distinção entre lestras maiúsculas e miúsculas). Na verdade ela é fruto de uma padronização resultante de um esforço da Netscape foi junto a um corpo internacional de padrões chamado ECMA, ao qual a Netscape submeteu a especificação da linguagem JavaScript. A ECMA, em junho de 1997, produziu um padrão chamado ECMA-262 (também conhecido como ECMAScript). [2]

As instruções de repetição e execução condicional, normalmente encontradas em C e Java também estão presentes em JS, havendo algumas diferenças quanto a declaração de variáveis e seus tipos.

Prática 01 – A Função document.write()

Toda página JS apresenta um objeto implícito chamado document. Uma de suas principais funções é write(), usada para escrever no corpo da página.

Teste o código abaixo no seu browser:

```
<html>
  <head>
        <title>Primeiro JS</title>
        <script>

              var browserName browserName = navigator.appName;
              document.write("seu browser é " + browserName);
        </script>
  </head>
  <body>
  </body>
</html>
```

Ao executar o código acima no navegador e clicar no botão exibido, resultado esperado deve ser semelhante ao da figura a seguir:

Figura 1 – Resultado do clique com JS

3. Operadores Lógicos e Matemáticos

3.1 Operadores Lógicos

JavaScript suporta os seguintes operadores lógicos:

Operador	Significado
==	Igual
!=	Diferente
>	Maior
<	Menor
>=	Maior ou Igual
<=	Menor ou Igual
&&	"E" lógico
||	"Ou" lógico

3.2 Operadores Matemáticos

Os seguintes operadores matemáticos são suportados:

Operador	Significado
+	Serve para adição de valores e ao mesmo tempo pode ser usado na concatenação de strings.
-	Utilizado na subtração de valores
*	Utilizado em multiplicação de valores
/	Utilizado para divisão de valores
%	Obtém o resto de uma divisão

Usando Variáveis

Uma variável pode ser chamada de um identificador. Em JS use a palavra var para declarar suas variáveis, podendo ela ser de qualquer tamanho, conter números, dígitos e underscores/underlines ('_'), sem a necessidade de especificar um tipo. Exemplo:

```
var minha_idade = 10;
var cor = 'azul';
var altura = 1.75;
```

* Nota: Não se esqueça: JavaScript é case-sensitive!

3.3 Estruturas Condicionais

JavaScript incorporou as principais estruturas de decisão do C/C++.

3.3.1 A instrução IF

Usada para implementar tratamento condicional diante de uma decisão a ser tomada, exemplo:

```
// exibe uma mensagem com "Maior de idade!", caso a idade seja maior que 18.
if (idade >= 18) {
   alert("Maior de idade!";
}
```

Essa instrução pode vir acompanhada de sua complementar else, como é visto a seguir:

```
// exibe uma mensagem com "Maior de idade!", caso a idade seja maior que 18.
// Caso contrário, exibe "Ainda menor de idade!"
if (idade >= 18) {
  alert("Maior de idade!");
} else {
  alert("Ainda menor de idade!");
}
```

3.3.2 A instrução Switch

Usada quando se tem mais de duas opções de escolha baseadas em valores predefinidos. Exemplo:

```
switch (param) {
  case '1':
        alert('Opção 1 - listar');
        break;
  case '2':
        alert('Opção 2 - Adicionar');
        break;
  case '3':
        alert('Opção 3 - Remover');
        break;
  default:
        alert('Sem opção - Sair!');
        break;
}
```

3.4 Estruturas de Repetição

3.4.1 A instrução for

Para execução de ações repetitivas é possível usar a instrução for. Ele é composta de três partes distintas: inicialização, teste e incremento. A seguir é possível observar seu uso:

```
for(var i = 1; i < 10; i++) {
        document.write(i + “<br />”);
}
```

3.4.2 A instrução while

Assim como a instrução for, while é usada para repetitivas. A idéia é praticamente a mesma que a instrução anterior como pode ser visto a seguir:

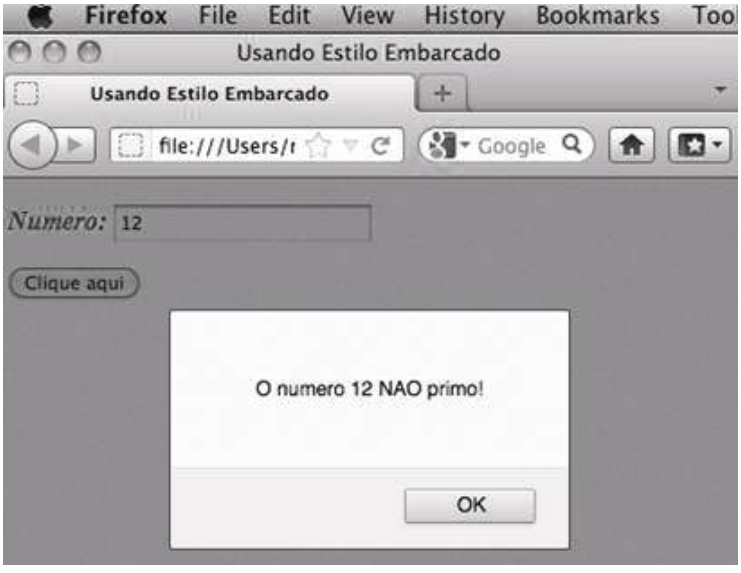
```
var i = 0; // inicialização
while(i < 10){ // teste
        document.write(i + “<br />”);
            i++; // incremento
}
```

