

*Criação de Web Sites I*

2  
HTML

# Conteúdo

3. Fundamentos do HTML.....	26
3.1. Elementos e descritores (tags).....	27
3.2. Atributos.....	29
3.3. Caracteres de escape .....	29
3.4. Comentários.....	31
3.5. Estrutura do documento.....	31
3.6. Elementos do HEAD.....	32
TITLE.....	32
META.....	32
LINK.....	33
BASE.....	33
3.7. Elementos de BODY.....	33
4. Parágrafos e blocos.....	35
4.1. Parágrafos <P>.....	35
4.2. Cabeçalhos <H1> a <H6>.....	36
4.3. Quebras de linha  .....	37
4.4. Linhas horizontais <HR>.....	37
4.5. Citação de bloco <BLOCKQUOTE>.....	37
4.6. Texto pré-formatado <PRE>.....	38
4.7. Bloco genérico de seção <DIV>.....	39
4.8. Bloco de endereço <ADDRESS>.....	39
5. Listas.....	40
5.1. Listas não-ordenadas <UL>, <LI>.....	40
5.2. Listas ordenadas <OL>, <LI>.....	41
5.3. Listas de definições <DL>, <DT>, <DD>.....	41
6. Imagens.....	43
6.1. O objeto <IMG>.....	43
URIs relativas.....	43
6.2. Alinhamento de imagens.....	44
6.3. Dimensionamento.....	44
7. Formatação de texto ( <i>inline elements</i> ).....	46
8. Âncoras e Vínculos.....	48
9. Tabelas.....	49
9.1. Principais elementos estruturais <TABLE>, <TR>, <TD> ou <TH>.....	49
9.2. Tabela sem bordas.....	50
9.3. Bordas.....	51
9.4. Espaçamento.....	51

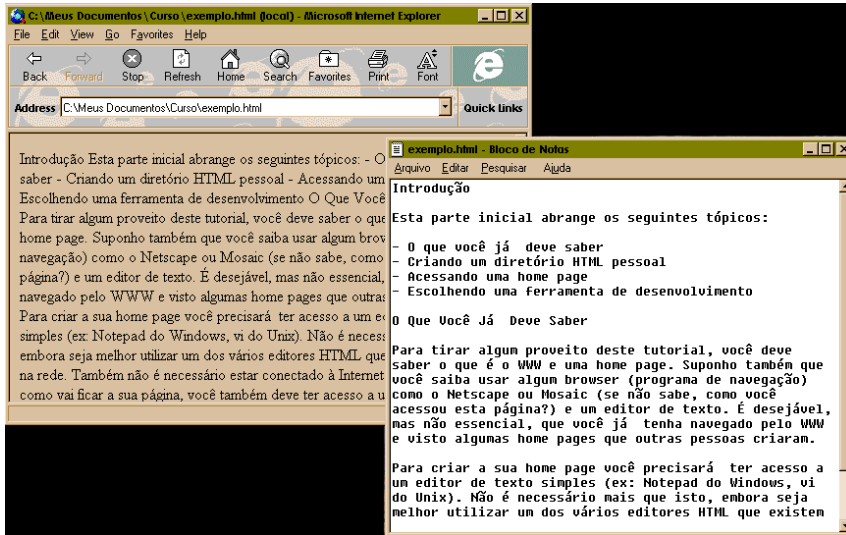
9.5.	Largura da tabela .....	52
9.6.	Altura, posicionamento e cores.....	53
9.7.	Alinhamento de células .....	54
9.8.	Combinação de células .....	55
9.9.	Otimização <THEAD>, <TBODY>, <COLGROUP>, <COL> .....	55
9.10.	Legenda <CAPTION> .....	56
10.	<i>Elementos deprecados</i> .....	57
10.1.	Blocos que alteram estilo (deprecados).....	57
10.2.	Bloco centrado <CENTER> .....	57
10.3.	Bloco sem quebras de linha <NOBR> e <WBR> .....	58
10.4.	Elementos de estilo (deprecados no HTML 4.0) .....	58
10.5.	Atributos de uma página <BODY atributos> .....	59
10.6.	Cores e fontes <FONT> .....	59
10.7.	Conclusão .....	60
11.	<i>Testes</i> .....	61

# 3. Fundamentos do HTML

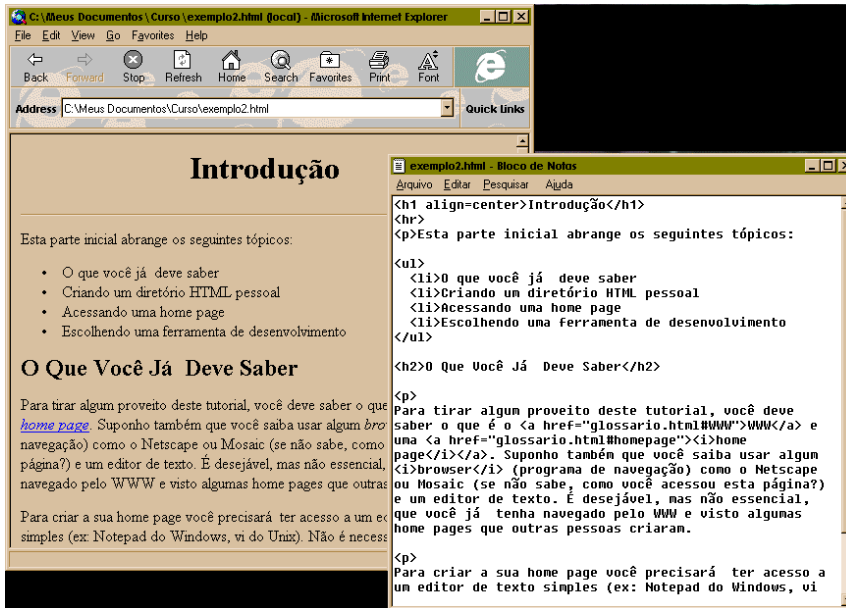
O arquivo usado para construir uma página Web é texto simples. Mesmo que a página visualizada no browser mostre imagens, applets Java, plug-ins (como *Flash*) e outros recursos, por trás de tudo existe uma página de texto e vários outros arquivos separados, que foram montados em uma página pelo browser. O código é texto, mas não é *só* texto. É texto *marcado* com HTML. HTML define *toda* a estrutura de uma página para que um browser possa formatá-la e produzir a apresentação desejada. HTML também *importa* as imagens, programas, sons e vídeos que uma página exhibe em seu interior. Mas como HTML é texto simples, pode ser editada em qualquer editor de texto (como o *Bloco de Notas* do *Windows*).

Ao ler um documento HTML, um browser tenta interpretar todas as seqüências de caracteres que ficam entre os símbolos "<" e ">". O browser entende que *qualquer coisa* que estiver entre esses caracteres é um *descriptor HTML (tag)* e não deve ser mostrado na tela. Se o descriptor for desconhecido ele simplesmente o *ignora* (e não mostra o conteúdo na tela) mas se realmente for um *elemento HTML* (definido na especificação suportada pelo browser), ele usará as informações contidas entre os símbolos estruturar a página, formatando-a de acordo com alguma regra de estilo previamente definida.

Toda a formatação de um arquivo HTML é feita *exclusivamente* através dos descritores. O browser sabe que o arquivo contém código HTML através de sua extensão (ou tipo MIME enviado pelo servidor). Se um arquivo de texto com extensão HTML mas sem descritores for carregado, no browser, toda a sua informação aparecerá em um único parágrafo, sem destaques, sem formatação alguma. Veja as figuras abaixo. Elas ilustram um mesmo texto de um documento HTML digitado no *Bloco de Notas* do *Windows* e lidos através do *Internet Explorer*. Na primeira figura, não há descritores HTML e, apesar dos espaços, tabulações e novas-linhas existente no arquivo de texto visualizado no editor de textos, o browser ignora toda a formatação. Já na segunda figura descritores HTML foram incluídos e o browser foi capaz de formatar a página de uma maneira estruturada.



