Silvio Ferreira

Guia prático de HTML

Guia completo de comandos • Exemplos práticos

Novos recursos • Passo a passo detalhado

UNIVERSO DOS LIVROS

Guia prático de HTML5

Universo dos Livros Editora Ltda.
Rua do Bosque, 1589 — Bloco 2 — Conj. 603/606
CEP 01136-001 — Barra Funda — São Paulo/SP
Telefone/Fax: (11) 3392-3336
www.universodoslivros.com.br
e-mail: editor@universodoslivros.com.br
Siga-nos no Twitter: @univdoslivros

Guia prático de HTML5

Silvio Ferreira

© 2013 by Universo dos Livros
Todos os direitos reservados e protegidos pela Lei 9.610 de 19/02/1998.

Nenhuma parte deste livro, sem autorização prévia por escrito da editora, poderá ser reproduzida ou transmitida sejam quais forem os meios empregados: eletrônicos, mecânicos, fotográficos, gravação ou quaisquer outros.

Diretor editorial
Luis Matos

Coordenador editorial
Bóris Fatigati

Preparação
Adir de Lima

Revisão
Fabiana Chiotolli

Arte
Francine C. Silva
Karine Barbosa

Capa
Marcos Mazzei

1ª edição - 2013

Dados Internacionais de Catalogação na Publicação (CIP)
Angélica Ilacqua CRB-8/7057

F439g Ferreira, Silvio

Guia prático de HTML5 / Silvio Ferreira. – São Paulo: Universo dos Livros, 2013.
168 p.

ISBN: 978-85-7930-376-0

1. Informática 2. HTML5 3. Webdesign I. Título

13-0165 CDD 005.75

<Capítulo 1>
Introdução

Definições iniciais

HTML são siglas de **H**ypertext **M**arkup **L**anguage. A tradução em bom português é Linguagem de Marcação de Hipertexto. É uma linguagem usada para criar páginas para Web e com ela é possível criar as marcações no conteúdo de uma página.

Uma página da web pode possuir itens como imagens, parágrafos, títulos, subtítulos, vídeos, tabelas, listas etc. E para criar cada um deles utilizamos elementos HTML que irão marcar e definir o tipo de item em questão. Esses elementos HTML são chamados de tags.

Assim, existem tags para criar parágrafos, títulos, para definir imagens, quebra de linhas, enfim, para tudo que for feito em uma página web existe uma tag específica. Exemplos: para criar um parágrafo utiliza-se a tag ; para carregar uma imagem na página há a tag <img>; para criar tabelas há a tag <table>; para títulos existe a tag <h>; entre várias outras.

Como é possível observar, tags são digitadas entre os sinais **menor que** (<) e **maior que** (>). Grande parte das tags precisa, quando utilizada, ser fechada. A tag de fechamento utiliza o caractere barra (/). Veja abaixo o exemplo do uso da tag para definir um parágrafo.

```
<p> Curso de HTML5 - Com Silvio Ferreira </p>
```

Aqui há a tag definindo um parágrafo, o texto (Curso de HTML5 – Com Silvio Ferreira) e o fechamento da tag com o uso de . O fechamento da tag é necessário para definir onde termina o parágrafo.

Não é toda tag que precisa ser fechada. Por exemplo: a tag
 é usada para definir uma quebra de linha. Ela não precisa ser fechada. Apenas define uma quebra de linha. No decorrer do livro há explicações detalhadas sobre o uso de diversas tags.

Tags, atributos e valores

Uma tag pode possuir atributos e valores. Vejamos como exemplo o uso da tag <table> que é utilizada para criar tabelas:

```
<table border="1">
```

Nessa tag <table> digitamos um atributo, que é border. Com esse atributo, podemos especificar uma borda nas células da tabela. E o valor do atributo neste exemplo é 1. Valores de atributos são digitados sempre depois do sinal de igual (=).

É importante saber diferenciar esses três elementos (tag, atributo e valor), pois são usados comumente ao trabalharmos com HTML, principalmente no HTML 4.01.

No HTML5, muitos atributos (e consequentemente seus valores) de tags não são usados, pois foram excluídos. Isso ocorreu porque suas funções são facilmente realizadas por meio de CSS. Tais questões serão abordadas em detalhes no decorrer do livro.

Hipertexto e hiperlinks

Duas palavras muito comuns no vocabulário de desenvolvedores web, profissionais e estudantes do meio são hipertexto e hiperlinks. No próprio significado das siglas HTML aparece a palavra hipertexto (Hypertext Markup Language).

Hipertexto é uma palavra cujo significado está ligado à ideia de um texto que apresenta diversos caminhos diferentes de leitura, cujas partes estão interconectadas. De forma mais objetiva, ele é composto de páginas que irão possuir links. Um link (ou hiperlink) é um elemento clicável que nos permite acessar outras páginas ou arquivos na web. Um link pode ser um texto, uma imagem ou ambos.

Um novo HTML?

O HTML5 é um novo HTML? Tudo irá mudar a partir da adoção dessa nova versão? Essas são apenas algumas das dúvidas que desenvolvedores de websites e estudantes podem ter ao se aventurarem nessa nova versão do HTML.

O HTML5 é uma nova versão do HTML 4.01, que foi concebida para permitir que programadores possam gerar códigos mais organizados, bem estruturados, com a utilização das marcações de forma correta e separando definitivamente a marcação do conteúdo da formatação do layout. Além de permitir a construção de websites com mais acessibilidade.

A partir dessa nova versão, ele é usado somente para criar as marcações e estruturar o conteúdo do documento. O HTML5 nasceu para ser usado de forma semântica.

E a formatação do conteúdo, ou seja, o layout? Como serão definidos as cores dos textos, a fonte, fundos, posicionamentos dos elementos na página, uso de bordas, enfim, como será definida toda a aparência visual da página? Tudo isso passa a ser papel indispensável de CSS – **Cascading Style Sheets** (**Folhas de Estilo em Cascata**), e que pode ser chamada somente por Folhas de Estilo.

No HTML 4.01 não existia obrigatoriamente essa separação. O uso de CSS era comum, porém, o HTML ainda era usado para definir a aparência dos elementos da página. Vejamos alguns exemplos:

- O uso da tag <fonte> para definir a cor, família de fonte e tamanho dos textos.
- Tag <p> com atributo *align* para alinhar parágrafos (esquerda, direita, justificado, centro).
- Tag centraliza elementos variados (textos, imagens, tabelas etc.).
- Utilização de atributos que definem borda (border), alinhamento (align), cor de fundo (bgcolor), imagem de fundo (background), entre outros.

Esses são apenas alguns exemplos de HTML sendo usados para definir a aparência, o visual de elementos diversos. Com HTML5 isso não é mais permitido. A partir dessa versão, tags HTML são usadas para criar marcações no conteúdo e CSS para formatar e dar todo o visual da página web.

No HTML5, algumas tags que são comumente usadas no HTML 4.01 foram excluídas. Outras tiveram seus significados modificados e novas tags foram criadas.

Portanto, programar com HTML5 exige uma nova forma de pensar. Ao criar uma página para a web, tudo que for feito e construído nela passa a ter significado. A página irá possuir cabeçalhos, menus, artigos, conteúdo relacionado e rodapé. E cada parte dela possui novas tags que as identificam categoricamente no código.

O código passa a ser mais limpo e organizado. A sua interpretação também passa a ser mais fácil. Um programador, ao analisar o código, irá identificar e entender com mais rapidez cada parte e cada tag do código.

Não é somente o fator humano que passa a ser beneficiado. Sistemas baseados na web, que de alguma forma irão acessar a página web, também irão se beneficiar deste novo código. Sites de busca como o Google (www.google.com.br), Achei (www.achei.com.br), Alta Vista (www.altavista.com), entre outros, encontrarão as informações com mais facilidade. Se eles precisarem de uma informação do cabeçalho ou do rodapé, por exemplo, encontrarão facilmente porque essas partes da página estarão devidamente marcadas no código. E o que isso significa na prática? Na prática teremos sistemas de busca mais precisos. Ao fazer uma busca em um site, o resultado exibido será mais coerente e relevante em relação ao que procuramos e as informações estarão mais bem definidas. Teremos, assim, uma garantia maior de acessar sites que realmente

possuem aquilo que procuramos.

Todas as tags do HTML5 são novas?

Definitivamente não. O HTML5 usa diversas tags do HTML 4.01, entretanto, devemos estar atentos pois muitos atributos de tags não são compatíveis no HTML5. Mesmo que uma determinada tag seja compatível, seus atributos podem não ser mais usados. Mais adiante, retornaremos a este ponto ao particularizar as tags compatíveis e as incompatíveis com HTML5.

O que há de novo?

O HTML5 foi criado em prol de uma web mais semântica. Isso significa que teremos um código padronizado e organizado. Para isso ser possível foram criadas diversas novas tags enquanto outras foram excluídas.

Como o objetivo deste tópico é citar as novidades, vamos falar apenas das tags estruturais. Há diversas outras tags novas que não abordaremos por ora, mas, no decorrer do curso, iremos conhecê-las.

A estruturação de páginas é o tópico que mais sofreu mudanças no HTML5. Há agora tags específicas para cada parte de uma página. Essas partes podem ser chamadas de seções.

Como podemos observar na **Figura 1.1**, a página está dividida em diversas partes (ou seções). Cada uma delas é definida por uma nova tag. São elas:

- **<header>**: cabeçalho de uma seção. Como vemos, cabeçalhos podem ser usados no documento ou em seções específicas como a seção <article>;
- **<aside>**: cria uma seção de conteúdo relacionado. É qualquer conteúdo que tenha alguma relação com o conteúdo principal. Pode ser um menu secundário, banners, links ou imagens de parceiros, link do tipo “indique” ou “cadastre-se aqui”, anúncios em geral, entre vários outros exemplos;
- **<section>**: cria uma seção de conteúdo. É uma seção “genérica” e pode agrupar conteúdos diversos. Pode ser utilizada inclusive para agrupar outras seções que pertençam a um mesmo conteúdo. Exemplo: artigo, cabeçalho do artigo e rodapé do artigo;
- **<article>**: criar uma seção de artigo. É o conteúdo principal do documento. Pode possuir textos, imagens, vídeos etc.;
- **<nav>**: cria uma seção de menu. É o menu principal do site, que conterá links para acessar as outras páginas e/ou
- arquivos. Não deve ser usada para agrupar qualquer link, mas somente um menu. Um menu é composto por links (em modo texto ou imagem) que darão acesso às várias partes do site. Pode ser utilizada também para criar seções de menus secundários;
- **<footer>**: cria uma seção de rodapé. O rodapé pode ser do documento ou de determinadas seções (tal como o rodapé de <article>). O rodapé da página possui informações geralmente visíveis a todas as páginas do site, tais como telefones, e-mails, endereços, avisos de proteção e de direitos autorais etc.

Apesar de essas tags estruturais representarem uma das principais mudanças, há muitas novidades do HTML5, muitas tags novas e outras que foram excluídas. Tudo isso é mencionado e explicado em detalhes no decorrer do livro.

Figura 1.1.: *Estruturação de uma página com HTML5.*

HTML5 Semântico

Já mencionamos anteriormente um termo muito importante: semântica. Este termo está intimamente ligado ao HTML5. Como dissemos, ele nasceu para ser usado de forma semântica. Mas, o que é HTML semântico?

Ao pesquisar em dicionários a palavra semântica, encontramos "ramo da linguística que estuda o significado das palavras" ou ainda "a ciência das significações".

E onde o HTML5 se encaixa nisso? Estaremos programando com HTML semântico quando utilizarmos cada tag para dar um *significado* aos elementos da página. Devemos utilizar as tags de forma correta e para aquilo que ela realmente foi criada. Se formos criar parágrafos, devemos usar a tag <p> (tag para marcar parágrafos). Se formos criar listas, devemos usar <ul>, <ol> e <li> (tags para criar listas). E se na página houver endereço, abreviaturas e citações, também há tags para cada caso em particular.

Ao usar as tags para marcar cada tipo de elemento (parágrafos, títulos, citações, endereços, imagens, listas, tabelas, vídeos, áudio, siglas etc.) de nossa página, daremos um significado em termos de código para tudo. Navegadores, motores de busca, tradutores, leitores de tela e qualquer outro sistema de acesso a websites irão manipular essa página sem nenhum tipo de problema, encontrando as informações necessárias com precisão. Desse modo, construiremos um código semântico. Para facilitar a assimilação desse conceito, vejamos alguns pontos importantes:

1. **O conteúdo deve ser separado da apresentação visual**: como mencionamos, usamos as tags unicamente para criar as marcações e CSS para formatar toda a aparência visual.
2. **Usar as marcações para cada seção da página**: o HTML5 possui diversas tags estruturais, como <header>, <aside>, <section>, <article>, <nav> e <footer>. Cada tag dessa possui um significado semântico. Quando um navegador carregar uma página com essas marcações, ele saberá que em <header> há um cabeçalho, em <aside> o conteúdo relacionado da página, em <nav> há um menu, e assim sucessivamente.
3. **Não construir o layout da página em tabela**: esse é definitivamente um erro grotesco. É comum encontrar websites cujo layout foi todo construído usando uma tabela. Tabelas devem ser usadas somente para organizar *dados tabulares*, ou seja, qualquer tipo de informação que necessita ser apresentada em linhas e colunas de forma lógica e organizada. Quando utilizamos as tabelas para construir o layout, não estamos construindo um código semântico.
4. **Títulos**: as tags <h1> até <h6> são usadas para definir títulos. <h1> representa o título mais relevante da página, isto é, o título principal. Para o subtítulo usa-se <h2>, e assim sucessivamente. Essas tags não devem ser usadas unicamente para definir o tamanho de textos. Elas marcam títulos, e, por meio de CSS, podemos configurar o tipo de fonte, cor e até o tamanho desses títulos.

Figura 1.2.: *Títulos de h1 a h6.*

1. Abreviaturas e siglas: se na página for digitado alguma sigla, devemos usar a tag <abbr> para descrever o significado delas. Ao marcar as abreviaturas, essas informações passam a ser usadas pelos navegadores, sites de buscas, tradutores etc. Veja um exemplo no qual temos o uso das siglas HTML.

```
O <abbr title=”Hypertext Markup
Language”>HTML</abbr> é uma linguagem de…
```

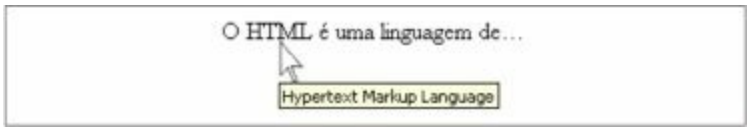


Figura 1.3.: *Resultado prático do uso da tag <abbr>. No navegador, quando o cursor do mouse é posicionado sobre a sigla, podemos ver o significado dela.*

1. Imagens: use sempre o atributo alt. Ele serve para especificar um texto *alternativo* para a imagem. Exemplo:

```
<img src=”logo.png” alt=”Estrela Informática” height=”42” width=”42”>
```

2. Frames: o uso de frames é totalmente incompatível com a acessibilidade. Leitores de tela apresentam dificuldades ao lidar com frames. O leitor de tela é um software usado para obter resposta do computador por meio sonoro. É usado principalmente por deficientes visuais. No HTML5 o uso de frames foi banido e, portanto, esse recurso totalmente dispensável não vai funcionar. Aqui podemos perceber que o HTML5 foi construído para ser semântico e para oferecer maior acessibilidade.

3. Informações de contato: se na página houver as informações de contato do autor, tais como telefone, e-mail, endereço, entre outras, devemos usar a tag <address> para marcar essas informações no código. Exemplo:

```
<address>
Postado por <a href=”mailto:lucasyyyy@00000.
com”>Silvio XX YY</a>.<br>
Website:<br>
www. 00000.com <br>
Rua Tal, nª999, Centro, Cidade/Estado/
CEP<br>
Tel.: (00) 0000-0000 <br>
</address>
```

4. Bloco de citações: se na página web houver algum bloco de citação devemos marcá-lo com a tag <blockquote>. Usamos essa tag para marcar citações longas, que podem possuir um ou mais parágrafos. São citações extraídas de revistas, jornais, livros ou outros sites.

5. Listas: listas podem ser numeradas ou não. Ao construir uma lista, devemos utilizar as tags <ul> e <li> para listas não numeradas, e, <ol> e <li> para listas numeradas. É comum, por desconhecimento ou “preguiça”, programadores de sites criarem listas sem usar essas tags (usando

somente a tag <p>). Entretanto, ao não usá-las nas listas o HTML perde sua semântica.

Esses dez itens citados são apenas um resumo. Programar HTML semântico requer mais do que isso. Requer compreender o conceito em sua plenitude, estudo constante e prática. Ao construir páginas para web, use o que aprendeu, ponha em prática! Pense sempre em como melhorar o código, como torná-lo, da melhor forma possível, semântico.

HTML5 versus navegadores

Se você está iniciando seus estudos agora no HTML5, com certeza terá esta dúvida: todo e qualquer navegador suporta o HTML5? No momento em que escrevo este livro, praticamente todos os navegadores atuais suportam boa parte do HTML5. É preciso estar atento a duas questões: recurso do HTML5 e versão do navegador.

Versões mais antigas de navegadores não suportam o HTML5. E há determinados recursos do HTML5 que não são suportados por um determinado navegador (mesmo que atual), mas, podem ser suportados por outros navegadores. Há expectativa, entretanto, que logo todos os navegadores passarão a suportar o HTML5 em sua total plenitude.

Enquanto isso não acontece, é preciso descobrir quais são os navegadores que o suportam. Existe na web vários sites que podem ser usados como referência para responder a essa questão.

Selecionamos dois deles:

- http://www.findmebyip.com/litmus
- http://html5test.com/

O primeiro (http://www.findmebyip.com/litmus) é um site bem completo e apresenta informações sobre a compatibilidade de HTML5 e CSS3 com os navegadores em várias versões para MAC e Windows.