O exemplo a seguir demonstra muitos dos conceitos abordados até agora, dentre eles, variáveis, operadores e instruções de repetição e condicionais:

```
<html>
  <head>

 <title>Usando Operadores e Instrucoes </title>

 <script>
        function validarPrimo(){
            // Recuperando o valor digitado no campo de texto.
            var valor = document.forms[0].numero.value;

            var ehPrimo = true;

            for(var i = 0; i < valor; i++) {
                if ((valor % 2) == 0) {
                  ehPrimo = false; // o numero tem divisor
                  break;
                }
            }

            // Verificando e exibindo o resultado.
            if (ehPrimo){
```

```
                    alert("O numero " + valor + " primo!");
                  } else {
                    alert("O numero " + valor + " NAO primo!");
                  }
            }
        </script>
          </head>

          <body>
              <form>
              <p>
                Numero: <input type="text" name="numero" />
        </p>
        <p>
              <input type="button" onclick="javascript:validarPrimo();"
value="Clique aqui" />
        </p>
        </form>
          </body>
      </html>
```

O resultado esperado pode ser visto a seguir:



Objetos

Em JavaScritp é possível trabalhar com objetos sendo alguns destes pre-definidos na linguagem e acessíveis para uso em páginas HTML. Dentre esses objetos merecem destaque window e forms, que representam a janela atual e o conjunto de formulários HTML de uma dada página, respectivamente. O exemplo a seguir mostra como é possível acessar tais objetos de forma indexada:

```
<form method=”post”>
        Nome: <input type=”text” name=”nome” />
        Idade: <input type=”text” name=”idade” />
        <input type=”button” name=”enviarBtn” />
</form>

<script type=”text/javascript”>
            // acessando o valor do campo nome
        var nome = document.forms[0].nome.value;
        alert(nome);
</script>
```

No exemplo acima são utilizados dois objetos implícitos a toda página HTML: document que representa a própria página e forms. Esse código acessa o primeiro formulário presente na página e em seguida busca o conteúdo de um campo de formulário chamado “nome”.

No exemplo a seguir é usado um nome específico para acessar o formulário, baseado em seu atributo name:

```
<form name=”dados” method=”post”>
        Nome: <input type=”text” name=”nome” />
        Idade: <input type=”text” name=”idade” />
        <input type=”button” name=”enviarBtn” />
</form>

<script type=”text/javascript”>
            // dados é o nome do form
        var nome = document.dados.nome.value;
        alert(nome);
</script>
```

Dessa forma fica fácil perceber que os objetos podem ser tratados de duas formas: uma delas é chamando por um nome ordenado de uma propriedade do documento e a outra é chamando pelo próprio nome.

Outro detalhe interessante é que esses objetos podem ter propriedades (características) e métodos (comportamentos) e esses dois conceitos são bastante explorados durante a programação JS.

Eventos

O browser interpreta o JavaScript esperando que eventos aconteçam. Esses eventos podem vir a acontecer quando a página termina de ser carregada ou quando o usuário executa alguma interação com a página, como por exemplo mover o mouse sobre um link ou clicar em um botão. A associação de eventos com funções tratadoras desses é feita inserindo-se dentro das tags HTML instruções que ligam os dois. A maioria dos eventos é precedido da palavra ON, como por exemplo onMouseOver.

Cabe ao programador definir funções que contenham as devidas ações a serem tomadas quando um dado evento é disparado. A seguir é possível ver uma tabela que ilustra que elementos são capazes de disparar eventos, que tipo de eventos e em que situação isso acontece:

Tag	Evento	Quando é chamado
a	onClick	O usuário clica no link.
	onMouseOver	O ponteiro do mouse é colocado em cima do link.
	onMouseOut	Quando o mouse sai fora do link.
body	onBlur	A janela ou frame contendo esta página perde o fócus.
	onFocus	Ao contrário do onBlur a página recebe o fócus.
	onLoad	Quando a página foi terminada de carregar.
	onUnlod	Quando o usuário sai da página.
form	onReset	Quando o botão RESET é pressionado.
	onSubmit	Quando o Botão submit é pressionado.
Tag	**Evento**	**Quando é chamado**
img	onAbort	A leitura da imagem foi parada por uma ação do usuário
	onLoad	Quando a leitura da imagem é terminada.
	onError	Quando um erro ocorreu enquanto a imagem foi lida.
input:button ou checkbox ou radio ou reset	onClick	Quando um elemento do formulário é clicado com o mouse.
input:text ou textarea	onBlur	Quando o elemento do formulário perde o fócus.
	onChange	Quando o usuário muda o item selecionado.
	onFocus	Quando um usuário clica em um item.
	onSelect	Quando um elemento do formulário recebe o fócus