Arquivo de texto (com extensão .html) sem marcação HTML ao ser visualizado no browser e no editor Bloco de Notas (Windows).



Arquivo de texto (com extensão .html) contendo marcação HTML ao ser visualizado no browser e no editor Bloco de Notas (Windows).

### 3.1. Elementos e descritores (tags)

A maioria dos *elementos HTML* possui um *descritor inicial* e um *descritor final*, cercando o texto que é marcado por eles. A sintaxe básica é

**<Elemento>** Texto marcado por Elemento **</Elemento>**

O texto marcado passa a ter uma *função*, determinada pelo Elemento. Nem sempre isto significa que o texto marcado terá sua aparência alterada quando a página HTML aparecer no browser. Há browsers orientados a caractere que não são capazes de exibir tamanhos de fonte, por exemplo. Outros podem não ter uma forma associada a um elemento. Neste caso, o autor

da página poderá definir a forma através de uma folha de estilos usando CSS, caso o browser a suporte. Os elementos também podem influenciar o comportamento de mecanismos de busca, que atribuem maior importância ou filtram dados de acordo com a sua função determinada pelos elementos HTML. O texto marcado e o elemento HTML também assumem o papel de *objeto abstrato* utilizável por linguagens embutidas na página como CSS e JavaScript. Essas linguagens interagem com o HTML através de um modelo de objetos que relaciona as estruturas HTML com comportamentos interativos (em JavaScript e DHTML) e forma gráfica (CSS).

Veja alguns exemplos de texto rotulado com descritores HTML:

```
<b> Texto em negrito </b>
<h2> Subtítulo </h2>
```

Observe que o descritor final é praticamente idêntico ao descritor inicial. A única diferença é que este último é precedido por uma barra ("/"). Não faz diferença utilizar letras maiúsculas ou minúsculas para os descritores HTML.

Os blocos de texto marcados pelo HTML podem conter outros blocos. Sempre que isto ocorrer, o bloco mais interno deverá ser fechado antes do descritor de fechamento do bloco externo. Veja um exemplo:

```
<h1> Texto <b> muito <i> importante</i></b> para todos</h1>
```

Este outro exemplo está incorreto pois o bloco <b> fecha sem que o bloco <i> tenha fechado.

```
<h1> Texto <b> muito <i> importante</b></i> para todos</h1>
```

Nem todo elemento HTML pode conter outros descritores e mesmo os que podem têm regras que definem quais os elementos permitidos.

Existem elementos que não precisam ter descritores de fechamento. O parágrafo, por exemplo, marcado por <P> e </p> pode omitir o descritor final. Isto é permitido porque não pode haver um parágrafo dentro de outro, portanto, se o browser encontra um <p> pouco depois de outro <p> ele automaticamente coloca um </p> antes.

```
<p>Primeiro parágrafo. <p>Início de outro parágrafo.</p>
```

Finalmente, há também elementos que não podem conter texto ou quaisquer descritores HTML. Eles precisam ser vazios. O descritor final nesses casos sequer é permitido (a não ser que apareça logo após o descritor inicial).

```
<p>Esta é uma frase<br>que ocupa duas linhas.
```

No exemplo acima <br> (quebra de linha) marca a posição onde a linha deve ser quebrada (não contém ou marca texto algum).

## 3.2. Atributos

Alguns elementos HTML podem ter um ou mais *atributos*, opcionais ou não, que modificam alguma propriedade do texto marcado ou acrescentam alguma informação necessária. Geralmente os atributos são pares do tipo:

```
propriedade="valor"
```

Quando presentes, os atributos aparecem apenas no descritor inicial separados por espaços, logo após o nome do elemento:

```
<h1 align="center">Título centralizado</h1>
```

Um descritor pode ter vários atributos. A *ordem* em que aparecem *não faz diferença* desde que apareçam no *descritor inicial* e estejam separados dos outros atributos e do nome do elemento por *espaços*. Se um atributo for escrito incorretamente ou se o browser não o reconhecer, ele será ignorado. Diversos atributos são comuns a todos os elementos HTML. Outros só fazem sentido em certos elementos.

O valor do atributo não precisa estar entre aspas se contiver apenas letras (A-Z, a-z), números, ponto (.), dois pontos (:), hífen (-) e sublinhado (\_). Se o valor atribuído à propriedade tiver espaços, é preciso colocá-lo entre aspas ou apóstrofes. Não pode haver espaço entre a propriedade e o "=" nem entre o "=" e o valor:

```
propriedade="Valor com espaços"
propriedade = "Sintaxe incorreta"
```

O segundo exemplo acima será interpretado incorretamente pelo browser como tendo três atributos e não um (não pode haver espaços!).

Atributos geralmente são opcionais. Em alguns casos são obrigatórios, como os atributos HREF e SRC em vínculos de hipertexto e imagens, respectivamente. Ambos informam uma URI onde se encontra um recurso na Web.

```
<a href="http://www.ibpinet.net/"
  title="Vai logo! Clica!"> Clique aqui </a>


```

## 3.3. Caracteres de escape

Os caracteres "<" e ">", por definirem o início e final dos descritores, não podem ser impressos na tela do browser. Quando é necessário produzi-los, deve-se utilizar uma *seqüência de escape*. Esta seqüência é iniciada por um "&" seguido de uma abreviação e um ponto-e-vírgula, que indica o final da seqüência. Como o "&" também é caractere especial, há também

uma seqüência para produzi-lo. As principais seqüências de escape, necessárias para produzir "<", ">", "&" e aspas (quando necessário) são:

<i>Caractere</i>	<i>Seqüência de Escape</i>
<	&lt;
>	&gt;
&	&amp;
"	&quot;

Por exemplo, para produzir as seguintes linhas no browser:

```
144 < 25 + x < 36 + y
Fulano, Sicrano & Cia.
```

é preciso digitar no editor de textos (código HTML):

```
144 &lt; 25 + x &lt; 36 + y
Fulano, Sicrano &amp; Cia.
```

As seqüências de escape também são usadas para produzir caracteres que não são encontrados no teclado, como letras acentuadas, símbolos de *copyright*, etc. Por exemplo, para produzir, no browser, as palavras:

```
Plantação
maßgebend
hândbfger
enciclopædia
©
sueño
```

pode-se usar, no editor de textos:

```
Planta&ccedil;&atilde;o <BR>
ma&szlig;gebend <BR>
h&aring;ndb&oslash;ger <BR>
encilop&aelig;dia <BR>
&copy; <BR>
sue&ntilde;o
```

É possível usar também códigos numéricos, porém eles são dependentes do alfabeto utilizado. Consulte a documentação HTML para uma lista completa de todos os códigos.

Os editores mais sofisticados, como por exemplo o *HomeSite*, facilitam a entrada de caracteres especiais e geram automaticamente as seqüências de escape. Se você está usando um editor comum, como o *Bloco de Notas*, deve ter o cuidado de digitar corretamente as seqüências. Alguns browsers simplesmente ignoram seqüências que não conhecem, outros engolem linhas e palavras. Em qualquer um dos casos há perda de informação. Nunca é necessário digitar seqüências de escape se:

- você tem os caracteres no seu teclado



- eles fazem parte do conjunto de caracteres ISO-Latin-1 (alfabeto *default* do HTML)
- alfabeto default da página não foi alterado através de um descritor <META> ou de um cabeçalho HTTP.

Todos os browsers há mais de 4 anos entendem acentos e cedilha sem que seja necessário digitar os códigos.