Figura 1.4.: *Website http://www.findmebyip.com/litmus*

Como vemos na **Figura 1.4**, HTML5 Graphics & Embedded Content não é suportado somente pelos navegadores Internet Explorer 8, 7 e 6. Quanto ao HTML5, veremos informações de compatibilidade nas seguintes categorias:

- HTML5 Web Applications.
- HTML5 Graphics & Embedded Content.
- HTML5 Audio Codecs.
- HTML5 Video Codecs.
- HTML5 Forms Inputs.
- HTML5 Forms Attributes.

Já o site http://html5test.com/ faz um teste bem interessante. Ao acessá-lo, ele mostrará a compatibilidade de HTML5 com o nosso próprio navegador. Ou seja, o resultado exibido vale somente para o navegador que estivermos usando no momento.

Ao acessá-lo, ele nos mostra a pontuação do nosso navegador. Essa pontuação vai de 0 a 500 pontos. Quanto mais próximo de 500 pontos melhor é a compatibilidade.

Em nosso caso, vemos na **Figura 1.5** que o nosso navegador usado para acessar o site obteve 448 pontos. E mais abaixo na página há um relatório completo de todos os recursos suportados ou não.

Figura 1.5.: *Website http://html5test.com/*

Aplicativos para programar com HTML5

Para programar com HTML5 podemos usar até um simples bloco de notas. Mas existem softwares que facilitam a programação, oferecendo recursos tais como a numeração de linhas, uso de cores diferente na sintaxe da linguagem etc. Recomendamos o uso de dois programas bem famosos e, melhor ainda, ambos gratuitos:

- **Notepad++**: pode ser baixado em http://notepad-plus-plus.org/.
- **Aptana Studio Community Edition**: pode ser baixado em www.aptana.com.

Por motivos didáticos, neste livro é usado somente o bloco de notas (do Windows). Com isso, usuário de Windows, Linux, Mac ou qualquer outro sistema operacional conseguirá acompanhar os exercícios da mesma forma apresentada neste livro.

Entretanto, nada impede que você faça o download de um dos softwares apresentados anteriormente, instale-o e use-o para pôr em prática os ensinamentos desta obra.

Como salvar um arquivo HTML

Se for utilizar o bloco de notas para digitar os códigos, a primeira providência é salvar um arquivo HTML. É nesse arquivo que digitará os códigos e testará os resultados em um navegador. Para salvar um arquivo HTML dê os seguintes passos:

1. No bloco de notas, clique em **Arquivo – Salvar Como**.

Figura 1.6.: *Arquivo – Salvar Como.*

2. Na janela que é aberta, na opção **Salvar Como Tipo**, seleciona-se a opção **Todos os Arquivos**.

Figura 1.7.: *Salvar Como Tipo – Todos os Arquivos.*

3. Em **Nome do Arquivo**, digite um nome seguido da extensão HTML.

Figura 1.8.: *Digite um nome seguido da extensão HTML.*

4. Escolha um local onde deseja salvar o arquivo e clique no botão **Salvar**.

Ao digitar os códigos, clique sempre na opção **Arquivo – Salvar**. E para testar cada código, abra o arquivo em um navegador que suporte HTML5. Neste livro usamos, preferencialmente, para testar os códigos o navegador Google Chrome.

<Capítulo 2>
O que ainda funciona

Introdução

O objetivo deste capítulo é listar todas as tags existentes no HTML 4.01 e que são compatíveis no HTML5. Com isso, se você for um estudante de desenvolvimento web e não conhecer a linguagem, poderá usar este livro para fazer um estudo bem completo. Nas páginas seguintes, há explicações de cada tag e exemplos de código de cada caso separadamente.

Comentários

Um comentário é um texto explicativo que podemos inserir no código. Esse texto não interfere no código e não é executado e nem exibido pelo browser. Sua função é apenas servir como um lembrete ou explicação de um código ou linha de código.

Suponhamos que você está trabalhando com um código. Para que você não se esqueça a que se refere determinadas linhas já digitadas, digite comentários como forma de descrever essas linhas ou até mesmo bloco de códigos.

Os comentários serão úteis não somente a você, mas também ajudará a qualquer outro programador que estiver trabalhando com o código.

No entanto, comentários não podem ser simplesmente digitados de qualquer forma. Para que o navegador saiba que o texto digitado trata-se de um comentário, é necessário demarcar esse texto. Eles devem ser digitados entre os caracteres <!-- e -->

Exemplo de uso:

```
<!-- Este é um exemplo de comentário -->
```

<HTML>

Esta é a primeira tag que digitamos. Representa a raiz do documento HTML e é a partir dela que todas as demais tags serão digitadas. Ao final de todo o código ela é fechada.

O site propriamente dito é construído entre <html> e </html>. Ela cria uma seção (ou recipiente), e dentro dela digitaremos todas as tags da página. Ao abrir a página em um navegador, ele irá processar tudo que estiver entre essas tags e exibir o conteúdo na tela.

Exemplo de uso:

```
<html>
<!-- Demais tags e conteúdo aqui -->
</html>
```

<head>

A tag <head> é usada para construir o cabeçalho do documento (ou da página, como queiram chamar) HTML. Trata-se do cabeçalho principal, onde iremos especificar o título da página, tags META, carregamento de arquivos externos (como arquivos CSS) entre outras informações.

Exemplo de uso:

```
<html>
<head>
<!--Cabeçalho aqui-->
</head>
</html>
```

Não podemos confundir esse cabeçalho com a tag <header>, que é uma nova tag do HTML5 utilizada para construir cabeçalhos de artigos por exemplo.

<title>

Toda página HTML possui um título que é exibido na parte superior do navegador (na barra de títulos). Essa é uma informação que pertence ao cabeçalho do documento HTML. O título é digitado entre as tags <title> e </title>.

Exemplo de uso:

```
<html>
<head>
    <title>
        Página do Silvio Ferreira
    </title>
</head>
</html>
```

Este título é fundamental ao construir uma página HTML. Ele serve como referência para o usuário que está lendo a página. Pode ser o título do assunto principal, uma leve descrição ou pode conter até palavras-chave que tenham relação com o conteúdo.

<body>

Após o cabeçalho vem a tag <body>. Ela define o corpo do documento. É entre as tags <body> e </body> que todo o conteúdo da página é inserido. Tudo que vemos na página (textos, figuras, tabelas, listas, vídeos etc.) é digitado entre elas.

Essa tag deve ser fechada no ponto onde termina o corpo do documento, sempre antes de </html>.

Exemplo de uso:

```
<html>
<head>
    <title>
        Página do Silvio Ferreira
    </title>
</head>
<body>
<!-- Corpo do documento aqui -->
</body>
</html>
```

<h>

A tag <h> faz marcações de títulos, tais como títulos de cabeçalhos de artigos. Existe uma classificação quanto ao nível de relevância, que vai de <h1> até <h6>. O título mais importante da página é marcado com a tag <h1> e o menos importante com a tag <h6>. Ou seja, se a página possui um título principal, ele deve ser marcado com <h1>. E se possui um subtítulo ele deve ser marcado com <h2>.

Não confunda esses títulos com o título do cabeçalho. O título do cabeçalho é marcado com a tag <title> e é usado pelo navegador, sendo exibido na barra de títulos. Já os títulos marcados com as tags <h> pertencem ao conteúdo, como, por exemplo, o título principal de um artigo.

Exemplo de uso:

```
<html>
<head>
    <title>
        Página do Silvio Ferreira
    </title>
</head>
<body>
<h1> Título Principal de um Artigo </h1>
</body>
</html>
```

<p>

Usamos esta tag para definir parágrafos. Sempre que inserir textos em uma página, essa tag deve ser usada. Em cada parágrafo que ela for empregada haverá uma quebra de linha entre um parágrafo e outro. Isso significa que quando um parágrafo for finalizado, o próximo se inicia na linha de baixo.

Exemplo de uso:

```
<html>
<head>
    <title>
        Página do Silvio Ferreira
    </title>
</head>
<body>
<h1> Título Principal de um Artigo </h1>
<p>  Texto Texto Texto Texto Texto Texto Texto
</p>
</body>
</html>
```

*
*

Esta tag insere uma quebra de linha. Quando colocada após um texto, provocará uma quebra de linha e os textos seguintes saltarão para a linha de baixo. Não precisa ser fechada, apenas indica que houve uma quebra de linha.

Exemplo de uso:

```
<html>
<head>
    <title>
        Página do Silvio Ferreira
</head>
<body>
<h1> Título Principal de um Artigo </h1>
<p>
Texto Texto Texto Texto Texto Texto <br>
Texto Texto Texto Texto Texto Texto
</p>
</body>
</html>
```

<hr>

Ao usar a tag <hr>, o resultado prático é que veremos uma linha horizontal na página. Melhor dizendo, esse é o significado dessa tag no HTML 4.01, no entanto, no HTML5 essa tag possui um significado semântico, que é definir a mudança de tema, de assunto. Por exemplo: separar um parágrafo que se refere a HTML de um que fala sobre CSS.

Exemplo de uso:

```
<html>
<head>
    <title>
        Página do Silvio Ferreira
    </title>
</head>
<body>
<h1> Título Principal de um Artigo </h1>
<p>
HTML é uma sigla inglesa da expressão Hypertext Markup Language.
</p>
<hr>
<p>
CSS é uma sigla inglesa da expressão Cascading Style Sheets.
</p>
</body>
</html>
```

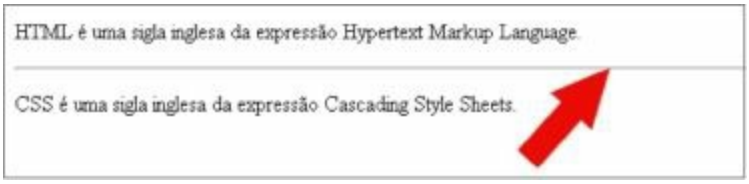


Figura 2.1.: *Uso de <hr>.*

<abbr>

Usada para marcar abreviaturas ou acrônimos e descrever o significado delas. Os significados das abreviaturas passam a ser reconhecidos pelos navegadores, sites de buscas, tradutores etc.

Exemplo de uso:

```
<html>
<head>
    <title>
        Página do Silvio Ferreira
    </title>
</head>
<body>
<p>
O <abbr title=”Hypertext Markup
Language”>HTML</abbr> é uma linguagem de…
</p>
</body>
</html>
```

<address>

tag que possui a função de marcar endereços e informações de contato com o autor da página ou do conteúdo (artigo).

Exemplo de uso:

```
<html>
<head>
    <title>
        Página do Silvio Ferreira
    </title>
</head>
<body>
<address>
Postado por <a href=”mailto:lucasyyyy@00000.
com”>Silvio XX YY</a>.<br>
Website:<br>
www. 00000.com <br>
Rua Tal, no999, Centro, Cidade/Estado/
CEP<br>
Tel.: (00) 0000-0000 <br>
</address>
</body>
</html>
```

<b>

A função original desta tag é aplicar negrito ao texto que ela estiver marcando. A letra “b” vem de bold, que é traduzido como negrito.

Mas, no HTML5, esta tag passa a ter significado semântico. Agora, ela deve ser usada em textos que não terão nenhuma importância extra, mas que estilisticamente são destacados, tais como nomes de produtos e palavras-chave de um documento.

Exemplo de uso:

```
<html>
<head>
    <title>
        Página do Silvio Ferreira
    </title>
</head>
<body>
<p> Para ajudar a associação <b>Abraço</b>
entre em contato no telefone ... </p>
</body>
</html>
```

<bdo>

Basicamente, esta tag define a direção do texto na tela, que pode ser da esquerda para a direita, ou da direita para a esquerda. Se utilizada, obrigatoriamente devemos usar o atributo dir (direção) que possui os seguintes valores:

- ltr = left to right = da esquerda para a direita;
- rtl = right to left = da direita para a esquerda.

Exemplo de uso:

```
<html>
<head>
    <title>
        Página do Silvio Ferreira
    </title>
</head>
<body>
<p>Texto Texto Texto.</p>
<p><bdo dir="rtl"> Texto Texto Texto.</
bdo></p>
</body>
</html>
```

Além do atributo dir, há também o atributo lang (linguagem) que é opcional. Com ele podemos definir o idioma do texto. Exemplo de valores: en (inglês), pt-br (português do Brasil), sa (sânscrito).

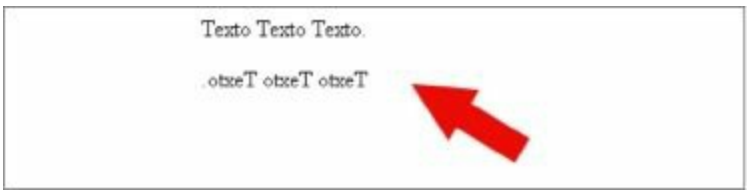


Figura 2.2.: *Exemplo de uso da tag <bdo>.*

<blockquote>

Esta tag é usada para marcar alguma seção de citação longa. São citações que foram extraídas de revistas, jornais, livros ou outros sites. Possui o atributo cite, onde podemos especificar a fonte da citação.

Exemplo de uso:

```
<html>
<head>
    <title>
        Página do Sílvio Ferreira
    </title>
</head>
<body>
<blockquote cite=" http://pt.wikipedia.org/
wiki/Coliseu_de_Roma">
O Coliseu, também conhecido como Anfiteatro Flaviano ou Flávio (em latim: Amphitl
no período da Roma Antiga. Deve seu nome à
expressão latina Colosseum (ou Coliseus, no
latim tardio), devido à estátua colossal do
imperador romano Nero, que ficava perto da
edificação. Localizado no centro de Roma, é  uma exceção entre os anfiteatros pel
volume e relevo arquitetônico. Originalmente
capaz de abrigar perto de 50.000 pessoas[1],
e com 48 metros de altura, era usado para
variados espetáculos. Foi construído a leste
do Fórum Romano e demorou entre oito a dez
anos para ser construído.
</blockquote>
</body>
</html>
```

<cite>

Esta tag deve ser usada para marcar títulos de obras, como livros, revistas, jornais, programas de TV, música, filme, escultura, pintura etc.

Exemplo de uso:

```
<html>
<head>
    <title>
        Página do Sílvio Ferreira
    </title>
</head>
<body>
<blockquote cite="http://pt.wikipedia.org/
wiki/Guinness_World_Records">
O <cite> Guinness World Records</cite> (antigo
<cite>Guinness Book of Records</cite>, lançado
em português como <cite>Livro Guinness dos Recordes</ cite>) é uma edição publica
 </blockquote>
</body>
</html>
```

<code>

Esta tag quando usada marcando um texto faz com que ele seja exibido com fonte monoespaçada. Uma fonte monoespaçada é aquela cujos caracteres possuem o mesmo tamanho.

Exemplo de uso:

```
<html>
<head>
    <title>
        Página do Sílvio Ferreira
    </title>
</head>
<body>
<code>
HTML é uma linguagem de marcação de texto.
</code>
</body>
</html>
```

Figura 2.3.: *Exemplo do uso da tag <code>.*

del

O efeito básico que esta marcação faz é riscar os textos, passando para o usuário a compreensão de que esses textos estão errados, devem ser apagados (deletados) ou não devem ser considerados.

Exemplo de uso:

```
<html>
<head>
    <title>
        Página do Sílvio Ferreira
    </title>
</head>
<body>
<del>
Este texto está riscado.
</del>
</body>
</html>
```

Figura 2.4.: *Exemplo de uso da tag <del>.*

Esta tag possui alguns atributos que podem ser usados:

- **cite**: serve para especificar um endereço (URL) onde o usuário poderá ler os motivos do texto ter sido apagado;

Exemplo de uso:

```
<del cite="saibamais.html">
Este texto está riscado.
</del>
```

- **datetime**: serve para especificar a data e hora em que o texto foi excluído. O formato é YYYYMM-DDThh:mm:ssTZD. YYYY especifica o ano (ex.: 2012); MM é o mês (Ex.: 11); DD é o dia (Ex.: 20); T é apenas um separador obrigatório; hh é hora (Ex.: 17); mm é minutos (Ex.: 09); SS é segundos (Ex.: 10); TZD (Time Zone Designator) indica a variação da hora local em relação ao horário UTC.

Exemplo de uso:

```
<del datetime="2012-11-20T17:09:10Z">
```

```
Este texto está riscado.
</del>
```

Os valores para TZD podem ser:

- **Z:** horário UTC;
- **+hh:mm**: hora local adiantada em relação ao UTC;
- **-hh:mm:** hora local atrasada em relação ao UTC.

<ins>

Esta tag pode ser usada em conjunto com a tag del. Enquanto a tag del define o texto que está sendo excluído, a tag <ins> define o texto que será inserido. Também é permitido usar os atributos cite e datetime explicados anteriormente.

Exemplo de uso:

```
<p>
Minha comida preferida é <del>bata frita</
del> <ins>peixe assado.</ins>
</p>
```

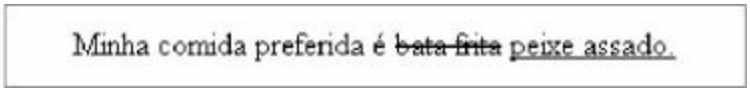


Figura 2.5.: *Resultado no navegador.*

<dfn>

Todo texto que for marcado pela tag <dfn> será interpretado como uma definição. No navegador, essa marcação resultará num texto geralmente exibido em itálico.

Exemplo de uso:

```
Gelo: <dfn> Água em estado sólido.</dfn>
```

**

Serve para dar ênfase a um texto. Essa tag deve ser usada quando desejamos aplicar negrito no texto e passar o sentido semântico de ênfase para navegadores, leitores de telas etc.

Exemplo de uso:

```
Você vai <em> Vencer!</em>
```

<i>

O significado desta tag no HTML 4.01 é simplesmente colocar o texto em itálico. No HTML5 o efeito visual no navegador é que o texto fica em itálico. Mas, semanticamente, o significado da tag <i> foi mudado.

No HTML5 a tag <i> deve ser usada em textos que devem ter destaque de entonação de voz quando lidos. Exemplos: termos técnicos, frases ou palavras em outros idiomas etc.