Práticas

Prática 01 – Validando dados do Formulário

Crie a seguinte página HTML:

```
<html>
  <head>
  <title>Prática-01</title>
        <script type="text/javascript">
            function validarCampos() {
                var nome = document.dados.nome.value;
                var idade = document.dados.idade.value;

                if ((nome == null) || (nome == '')){
                    alert('O campo nome é obrigatório!');
                }

                if ((idade == null) || (idade == '')){
                    alert('O campo idade é obrigatório!');
                }
            }
        </script>
  </head>
  <body>
        <form name="dados" method="post">
            Nome: <input type="text" name="nome" />
            Idade: <input type="text" name="idade" />
            <input type="button" name="enviarBtn"
onclick="validarCampos();"/>
        </form>
  </body>
</html>
```

O código acima faz uso do evento onClick no elemento button e chama a função JS denominada validarCampos() que é responsável por verificar se todos os campos do formulário foram preenchidos.

Síntese do Capítulo

Esse capítulo trata da linguagem JavaScript, seus fundamento s principais elementos de programação. Os exemplos contidos nele ajudam a ter um entendimento inicial de como criar tratadores de eventos em páginas HTML.

Atividades de avaliação

1. Crie uma página com um formulário contendo os seguintes campos:
   a. Nome
   b. Endereço
   c. Cidade
   d. Cep
   e. Cidade
   f. UF
   g. Data de Nascimento

   Valide o formulário como demonstrado na prática 01.
2. Crie uma página com dois formulários independentes, cada um contendo campos HTML distintos. Crie duas funções de validação independentes para cada formulário, baseado no exercício anterior.

Referências

[1] w3schools/JS. Disponível em: **http://www.w3schools.com/js/**

[2] ECMA-262. Disponível em: **http://developer.netscape.com/docs/javascript/e262-pdf.pdf**

[3] JavaScript Tutorial, disponível em: **www.w3schools.com/js/**

[4] JavaScript - The W3C DOM, disponível em: **http://www.tutorialspoint.com/javascript/javascript_w3c_dom.htm**

Sobre os autores

Joaquim Celestino Júnior: Possui graduação em Engenharia Eletrônica pela Pontifícia Universidade Católica do Rio de Janeiro (1985), mestrado em Ciência da Computação pela Universidade Federal da Paraíba/Campina Grande(1989), doutorado em Redes de Computadores - Universite de Paris VI (Pierre et Marie Curie) (1994) e pós-doutorado em Redes Veiculares pela Columbia University in New York City (2010). É professor adjunto da Universidade Estadual do Ceará (UECE). Tem experiência na área de Ciência da Computação, com ênfase em Teleinformática, atuando principalmente nos seguintes temas: redes de computadores, gerenciamento de redes e segurança.

Robério Gomes Patricio: É especialista em Desenvolvimento de Soluções Web usando a Plataforma Java EE pela Faculdade de Juazeiro do Norte, possui graduação em Ciência da Computação pela Universidade Estadual do Ceará e é Mestrando em Ciência da Computação pela Universidade Estadual do Ceará. Atuou como professor em Instituições de Ensino Público e Privado de renome no estado do Ceará, dentre elas a Instituto Federal do Ceará, Universidade Estadual do Ceará(UAB), Universidade de Fortaleza, Faculdade Sete de Setembro, Faculdade Leão Sampaio, Faculdade de Juazeiro do Norte em cursos de Graduação e Pós-Graduacão. Atualmente trabalha na Faculdade Sete de Setembro como Vice-Coordenador do Curso de Bacharelado em Sistemas de Informação e atua como Arquiteto de Software/Consultor em Fortaleza(desde 2000) em projeto na area de TIC, exercendo atividades de Consultor de TI e Instrutor de Cursos de Extensão nos mais diversos temas tais como: Java, OOAD, SOA, SCRUM, Java EE, Java ME, Games, UML e Design Patterns.

Computação

Fiel a sua missão de interiorizar o ensino superior no estado Ceará, a UECE, como uma instituição que participa do Sistema Universidade Aberta do Brasil, vem ampliando a oferta de cursos de graduação e pós-graduação na modalidade de educação a distância, e gerando experiências e possibilidades inovadoras com uso das novas plataformas tecnológicas decorrentes da popularização da internet, funcionamento do cinturão digital e massificação dos computadores pessoais.

Comprometida com a formação de professores em todos os níveis e a qualificação dos servidores públicos para bem servir ao Estado, os cursos da UAB/UECE atendem aos padrões de qualidade estabelecidos pelos normativos legais do Governo Federal e se articulam com as demandas de desenvolvimento das regiões do Ceará.

ISBN 978-85-78264-18-5
9 788578 264185

Universidade Estadual do Ceará

UAB UNIVERSIDADE ABERTA DO BRASIL

CAPES