É indiferente usar letras maiúsculas ou minúsculas para o nome do descritor ou seus atributos. Tanto faz usar <BODY>, <body>, <Body> ou <bOdY>; <a HREF="a.html"> ou <A href="a.html">. Deve-se tomar cuidado, porém, com o *conteúdo* dos atributos (o valor que às vezes precisam vir entre aspas). Em alguns casos, principalmente quando se lida com URIs ou quando se usa JavaScript, o formato maiúsculo ou minúsculo *faz* diferença e deve ser respeitado.

### 3.4. Comentários

Elementos desconhecidos pelo HTML são ignorados quando colocados entre sinais de < e >. Se você deseja fazer isto intencionalmente para acrescentar comentários no código (que não aparecerão na página), deve usar os comentários HTML, identificados entre <!--e -->. Qualquer texto ou código entre esses símbolos será ignorado na formatação da página pelo browser.

### 3.5. Estrutura do documento

A maioria dos elementos HTML segue uma estrutura hierárquica. Há uma estrutura básica (mínima) para uma página HTML, construída de acordo com a especificação padrão, que deve ser respeitada para garantir uma compatibilidade com o maior número de browsers possível. Esta estrutura está mostrada na figura abaixo:

```
<!DOCTYPE HTML Public "-//IETF//DTD HTML 4.0//EN" -->
<HTML>
  <HEAD>
    <TITLE> Descrição do documento </TITLE>
    Aqui ficam blocos de controle, folha de estilo, funções de
    scripts, informações de indexação, atributos que afetam todo
    o documento, etc.
  </HEAD>
  <BODY>
    Toda a informação visível da página vem aqui.
  </BODY>
</HTML>
```

A primeira linha: <!DOCTYPE HTML Public "-//IETF//DTD HTML 4.0//EN" --> é um *descritor SGML* que informa ao browser que ele deve interpretar o documento de acordo com a definição do HTML versão 4.0, se possível. HTML é uma linguagem derivada e comple-

tamente especificada a partir da linguagem SGML – *Standard Generalized Markup Language*. Você não precisa saber SGML para usar HTML, basta recortar e colar a linha acima que serve para orientar o browser e permitir a validação do código de sua página.

O elemento `<HTML>...</HTML>` marca o início e o final do *documento HTML*. Deve conter duas sub-estruturas distintas: o *cabeçalho*, delimitado por `<HEAD>` e `</HEAD>`, e o *corpo do documento*, entre os descritores `<BODY>` e `</BODY>`.

### 3.6. Elementos do HEAD

O bloco do cabeçalho, marcado por `<HEAD>` e `</HEAD>` pode conter informações *sobre* o conteúdo do documento utilizada para fins de indexação e organização. Não contém informação que será exibida na página.

#### TITLE

`<TITLE>` é o único elemento obrigatório do bloco do cabeçalho. Deve conter o título do documento que aparece fora da página, na barra de título do browser. É o que aparece também nos *hotlists* e *bookmarks*. O título deve conter informações que descrevam o documento.

```
<TITLE>HTML e CSS: Introdução</TITLE>
```

#### META

`<META>` é usado para incluir meta-informação como palavras-chave, descrições, etc. que podem ser usadas por mecanismos de busca, softwares de pesquisa e catalogação. A informação adicional deve vir nos atributos NAME (descreve o tipo de meta-informação, por exemplo *Keywords*) e CONTENT (descreve o conteúdo da meta-informação, por exemplo, uma lista de palavras-chave separadas por vírgula). `<META>` também pode ser usado para adicionar ou redefinir *cabeçalhos HTTP*. Isto é feito através do atributo HTTP-EQUIV. Neste caso, o CONTENT deve conter o conteúdo do cabeçalho.

```
<META HTTP-EQUIV="Set-Cookie" CONTENT="pag=12">
```

```
<META NAME="Keywords" CONTENT="html, css, folhas de estilo, estilo">
```

```
<META NAME="Description" CONTENT="Esta página explica os fundamentos básicos de HTML e folhas de estilo usando a linguagem CSS.">
```

```
<META HTTP-EQUIV="Refresh" CONTENT="10;url=pag13.html">
```

```
<META HTTP-EQUIV="Content-Type" CONTENT="text/html; charset=iso-8859-1">
```

## LINK

<LINK> é usado para vincular uma página a outro recurso. Na maioria dos casos, o vínculo é simbólico e não tem outra finalidade a não ser facilitar a indexação e organização do site por ferramentas de validação e busca. Vínculos a folhas de estilo incorporam o arquivo que é interpretado pelo browser antes de formatar a página..

```
<LINK REL="StyleSheet" HREF="../estilos/default.css">
```

## BASE

<BASE> altera os vínculos de origem e destino da janela, ou seja, a parte absoluta da URL das imagens e vínculos, especificados com URLs relativas, e a janela onde o resultado dos vínculos será mostrada. Normalmente a URL base de origem é o local onde a página se encontra no momento em que carrega a imagem ou o usuário clica no vínculo (protocolo atual, máquina atual e diretório atual ou "/"). Com <BASE>, o autor pode alterar a URL base para que as imagens e links sejam buscados em outros lugares. Normalmente a janela atual (\_self) é a responsável por receber o resultado dos vínculos. O autor também pode, com <BASE>, fazer com que links abram em outra janela, em novas janelas (\_blank) ou em outras partes de estrutura de *frames* (\_top e \_parent).

```
<BASE HREF="." TARGET="_self"> <!--BASE default -->
```

```
<BASE HREF=" http://www.a.com/dados/">
```

```
<!--Os links relativos serão procurados em http://www.a.com/dados/-->
```

```
<BASE TARGET="lateral2"> <!-- links abrirão na janela lateral2 -->
```

## 3.7. Elementos de BODY

O bloco marcado por <BODY> e </BODY> contém a parte do documento onde será colocada a informação que efetivamente será mostrada e formatada na tela pelo browser. Todos os elementos que serão apresentados daqui em diante tratam da formatação do texto (ou imagens) da página e devem, portanto, estar presentes no bloco <BODY>. Elementos HTML podem ser de vários tipos e têm regras específicas sobre o que podem conter e onde podem ser usados. Quanto à estrutura que assumem na página, podem ser classificados da seguinte forma:

- Elementos de *bloco* (contém texto ou agrupam outros elementos de bloco)
- Elementos *inline* (em linha – usados dentro do texto)
- Elementos de *lista* (usados para construir listas)
- Elementos de *tabela* (usados para construir tabelas)
- Elementos de *formulário* (usados para construir formulários)
- Elementos para embutir *objetos* (usados para incluir imagens, applets, vídeos, etc.)

Entre os elementos *inline* estão os vínculos (*links*) de hipertexto, que permitem marcar texto que servirá de ligação a outra página ou recurso na Internet.

O objetivo das seções seguintes será apresentar uma introdução básica aos principais elementos HTML. Não é uma introdução exaustiva. Consulte as referências e documentação oficial para uma abordagem completa de cada elemento HTML.

## 4. Parágrafos e blocos

*Blocos* são elementos que podem ocorrer dentro de <BODY>. Não é permitido existir texto diretamente abaixo de <BODY>. O texto deverá estar dentro de um bloco, tabela, lista ou formulário. Alguns blocos HTML só permitem que haja outros blocos dentro deles (é o caso de <BODY>). Outros só admitem texto e elementos *inline* (*em linha* – elementos que não são blocos e podem ser usados dentro de parágrafos para formatar texto, marcar posições ou incluir imagens). Alguns admitem blocos e texto (<TD>, <LI>, etc.).

### 4.1. Parágrafos <P>

O descritor <P> abre um parágrafo em HTML. O descritor </P> o fecha. Tudo o que está entre <P> e </P> é tratado como um bloco. Se esse nosso parágrafo tiver atributos, eles afetarão todo o bloco.