Exemplo de uso:

```
<p>
HTML é uma sigla inglesa da expressão
<i>Hypertext Markup Language</i>.
</p>
```

<kbd>

Podemos utilizar esta tag em situações nas quais precisamos indicar ao usuário que ele deve pressionar alguma tecla para executar uma determinada ação.

Exemplo de uso:

```
<p>
Pressione a tecla <kbd>ESC</kbd> para sair.
</p>
```

<pre>

Pré-formatação. Todo texto que estiver entre <pre> e </pre> terá a sua formatação respeitada, ou seja, da mesma forma que ele for digitado no bloco de notas, será exibido no browser. Isso inclui quebras de linha, espaços a mais entre uma palavra e outra etc.

Exemplo de uso:

```
<pre>
O conhecimento é
a base              para o desenvolvimento
de
uma             sociedade
                                Moderna.
</pre>
```

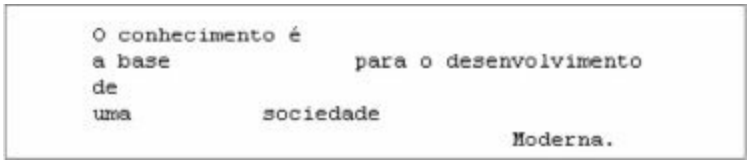


Figura 2.6.: *Resultado do uso da tag <PRE> no navegador.*

<q>

Usada para marcar citações curtas, como uma citação famosa de alguém conhecido. No navegador, a citação é exibida entre aspas.

Exemplo de uso:

```
<p><q>Triste época! É mais fácil desintegrar
um átomo do que um preconceito.</q> - Albert
Einstein </p>
```

Figura 2.7.: *Resultado do uso da tag <q> no navegador.*

Pode-se usar o atributo cite para especificar o endereço (URL) da fonte da citação, caso exista.

Exemplo de uso:

```
<p><q cite="http://www.xxxv.com.br">Triste
época! É mais fácil desintegrar um átomo do
que um preconceito.</q> - Albert Einstein </p>
```

<s>

O significado desta tag no HTML 4.01 é simplesmente texto tachado (riscado). Essa tag visualmente faz o mesmo efeito que a tag <del>, ou seja, risca o texto que ela estiver marcando. Porém, o sentido semântico no HTML5 é diferente.

Enquanto a <del> representa o sentido de um texto que está sendo apagado ou deletado (aquele texto não pode ou não deve ser usado novamente), a tag <s> indica um texto que não é mais válido, correto ou relevante. Um exemplo típico do uso dessa tag está nas promoções das lojas virtuais, como vemos na **Figura 2.8**. Perceba que nesse caso o valor que é riscado poderá ser usado novamente (ao término da promoção, por exemplo).

Figura 2.8.: *Exemplo de uso prático em uma loja virtual.*

Exemplo de uso:

```
<p>Promoção: de <s>R$99,00</s> por apenas R$58,00 </p>
```

<samp>

A tag <samp> é usada para marcar amostras de códigos fonte de programa de computador. Essa amostra de código fonte pode ser de qualquer linguagem.

Exemplo de uso:

```
<samp>
var n1 = 10; <br>
var n2 = 20; <br>
var n3 = n1 + n2; <br>
window.alert("n1 mais n2 é igua a: " + n3); <br>
</samp>
```

<small>

Esta tag reduz o tamanho dos textos que ela estiver marcando em relação aos demais textos do parágrafo.

Figura 2.9.: *Exemplo do uso da tag <small>.*

Exemplo de uso:

```
Este é um exemplo de <small>texto pequeno.
</small>
```

**

Serve para definir um texto importante, com forte ênfase. No navegador, o texto marcado por é exibido em negrito.

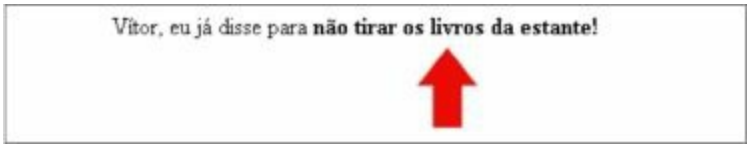


Figura 2.10.: *Exemplo do uso da tag .*

Exemplo de uso:

```
Vítor, eu já disse para <strong> não tirar
os livros da estante! </strong>
```

<sub>

Textos marcados pela tag <sub> ficam subscritos. Texto subscrito é aquele que está escrito sob outro, ou seja, seu alinhamento está mais para baixo do alinhamento de um texto normal. Um exemplo muito conhecido é o numero 2 da fórmula da água: H_2O.

Exemplo de uso:

```
H<sub>2</sub>O.
```

<sup>

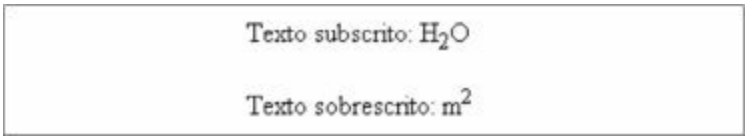


Figura 2.11.: *Texto subscrito e sobrescrito.*

Textos marcados pela tag <sup> ficam sobrescrito. Texto sobrescrito é aquele que está escrito com um alinhamento mais para cima do alinhamento de um texto normal. Por exemplo: em m2 (metro quadrado), o número dois está sobrescrito, pois, ele está escrito sobre a letra m.

Exemplo de uso:

```
m<sup>2</sup>
```

<u>

O resultado prático do uso desta tag é que o texto marcado por ela ficará sublinhado.

Exemplo de uso:

```
<u> Este texto está sublinhado. </u>
```

<var>

Esta tag pode ser utilizada para marcar variáveis dentro de um código fonte de um programa, independente da linguagem.

Exemplo de uso:

```
<script type=”text/javascript”> <br>
<var> var nome1 </var>= “Silvio Ferreira”;
<br>
window.alert(nome1); <br>
</script> <br>
```

<form>

Essa é a tag principal para se começar a criar formulários eletrônicos. Deve ser fechada por </form>.

Exemplo de uso:

```
<form>
<!—O formulário é construído aqui-->
</form>
```

<iframe>

O iframe é mais um recurso muito interessante. Um webdesigner ou webmaster experiente pode descobrir funcionalidades muito úteis desse recurso.

O iframe é um quadro que pode ser colocado em qualquer lugar de uma página. Por exemplo: você construiu uma página HTML contendo textos e imagens. Nessa página você pode colocar um quadro iframe, do tamanho e onde você quiser.

Exemplo de uso:

```
<iframe width=”400” height=”168” src=”b.
html” name=”I1” >
</iframe>
```

Os atributos válidos no HTML5 são:

- **width**: largura do iframe;
- **height**: altura do iframe;
- **name**: um nome para o iframe;
- **src**: a URL completa da página a ser carregada.

<*img*>

Para inserir uma imagem é usada a tag <img>.

Exemplo de uso:

```
<img src=”silvio.png”>
```

Observe que o local onde se encontra a imagem e o nome da imagem (mais extensão) é indicada em src=” “ (entre aspas).

Os atributos válidos em HTML5 são:

- **alt**: serve para especificar um texto alternativo para a imagem;
- **width**: largura da imagem;
- **height**: altura da imagem;
- **src**: especifica o endereço da imagem;
- **usemap**: indica que aquela imagem é mapeada com as coordenadas que virão na tag <map>.

<map>

tag usada para criar mapeamentos de imagens. Para entender melhor, imagine uma imagem do mapa do Brasil. Ela possui vários estados, como o Amazonas, Alagoas, Amapá, Bahia, Distrito Federal, Goiás, Mato Grosso, Minas Gerais, Rio de Janeiro, São Paulo etc. Será possível transformar cada um desses estados em uma área clicável, ou seja, em um link? Sim. Veja que, nesse caso, uma mesma imagem teria vários links em pontos diferentes.

Exemplo de uso:

```
<map name="mapaMenu">
<area shap="rect" coords="75, 86, 110, 105"
href="amazonas.html">
<img src="mapa.jpg" usemap="#mapaMenu">
```

O código acima se inicia com map name. Map é a tag de abertura e name é um nome que damos a ela.

Na segunda linha temos a especificação area shap="rect". O termo rect vem de rectangle, ou seja, indica que a forma (a área clicável) será um retângulo. E coords indica as coordenadas capturadas. Em src especificamos o endereço (URL).

As coordenadas são capturadas com o emprego de um programa gráfico (pode ser o Paint do Windows). Primeiro define-se uma região da figura que será clicável. Em seguida, anotam-se as coordenadas "x" e "y" superior esquerdo e "x" e "y" inferior direito. Mas espera aí, isso forma um quadrado ou retângulo! É isso mesmo, você já entendeu.

Figura 2.12.: *Anotando as coordenadas.*

Para anotar as coordenadas é simples. Abra a imagem no Paint. Selecione uma ferramenta, como o lápis, por exemplo. Posicione a ferramenta no ponto superior esquerdo. Observe que na barra abaixo, no Paint, serão mostrados dois números (**Figura 2.12**). São as coordenadas "x" e "y" desse ponto. Anote-os. Faça o mesmo com o ponto inferior direito.

Para construir os outros links basta repetir a linha <area shap="rect" coords=" " href=" "> para cada nova área, ou seja, pegue as novas coordenadas e digite a nova linha, apontando para uma nova página.

<a>

tag usada para criar hiperlinks. Os hiperlinks, que podem ser chamados somente por links, são ligações que permitem ao internauta “pular” de um ponto para outro no site. Eles podem ser de dois tipos:

- **Externo**: o link aponta para outra página, irá abrir outro arquivo.
- **Interno**: chamado de indicador, esse tipo de link aponta para uma região bem definida na mesma página onde ele se encontra. O indicador é ideal para ser usado em páginas com textos muito longos.

Existe também o link usado para abrir um programa de e-mail instalado na máquina do usuário, como o Outlook, mas, ele nada mais é que um link externo. É preciso entender que esse link não abrirá um formulário eletrônico para envio de e-mails, ele irá abrir o programa de envio de e-mails da máquina do usuário, e, se ela não tiver nenhum programa de e-mail configurado, o internauta não conseguirá enviar e-mail através deste recurso.

Links externos

A tag usada para fazer um link é <a> e esse link deverá ser fechado por . É necessário definir dois itens: quem será o link (que texto ou imagem o internauta verá no navegador como um link) e para onde esse link irá apontar (aonde o internauta será redirecionado ao clicar no link).

Primeiro é definido para onde o link aponta e isso é feito dentro da tag <a>, na frente de href.

Exemplo de uso:

```
<a href=”pagina_a.html”>
```

Lembre-se que a página “pagina_a.html” é apenas o nosso exemplo, pois o link pode apontar para qualquer outra página. Esse endereço é chamado de endereço local, ou seja, o arquivo que o link aponta está no mesmo nível de diretório da página.

Mas, ele pode ser também um endereço web. Nesse caso, ele se inicia com http://. Exemplo: http://www.google.com.br/.

Após esse procedimento, o navegador já sabe para onde o link aponta, mas, qual é o link? É preciso defini-lo para que o internauta possa clicar. Isso é feito da seguinte forma:

- Tudo que estiver entre e (o fechamento), se torna um link. Se escrevermos “Link para a página A” entre essas duas tags, esse texto se transforma em um link.

Exemplo de uso:

```
<a href=”pagina_a.html”> Clique-me. </a>
```

O link pode inclusive ser uma imagem. Para isso, basta usarmos a tag <img>.

Exemplo de uso:

```
<a href=”pagina_a.html”>
<img src=”logo.png” alt=”Estrela Informática” height=”42” width=”42”>
</a>
```

Podemos especificar também onde a página será carregada. Por exemplo: ao clicar no link o documento vinculado pode abrir na mesma janela ou em uma nova janela do navegador. Tal opção é configurada por meio do atributo target. Os valores para target são:

- **_blank**: abre o documento em uma nova janela;
- **_self**: abre o documento no mesmo quadro;
- **_parent**: abre o documento no quadro pai;
- **_top**: abre o documento em todo o corpo da janela.

Indicadores

Os indicadores são links internos. Em páginas muito extensas, o uso de indicadores é imprescindível. Nesses casos, é possível criar um índice no topo da página. Feito o índice, basta o internauta clicar no link que lhe interessar e o navegador deslizará a página até parar exatamente no item referente ao link clicado.

Para tornar possível essa operação, cada item na página deve possuir um nome. Suponhamos que o item é um texto com um título. Podemos dar um nome a esse título; fazemos isso usando a tag <a> com o atributo id.

Nota: No HTML 4.01 é usado o atributo name, mas, no HTML5 ele é incompatível. Por isso em HTML5 usa-se o atributo ID.

Exemplo de uso:

```
Clique-me para ir até o texto Vencer!
<!--Conteúdo aqui-->
<a id=”artigo1”> <h1> Vencer </h1></a>
<p> Texto Vencedor Texto Vencedor Texto Vencedor </p>
```

E para criar o link basta indicar o nome ID (artigo1). Veja no código seguinte, onde criamos um link no texto “Clique-me para ir até o texto Vencer!”. Atenção ao uso de # antes do nome ID.

Exemplo de uso:

```
<a href=“# artigo1”> Clique-me para ir até o
texto Vencer!</a>
<!--Conteúdo aqui-->
<a id=”artigo1”> <h1> Vencer </h1></a>
<p> Texto Vencedor Texto Vencedor Texto Vencedor </p>
```

Link para envio de e-mail

Como já mencionado anteriormente, é possível criar um link que ao ser clicado abre o programa de e-mail da máquina do usuário para enviar e-mails.

Exemplo de uso:

```
<a href=”mailto:contato@xxxxxxxxx.com.br?
subject=Aqui vem o assunto”>Contato</a>
```

Onde temos:

- **em href=”mailto**: você coloca o seu e-mail;
- **subject**: você coloca o assunto.

Atenção ao uso de ? separando href=”mailto: de subject:.

Colocamos o texto Contato que é onde o internauta vai clicar. Pode ser Entre em contato, Envie um

e-mail para gente etc. Por fim, fechamos o código com

<link>

A tag link é usada para carregar arquivos externos, como arquivos CSS. Quando utilizamos regras CSS importadas, os códigos digitados ficam em um arquivo externo e não no documento HTML. Com esse método é possível associar quantas páginas HTML forem necessárias a um único arquivo contendo as regras CSS. Mudando as regras desse arquivo externo, o site inteiro se altera automaticamente.

Esse arquivo externo que irá conter as regras CSS deve possuir a extensão css. Ele nada mais é que um arquivo txt com a extensão renomeada para css.

A tag <link> é digitada no cabeçalho do documento, entre as tags <head> e </head>.

Exemplo de uso:

```
<html>
<head>
    <link rel=”stylesheet” type=”text/css”
href=”default.css”>
    <title>
        Página do Sílvio Ferreira
    </title>
</head>
<body>
</body>
</html>
```

Neste exemplo podemos observar a linha: <link rel=”stylesheet” type=”text/css” href=”default.css”>

O parâmetro rel indica que o arquivo é do tipo stylesheet, ou seja, são folhas de estilo. E o parâmetro type indica que esse arquivo é do tipo texto CSS. O parâmetro href indica a URL do arquivo.

* e *

A tag <ul> inicia uma lista não numerada. Listas não numeradas usam marcadores (bullet). Cada linha da lista é construída com a tag <li>, da mesma forma que é feito nas listas numeradas.

Cada linha que contenha uma bullet deve ser iniciada por <li> e fechada por . No final de toda a lista devemos inserir a tag de fechamento .

Exemplo de uso:

```
<ul>
    <li> Uma linha da lista aqui; </li>
     <li> Outra linha aqui; </li>
     <li> Mais outra linha aqui. </li>
</ul>
```

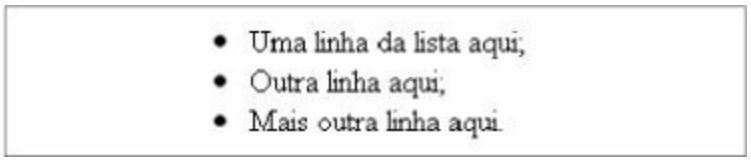


Figura 2.13.: *Uma lista não numerada.*

* e *

A tag <ol> é usada para criar listas numeradas. Da mesma forma que nas listas não numeradas, para cada linha da lista devemos usar a tag <li>.

Exemplo de uso:

```
<ol>
    <li> Primeira linha da lista aqui; </li>
     <li> Segunda linha aqui; </li>
     <li> Terceira linha aqui. </li>
</ol>
```

1. Primeira linha da lista aqui;
2. Segunda linha aqui;
3. Terceira linha aqui.

Figura 2.14.: *Uma lista numerada.*

<dl>, <dt> e <dd>

A tag <dl> inicia a construção de uma lista de definição. Deve ser fechada com a tag </dl>, indicando o fim da lista.

A tag <dt> é usada para definir os termos dessa lista. Cada termo deve ser digitado entre <dt> e </dt>.

Por fim, a tag <dd> é usada para definir a descrição dos termos. Cada descrição deve ser digitada entre <dd> e </dd>.

Exemplo de uso:

```
<dl>
    <dt> Muralha da China<dt>
    <dd>
 É uma impressionante estrutura de arquitetura militar construída durante a China
    <dd>
    <dt> Coliseu de Roma<dt>
    <dd>
 É um anfiteatro construído no período da Roma Antiga.
    <dd>
</dl>
```

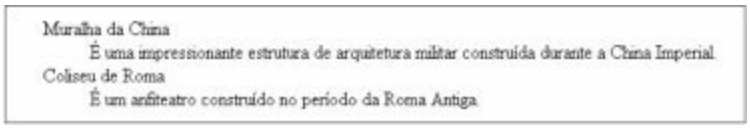


Figura 2.15.: *Uma de definição.*

<table>, <tr>, <th> e <td>

A tag <table> é usada para criar tabelas. Tabelas devem ser usadas somente para organizar dados tabulares, ou seja, qualquer tipo de informação que necessita ser apresentada em linhas e colunas de forma lógica e organizada.