Se seu arquivo HTML contiver:

```
<P>Um parágrafo</P> <P>Outro parágrafo</P>
```

um browser como o *Netscape* ou o *Internet Explorer* mostrará na tela algo parecido com:



<P> pode ter atributos. Um deles é o atributo ALIGN. Se ele não for usado, cada parágrafo alinha pela esquerda. Ele pode ser usado para fazer o parágrafo alinhar pelo centro ou pela direita, por exemplo:

```
<P ALIGN=center>Um parágrafo</P> <P ALIGN=right>Outro parágrafo</P>
```

vai resultar em:



O atributo ALIGN, como qualquer outro atributo que altera o posicionamento ou aparência através do HTML, foi *deprecado* na especificação HTML 4 em benefício das folhas de estilo. Nem todos os browsers suportam, porém, os recursos de folha de estilo que substituem o ALIGN. Mas as folhas de estilo, quando presentes, sempre têm precedência. Se uma folha de estilos ditar um alinhamento, aquele proposto pelo atributo ALIGN é ignorado.

O elemento </P> de um parágrafo é opcional. Como não pode haver um parágrafo dentro de outro ou qualquer *elemento de bloco* dentro de um parágrafo, o browser automaticamente fecha o parágrafo quando encontra um <P>, <H1>, <DIV>, <BLOCKQUOTE>, <PRE> ou outro elemento de bloco. Blocos em HTML nunca podem ocorrer dentro do texto de um parágrafo.

## 4.2. Cabeçalhos <H1> a <H6>

Um outro bloco importante na organização do texto é o cabeçalho. Há seis níveis diferentes. O mais importante é definido com <H1> e pode ser usado para destacar, por exemplo, o título de um documento. Não confunda esse título com <TITLE>, elemento de <HEAD>, que define um título para a página mas não aparece na mesma.

```
<h1>La Cosa Nostra Pizzeria i Ristorante</h1>
```

Observe no seu browser que o texto aparece grande e em negrito (por causa da folha de estilos do browser).

Assim como <P>, <H1> também suporta o atributo ALIGN, portanto, pode-se usá-lo para colocar o título alinhado pelo centro:

```
<h1 align=center>La Cosa Nostra Pizzeria i Ristorante</h1>
```

Além do H1, há mais 5 níveis de cabeçalho que podem ser usados para organizar os parágrafos de uma página em seções. Veja na figura abaixo o código e o resultado no browser de textos marcados com H1, H2, H3, H4, H5 e H6.



### 4.3. Quebras de linha <BR>

Os parágrafos sempre têm uma linha em branco separando-os dos outros blocos (por causa da folha de estilos do browser). Pode-se quebrar uma linha e ainda assim se manter no mesmo parágrafo usando o marcador <BR>. <BR> é um marcador vazio. Não pode ter descritor de fechamento. <BR> sempre quebra a linha na posição onde for colocado:

```
<p align=center>"Tradição, qualidade e<br>
segurança são nossas<br>
prioridades"</p>
```

Quebras de linha só podem ser usadas dentro de parágrafos, cabeçalhos e outros elementos de bloco que contém texto.

### 4.4. Linhas horizontais <HR>

Linhas horizontais decorativas podem ser produzidas pelo <HR>. <HR> é como <BR>. Não tem marcador final de fechamento. Acontece no lugar onde o colocarmos e ainda introduz uma quebra de linha antes e depois.:

```
<h1 align=center>La Cosa Nostra Pizzeria i Ristorante</h1>
<hr>
<h2 align=center>Una impresa de la famiglia Corleone</h2>
```

A linha atravessa toda a largura do browser. Podemos usar o atributo WIDTH para torná-la menor. WIDTH aceita um número absoluto em pixels ou uma porcentagem:

```
<hr width=50%>
```

Há vários outros atributos. Consulte a documentação (ou digite a tecla F1 quando estiver com o cursor dentro do elemento <HR> no *HomeSite*) e teste os possíveis resultados no seu browser.

Linhas horizontais são elementos de bloco, ou seja, eles quebram a linha antes e depois. Você não pode ter uma linha horizontal dentro de um parágrafo ou cabeçalho.

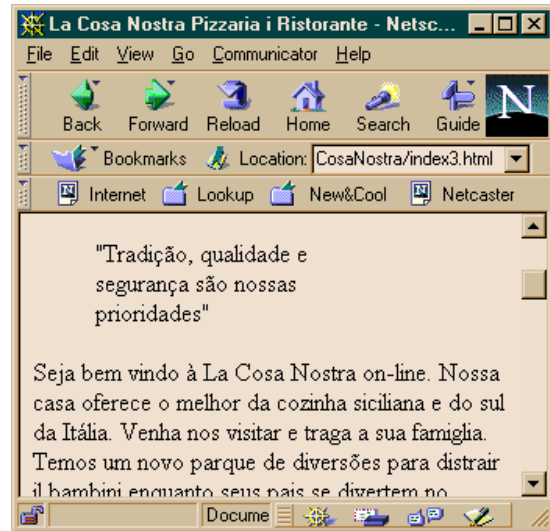
### 4.5. Citação de bloco <BLOCKQUOTE>

Com a função de dar o destaque de um bloco de *citação* em um texto, foi criado o descritor <BLOCKQUOTE>. Na época em que havia mais de dois browsers dividindo as preferências na Web, alguns browsers o formatavam em itálico, outros em fonte menor, outros endentavam. Hoje, com o monopólio *Microsoft-Netscape*, todos os browsers formatam <BLOCKQUOTE> endentado e com uma linha em branco antes e depois. O que era uma questão de estilo acabou virando regra de formatação. Na falta de um mecanismo de controle

gráfico como as folhas de estilos, a função de <BLOCKQUOTE> acabou reduzida a “bloco endentado”.

```
<blockquote>
<p>"Tradição, qualidade e<br>
segurança são nossas<br>
prioridades"</p>
</blockquote>
```

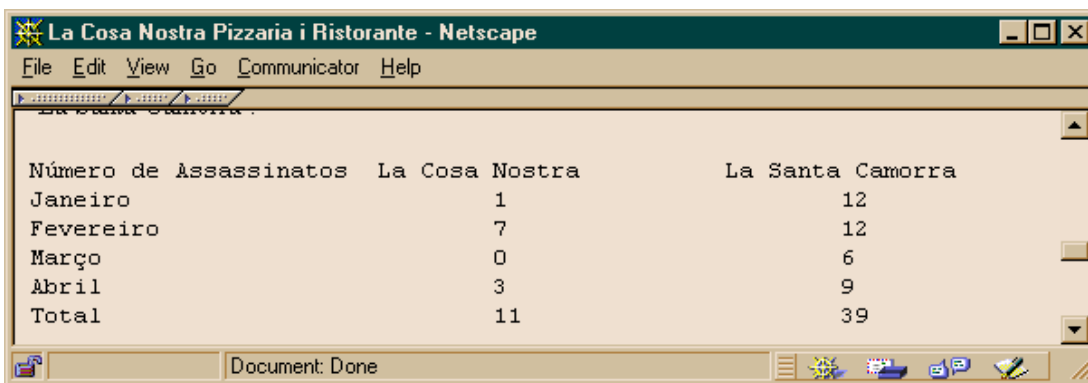
Mas <BLOCKQUOTE> não deve ser usado pensando só na aparência. Qualquer folha de estilos pode endentar um parágrafo ou mudar a aparência de <BLOCKQUOTE>. Com o suporte crescente de CSS, o uso de <BLOCKQUOTE> apenas com a finalidade de endentar o texto deve diminuir.



#### 4.6. Texto pré-formatado <PRE>

Vimos que o HTML ignora espaços, tabulações, novas-linhas e coisas parecidas. Se isto for necessário, podemos resolver o problema usando o descritor <PRE>. Esse descritor preserva as características do texto previamente formatado e leva em considerações as novas-linhas, espaços e tabulações:

```
<pre>
<p>Número de Assassinatos      La Cosa Nostra      La Santa Camorra
Janeiro                       1                   12
Fevereiro                     7                   12
Março                         0                   6
Abril                         3                   9
Total                         11                  39</p></pre>
```



Se a largura da tabela passar da largura da página, é só ajustar diminuindo um pouco o número de tabulações ou trocando por espaços que o ajuste fica perfeito.

Há formas melhores de criar tabelas em HTML, como veremos adiante.



## 4.7. Bloco genérico de seção <DIV>

Podemos decidir criar divisões na página pelos mais diversos motivos. Até por motivos de estrutura lógica (mesmo que o resultado não apareça visualmente, pode ser usado por um mecanismo de busca ou indexação). Com folhas de estilo, podemos atribuir um estilo totalmente diferente a uma seção da página, como mudar fundo, fontes, cores, etc.

O descritor <DIV> é usado para definir uma nova seção (ou divisão). Este descritor não faz nada se não tiver atributos. Um dos atributos é ALIGN, que tem a mesma função que o ALIGN de <P> e <H1> só que afeta um bloco inteiro, com todos os descritores que houver dentro. A precedência ocorre de dentro para fora. Se houver um <P ALIGN=center> dentro de um <DIV ALIGN=right>, o alinhamento de <P> prevalece.

<DIV> é um dos descritores mais úteis em páginas formatadas por folhas de estilo.

## 4.8. Bloco de endereço <ADDRESS>

O bloco <ADDRESS> tem apenas finalidade funcional. Indica um bloco que contém um endereço. O browser apenas formata o texto contido em <ADDRESS> de forma diferente (em itálico, por exemplo). Como tem pouco valor para controlar a aparência de uma página, é pouco usado e raramente é gerado por ferramentas gráficas como *FrontPage* ou *DreamWeaver*. É útil em clientes não gráficos como mecanismos de busca, filtros, etc.

**<ADDRESS>**

<P>Copyright &copy 2000,

<a href="mailto: fulano@tal.com">Fulano de Tal</a>.

<P>Rua Agá tê Eme-éle, 4, Cê-ésse Ésse, WebCity, Internet.

**</ADDRESS>**

# 5. Listas

HTML define várias formas de apresentar listas de informação em um documento. Toda lista é um elemento de bloco que possui um descritor inicial e final e só pode conter "itens de lista". Os itens de lista também são blocos que podem conter texto, parágrafos ou ainda outras listas. HTML define três tipos de listas: as listas ordenadas, marcadas pelos descritores `<OL>` e `</OL>`; as listas não-ordenadas marcadas por `<UL>` e `</UL>` e as listas de definições, marcadas por `<DL>` e `</DL>`.

## 5.1. Listas não-ordenadas `<UL>`, `<LI>`

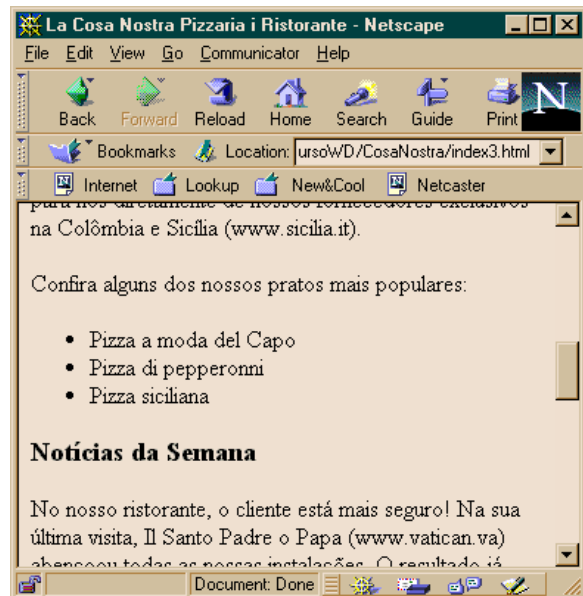
Nas listas UL, cada item da lista é contido dentro de `<LI>` e `</LI>`. O browser geralmente formata os itens com marcadores (bolinhas pretas). Veja a imagem ao lado e o código abaixo:

```
<ul>
  <li>Pizza a moda del Capo</li>
  <li>Pizza di pepperonni</li>
  <li>Pizza siciliana</li>
</ul>
```

Você também pode criar listas de nível hierárquico inferior. O browser formatará a sub-lista de uma forma diferente (geralmente endentada e com um marcador diferente). Por exemplo, para fazer a seguinte lista:

- item 1
- item 2
  - item 2.1
  - item 2.2
- item 3

pode-se utilizar o seguinte código:



```

<ul>
  <li>item 1</li>
  <li>item 2</li>
  <ul>
    <li>item 2.1</li>
    <li>item 2.2</li>
  </ul>
  <li>item 3</li>
</ul>

```


## 5.2. Listas ordenadas <OL>, <LI>

As listas ordenadas são marcadas pelos descritores <OL> e </OL>. Da mesma forma que nas listas não-ordenadas, cada item é contido dentro de <LI>. Poderíamos numerar as pizzas trocando os UL por OL:

```

<ol>
<li>Pizza a moda del Capo</li>
<li>Pizza di pepperonni</li>
<li>Pizza siciliana</li>
</ol>

```

- 
1. Pizza a moda del Capo
  2. Pizza di pepperonni
  3. Pizza siciliana

As listas e seus itens podem ter atributos que mudam suas características como tipo de marcador, início da contagem (para listas ordenadas), tipo de numeral (arábico, latino, etc.) A maior parte dessas alterações pode também ser realizada em folhas de estilo.

## 5.3. Listas de definições <DL>, <DT>, <DD>

Finalmente, existem as listas de definições. Esse tipo de lista é usado quando há vários tópicos curtos acompanhados de uma definição mais longa (em um glossário, por exemplo). A lista de definições consiste de um conjunto de três descritores:

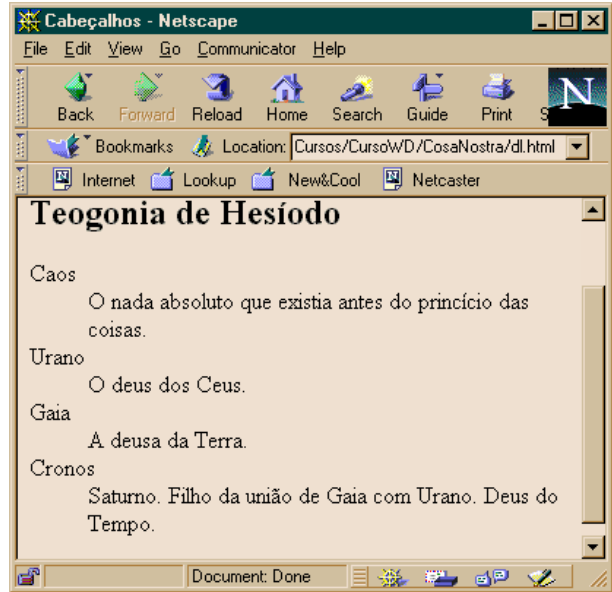
- <DL> e </DL> que delimitam toda a lista;
- <DT></DT> que marca o termo; e
- <DD></DD> marca a definição.