Uma tabela deve ser iniciada com a tag <table> e finalizada com a tag </table>. Uma tabela é composta por linhas, células e colunas.

Para criar uma linha usamos a tag <tr> no início e </tr> no fim da linha.

Com uma linha criada, podemos criar as células. Usamos a tag <th> para criar células de cabeçalhos e <td> para as células normais.

Exemplo de uso:

```
<table>
     <tr>
        <th> Nome</th>
        <th> Telefone</th>
        <th> Valor pago</th>
    </tr>
     <tr>
        <td> João</td>
        <td> (xx) xxxx-xxxx</td>
        <td> R$22.000,00</td>
    </tr>
     <tr>
        <td> Daniel</td>
        <td> (xx) xxxx-xxxx</td>
        <td> R$42.000,00</td>
    </tr>
</table>
```

Nome	Telefone	Valor pago
João	(xx) xxxx-xxxx	R$22.000,00
Daniel	(xx) xxxx-xxxx	R$42.000,00

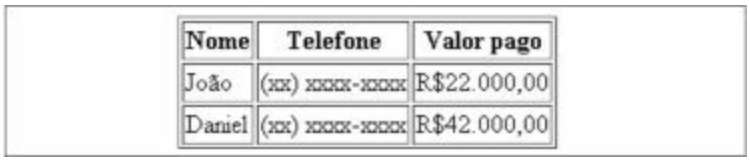
Figura 2.16.: *Exemplo de tabela.*

Diversos atributos da tag <table> existentes no HTML 4.01 não são compatíveis no HTML5. Um que pode ser usado, entretanto, é o atributo border, que especifica uma borda para as células da tabela.

Exemplo de uso:

```
<table border="1">
    <tr>
        <th> Nome</th>
        <th> Telefone</th>
        <th> Valor pago</th>
    </tr>
     <tr>
        <td> João</td>
```

```
        <td> (xx) xxxx-xxxx</td>
        <td> R$22.000,00</td>
    </tr>
     <tr>
        <td> Daniel</td>
        <td> (xx) xxxx-xxxx</td>
        <td> R$42.000,00</td>
    </tr>
</table>
```



Nome	Telefone	Valor pago
João	(xx) xxxx-xxxx	R$22.000,00
Daniel	(xx) xxxx-xxxx	R$42.000,00

Figura 2.17.: *Tabela com borda.*

<caption>

Essa tag pode ser usada para definir legendas em tabelas. Na prática, o usuário irá ver um título (ou legenda explicativa) sobre a tabela.

Exemplo de uso:

```
<table border=”1”>
<caption>Lista de Compradores</caption>
    <tr>
        <th> Nome</th>
        <th> Telefone</th>
        <th> Valor pago</th>
    </tr>
     <tr>
        <td> João</td>
        <td> (xx) xxxx-xxxx</td>
        <td> R$22.000,00</td>
    </tr>
     <tr>
        <td> Daniel</td>
        <td> (xx) xxxx-xxxx</td>
        <td> R$42.000,00</td>
    </tr>
</table>
```

Lista de Compradores

Nome	Telefone	Valor pago
João	(xx) xxxx-xxxx	R$22.000,00
Daniel	(xx) xxxx-xxxx	R$42.000,00

Figura 2.18.: *Uso de <caption>.*

e

Ambas são usadas para agrupar e estruturar elementos. São tags HTML que não possuem significado. Por exemplo: a tag <p> significa parágrafo, a tag quer dizer centralizado etc. Já as tags e são elementos neutros.

No entanto, em conjunto com classes e IDs, podem ser usadas para aplicar efeitos em partes bem definidas do seu texto. Basicamente, elas agrupam um bloco HTML e aplicam a ele algum efeito visual.

A diferença básica entre os dois é que o é usado em pequenas partes do HTML (como mudar a cor de uma palavra ou frase) e o é destinado a blocos maiores, como grupos de parágrafos inteiros.

<meta>

A tag <meta> serve para fornecer palavras-chave para os mecanismos de busca automático, além de descrever o seu site. Um exemplo de mecanismo de busca automático é o tão conhecido Google (www.google.com.br). Na verdade ela faz mais do que isso e, em resumo, serve para fornecer dados sobre o conteúdo de um documento. Veremos mais adiante quais são esses dados.

Essa tag deve ser inserida no cabeçalho da página, entre <head> e </head>. As principais tags <meta> são de descrição (description) e de palavras-chave (Keywords). Ela não precisa ser fechada e possui dois atributos:

- **name**: fornece meta informações sobre um nome exclusivo;
- **content**: texto ou outros valores a serem associados com o name.

Tag <meta> de descrição

Vamos supor que você quer colocar uma descrição do seu site. Veja no exemplo a seguir como fica o uso da tag <meta> para descrição.

Exemplo de uso:

```
<html>
<head>
<meta name="description" content="Website
pessoal do Silvio Ferreira. Aqui você encontrará informações sobre hardware, rede
programação e muito mais.">
    <title>
        Página do Silvio Ferreira
    </title>
</head>
```

Quando o seu site aparecer na busca do Google, por exemplo, essa será a descrição que vai aparecer na lista. Essa tag é muito importante.

Tag <meta> de palavras-chave

Vamos ao exemplo a seguir, no qual além da tag <meta> anterior, temos também o uso da tag keywords, ou seja, palavras-chave, que são os termos usados pelas pessoas nos mecanismos de busca. Se seu site é sobre hardware, como já exemplificado, você pode colocar a palavra-chave "hardware". Assim, quando uma pessoa digitar "hardware" no mecanismo de busca, seu site poderá aparecer na lista.

Exemplo de uso:

```
<html>
<head>
<meta name="description" content="Website
pessoal do Silvio Ferreira. Aqui você encontrará informações sobre hardware, rede
programação e muito mais.">
<meta name="keywors" content= "hardware, PC,
computador, placa-mãe, HD, RAM, memória, manutenção">
```

```
    <title>
        Página do Silvio Ferreira
    </title>
</head>
```

Tag <meta> de data de criação

Sim, é possível indicar a data de criação do documento pela tag <meta>. O formato é YYYY-MM-DDThh:mm:ssTZD. Nesse formato, YYYY especifica o ano (ex.: 2012); MM é o mês (Ex.: 11); DD é o dia (Ex.: 20); T é apenas um separador obrigatório; hh é hora (Ex.: 17); mm é minutos (Ex.: 09); SS é segundos (Ex.: 10); TZD (Time Zone Designator) indica a variação da hora local em relação ao horário UTC.

Os valores para TZD podem ser:

- **Z**: horário UTC;
- **+hh:mm**: hora local adiantada em relação ao UTC;
- **-hh:mm**: hora local atrasada em relação ao UTC.

Exemplo de uso:

```
<html>
<head>
<meta name="description" content="Website
pessoal do Silvio Ferreira. Aqui você encontrará informações sobre hardware, rede
programação e muito mais.">
<meta name="keywors" content="hardware, PC,
computador, placa-mãe, HD, RAM, memória, manutenção">
<meta name="dc.date.created" content="2012-
11-20T17:09:10Z">
    <title>
        Página do Silvio Ferreira
    </title>
</head>
```

Tag <meta> de data de expiração

Continuando a trabalhar com datas, é possível indicar também o dia em que o conteúdo expira, informando, dessa forma, ao mecanismo de busca quando ele deverá remover os documentos antigos de seu banco de dados. Veja o exemplo a seguir.

Exemplo de uso:

```
<meta http-equiv="expires" content="201212-
20T17:09:10Z">
```

Tag <meta> de refresh

Força um documento a ser recarregado ou a abertura de um novo documento depois de um número específico de segundos. Observe o exemplo a seguir. Em content="5" temos a indicação da

quantidade de segundos que vai transcorrer para o novo documento ser aberto. URL é o endereço da página.

Exemplo de uso:

```
<meta http-equiv="refresh" content="5"
url="http://www.xxx.com">
```

Tag <meta> de nome do autor

Fornece o nome do responsável ou autor da página.

Exemplo de uso:

```
<meta name="author" content="Silvio Ferreira">
```

Tag <meta> de endereço de e-mail

O autor da página ou responsável por ela pode fornecer, através desta tag, o seu e-mail de contato. Veja o exemplo a seguir:

Exemplo de uso:

```
<meta name="DC.creator.address"
content="silvio@xxxxx.com">
```

no-cache

Ao usar esta meta tag o navegador não irá armazenar a página em cache.

Exemplo de uso:

```
<meta http-equiv="pragma" content="nocache">
```

copyright

Insere informações de copyright, ou seja, informações de direito autoral da página.

Exemplo de uso:

```
<meta name="copyright" content="© 2005-
2013 Silvio Ferreira">
```

robots

Nesta meta tag hevará informações úteis a motores de buscas de sites como o Google.

Exemplo de uso:

```
<meta name="robots" content="all">
```

O atributo content pode possuir os seguintes valores:

- **all**: valor padrão. O robô de busca não receberá nenhuma informação;
- **index**: o robô de busca irá incluir a página;
- **follow**: o robô de busca irá incluir a página e seguir os links existentes nela;
- **noindex**: o robô de busca não irá incluir a página, mas poderá seguir os links existentes nela;
- **nofollow**: o robô de busca irá incluir a página, mas não poderá seguir os links existentes nela;
- **none**: o robô de busca irá ignorar a página;
- **noarchive (Google)**: não arquiva a página.

generator

Através desta meta tag podemos informar o software usado para construir a página.

Exemplo de uso:

```
<meta name="generator" content="Microsoft
Bloco de Nota 5.1">
```

revisit-after

Com esta meta tag podemos especificar um tempo para que servidores Proxy refaçam o cache da página.

Exemplo de uso:

```
<meta name="revisit-after" content="8 days">
```

rating

Esta meta tag é uma forma de classificar a página quanto à censura, ou seja, quanto às idades dos usuários.

Exemplo de uso:

```
<meta name="rating" content="general">
```

Os valores para content podem ser:

- **general**: não há censura. Pessoas de qualquer idade podem acessar a página;
- **14 years**: página recomendada para maiores de 14 anos;
- **mature**: página recomendada para maiores de 18 anos.

<base>

Esta tag é inserida no cabeçalho do documento <head> e não possui fechamento. Cada página HTML pode possuir apenas uma tag <base> em seu cabeçalho.

Ela serve para especificar a URL base de todos os demais links existentes, bem como figuras, formulários e demais elementos. Veja no exemplo como usá-la.

Exemplo de uso:

```
<html>
<head>
     <base href="http://www.nomedositeaquixxx.com/images/" target="_blank">
<link rel="stylesheet" type="text/css"
href="theme.css">
    <title>
        Página do Silvio Ferreira
    </title>
</head>
<body>
<img src="silviof.gif" width="100" height=" 100" alt="Silvio Ferreira">
</body>
</html>
```

Neste exemplo há uma tag <img> carregando na página a **Figura silviof.gif**. Neste ponto você pode estar se perguntando em qual diretório essa imagem será buscada pela tag <img>. E a resposta será no diretório base, que está informado na tag <base>: <base href="http://www.nomedositeaquixxx.com/images/" target="_blank">.

Nessa informação, vemos que além do atributo href, é usado também o atributo *target*. Ele serve para especificar o destino padrão para todos os hiperlinks, imagens etc. Nesse caso, o destinado padrão é *_blank*, ou seja, tudo será acessado, carregado e direcionado para novas janelas em branco.

<scritpt>

Esta tag é usada para digitar códigos JavaScript. Códigos JavaScript podem ser inseridos dentro da própria página HTML. Mas para isso funcionar, precisamos obedecer algumas regras. Para começar, o código JavaScript deve ser digitado entre as tag <script> e </script>. Além disso, devemos informar o tipo de script que vamos trabalhar através do parâmetro type. E qual é o tipo de script? Ora! É JavaScript!

Exemplo de uso:

```
<scritpt type=”text/javascript”>
```

Isso fará com que o navegador reconheça esses códigos como códigos JavaScript. E onde inserir o código? A melhor prática é inserir o código no cabeçalho do HTML, entre <head> e </head>. Isso fará com que o código seja carregado antes mesmo que o usuário do site tente usar o recurso em JavaScript.

Exemplo de uso:

```
<html>
<head>
    <link rel=”stylesheet” type=”text/css”
href=”theme.css”>
    <title>
        Página do Silvio Ferreira
    </title>
<script type=”text/javascript”>
var nome1 = “Silvio Ferreira”;
window.alert(nome1);
</script>
</head>
<body>
<img src=”silviof.gif” width=”100” height=” 100” alt=”Silvio Ferreira”>
</body>
</html>
```

Neste caso, o código é todo interno, ou seja, está dentro da página HTML. Mas há como importar o código de um arquivo externo.

No caso de arquivos externos, que são chamados de arquivos de JavaScript, eles devem possuir a extensão js. Esse arquivo conterá o código JavaScript e, na página HTML, iremos inserir apenas uma referência para essse arquivo externo. Para ficar mais fácil entender, vamos a um exemplo:

1. Suponhamos que temos um arquivo JavaScript que possui todo o código Javascript. Este arquivo é o código js.
2. E precisamos usar esse código em uma página HTML.
3. Nesse caso, vamos apenas criar uma referência para ele no arquivo HTML.

```
<html>
<head>
    <link rel=”stylesheet” type=”text/css”
href=”theme.css”>
```

```
    <title>
        Página do Silvio Ferreira
    </title>
<script src=”código.js”></script>
</head>
<body>
<img src=”silviof.gif” width=”100” height=” 100” alt=”Silvio Ferreira”>
</body>
</html>
```

Perceba que utilizamos o parâmetro src para digitar o nome do arquivo JavaScript. Esse parâmetro é usado para digitar não somente o nome do arquivo, mas também a sua localização completa, ou seja, em qual pasta ele se encontra.

Uso da tag <noscript>

Esta tag é usada para exibir um conteúdo alternativo, como um texto, caso o navegador do usuário não suporte JavaSscript.

Exemplo de uso:

```
<html>
<head>
    <link rel=”stylesheet” type=”text/css”
href=”theme.css”>
    <title>
        Página do Silvio Ferreira
    </title>
<script src=”código.js”></script>
<noscript>
Seu navegador não suporta JavaScript
</noscript>
</head>
<body>
<img src=”silviof.gif” width=”100” height=” 100” alt=”Silvio Ferreira”>
</body>
</html>
```

<object>

Esta tag serve para incorporar em uma página HTML objetos de mídia externos, como arquivos swf, pdf, vídeo, jpg, png etc.

Exemplo de uso:

```
<object width=”500” height=”500”
data=”silvio.swf”></object>
```

Figura 2.19.: *Arquivo SWF inserido em uma página HTML.*

Essa tag é fácil de utilizar e útil para incorporar arquivos PDF à página.

Exemplo de uso:

```
<object width=”800” height=”500”
data=”silvio.pdf”></object>
```

Figura 2.20.: *Exemplo de um arquivo PDF incorporado a uma página HTML.*

A tag <object> possui os seguintes atributos:

- **width**: largura da imagem;
- **height**: altura da imagem;
- **data**: especifica o endereço (URL) do arquivo;
- **name**: especifica um nome para o elemento <object>;

- **type**: especifica o tipo MIME dos dados passados para o objeto. Veja a seguir um exemplo de uso.

Exemplo de uso:

```
<object width=”800” height=”500”
data=”silvio.pdf” type=”application/pdf”></
object>
```

MIME Types comuns:

- Arquivos swf: application/x-shockwave-flash.
- Arquivos flv: video/x-flv.
- Arquivos pdf: application/pdf.
- Arquivos tif ou tiff: image/tiff ou image/x-tiff.
- Arquivos txt: text/plain.
- Arquivos avi: application/x-troff-msvideo, video/avi, video/msvideo ou video/x-msvideo.
- Arquivos bmp: image/bmp ou image/x-windows-bmp.
- Arquivos CSS: text/css ou application/x-pointplus.
- Arquivos gif: image/gif.
- Arquivos htm, html e htmls: text/HTML.
- Arquivos Java: text/plain ou text/x-java-source.
- Arquivos jpg: image/jpeg.
- Arquivos midi: audio/midi, audio/x-mid, audio/x-midi, application/x-midi.
- Arquivos mjpg: video/x-motion-jpeg.
- Arquivos .mov: video/quicktime.
- Arquivos .mp3: audio/mpeg3, audio/mp3.
- Arquivos .mp4 video/mp4.
- Arquivos .rm: application/vnd.rn-realmedia.
- Arquivos .wav: audio/wav.

<Capítulo 3>
Novas tags

Introdução

A criação e desenvolvimento do HTML preveem diversas novas tags, incluindo as tags estruturais. Essas tags, bem como as novas tags para criação de formulários serão estudadas nos capítulos seguintes. Neste capítulo, estudaremos uma série de novas tags criadas para marcar diversos elementos nas páginas, tais como textos, imagens, áudio e vídeo etc.

Sobre a declaração doctype

A declaração doctype não é exatamente uma novidade. Mas resolvemos comentar sobre ela somente neste capítulo porque a forma de trabalhar com ela é que é uma novidade.

Esta declaração serve para indicar o tipo de código utilizado na página, que é HTML. Ela é sempre a primeira linha de código na página. Já no HTML 4.01 essa linha era algo como o que é mostrado a seguir.