A imagem ao abaixo foi produzida com os descritores ao lado:

```

<dl>
<dt>Caos</dt>
  <dd>O nada absoluto que existia antes do princípio das coisas.</dd>
<dt>Urano</dt>
  <dd>O deus dos Céus.</dd>
<dt>Gaia</dt>
  <dd>A deusa da Terra.</dd>
<dt>Cronos</dt>
  <dd>Saturno. Filho da união de Gaia com Urano. Deus do Tempo.</dd>
</dl>

```



# 6. *Imagens*

Já vimos que uma página HTML é uma mera página de texto. Mesmo com imagens, ela continuará a ser uma mera página de texto, não vai aumentar de tamanho pois as imagens sempre ficarão separadas. As imagens são carregadas na hora em que o browser lê a página, através de um vínculo (link) para a sua localização através de uma URL. Ela pode estar no mesmo diretório que a página ou em outra parte do mundo. Desde que se informe o seu endereço corretamente e que a rede não esteja sem comunicação, o browser localizará a imagem e a colocará na página no local onde estiver o descritor <IMG> que a solicitou.

## 6.1. *O objeto <IMG>*

<IMG> é como <BR> e <HR> que não marcam texto, mas uma posição, por isso não têm descritores de fechamento. No caso de <IMG>, o atributo SRC é obrigatório pois é ele quem informa a localização da imagem através de sua URL. A URL pode ser tanto um endereço absoluto (<http://alguma.coisa/>) como um endereço relativo à URL da própria página. Por exemplo, se a URL página está em <http://blablabla.com/dirdir/pagina.html> e a URL da imagem que a página carrega é <http://blablabla.com/dirdir/imagem.gif>, pode-se simplesmente chamar a imagem pelo endereço relativo "imagem.gif", usando <IMG SRC="imagem.gif"> na posição, relativa ao texto, onde ela deva aparecer.

### *URIs relativas*

O browser, como está remoto, precisa utilizar uma *URI absoluta* que contenha o protocolo, o nome da máquina onde roda o servidor e a porta de serviços (se não for a porta padrão). A URI usada internamente no servidor (o sistema de arquivos virtual) contém apenas o caminho absoluto dentro do mesmo (/, /Imagens ou /Videos). As páginas HTML disponíveis via servidor Web estão armazenadas em determinada máquina e diretório e podem usar, para comunicação entre si, *URIs relativas* à sua localização.

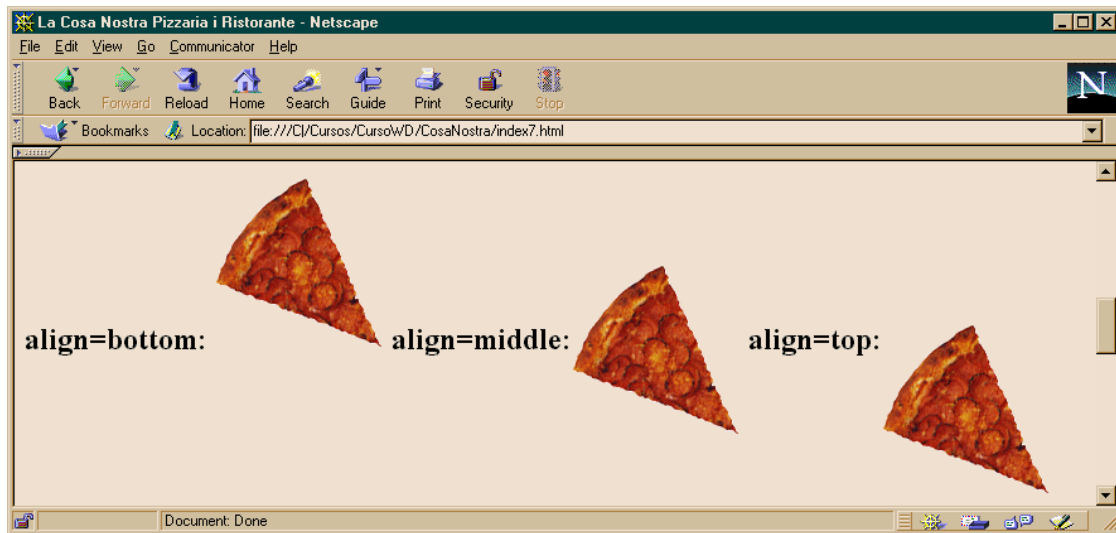
As URIs relativas são úteis em HTML. Um vínculo em uma página pode se referir a um arquivo que ocupa o mesmo diretório que ela simplesmente usando o nome do arquivo como URI (e omitindo <http://maquina:porta/caminho/>). Para ter acesso a um diretório anterior, usa-se a notação Unix: `../arquivo.txt` refere-se a um arquivo chamado `arquivo.txt` localizado no diretório anterior..

Resumindo: servidores e browsers só localizam recursos usando URIs. Não faz sentido escrever `C:\qualquercoisa\x.txt` no browser a não ser para utilizá-lo como visualizador de páginas HTML armazenadas *localmente*. Também não faz sentido usar caminhos desse tipo em arquivos HTML (como veremos). Eles podem até funcionar localmente, sem servidor, mas não mais funcionarão quando forem publicados em um servidor Web. O ideal é usar URIs relativas que funcionam em qualquer situação.

## 6.2. Alinhamento de imagens

A imagem pode ser alinhada de várias formas em relação ao texto usando HTML (ou folhas de estilo). Veja abaixo alguns exemplos.

`` e outros...

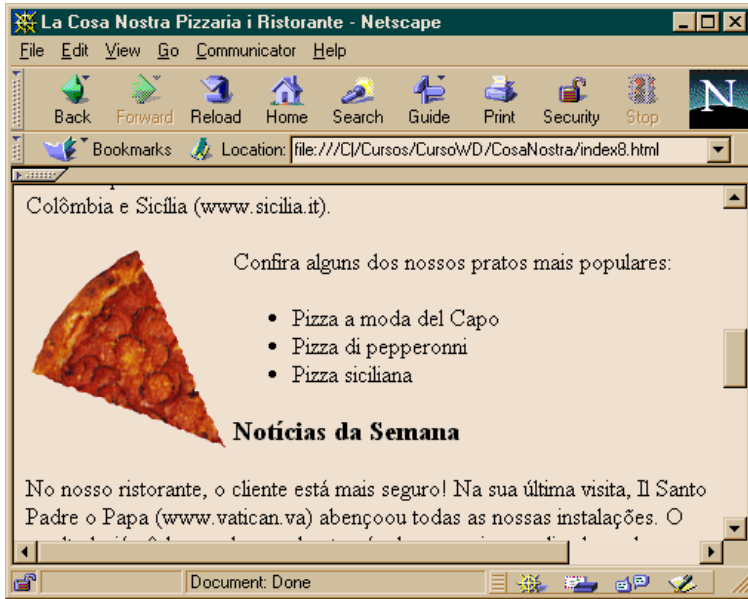


Use `ALIGN=left` ou `ALIGN=right` para fazer com que o texto corra à esquerda ou à direita da imagem:

``

## 6.3. Dimensionamento

Se a imagem ficar muito grande, podemos reduzi-la um pouco usando os atributos `HEIGHT` e `WIDTH`. Através deles podemos mudar quaisquer dimensões da imagem ao ser exibida, em pixels. Mas cuidado! A qualidade da imagem e seu tamanho continuam os mesmos. Carregar uma imagem gigantesca para depois reduzi-la é desperdício assim como há perda de qualidade ao se ampliar uma imagem pequena.



HEIGHT e WIDTH não servem só para redimensionar imagens. Usando esses atributos em todas as suas imagens com as suas dimensões verdadeiras torna o carregamento da página mais rápido. Por que isto? Porque o browser não vai precisar calcular as dimensões da imagem para saber onde colocar o texto. Se não houver HEIGHT e WIDTH ele só pode mostrar o texto depois de saber o tamanho das imagens. Para descobrir o tamanho da imagem, use um

editor gráfico como o *Lview* ou *PhotoShop*. Que tal uma pizza esticada?

```

```

Se você não gostou da pizza esticada, use as dimensões corretas em pixels que são HEIGHT=137 e WIDTH=134.