Exemplo de uso:

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0
Transitional//EN" "http://www.w3.org/TR/
xhtml1/DTD/xhtml1-transitional.dtd">
```

Como vemos, é um código enorme e nada amigável. Já no HTML5 esse mesmo código foi reduzido para a linha mostrada a seguir.

Exemplo de uso:

```
<!doctype html>
```

Prático, não?

O elemento HTML

Esta é a primeira tag digitada ao criarmos documentos HTML. Você pode até questionar: Mas Silvio, a primeira tag não seria a <!doctype html>? Como já dissemos, <!doctype html> é uma declaração e não uma tag.

O código HTML é composto por diversos elementos em árvore, em que alguns são filhos de outros. A tag <HTML> é o início dessa árvore, mais exatamente é a tag <html>. É a partir desse ponto que o navegador começará a realizar a interpretação da página e exibição do conteúdo na tela.

Essa tag desempenha o papel de definição da linguagem padrão do documento (além de iniciar essa grande árvore, claro). Isso é feito através do atributo lang.

Exemplo de uso:

```
<html lang="pt-br">
```

Veja alguns idiomas:

- Português Brasil: pt-br.
- Português Portugal: pt-pt.
- Inglês Internacional: en.
- Inglês Americano: en-us.
- Espanhol: es.
- Italiano: it.
- Francês: fr.
- Alemão: de.

meta charset

Através da tag meta charset defimos a tabela de caracteres que a página está utilizando. É graças ao uso correto da tabela de caracteres que a acentuação e os caracteres especiais serão exibidos corretamente na página. A meta charset deve ser digitada no cabeçalho do documento, entre <head> e </head>.

Exemplo de uso:

```
<head>
    <meta charset=”iso-8859-1”>
</head>
```

<mark>

Esta tag produz um efeito bem interessante nos textos. Mark significa marcar. E é exatamente isso que essa tag faz, ou seja, marcar o texto. Ela marca, destaca uma frase ou até mesmo uma palavra na cor amarela. É como se estivéssemos usando uma caneta marca-texto, aquelas compradas em qualquer papelaria.

Exemplo de uso:

```
<p> texto sem destaque</p>
<p><mark> Texto destacado</mark> </p>
```

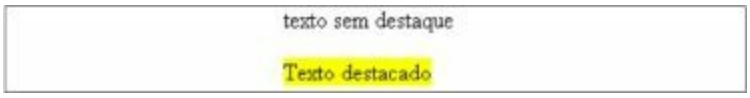


Figura 4.1.: *Texto marcado com <Mark>.*

<meter>

Usada para criar e definir medições. O resultado prático é que o usuário verá um pequeno gráfico na página, semelhante a uma barra de progresso. Mas não podemos usar esta tag para construir barras de progresso (para isso existe a tag <progress>). Ela é usada se baseando em valores pré-definidos: um valor mínimo, um valor máximo e um valor atual.

Exemplo de uso:

```
<p> Estoque</p>
<p> HDs <meter value=”8” min=”0”
max=”10”>Teste 1</meter></p>
<p> CPU <meter value=”5” min=”0”
max=”10”>Teste 1</meter></p>
<p> Pasta Térmica <meter value=”2” min=”0”
max=”10”>Teste 1</meter></p>
<p> Mouse <meter value=”10” min=”0”
max=”10”>Teste 1</meter></p>
```

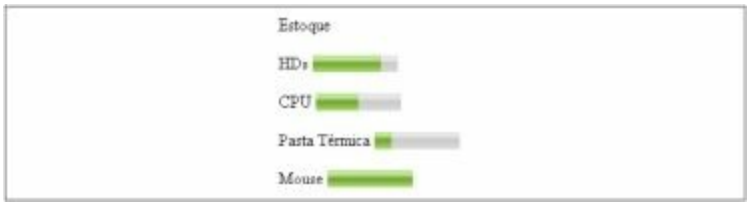


Figura 4.2.: *Exemplo do uso de <meter>.*

Como vemos, há alguns atributos importantes a saber:

- **value**: é aqui que definimos o valor atual do indicador gráfico;
- **min**: especifica o valor limite mínimo;
- **max**: especifica o valor limite máximo.

<progress>

Esta é a tag usada para criar barras de progresso. Com ela, podemos construir barras que vão representar o progresso de uma tarefa. Se você leu o tópico anterior, percebeu que dissemos que não podemos usar a tag <meter> para representar barras de progresso. Isso pode ocorrer graças à semelhança visual que <meter> possui com uma barra de progresso. Mas, se isso for feito, o código perde sua semântica, pois estaremos usando uma tag que possui um objetivo para expressar uma finalidade totalmente diferente.

Portanto, barras de progresso devem ser construídas com a tag <progress>. Barras de progresso são componentes gráficos utilizados para representar o progresso de uma determinada tarefa. Na web é muito comum vê-las na transferência de arquivos, carregamentos de páginas etc.

Exemplo de uso:

```
<p> Curso de HTML5 - Silvio Ferreira - Meu
Progresso</p>
<progress value=”22” max=”100”></progress>
```

Figura 4.3.: *Exemplo de uso de <progress>.*

Possui alguns atributos a saber:

- **max:** especifica o valor total da tarefa;
- **value:** especifica o quanto já foi concluído.

<time>

Sempre que no texto houver datas ou horas, podemos usar a tag <time> para marcá-las e defini-las.

Exemplo de uso:

```
<p> Chegaremos às <time>11:30</time></p>
```

Neste exemplo, a data é dada mediante o próprio conteúdo. Mas, podemos dar a data por meio de código, dentro da tag <time>. Para isso, podemos utilizar o atributo datetime para especificar data e hora. Esse atributo é útil em vários casos, como, por exemplo, especificar uma data e/ou hora em que um artigo foi incluído ou excluído. E essa informação será usada por buscadores, tradutores etc.

- **datetime:** o formato é YYYY-MM-DDThh:mm:ssTZD. Nesse formato, YYYY especifica o ano (ex.: 2012); MM é o mês (ex.: 11); DD é o dia (ex.: 20); T é apenas um separador obrigatório; hh é hora (ex.: 17); mm é minutos (ex.: 09); SS é segundos (ex.: 10); TZD (Time Zone Designator) indica a variação da hora local em relação ao horário UTC.

Exemplo de uso:

```
<p> Artigo inserido por Silvio -<Time
datetime=”2012-11-20T17:09:10Z”> 20-11-2012
</time></p>
```

Os valores para TZD podem ser:

- **Z:** horário UTC.
- **+hh:mm**: hora local adiantada em relação ao UTC.
- **-hh:mm:** hora local atrasada em relação ao UTC.

Caso a data represente a data de publicação do conteúdo em questão, podemos usar ainda o atributo pubdate.

Exemplo de uso:

```
<p> Artigo inserido por Silvio -
<Time datetime=”2012-11-20T17:09:10Z”
pubdate=”pubdate”> 20-11-2012
</time></p>
```

<wbr>

A tag <wbr> é uma opção para especificar oportunidades de quebra de linhas em palavras. É um recurso que em editores de texto se chama hifenização. Ocorre quando uma palavra não cabe na linha e precisa ser dividida, ficando uma parte em uma linha e a outra parte na linha logo abaixo.

Ao usar <wbr> em uma palavra, esta será dividida somente se for necessário. A tag <wbr> não precisa ser fechada. Ela apenas informa um ponto (em uma palavra) onde pode ocorrer uma quebra de linha.

Exemplo de uso:

```
<p> Genu<wbr>flexório: Peça de madeira fixa
ou móvel usado nas igrejas, geralmente na
parte de trás do banco, para que os fiéis
possam ajoelhar-se e rezar.</p>
```

<figure>, <figcaption>

Figuras. Marcam e especificam imagens em geral, tais como fotos, desenhos, diagramas, ilustrações etc. Podemos usá-las com ou sem legendas.

Uma dúvida muito comum naqueles que estão iniciando seus estudos em HTML5 é se <figure> seria uma tag que substituiria a <img>. A resposta é não. A tag <img> é usada normalmente para carregar as imagens na página.

Exemplo de uso:

```
<figure>
    <img src=”silvio.jpg” alt=”Silvio Ferreira”
    width=”288” height=”432”>
</figure>
```

Para especificar uma legenda, usamos a tag <figcaption>.

Exemplo de uso:

```
<figure>
    <img src=”silvio.jpg” alt=”Silvio Ferreira”
    width=”288” height=”432”>
<figcaption> Minha Foto </figcaption>
</figure>
```

<canvas>

A tag <canvas> cria um recipiente para criação de gráficos através de scrips, tal como JavaScript. Precisamos entender que <canvas> apenas cria a área de desenho. Quem construirá os gráficos propriamente ditos é o JavaScript. Por isso, <canvas> possui apenas dois atributos:

- **height:** altura da área de desenho;
- **width:** largura da área de desenho.

Mas graças à <canvas> é possível construir gráficos diversos, jogos para web, efeitos dinâmicos em interfaces de usuário, editores gráficos e de código, efeitos 3D etc.

Exemplo de uso:

```
<canvas id="jk87"></canvas>
<script type="text/javascript">
var canvas=document.getElementById('jk87');
var ctx=canvas.getContext('2d');
ctx.fillStyle='#FF0000';
ctx.fillRect(22,45,180,180);
</script>
```

<audio>

Permite a inserção de arquivo de áudio em uma página web para que ele seja reproduzido. Pode ser uma música, áudio de alguma narração etc. Em geral, arquivos de áudio são usados como som de fundo. Os formatos mais comuns de arquivos de áudio executados na web são o .mp3 e .wav. Por questões de compatibilidade com qualquer sistema operacional e qualquer navegador, indicamos o formato .mp3.

Exemplo de uso:

```
<audio src="audio.mp3" controls="controls">
Seu navegador não suporta o elemento audio.
</audio>
```

Figura 4.4.: *Uso de <áudio> com controle.*

A tag <audio> possui atributos que podem ser usados. São eles:

- **src**: especifica o local (URL) onde se encontra o arquivo;
- **autoplay**: faz com que o arquivo de áudio seja reproduzido automaticamente assim que a página for carregada;
- **controls**: insere um controle com botões play, pause e controle de volume;
- **loop**: faz com que o áudio seja reproduzido novamente sempre que chegar ao fim;
- **preload**: significa pré-carregamento. Controla o comportamento do carregamento do arquivo de áudio enquanto a página é carregada. Os valores são: none (o navegador não carrega o arquivo de áudio enquanto a página é carregada), metadata (carrega os metadados enquanto a página é carregada) e auto (carrega o arquivo de áudio enquanto a página é carregada);
- **type**: especifica o tipo MIME dos dados passados para o objeto. Veja a seguir um exemplo de uso.

MIME Types de arquivos de áudios comuns:

- Arquivos .mp3: audio/mpeg3, audio/mp3.
- Arquivos .wav: audio/wav.

Exemplo de uso:

```
<audio src="audio.mp3" controls="controls"
autoplay="autoplay" loop="loop"
preload="none" type="audio/mp3">
Seu navegador não suporta o elemento audio.
</audio>
```

<source>

Uma situação que pode ocorrer é o navegador do usuário não suportar o formato de arquivo de áudio utilizado. Para resolver isso facilmente, nada mais justo que fornecer o mesmo arquivo de áudio em formatos alternativos. Desta forma, o navegador pode escolher o formato que ele é capaz de reproduzir. Isso é feito através da tag <source>.

Exemplo de uso:

```
<audio controls=”controls”
autoplay=”autoplay” loop=”loop”
preload=”none”>
   <source src=”audio.mp3” type=”audio/mp3”>
   <source src=”audio.mp3” type=” audio/wav”>
Seu navegador não suporta o elemento audio.
</audio>
```

Nota: veja que os parâmetros src e *type* ficam na tag <source>. Os demais em <audio>.

<video>

No tópico anterior abordamos a tag <audio> para inserção de arquivos de áudio em uma página web. Mas, será que o HTML5 possui uma nova tag para inserir vídeo? Sim. Os vídeos são um tipo de mídia cada vez mais usados em websites. Depois da explosão de grandes portais que fornecem o serviço de armazenamento de vídeos, como o Youtube (http://www.youtube.com/), o uso de vídeos na web se popularizou muito. E nada mais natural que o HTML5 fornecesse um suporte melhor aos vídeos.

A nova tag par inserir arquivos de vídeo em páginas web é a <vídeo>. Seu uso é tão simples quanto a tag <audio>. Os formatos de arquivos de vídeo mais comuns para serem usados na web são: flv, mov, MP4, AVI. E, evidentemente, o computador do usuário deve ter todos os softwares e plugins necessário para "rodar" cada formato. Geralmente, o formato AVI e MP4 é suportado por qualquer sistema operacional de forma nativa. O flv necessita do plugin do flash; e o mov necessita do QuickTime.

Veja agora, no exemplo a seguir, a sintaxe básica para inserir um vídeo em uma página web.

Exemplo de uso:

```
<video src="silvio.mp4" width="320" height=" 240" controls="controls">
Seu navegador não suporta o elemento video.
</video>
```

Os parâmetros que podemos usar são:

- **src:** especifica o local (URL) onde se encontra o arquivo;
- **height:** altura do vídeo;
- **width:** largura do vídeo;
- **autoplay:** faz com que o arquivo de vídeo seja reproduzido automaticamente assim que a página for carregada;
- **controls:** insere um controle com botões play, pause e controle de volume, entre outros;
- **loop:** faz com que o vídeo seja reproduzido novamente sempre que chegar ao fim;
- **preload:** significa pré-carregamento. Controla o comportamento do carregamento do arquivo de vídeo enquanto a página é carregada. Os valores são: none (o navegador não carrega o arquivo de vídeo enquanto a página é carregada), metadata (carrega os metadados enquanto a página é carregada) e auto (carrega o arquivo de vídeo enquanto a página é carregada);
- **muted:** especifica que o vídeo, ao iniciar, ficará em estado mudo (sem áudio);
- **poster:** indica uma imagem para ser exibida enquanto o vídeo é carregado ou enquando não for iniciado;
- **type:** especifica o tipo MIME dos dados passados para o objeto. Veja a seguir um exemplo de uso.

MIME Types de arquivos de vídeo comuns:

- Arquivos flv: video/x-flv.
- Arquivos avi: application/x-troff-msvideo, video/avi, video/msvideo ou video/x-msvideo.
- Arquivos .mov: video/quicktime.

- Arquivos mp4 video/mp4.

Veja agora, no exemplo, a sintaxe com o uso de todos os parâmetros.

Exemplo de uso:

```
<video src=”silvio.mp4” width=”320”
height=”240” autoplay=”autoplay”
controls=”controls” loop=”loop”
muted=”muted” poster=”ferreira.gif”
preload=”none” type=”video/mp4”>
Seu navegador não suporta o elemento video.
</video>
<source>
```

<source>

A tag <source> pode ser usada, da mesma forma que ocorre ao trabalhar com áudios, para fornecer ao navegador opções de escolha. Podemos colocar à disposição o arquivo de vídeo em formatos diferentes, para que o próprio navegador escolha aquele que for mais compatível.

Exemplo de uso:

```
<video width=”320” height=”240”
autoplay=”autoplay” controls=”controls”
loop=”loop” muted=”muted” poster=”ferreira.
gif” preload=”none”>
<source src=”silvio.mp4” type=”video/mp4”>
<source src=”silvio. avi” type=” video/avi”>
Seu navegador não suporta o elemento video.
</video>
```

<track>

Esta tag possui uma função muito útil. Com ela poderemos definir faixas de textos, como legendas, para a tag <video> ou <audio>. O problema, é que no exato momento em que escrevo este curso (meados de 2011), ela não é compatível com nenhum navegador. Mas, vale à pena conhecer essa tag, acompanhar a adoção inevitável do HTML5 e ficar sempre na torcida para que todos os navegadores adotem o HTML5 em sua total plenitude.

Exemplo de uso:

```
<video width=”640” height=”480”
controls=”controls”>
    <source src=”silvio.mp4” type=”video/mp4”>
    <source src=”silvio. avi” type=” video/avi”>
    <track src=”legenda_en.vtt”
kind=”subtitles” srclang=”en”
    label=”English”>
    <track src=”legenda_br.vtt”
kind=”subtitles” srclang=”pt”
    label=”Português”>
</video>
```

Os atributos da tag <track> são:

- **src:** indica o endereço (URL) da legenda;
- **kind:** especifica o tipo de texto. Os tipos são: captions, chapters, descriptions, metadata, subtitles;
- **srclang:** especifica o idioma da legenda;
- **label:** especifica o título.

<embed>

A tag <embed> define um recipiente para inserção de conteúdos e aplicações externas. O uso é bem simples, veja no exemplo.

Exemplo de uso:

```
<embed src="silvio.swf">
```

Os atributos são:

- **src:** especifica o local (URL) onde se encontra o arquivo;
- **height:** altura;
- **width:** largura;
- **type:** especifica o tipo MIME.

<Capítulo 4>
Novas tags estruturais

Introdução

Programar com HTML5 requer uma nova forma de pensar. É preciso pensar em cada parte de uma página e como defini-las via código. Uma página HTML ao ser construída, por mais simples que seja, irá possuir algumas partes bem distintas. E para padronizar essas partes, o HTML5 possui novas tags estruturais: <section>, <header>, <aside>, <nav> e <footer>. Todas essas novas tags criam uma seção e possuem significado semântico.

No HTML 4.01 havia a tag (que ainda é utilizada no HTML5), que agrupava blocos de conteúdos e aplicava neles alguma formatação via CSS. Vemos um exemplo na **Figura 5.1**.

Figura 5.1.: *A estrutura de uma página no HTML 4.01.*

Mas, o problema é que essas tags não possuem um significado semântico. Para um navegador, um tradutor, um leitor de tela ou um buscador, tudo é div. Veja isso na **Figura 5.2**.

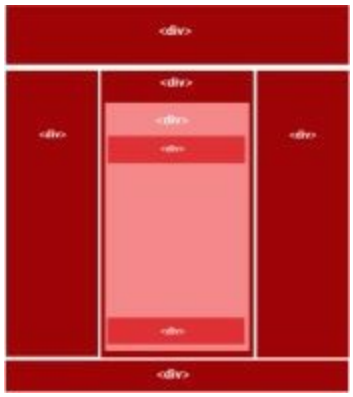


Figura 5.2.: *Tudo é div!*

Agora, no HTML5, cada parte ganha sentido. Um motor de busca (do Google, por exemplo) ao

analisar o código saberá onde está o cabeçalho, o conteúdo principal, o menu etc. Para entender melhor, veja como ficou a estruturação de uma página com HTML5 na **Figura 5.3**.

Figura 5.3: *Estrutura de uma página com HTML5.*

Perceba que cada parte possui uma tag específica. Perceba que nesse caso não estão sendo usadas tags , mas, entenda que as não foram excluídas do HTML5. Elas podem ser usadas, como containers genéricos, para marcar conteúdos que serão estilizados com CSS.

Além disso, nas regras CSS podemos usar como seletor cada tag estrutural do HTML5.

Exemplo de uso:

```
nav {
background: # 086aa3;
width: 200px;
position: absolute;
padding: 3px;
border: 1px none black;
margin: 0px;
top: 180px;
left: 700px;
height: 300px;
}
```

Nos tópicos seguintes há uma abordagem de cada nova tag estrutural do HTML5 e exemplos de uso isoladamente.

<header>

Esta tag estrutural serve para marcar seções de cabeçalho. No HTML5, textos, artigos e demais conteúdos que possuam pelo menos um título principal passam a ter um cabeçalho.

É preciso não confundir, entretanto, este cabeçalho com o cabeçalho do documento, onde é definido o título da página. Este cabeçalho é definido pela tag <head>, e é o cabeçalho principal da página. Esse tipo de cabeçalho já foi estudado anteriormente neste curso.

Já os cabeçalhos marcados pelas tags <header> estarão no conteúdo da página. E uma mesma página pode possuir um ou mais cabeçalhos. Nesses cabeçalhos estão incluídos: títulos, datas, horas, prévia do texto etc.

Exemplo de uso:

```
<header>
<h1> Sobre o Autor <h1>
<p> Artigo inserido por Silvio -
<Time datetime=”2012-11-20T17:09:10Z”
pubdate=”pubdate”> 20-11-2012</time></p>
<p> Neste Artigo Veremos Informações sobre o
autor Silvio</p>
</header>
```

A página pode possuir um cabeçalho principal, contendo o nome da empresa, logomarca, entre outras informações. E os artigos nela existentes também podem possuir cabeçalhos, contendo o título do artigo, datas etc.

<aside>

Serve para marcar seções de conteúdo relacionado. O conteúdo relacionado em uma página é qualquer texto, link, imagem ou outro item que tenha alguma relação ou ligação com o conteúdo (artigo) principal da página.

Pode ser anúncios, banners, menus secundários, links diversos (cadastre-se aqui, indique nosso site, receber newsletter, procurar etc.), pequenos textos, enfim, qualquer tipo de conteúdo que tenha relação com o conteúdo principal.

Logo, cada página poderá ter o seu próprio conteúdo relacionado. Não é regra. Se for necessário, todas as páginas do site podem possuir exatamente o mesmo tipo de conteúdo relacionado.

Exemplo de uso:

```
<aside>
<!--Conteúdo Relacionado Aqui-->
</aside>
```

<footer>

Esta tag serve para definir uma seção de rodapé. O rodapé é um conceito já usado no desenvolvimento de páginas para web antes do HTML5. Porém, era algo não definido como deveria ser. Não existia uma tag semântica para definir o rodapé de uma página. E no rodapé há informações importantes, principalmente para motores de busca (tal como do site Google), tais como as informações de direitos autorais.

Por isso, o HTML5 prevê uma tag para rodapés, que é a <footer>. E essa tag pode ser usada para marcar o rodapé principal da página ou rodapés de artigos.

Exemplo de uso:

```
<footer>
<p> Todos os direitos reservados © - Silvio
Ferreira </p>
<p> <a href=”#”> Entre em Contato </a>
</footer>
```

<article>

Define seções de conteúdos do tipo artigos. São textos, matérias, enfim, conteúdos compostos por títulos, parágrafos, imagens, vídeos etc. Uma seção <article> pode conter outras seções, como <header> e <footer>.

Exemplo de uso:

```
<article>
    <header>
        <h1> Sobre o Autor <h1>
        <p> Artigo inserido por Silvio
- <Time datetime=”2012-11-20T17:09:10Z”
pubdate=”pubdate”> 20-11-2012</time> </p>
        <p> Neste Artigo Veremos Informações
sobre o autor Silvio</p>
        </header>
<!--Conteúdo Aqui-->
<footer>
        <p> Todos os direitos reservados © -
Silvio Ferreira </p>
        <p> <a href=”#”> Entre em Contato </a>
        </footer>
</article>
```

<section>

Esta tag marca uma seção genérica. Para entendermos melhor, veja que as tags <header> e <aside> possuem um significado muito bem definido: <header> é cabeçalho e <aside> é conteúdo relacionado. Já a <section> não possui um significado semântico. Ela é genérica. Serve para marcar uma seção genérica, composta por conteúdos diversos. A <section> pode, inclusive, agrupar outras seções.

Exemplo de uso:

```
<section>
<article>
    <header>
        <h1> Sobre o Autor <h1>
        <p> Artigo inserido por Silvio
- <Time datetime=”2012-11-20T17:09:10Z”
pubdate=”pubdate”> 20-11-2012</time> </p>
                <p> Neste Artigo Veremos Informações sobre o autor Silvio</p>
    </header>
<!--Conteúdo Aqui-->
<footer>
    <p> Todos os direitos reservados © -
Silvio Ferreira </p>
    <p> <a href=”#”> Entre em Contato </a>
    </footer>
</article>
</section>
```

<nav>

Lembra navegação (navigation), não é verdade? E é exatamente essa a função dela. Esta tag serve para marcar uma seção de navegação, ou seja, um menu de navegação. Esse menu pode ser construído com links em modo texto ou imagens. Mas, não é um link qualquer, ou um pequeno grupo de links, que pode ser considerado um elemento pertencente à <nav>. Tem de ser os links principais, que formam um menu principal, um menu que dará acesso a outras páginas.

Exemplo de uso:

```
<nav>
<!—Links do menu aqui-->
</nav>
```

Menus podem ser construídos em outras seções, tais como na seção marcada por <aside>. Suponhamos que na seção de conteúdo relacionado haja um menu secundário. Dessa forma, esse menu pode (e deve) ser marcado por <nav>.

Exemplo de uso:

```
<aside>
<!--Conteúdo Relacionado Aqui-->
    <nav>
    <!—Links do menu secundário aqui-->
    </nav>
</aside>
```

Uma página básica

Para finalizar este capítulo, vejamos na tabela a seguir um código básico no qual são utilizadas todas as tags mencionadas até este ponto.

Exemplo de uso:

```
<!doctype html>
<html lang=”pt-br”>
<head>
    <meta charset=”iso-8859-1”>
    <title>Silvio Ferreira</title>
    <link href=”default.css”
rel=”stylesheet” type=”text/css”>
</head>
<body>
<header>
<h1> Página Pessoal do Autor Silvio>
</header>
<aside>
<!--Conteúdo Relacionado Aqui-->
    <nav>
    <!--Links do menu secundário aqui-->
    </nav>
</aside>
<section>
<article>
    <header>
        <h1> Sobre o Autor <h1>
        <p> Artigo inserido por Silvio
- <Time datetime=”2012-11-20T17:09:10Z”
pubdate=”pubdate”> 20-11-2012</time> </p>
        <p> Neste Artigo Veremos Informações
sobre o autor Silvio</p>
    </header>
<!--Conteúdo Aqui-->
<footer>
     <p> Todos os direitos reservados © -
Silvio Ferreira </p>
    <p> <a href=“#”> Clique para acessar o
próximo Artigo </a>
     </footer>
</article>
</section>
<nav>
<!—Links do menu aqui-->
</nav>
<footer>
<p> Todos os direitos reservados © - Silvio
Ferreira </p>
<p> <a href=”#”> Entre em Contato </a>
</footer>
</body>
</html>
```

<Capítulo 5>
Formulários

Introdução

Um formulário é uma parte do site que serve, geralmente, para captar informações dos usuários de um site ou de um serviço. Exemplos de formulários:

- Formulário de cadastro.
- Formulário de contato.
- Questionários.
- Pesquisas de satisfação.
- Formulário para cadastro de sites em mecanismos de busca.
- Formulário para cadastro em sistemas de e-mails.
- Formulários de orçamentos.

Bom, agora que já temos a definição, vamos às principais tags usadas para permitir a introdução de dados sob várias formas. Estudaremos aqui as formas de se construir um formulário por meio de códigos HTML, criando assim os campos visíveis pelo usuário, mas é preciso ressaltar que um formulário em HTML puro não funciona sozinho. Para que ele possa realmente enviar as informações pela web, é necessário empregar também o uso de linguagens dinâmicas tais como o PHP, CGI etc.

Os formulários HTML sofreram uma verdadeira revolução. No HTML5 há muito mais recursos para criação de formulários do que no HTML 4.01. Eles contam agora com novos tipos de campos e recursos. Um grande avanço sem dúvida alguma são os campos capazes de realizar a validação de dados sem a necessidade de intervenção de códigos JavaScript ou de outra linguagem. Há, por exemplo, campos para e-mails que já fazem nativamente a validação de e-mails. Nos tópicos seguintes estudaremos esses novos campos.

<form>

Para criar formulário devemos sempre começar com esta tag. Ela é a tag principal.

Exemplo de uso:

```
<form action=" form.php" method="get">
</form>
```

Possui alguns parâmetros básicos:

- **action:** é nesse atributo que informamos onde (URL) o formulário será processado;
- **autocomplete:** ativa (on) ou desativa (off) o recurso autocomplete. O autocomplete se baseia em valores que o usuário digitou anteriormente;
- **method:** especifica como será submetido o formulário, indica qual método de http será escolhido para enviar o conteúdo do formulário. São duas as possibilidades:
- **BB= POST:** a informação recolhida no formulário será incluída no corpo da página e enviada para o destino especificado em action.
- **BB= GET:** nesse caso, será adicionado à URL especificado pelo atributo action com um ponto de interrogação como separador e, em seguida, a URL é enviada. Escolhendo esse método, ao submeter o formulário podemos ver os dados enviados e até modificá-los. Veja na sequência um exemplo disso.

- **name:** este é fácil deduzir para que serve. Sua utilidade é nomear (entre aspas) uma determinada variável que receberá e armazenará o conteúdo de um campo;
- **novalidate:** especifica que o formulário não deve ser válido quando for enviado;
- **target:** especifica onde a resposta de envio do formulário deve ser exibida. Os valores são: _blank, _self, _parent e _top.

<button>

Há duas formas de criar botões em formulários. A primeira é por meio da tag <button> e a segunda é por meio da tag <input> (veremos isso mais adiante).

Botões são elementos básicos em um formulário. É por meio deles que iremos clicar e enviar o formulário para ser processado por um arquivo (pelo atributo action).

Exemplo de uso:

```
<form action=" form.php" method="get">
<button type="submit">Enviar</button>
</form>
```

Como vemos, o texto que aparece escrito dentro do botão é o texto que digitamos entre <button> e </button>. Vejamos alguns atributos:

Figura 6.1.: *Botão construído com <button>.*

- **type:** especifica o tipo de botão. Os valores são: button (um botão clicável), reset (botão que apaga tudo que estiver digitado nos campos) e submit (botão de envio de formulários);
- **name:** especifica um nome para o botão;
- **value:** serve para definir o valor para o botão;
- **formtarget:** especifica onde a resposta de envio do formulário deve ser exibida. Os valores são: _blank, _self, _parent e _top. Usar apenas quando o atributo type for submit;
- **formaction:** especifica onde (URL) o formulário será processado. Usar apenas quando o atributo type for submit;
- **autofocus:** especifica que o botão deve receber foco automaticamente assim que a página for aberta;
- **disabled:** mantém o botão desativado;
- **formmethod:** especifica como será submetido o formulário, indica qual será o método de http escolhido para enviar o conteúdo do formulário: get ou post. Usar apenas quando o atributo type for submit;
- **formnovalidate:** especifica que o formulário não deve ser válido quando for enviado. Usar apenas quando o atributo type for submit.

<input>

Permite a criação de campos. Esses campos serão usados pelo usuário para entrar com algum tipo de informação. Alguns atributos básicos:

- **value:** define o valor. Esse atributo é utilizado de forma diferente, de acordo com o tipo de campo. Por exemplo: para botões, value define o texto que é exibido no botão. Para caixas de textos, value define o valor padrão inicial. Exemplo: value=”Clique-me” (Para type=”button”);
- **autocomplete:** ativa (on) ou desativa (off) o recurso autocomplete. Exemplo: autocomplete=”on”;
- **autofocus:** especifica que o botão deve receber foco automaticamente assim que a página for aberta. Exemplo: autofocus=”autofocus”;
- **disabled:** mantém o objeto desativado. Exemplo: disabled=”disabled”;
- **name:** especifica um nome para o objeto. Exemplo: name=”btreset”;
- **min e max:** define valores mínimo e máximo para um determinado elemento input. Exemplo: min=”1” max=”5”.
- **maxlength:** define o número máximo de caracteres permitido em um determinado elemento input. Exemplo: maxlength=”100”;
- **placeholder:** insere um texto descritivo ou uma dica em campos de preenchimento (campos onde o usuário deve digitar informações). Exemplo: placeholder=”Digite Seu Nome Aqui”;
- **required:** faz com que um campo seja de preenchimento obrigatório. Exemplo: required=”required”.

Além desses atributos, outro muito importante é o type. Por meio do atributo type podemos determinar o tipo de objeto, ou seja, o tipo de campo (valores para type). **São eles: button**, checkbox, color, date, datetime, datetime-local, email, file, hidden, image, month, number, password, radio, range, reset, search, submit, tel, text, time, URL e week.

button

Define um botão clicável.

Exemplo de uso:

```
<input type="button" value="Clique-me">
```

Figura 6.2.: *Botão construído com input.*

submit

Também define um botão, porém, do tipo submeter. Um botão submeter é usado para “enviar” o formulário.

Exemplo de uso:

```
<input type=”submit”>
```

checkbox

Permite construir caixas de seleção. Com elas, podemos construir campos de múltipla escolha. Os atributos principais são name (define o nome) e value (define o valor que é enviado quando clicar no botão enviar).

Exemplo de uso:

```
<form>
<p> O que você prefere?</p>
<input type=”checkbox” name=”prefiro”
value=”cinema”> Prefiro cinema<br>
<input type=”checkbox” name=”prefiro”
value=”praia”> Prefiro praia <br>
<input type=”checkbox” name=”prefiro”
value=”natureza”> Prefiro natureza<br>
<input type=”checkbox” name=”prefiro”
value=”Casa”> Prefiro ficar em casa <br
<button type=”submit”>Enviar</button><br>
</form>
```

É permitido usar o atributo checked, que serve para especificar que um determinado elemento deve ficar pré-selecionado.

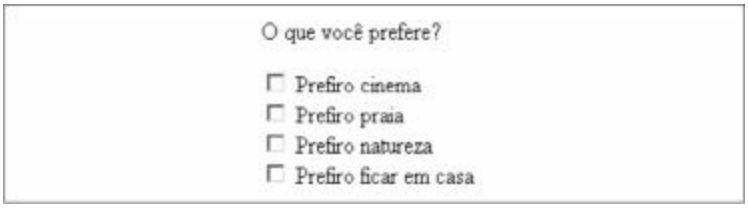


Figura 6.3.: *Uso de checkbox.*

Exemplo de uso:

```
<form>
<p> O que você prefere?</p>
<input type=”checkbox” name=”prefiro”
value=”cinema”> Prefiro cinema<br>
<input type=”checkbox” name=”prefiro”
value=”praia”> Prefiro praia <br>
<input type=”checkbox” name=”prefiro”
value=”natureza” checked=”checked”> Prefiro
natureza<br>
<input type=”checkbox” name=”prefiro”
value=”Casa”> Prefiro ficar em casa <br
<button type=”submit”>Enviar</button><br>
</form>
```

Neste exemplo, observe que o input type=”checkbox” com value natureza possui o atributo checked. Ao abrir este formulário em um navegador, a opção *Prefiro natureza* estará pré-selecionada.

Importante notarmos que o nome (valor do atributo name) tem que ser igual em todas as alternativas se elas (as alternativas) fizerem parte da mesma questão (pergunta). E o value tem que ser diferente em cada questão, pois, é ele que define e distingue o valor de cada entrada.

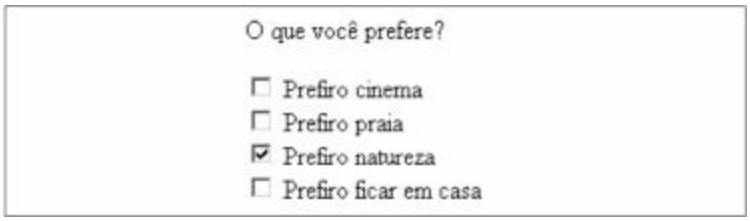


Figura 6.4.: *Opção pré-selecionada.*

radio

Continuando nossos estudos com campos do tipo escolha, veremos agora sobre o valor radio. Radio é um valor para o atributo type que permite construir questões com várias opções, mas em que é possível escolher somente uma resposta.

Exemplo de uso:

```
<form>
<p> Quanto é 1+1?</p>
<input type=”radio” name=”prefiro”
value=”r1”> 1<br>
<input type=”radio” name=”prefiro”
value=”r2”> 2<br>
<input type=”radio” name=”prefiro”
value=”r3”> 3<br>
<input type=”radio” name=”prefiro”
value=”r4”> 4 <br
<button type=”submit”>Enviar</button><br>
</form>
```

Figura 6.5.: *Uso do valor radio.*

O mesmo que foi dito no tópico anterior a respeito do nome (valor de name) e value deve ser aplicado aqui. Ou seja, se as questões fizerem parte de uma mesma questão, os nomes devem ser iguais e os valores de value devem ser diferentes.

text

O valor text define um campo simples, composto por uma única linha. Neste campo o usuário pode digitar textos que não necessitam de validação, tal como nome.

Exemplo de uso:

```
<form>
Digite Seu Primeiro Nome: <input type="text"
name="pnome"><br>
Digite Seu Segundo Nome: <input type="text"
name="snome"><br>
<button type="submit">Enviar</button><br>
</form>
```

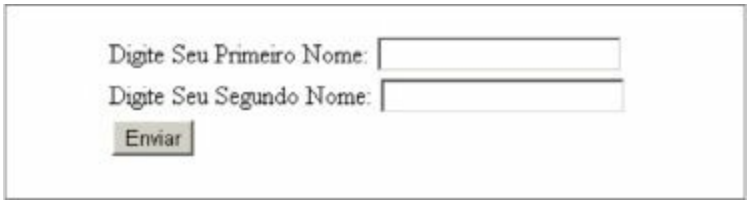


Figura 6.6.: *Input type text.*

reset

Um botão com função muito útil em formulários é o botão reset, usado para apagar tudo que foi digitado pelo usuário. Imagine a seguinte situação: você acessa uma página web para preencher um cadastro qualquer. Ao clicar no botão Cadastrar (ou outro semelhante), uma mensagem surge avisando que este cadastro já existe. Você resolve então cadastrar sua irmã. E, para agilizar, você pode clicar no botão reset que vai apagar todos os dados já preenchidos.

Exemplo de uso:

```
<form>
<input type="reset" value="Apagar">
</form>
```

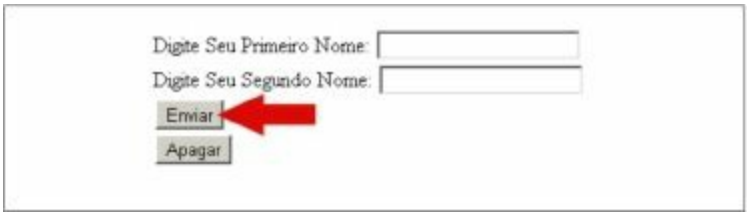


Figura 6.7.: *Botão reset.*

color

Este valor para type é uma novidade bem interessante. Ele exibe uma paleta de cores onde o usuário pode escolher uma cor. Essa paleta de cores não é construída pelo HTML. Ela é a paleta de cores do sistema operacional.

Esse recurso é bem útil em interfaces web tais como editores de texto. Até o momento em que

escrevemos este curso a compatibilidade com os browsers é pouca, restringindo-se somente ao Google Chrome e Opera. Por isso, é sempre importante testar nos navegadores para verificar a compatibilidade.

Exemplo de uso:

```
<form>
Selecione uma Cor: <input type="color"
name="pCores"> <br>
</form>
```

Figura 6.8.: *Paleta de cores.*

date

Mais uma novidade interessante e útil em diversos casos, tais como editores de textos baseados na web. O valor date serve para construir um campo com um controle de data, ou seja, um calendário, onde o usuário pode escolher uma data (dia, mês e ano).

A compatibilidade ainda é pequena: Google Chrome, Opera e Safari.

Exemplo de uso:

```
<form>
Escolha uma data: <input type="date"
name="calendario"> <br>
</form>
```

Figura 6.9.: *Calendário.*

Nota: Por que estamos usando a tag
 no final de cada campo? Para criar uma quebra de linha e

evitar que os campos fiquem na mesma linha. Dessa forma, um campo ficará debaixo do outro.

---

datetime

Este valor possui função semelhante ao anterior, com a diferença de ser possível escolher também uma hora (com fuso horário). Devemos ter atenção quanto à compatibilidade com os navegadores. No momento em que escrevemos este curso, somente os navegadores Safari e Opera suportam este recurso. Por isso, teste este código nos navegadores antes de pô-lo em execução, para verificar a compatibilidade.

Exemplo de uso:

```
<form>
Escolha uma data e Hora: <input
type="datetime" name="datahora"><br>
</form>
```

Figura 6.10.: *Exemplo do uso de datetime.*

Datetime-local

Continuando nossos estudos com datas e horas, conheceremos agora o valor datetime-local. Faz exatamente o mesmo que o valor anterior (datetime), porém, sem fuso horário. A compatibilidade é a mesma do datetime, por isso, as observações que fizemos anteriormente valem para o valor datetime-local.

Exemplo de uso:

```
<form>
Escolha uma data e Hora: <input
type="datetime" name="datahora"><br>
</form>
```

Figura 6.11.: *Data e hora local.*

email

Este valor para type permite a construção de campos para digitação de e-mails. Trata-se de uma novidade e uma excelente demonstração do avanço dos formulários. Isso porque este campo faz a validação automática de e-mails. Se o usuário digitar um e-mail com "sintaxe" errada, ao clicar no botão enviar um aviso (pedindo para digitar um e-mail válido) será emitido.

Exemplo de uso:

```
<form>
Digite seu E-mail: <input type="email"
name="usermail"><br>
</form>
```

Figura 6.12.: *Aqui um campo de e-mail.*

Figura 6.13.: *Observamos aqui a validação.*

image

Uma forma básica de estilizar formulários é personalizando o botão. Com HTML5 podemos usar uma imagem qualquer (jpg, gif, png etc.) como um botão. E isso é feito com muita facilidade e simplicidade.

O primeiro passo é selecionar uma imagem. Cuide para que o tamanho dela fique de acordo com o formulário: nem muito grande e nem muito pequena. Salve-a no diretório onde se encontra os arquivos HTML. Feito isso, verifique o exemplo de uso.

Exemplo de uso:

```
<form>
<input type="image" src="Play.jpg" alt="Submit">
</form>
```

Figura 6.14.: *Uso de imagem como botão.*

file

Este valor para type permite criar um botão do tipo “Escolha um Arquivo”. É muito útil em situações nas quais o usuário precisa preencher um formulário e escolher um arquivo em seu computador para ser feito o upload. Dessa forma, quando o usuário enviar o formulário o arquivo também será enviado.

Exemplo de uso:

```
<form>
Selecione um Arquivo: <input type=”file”
name=”arquivo”><br>
</form>
```

Figura 6.15.: *Selecione um arquivo.*

hidden

Hidden possui a função de criar um campo oculto, ou seja, que o usuário não irá ver ao acessar a página. Mas, para que serve um campo oculto? Serve para armazenar valores padrões, que poderão ser manipulados e modificados por JavaScript.

Exemplo de uso:

```
<form>
<input type=”hidden” name=”camp1”
value=”acessar”>
</form>
```

month

Este é mais um valor para o atributo type que permite trabalhar com datas. Porém, este campo permite que o usuário escolha somente um mês e ano. Por exemplo: 2011-11 (ano 2011 e mês 11). Mais uma vez, a dica é testar em cada navegador para verificar compatibilidade.

Exemplo de uso:

```
<form>
Escolha um mês e ano: <input type="month"
name="meseano"><br>
</form>
```

Figura 6.16.: *Exemplo do uso de month.*

number

Define um campo para entrada de números. Através do campo number o usuário pode escolher quantias. Esse tipo de campo é muito utilizado em lojas virtuais, onde o usuário escolhe a quantia de produtos que deseja comprar.

Exemplo de uso:

```
<form>
Escolha uma quantia: <input type="number"
name="quantia"><br>
</form>
```

No exemplo anterior o usuário poderá escolher qualquer quantia, partindo de um (1) até um número indefinido.

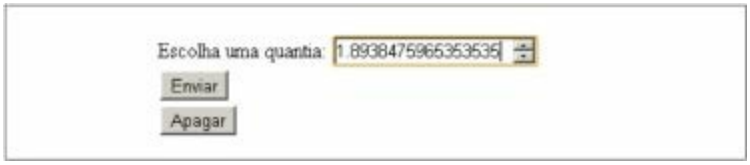


Figura 6.17.: *Campo number.*

No entanto, dependendo da situação, pode haver a necessidade de restringir a quantia máxima que o usuário pode escolher. Exemplo: em uma loja virtual onde cada usuário só pode comprar uma determinada quantia máxima do mesmo produto por vez. Para isso, devemos definir as restrições das quantias que se pode escolher. Isso é feito através de dois atributos:

- **min:** define a quantia mínima;
- **max:** define a quantia máxima.

Exemplo de uso:

```
<form>
Escolha uma quantia: <input type=”number”
name=”quantia” min=”1” max=”10”
value=”1”><br>
</form>
```

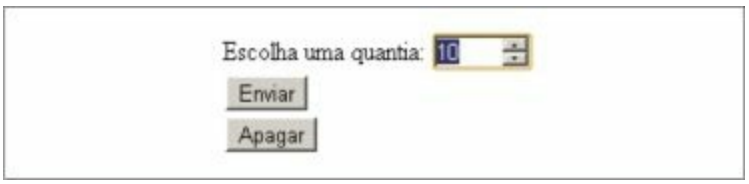


Figura 6.18.: *Neste exemplo, a quantia máxima que podemos escolher é 10.*

Além desses atributos (min e max), podemos usar o value, que nesse caso vai definir o valor padrão já selecionado. Exemplo: value=”1”.

E há também o atributo step, que define o intervalo de números. Exemplo: se definir step=“2”, a contagem será de 2 em 2. (Exemplo: 2, 4, 6, 8, 10 etc.)

Exemplo de uso:

```
<form>
Escolha uma quantia: <input type=”number”
name=”quantia” min=”1” max=”10” value=”1”
step=”2”><br>
</form>
```

password

Campo para digitação de senhas. Esse campo recebe tratamento automático. Isso significa que quando o usuário digitar uma senha, os caracteres não são mostrados. No lugar deles são exibidos apenas os famosos asteriscos (nesse caso, são pontos ou bullets, para ser mais preciso).

Exemplo de uso:

```
<form>
Digite uma Senha: <input type="password"
name="senha"><br>
</form>
```

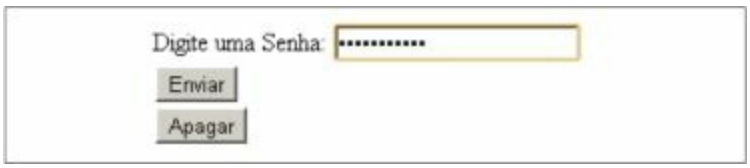


Figura 6.19.: *Campo password.*

range

Este valor para type permite construir um controle deslizante. Por meio dele o usuário pode definir valores em que a exatidão não é importante.

Exemplo de uso:

```
<form>
Defina um valor: <input type="range"
name="pontos"><br>
</form>
```

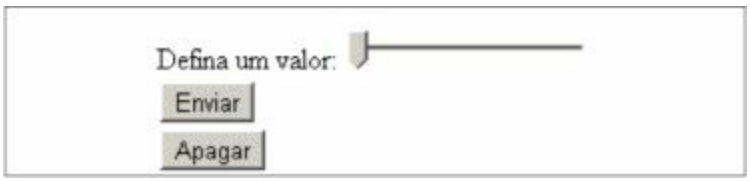


Figura 6.20.: *Uso de range.*

Também podemos definir restrições quanto aos valores aceitos. Isto é feito por meio dos atributos:

- **max:** define o valor máximo aceito;
- **min:** define o valor mínimo aceito.

E há também os seguintes atributos:

- **value:** define o valor padrão;
- **step:** define o intervalo de números.

Exemplo de uso:

```
<form>
Defina um valor: <input type="range"
name="pontos" min="1" max="10" value="1"
step="2"><br>
</form>
```

search

Search é um termo da língua inglesa que significa pesquisar. Este valor para type cria um campo para ser usado na realização de pesquisas em páginas web. Um campo de pesquisa é muito útil pois permite que o usuário encontre informações diversas com mais rapidez.

Até o momento em que escrevemos este curso, os navegadores que suportam esse recurso são o Google Chrome e o Safari. Por isso, sempre teste o código em navegadores diversos antes de colocá-lo em prática.

Exemplo de uso:

```
<form>
Procurar: <input type="search"
name="buscar"><br>
</form>
```

Figura 6.21.: *Uso de search.*

tel

Esta é mais uma novidade do HTML5. Este valor define um campo para digitação de números de telefones com validação. É um tipo de campo muito usado em formulários eletrônicos. Atualmente, quando este tipo de campo possui alguma validação ela é geralmente feita por JavaScript.

O valor tel para type chega exatamente para suprir essa necessidade: construir campos para digitação e validação de números de telefone sem a necessidade de JavaScript ou outras linguagens.

O problema é que no momento em que escrevemos este curso (meados de 2011), este campo não é compatível com nenhum navegador atual. Mas, vale a pena conhecer esse valor, e continuar acompanhando a evolução do HTML5 e sua crescente adoção por parte dos navegadores.

Exemplo de uso:

```
<form>
Telefone: <input type=”tel”
name=”telefone”><br>
</form>
```

time

O valor time define um campo onde o usuário pode definir uma hora sem fuso horário.

Exemplo de uso:

```
<form>
Telefone: <input type=”tel”
name=”telefone”><br>
</form>
```

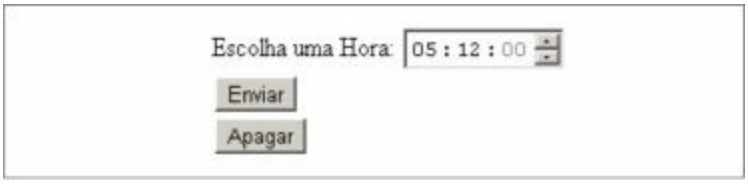


Figura 6.22.: *Uso do valor time.*

url

Este novo valor para type existente no HTML5 permite a criação de campos para digitar endereços de websites (URL). E o melhor, com validação. Isso significa que se o usuário digitar uma URL com erro de "sintaxe", ao clicar no botão enviar uma mensagem pedindo para corrigir a URL será exibida.

Exemplo de uso:

```
<form>
Website: <input type="url"
name="website"><br>
</form>
```

Figura 6.23.: *Campo URL. Veja que possui validação.*

week

Cria um campo para escolha de uma data de um mês, sem fuso horário.

Exemplo de uso:

```
<form>
Escolha uma Data: <input type="week"
name="data"><br>
</form>
```

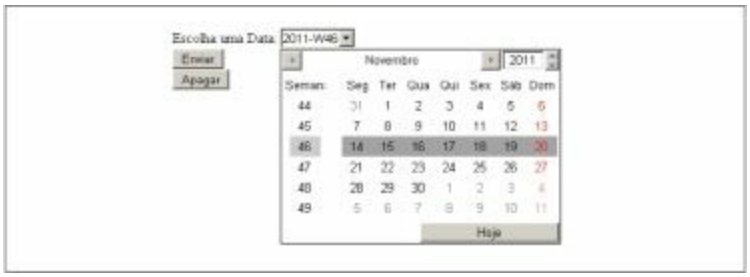


Figura 6.24.: *Uso de week.*

<textarea>

A tag <textarea> constrói áreas multilinhas para digitação de textos diversos. É um tipo de campo importante em formulários. Geralmente é disposto no final do formulário e é uma oportunidade de o visitante ou cliente expor suas opiniões, enviar dicas e sugestões, pedir maiores esclarecimentos a respeito de algo, enviar maiores informações etc.

Exemplo de uso:

```
<form>
<textarea>
Deixe aqui o seu recado...
</textarea>
</form>
```

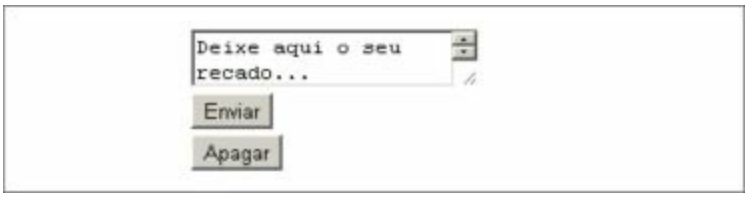


Figura 6.25.: *Uma área de texto básica.*

Como podemos observar no código e na **Figura 6.25**, construímos uma área de texto, porém, sem configurar sua altura e largura. Isso pode ser feito facilmente graças aos atributos:

- **rows**: define a quantidade de linhas que a área de texto irá possuir;
- **cols**: define a quantidade de colunas que a área de texto irá possuir.

Exemplo de uso:

```
<form>

<textarea rows=”10” cols=”35”>
Deixe aqui o seu recado...
</textarea>

</form>
```

Figura 6.26.: *Área de texto com rows e cols configurado.*

Há outros atributos importantes:

- **autofocus**: especifica que o botão deve receber foco automaticamente assim que a página for aberta. Exemplo: autofocus=”autofocus”;
- **disabled**: mantém o objeto desativado. Exemplo: disabled=”disabled”;
- **maxlength**: define o número máximo de caracteres permitido em um determinado elemento input. Exemplo: maxlength=”100”;
- **name**: especifica um nome para o objeto. Exemplo: name=”btreset”;
- **placeholder**: insere um texto descritivo ou uma dica em campos de preenchimento (campos onde o usuário deve digitar informações). Exemplo: placeholder=”Digite Seu Nome Aqui”;
- **required**: faz com que um campo seja de preenchimento obrigatório. Exemplo: required=”required”.

<select> e <option>

Esta tag permite a criação de um campo chamado lista drop-down. É um tipo de lista onde o usuário pode escolher um item entre várias opções.

Exemplo de uso:

```
<form>
<p> O que você prefere?</p>
<select>
    <option value=”praia”>Praia</option>
    <option value=”cinema”>Cinema</option>
    <option value=”natureza”>Natureza</option>
    <option value=”casa”>Ficar em Casa</option>
</select>
<br>
</form>
```

Observe que além da tag <select> é usado a <option> para criar cada item da lista. E cada item possui um atributo value.

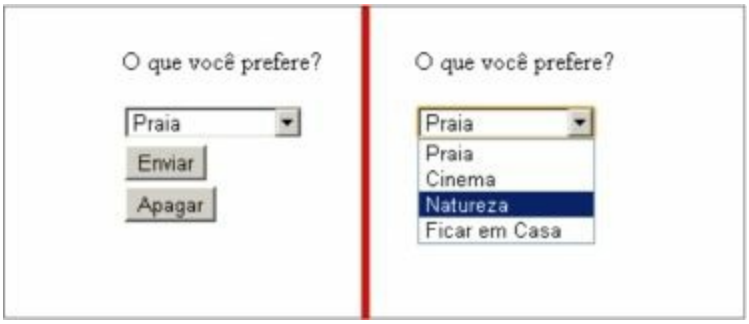


Figura 6.27.: *Campo construído com <select>.*

Alguns atributos básicos que podem ser usados: autofocus, disabled e name.

<optgroup>

Esta tag é usada para criar grupos de opções em uma lista drop-down. É uma tag que ajuda a organizar listas criadas pela tag <select>. Pode ser usada principalmente em listas grandes, onde podemos separar os itens por grupos, facilitando a visualização e escolha por parte do usuário.

Exemplo de uso:

```
<form>
<p> O que você prefere?</p>
<select>
    <optgroup label=”Carros”>
    <option value=”blazer”>Blazer</option>
    <option value=”s10”>S10</option>
    <option value=”uno”>Fiat Uno</option>
    </optgroup>
    <optgroup label=”Motos”>
    <option value=”blazer”>Honda</option>
    <option value=”s10”>Yamaha</option>
    <option value=”uno”>Suzuki</option>
    </optgroup>
</select>
<br>
<br>
</form>
```

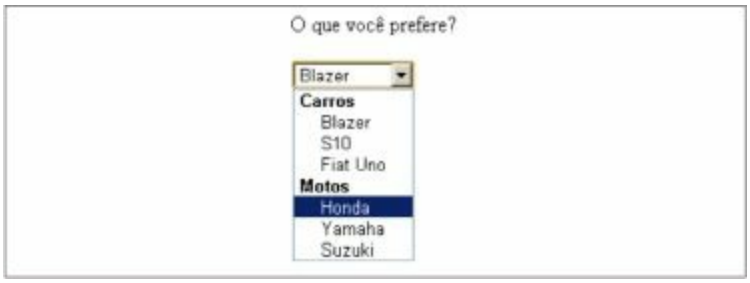


Figura 6.28.: *Uma lista com uso de <select> e <optgroup>.*

<fieldset>

Esta tag possui a função de agrupar todos os elementos de um formulário, ou seja, tudo que estiver entre <form> e </form>. Visualmente, ela cria uma caixa com um contorno em volta do formulário.

Exemplo de uso:

```
<form>
<fieldset>
<p> O que você prefere?</p>
<select>
    <optgroup label=”Carros”>
    <option value=”blazer”>Blazer</option>
    <option value=”s10”>S10</option>
    <option value=”uno”>Fiat Uno</option>
    </optgroup>
    <optgroup label=”Motos”>
    <option value=”blazer”>Honda</option>
    <option value=”s10”>Yamaha</option>
    <option value=”uno”>Suzuki</option>
    </optgroup>
</select>
<br>
<button type=”submit”>Enviar</button><br>
<input type=”reset” value=”Apagar”> <br>
</fieldset>
</form>
```

Figura 6.29.: *Observe o contorno em volta do formulário. Ele é criado graças ao <fieldset>.*

<legend>

A legenda é um título ou descrição do formulário. É usado em conjunto com <fieldset>.

Exemplo de uso:

```
<form>
<fieldset>
<legend>O que você prefere?</legend>
<select>
    <optgroup label=”Carros”>
    <option value=”blazer”>Blazer</option>
    <option value=”s10”>S10</option>
    <option value=”uno”>Fiat Uno</option>
    </optgroup>
    <optgroup label=”Motos”>
    <option value=”blazer”>Honda</option>
    <option value=”s10”>Yamaha</option>
    <option value=”uno”>Suzuki</option>
    </optgroup>
</select>
<br>
<button type=”submit”>Enviar</button><br>
<input type=”reset” value=”Apagar”> <br>
</fieldset>
</form>
```

<Capítulo 6>
HTML5 e Cascading style sheet

Introdução

CSS é a abreviatura para Cascading Style Sheets. Em bom português é Folha de Estilos em Cascata. Foi criada pelo W3C, sigla de World Wide Web Consortium, um consórcio formado por empresas de tecnologia que criam padrões para apresentação de conteúdo na web. O website do W3C é www.w3.org.

Para quê serve CSS?

Essa linguagem serve para construir layouts de sites, ou seja, é usada para definir toda a aparência do site. Por exemplo: ela pode definir a cor ou imagem de fundo, cor dos textos, fontes e tamanhos das fontes, margens, alturas, linhas etc.

Qual a diferença entre HTML e CSS?

Você deve estar se perguntando: mas, qual a diferença entre HTML e CSS? Afinal, com HTML também podemos construir layouts para sites. Bom, isso é verdade. Mas, de início, vamos ressaltar três pontos:

1. A CSS é muito mais sofisticada do que HTML.
2. Para aprender CSS é necessário ter algum conhecimento de HTML.
3. HTML é usado para estruturar conteúdos. Já a CSS é usado para formatar conteúdos estruturados.

Além desses pontos, é importante saber que a linguagem HTML foi projetada originalmente para estruturas e não para layouts. Por exemplo: o uso de <table>, que deveria ser usado para organizar dados tabulares (por exemplo: uma lista de alunos, de produtos, de preços etc.) passou a ser usado para construir layouts. Definir layouts em HTML funciona, é uma técnica que pode ser usada. Mas, ao definir o layout com CSS você ganha mais opção, precisão, sofisticação, rapidez etc.

Exemplos:

- **Precisão**: ao usar CSS você pode definir se a imagem de fundo irá ficar expandida na tela ou se irá se repetir várias vezes. Caso defina que a imagem de fundo irá se repetir na tela, você pode definir se isso ocorrerá somente na horizontal, na vertical ou de ambos os modos. É possível definir posicionamentos absolutos, como, por exemplo, definir um ponto exato da tela onde uma determinada imagem deve aparecer.
- **Rapidez**: um único arquivo externo pode definir a aparência de todo o seu site, não importando quantas páginas ele tenha. Ao mudar esse único arquivo, as aparências de todas as páginas

mudam automaticamente.

Além do mais, a linguagem CSS é reconhecida por praticamente qualquer browser. Já no HTML alguns tags podem não funcionar em determinados browsers.

Vantagens

CSS é uma das linguagens mais sensacionais que já foram inventadas para webdesign. Ao estudar CSS você irá aprender a criar sites com identidade visual unificada e coerente. Entre as vantagens dessa poderosa linguagem, citamos: facilidade em mudar o visual do site, criação de menus, permite colocar a imagem de fundo no modo extendido (no tradicional HTML, ela se repete no fundo), as páginas ficam mais leves e são reconhecidas por praticamente qualquer browser e muito mais.

Quais programas são necessários?

Tal como ocorre com a HTML, um simples bloco de notas já é suficiente para trabalhar com CSS. Nada mais do que isso. Além disso, todos os códigos podem ser testados localmente, no seu próprio micro.

Onde e como inserir os códigos?

Os códigos CSS podem ser aplicados a um documento de três formas bem distintas: In-line, Interno e Externo (Importado).

A forma mais interessante é a externa. Usando-a podemos deixar todos os códigos CSS em um arquivo externo. Se for necessário mudar alguma coisa na aparência do site, basta mudar as configurações desse único arquivo. Vejamos a seguir as peculiaridades de cada um.

In-line

Este método faz uso do atributo style do HTML. Os códigos são colocados dentro da tag em que se deseja aplicar a regra. Veja o exemplo a seguir.

Exemplo de uso:

```
<P style="color: #09873b; Font-family: verdana; Font-size: 16px;">
```

Interno

Este método faz uso da tag <style>. Preste atenção: no método anterior, style era somente um atributo, inserido em uma tag HTML. Agora ele já é uma tag.

Pois bem, usando a tag <style> podemos escrever todos os códigos CSS dentro da página em questão. As regras serão, desta forma, aplicadas somente a uma página. As regras CSS devem ser inseridas depois de <Head> e antes de </Head> do código HTML. Vejamos o exemplo a seguir.

Exemplo de uso:

```
<head>
<title>Exemplo
</title>
<style type="text/css">
```

```
Body {
Background: #FFFFCC;
}
H1 {
color: #FF0000;
}
H2 {
color: #008000;
}
H3 {
color: #008080;
}
</style>
</head>
<H1> Texto Vermelho </H>
<H2> Texto Verde </H>
<H3> Texto Azul-Petróleo </H>
```

Importado

Quando utilizamos regras CSS importadas, os códigos digitados ficam em um arquivo externo, e não no documento HTML. Com esse método é possível associar quantas páginas HTML forem necessárias a um único arquivo contendo as regras CSS. Mudando as regras desse arquivo externo, o site inteiro se altera automaticamente.

Esse arquivo externo, que irá conter as regras CSS, deve possuir a extensão ".css". Ele nada mais é que um arquivo ".txt" com a extensão renomeada para ".css". Esse método é o que recomendamos, pois torna o trabalho muito mais dinâmico e rápido. A sintaxe básica para importar o arquivo ".css" pode ser vista no exemplo a seguir.

Exemplo de uso:

```
<head>
<title>Exemplo
</title>
<link rel=”stylesheet” type=”text/css”
href=”default.css”>
</head>
```

Seletor

CSS, assim como a HTML, possui uma sintaxe própria. Essa sintaxe possui partes bem definidas e regras exatas para o seu uso. Erros de sintaxe podem fazer com que o código não funcione, por isso, tudo deve ser feito sempre com atenção.

Para explicar melhor, vamos primeiramente apresentar cada parte dessa sintaxe, que são três: seletor, propriedade e valor. Ao escrever os códigos, é preciso adotar o seguinte procedimento:

Seletor {propriedade: valor}

Observe que primeiro vem o seletor, e entre chaves ficam a propriedade e o valor, ambos separados por dois pontos. Por questões de organização do código, você pode escrever da seguinte forma:

Exemplo de uso:

```
Seletor {
propriedade: valor
}
```

Mas, o que é um seletor, uma propriedade e um valor?

Vamos responder a essa questão em três partes:

- **Seletor**: ele indica qual tag HTML será aplicada a propriedade. Essa tag pode ser identificada pela tag em si, por um ID ou por uma classe. Não se preocupe que no momento certo iremos abordar o que são e como usar IDs e Classes.
- **Propriedade**: é um elemento que está ligado à tag. No HTML o chamamos de “atributo”. Exemplos: “font”, “face”, “color”, “background” etc.
- **Valor**: tal como ocorre no HTML, o valor é ligado à propriedade. Por exemplo: o valor do atributo “color” são as cores, que podem ser inseridas em números hexadecimais ou, para as principais, em inglês (Não é recomendável. Prefira sempre valores em hexadecimal, pois é uma garantia de que esse código será reconhecido por qualquer browser e em qualquer país). Podemos perceber então que o valor é característica que será atribuída à propriedade.

No HTML5 podemos usar como seletores as tags estruturais <header>, <aside>, <footer>, <section>, <article> e <nav>.

Algumas regras básicas

Vejamos agora algumas regras básicas.

Cor de fundo

Para definir a cor de fundo de uma página ou de algum elemento (tabela, seção header etc.), usa-se a tag Body e a propriedade background-color. O valor são as cores expressas, de preferência, em números hexadecimais.

Exemplo de uso:

```
Body {
Background-color: #FFCC55
}
header{
Background-color: #FFFFCC;
}
```

Cor de textos

Para definir as cores de um texto é muito simples. Primeiramente basta você definir qual seletor irá usar. Alguns exemplos: p (de parágrafo), h1 (de texto com tamanho h1), h2 (de texto com tamanho h2) etc. Para aplicar a cor, usamos o atributo color.

Exemplo de uso:

```
h1 {color: #ffaa44; }
h4 {color: #11aa44; }
p {color: #001166}
```

Fonte

Para definir a fonte a usar (exemplo: verdana, arial, times news roman etc.) é usada a propriedade font-family. Essa propriedade serve para definir uma lista de fontes que devem ser usadas. O navegador do usuário irá usar sempre a primeira. Caso o micro do usuário não tiver disponível a primeira, ele assume a segunda, e assim sucessivamente. Nessa lista, cada fonte é separada por uma vírgula.

Exemplo de uso:

```
h1 {font-Family: verdana, arial, helvetica,
sans-serif;}
h4 {font-Family: arial, verdana, sansserif;}
p {font-Family: “times new roman”, verdana,
arial;}
```

Tamanho da fonte

O tamanho da fonte é definido pelo atributo font-size. Existem vários valores que podem ser

usados, tais como px, pt, % ou a descrição em inglês (xx-small, x-small, small, medium, large, x-large, xx-large).

Exemplo de uso:

```
h1 {font-size: 14pt;}
h4 {font-size: 10pt;}
p {font-size: 8pt;}
```

Estilo de bordas

Esta é a propriedade básica para se colocar uma borda. Sem ela você não irá conseguir inseri-la. Uma borda pode ter vários estilos. O comum é o sólido, em que a borda é formada por uma linha simples, na cor e espessura que você escolher (ou em um valor padrão). A propriedade usada é a border-style. Os estilos comuns são: solid, dotted, dashed, inset, outset, groove, double.

Exemplo de uso:

```
h1 {border-style: dotted;
}
p {border-style: double;
}
```

Espessura de bordas

A espessura é a grossura da borda. É definida pela propriedade border-width. Os valores podem ser usados em "px" ou por meio de uma das seguintes definições: thin (fina), medium (media) e thick (grossa). Veja a seguir como usar.

Exemplo de uso:

```
h1 {
border-style: dotted;
border-width: thin;
}
p {
border-style: double;
border-width: thick;
}
```

Cor de bordas

Por meio da propriedade border-color podemos escolher uma cor. Os valores mais comuns usados são as cores em hexadecimal ou em inglês (black, red, yellow etc.), mas, prefira a primeira opção. Veja a seguir o exemplo de uso.

Exemplo de uso:

```
h1 {
border-style: dotted;
```

```
border-width: 12px;
border-color: #5CE2BA;
}
```

Margens

Podemos definir as margens como as bordas de um documento, ou a distância entre elementos vizinhos. Seja para definir a mergem de um documento inteiro ou apenas de elementos em uma página, uma borda possui quatro lados:

- superior (top);
- esquerda (left);
- direita (right);
- inferior (bottom).

E para cada lado há um atributo:

- margin-top;
- margin-right;
- margin-bottom;
- margin-left.

Desse modo, é fácil definir as margens de um documento ou de elementos dentro de um documento.

Exemplo de uso:

```
body {
margin-top: 50px;
margin-right: 40px;
margin-bottom: 80px;
margin-left: 40px;
}
```

Enchimento

O enchimento define a distância entre um elemento e a sua borda, ou seja, define o espaçamento entre conteúdo e borda. Para isso, usamos o atributo padding e podemos controlar os quatros lados:

- superior: padding-top;
- direito: padding-right;
- inferior: padding-bottom;
- esquerdo: padding-left.

Para ficar mais fácil visualizar, no exemplo a seguir vamos usar o enchimento com os atributos para construir bordas. Perceba a diferença em comparação com todos os exemplos que já demos anteriormente.

Exemplo de uso:

```
H1 {
border: 12px dotted #5CE2BA;
Padding-top: 40px;
Padding-right: 29px;
Padding-bottom: 40px;
Padding-left: 29px;
}
p {
border: 12px double #5CE2BA;
Padding-top: 40px;
Padding-right: 100px;
Padding-bottom: 40px;
Padding-left: 100px;
}
```

Listas

Com CSS podemos construir listas de forma muito mais fácil do que com HTML puro e com mais opções. Podemos usar facilmente imagens como marcadores ou escolher um tipo de marcador. É possível também escolher a posição do marcador. Acompanhe nos tópicos que se seguem os exemplos, na prática.

- **Listas não numeradas**: vamos começar nossos trabalhos com listas exatamente com os tipos de marcadores. Para isso, o atributo que vamos usar é o list-style-type. O nosso seletor HTML será o <ul>. Dessa forma, veja o exemplo a seguir. Perceba que para cada item da lista usamos a tag <li> (no arquivo HTML).

Exemplo de uso:

```
ul{ list-style-type: disc;
}
```

- **Listas numeradas**: Se precisa ordenar algo, então você precisa trabalhar com listas ordenadas. E é isso que veremos agora. O atributo que vamos usar é o mesmo, ou seja, list-style-type. O nosso seletor HTML também será <ul>. E no arquivo HTML, também iremos colocar a tag <li> em cada item da lista. O que muda são os valores, que podem ser: decimal, lower-roman, upper-roman, lower-alpha e upper-alpha.

Exemplo de uso:

```
ul{ list-style-type: decimal;
}
```

Links

É possível aplicar regras CSS também em links. Podemos, por exemplo, escolher as cores que eles usarão nos mais variados estados (visitado, não visitado etc.), remover o sublinhado etc.

O link é construído por uma tag HTML (<a>), mas, podemos estilizá-los com CSS. Como o objetivo deste curso é ser o mais prático possível, vamos, nos tópicos seguintes, partir logo para a prática e criar regras CSS para aplicar cores em links nos três estados: não visitado, visitado e ativo.

O link não visitado é, obviamente, aquele que o internauta ainda não clicou, ou seja, que não acessou a página correspondente a ele. Para isso, usaremos a pseudoclasse "link". A tag HTML que iremos usar é "a", pois, é a tag usada para criar links no HTML (<a>). Acompanhe o exemplo.

Exemplo de uso:

```
a:link {
color: #FF6600;
}
```

O link visitado é aquele que o internauta já clicou, acessou, abriu a página correspondente a ele. É comum configurar em sites uma cor diferente dos links "não visitados" para os que já são "visitados". É uma forma de sinalizar para o internauta as páginas que ele já abriu. Para isso, usaremos a pseudoclasse "visited". Acompanhe o exemplo a seguir. Observe que iremos manter o código para link não visitado.

Exemplo de uso:

```
a:link {
color: #FF6600;
}
a:visited {
color: #FFCC00;
}
```

O link ativo é o estado que ocorre quando estamos clicando. Para isso, usaremos a pseudoclasse "active". Acompanhe o exemplo a seguir. Observe que iremos manter o código para link não visitado e visitado.

Exemplo de uso:

```
a:link {
color: #FF6600;
}
a:visited {
color: #FFCC00;
}
a:active {
color: #008000;
}
```

Este livro foi composto na fonte Minion Pro e
impresso em papel Norbrite 66.6g/m^2 na Imprensa da Fé.