Ainda há alguns atributos que permitem que se empurre o texto para longe da figura. Na verdade, criam uma margem invisível ao redor dela. HSPACE e VSPACE, respectivamente acrescentam margens horizontais e verticais, em pixels, em volta da figura. Todos os atributos de alinhamento podem ser substituídos por folhas de estilo.

```

```

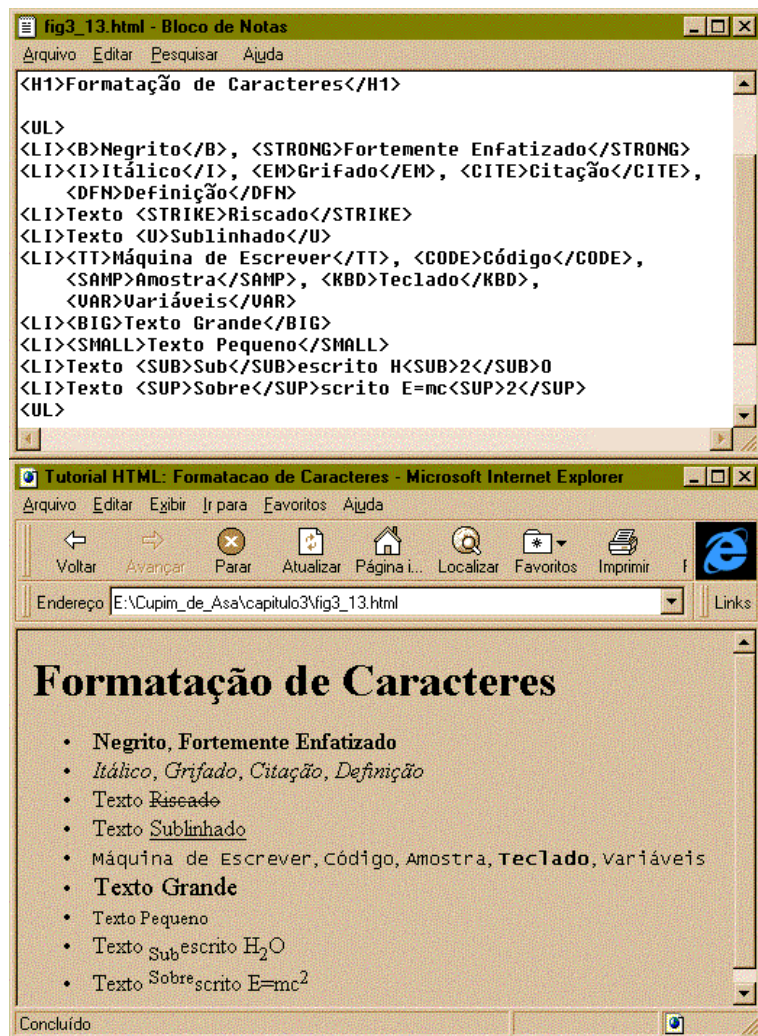
# 7. Formatação de texto

## (inline elements)

Há vários descritores que têm a função de atribuir um papel a trechos de texto. Na prática, este "papel" se reflete na forma de um destaque diferente. Somente com folhas de estilo o "papel" é aproveitado totalmente.

Os descritores de formatação de caracteres são de dois tipos:

- descritores *lógicos* (aqueles que tem uma *função*): `<STRONG>`, `<CITE>`, `<DFN>`, `<CODE>`, `<Q>`, `<VAR>`, `<KBD>`, `<SAMP>`, `<EM>`, `<SPAN>`, `<ABBR>` e `<ACRONYM>`
- os *físicos* (aqueles que não tem função estrutural, só *aparência*): `<B>`, `<I>`, `<BIG>`, `<SMALL>`, `<S>`, `<STRIKE>`, `<SUB>`, `<SUP>`, `<TT>`, `<U>`



Esses elementos servem para marcar texto dentro de blocos que os contém como parágrafos, cabeçalhos, células de tabelas, etc. Os descritores físicos têm um nome associado à sua



aparência, mas nada impede que uma folha de estilos mude isto. Os descritores lógicos têm uma aparência *default* mas o que interessa é a sua função (CITE de citação, ABBR de abreviação, VAR de variável). Alguns sequer mudam a aparência do texto a não ser que um estilo seja atribuído através de CSS (<SPAN>, <ABBR> e <ACRONYM>, por exemplo, não mudam a aparência do texto).

A figura acima mostra a aparência *default* de alguns dos elementos. A sua função está descrita abaixo:

Os elementos abaixo significam mais pela sua função que pela sua aparência. São elementos lógicos.

EM	texto grifado
STRONG	texto fortemente grifado
DFN	definição
CODE	código
SAMP	amostra
KBD	teclado
VAR	variável
CITE	citação
SAMP	amostra
ABBR	abreviação
ACRONYM	sigla
SPAN	container genérico
Q	texto de citação (entre aspas)

Estes outros elementos têm um nome que enfatiza a sua aparência (são os elementos físicos). Folhas de estilo, porém, podem alterar tudo.

TT	texto de espaçamento fixo
I	texto em itálico
B	texto em negrito
S	texto riscado (mesmo que STRIKE)
STRIKE	texto riscado
U	texto sublinhado
BIG	texto em fonte maior
SMALL	texto em fonte menor
SUB	texto sub-escrito
SUP	texto sobre-escrito

## 8. Âncoras e Vínculos

Uma âncora é qualquer um dos lados de um vínculo de hipertexto. Em outras palavras, tanto a origem quanto o destino de um link são âncoras. Você pode criar uma âncora destino em uma página HTML usando o elemento <A> com atributo NAME (para dar um nome à âncora):

```
<A NAME="cap1"></A> <H1>Capítulo 1 </H1>
```

O descritor pode ficar vazio, já que só marca uma posição, mas o marcador final é necessário.

Para criar um vínculo à âncora, utiliza-se o mesmo elemento <A>, em outro lugar da página ou em outra página, marcando um texto ou imagem que servirá de ponto de partida, indicando o destino através do atributo HREF:

```
<A HREF="#cap1">Ir até o capítulo 1</A>
```

O trecho acima vincula a frase "Ir ao capítulo 1" com a posição onde está a âncora "cap1", desde que ela esteja definida no mesmo arquivo.

Para especificar um destino externo, é preciso usar uma URI (relativa ou absoluta). A URI pode apontar para qualquer coisa (não precisa ser uma página HTML).

```
<A HREF="http://www.abc.com/dados/index.html">Ir até o arquivo Index.html</A>
```

Se o destino for uma determinada seção da página, pode-se complementar a URI com o fragmento correspondente ao nome da âncora definida no arquivo:

```
<A HREF="http://www.abc.com/dados/index.html#cap1">Ir até o capítulo 1 no arquivo Index.html</A>
```

# 9. Tabelas

Até a versão 2.0, HTML só fornecia meios de estruturar informação que fosse formatada seqüencialmente, ou seja, o fluxo do texto e imagens era único e acontecia da esquerda para a direita e de cima para baixo. Não havia meios de dispor, por exemplo, dois parágrafos lado-a-lado a não ser que todo o texto fosse formatado usando espaços e tabulações e posteriormente incluído em um bloco `<PRE>`. A introdução das tabelas, proposta no HTML 3.0 (mas adotada pelos browsers mais populares como uma extensão do HTML 2.0), tornou possível a classificação da informação com marcadores HTML que seria formatada no browser em linhas e colunas.

Os marcadores eram inicialmente destinados à organização de dados em forma de tabelas, mas em pouco tempo passaram a ser usados pelos Web designers para obter maior controle sobre a apresentação do conteúdo das suas páginas. Imagens com partes dinâmicas, páginas de mais de uma coluna e controle da largura da página são algumas das soluções possíveis em HTML usando tabelas.

Por ter tantas utilidades, as tabelas são um tópico complexo em HTML, mas as possibilidades que oferecem compensam o esforço de conhecer bem cada um dos seus atributos.

## 9.1. Principais elementos estruturais `<TABLE>`, `<TR>`, `<TD>` ou `<TH>`

Uma tabela precisa ter no mínimo três elementos HTML. O elemento `<TABLE>` representa a tabela e contém uma ou mais linhas, representadas pelo elemento `<TR>`. O elemento `<TR>` é o único, dentro de um bloco `<TABLE>...</TABLE>` que contém a informação da tabela. Essa informação *precisa* estar dentro de elementos `<TD>` ou `<TH>`, que são os únicos permitidos dentro de um `<TR>`.

`<TR>...</TR>` representa uma linha da tabela (*Table Row*). Cada linha pode ter uma ou mais células de dados em blocos `<TD>...</TD>` (*Table Data*) ou `<TH>...</TH>` (*Table Header*). O número de colunas da tabela é, portanto, derivado do número de células contidas no bloco `<TR>...</TR>` que tiver mais blocos `<TH>...</TH>` ou `<TD>...</TD>`.

Como os elementos `<TD>`, `<TH>` e `<TR>` não podem ocorrer dentro de blocos do mesmo tipo, os descritores finais `</TD>`, `</TH>` e `</TR>` são opcionais.

Os exemplos abaixo ilustram tabelas simples. O código HTML:

```
<p><table border>
<tr> <td>1, 1</td> <td>1, 2</td> <td>1, 3</td> </tr>
<tr> <td>2, 1</td> <td>2, 2</td> <td>2, 3</td> </tr>
<tr> <td>3, 1</td> <td>3, 2</td> <td>3, 3</td> </tr>
</table>
```

resulta na seguinte formatação:

1,1	1,2	1,3
2,1	2,2	2,3
3,1	3,2	3,3

Este outro exemplo

```
<TABLE BORDER=1>
<TR><TD>Mercúrio</TD><TD>Venus</TD><TD>Terra</TD></TR>
<TR><TD>Marte</TD><TD>Júpiter</TD><TD>Saturno</TD></TR>
<TR><TD>Urano</TD><TD>Netuno</TD><TD>Plutão</TD></TR>
</TABLE>
```

produz:

Mercurio	Venus	Terra
Marte	Jupiter	Saturno
Urano	Netuno	Plutão

## 9.2. Tabela sem bordas

BORDER é um atributo de <TABLE> que faz com que a tabela seja desenhada com uma borda *default* (1 pixel). Se o nome BORDER não estiver presente, a borda não aparecerá, embora o espaço de 1 pixel em volta da tabela permaneça. Veja o exemplo abaixo (tabela simples sem bordas):

```
<TABLE>
<TR><TD>Maranhão</TD><TD>Piauí</TD><TD>Ceará</TD></TR>
<TR><TD>Rio Grande do Norte</TD><TD>Paraíba</TD>
<TD>Pernambuco</TD></TR>
<TR><TD>Alagoas</TD><TD>Sergipe</TD><TD>Bahia</TD></TR>
</TABLE>
```

Maranhão	Piauí	Ceará
Rio Grande do Norte	Paraíba	Pernambuco
Alagoas	Sergipe	Bahia

Os elementos <TD> e <TH> de <TABLE> podem conter qualquer texto ou elementos HTML utilizados dentro de <BODY>, como imagens:

- Altímetro
- Taquímetro
- Indicador de velocidade do vento
- Bússola
- Nível de combustível e óleo
- Termômetros

```
<table>
<tr><td></td> <td>Altímetro </td></tr>
<tr><td></td> <td>Taquímetro </td></tr>
<tr><td></td> <td>Indicador ...</td></tr>
<tr><td></td> <td>Bússola </td></tr>
<tr><td></td> <td>Nível de ... </td></tr>
<tr><td></td> <td>Termômetros </td></tr>
</table>
```

### 9.3. Bordas

O elemento <TABLE> contém vários outros atributos que alteram a aparência de toda a tabela. BORDER=valor permite especificar a largura da borda da tabela através de um valor absoluto em pixels. A tabela abaixo possui uma borda de 10 pixels:

<a href="#">Estate</a>	<a href="#">Autunno</a>
<a href="#">Primavera</a>	<a href="#">Inverno</a>

```
<TABLE BORDER=10>
<TR><TD><a href="estate.html">Estate</a></TD>
<TD><a href="autunno.html">Autunno</a></TD></TR>
<TR><TD><a href="primavera.html">Primavera</a></TD>
<TD><a href="inverno.html">Inverno</a></TD></TR>
</TABLE>
```

### 9.4. Espaçamento

CELLSPACING=valor controla a quantidade de espaço entre as células de uma tabela. Normalmente esse espaço é de 1 pixel. CELLPADDING=valor controla a quantidade de espaço entre o texto da célula e suas bordas (margens internas das células). O valor *default* é de 2 pixels, portanto, se as bordas forem zero, haverá 3 pixels entre os elementos da tabela a não ser que esse espaço seja alterado com CELLSPACING e CELLPADDING. Veja os exemplos abaixo:

Aranha	Lacrau	Escorpião
Barata	Formiga	Mosca

```
<TABLE BORDER CELLPADDING=10 CELLSPACING=0>
  <TR><TD>Aranha</TD><TD>Lacrau</TD><TD>Escorpião</TD></TR>
  <TR><TD>Barata</TD><TD>Formiga</TD><TD>Mosca</TD></TR>
</TABLE>
```

Aranha	Lacrau	Escorpião
Barata	Formiga	Mosca

```
<TABLE BORDER CELLPADDING=0 CELLSPACING=10>
  (...) </TABLE>
```

Aranha	Lacrau	Escorpião
Barata	Formiga	Mosca

```
<TABLE BORDER CELLPADDING=10 CELLSPACING=10>
  (...) </TABLE>
```

Se os atributos `BORDER`, `CELLPADDING` e `CELLSPACING` forem definidos com o valor "0", não haverá espaço algum entre os elementos da tabela a não ser que o espaço seja adicionado de forma explícita (linhas em branco, `&nbsp;`; tabulações ou espaços dentro do bloco `<TD>`).

## 9.5. Largura da tabela

O atributo `WIDTH=numero` ou porcentagem permite especificar a largura da tabela, através de uma medida absoluta em pixels ou através de uma porcentagem da largura visível da página. Sem `WIDTH`, a tabela ocupará apenas o espaço horizontal necessário para conter todos os seus elementos, as bordas e margens. Se uma tabela for definida com `WIDTH=100%`, ela ocupará toda a largura visível da página e mudará de tamanho caso a janela seja redimensionada (e a área visível seja alterada). Se for definida com `WIDTH=500`, terá uma largura fixa e que não mudará mesmo com o redimensionamento da janela.

No uso de tabelas como ferramenta para o *layout* das páginas, `WIDTH` é usado para determinar a largura da página que está contida em uma tabela de única célula. Páginas criadas para a resolução de 640x480 são tipicamente definidas dentro de tabelas com largura de 600 pixels. Para 800x600, é comum usar 750 pixels:

```

<body>
<table width=750>
<tr><td> (... a página inteira ...) </td></tr>
</table>
</body>

```

No exemplo abaixo há duas tabelas. A primeira não possui o atributo WIDTH. A segunda possui o atributo WIDTH=50%.

16	25	26
34	40	43

16	25	26
34	40	43

```

<TABLE BORDER>
<TR><TD>16</TD> <TD>25</TD> <TD>26</TD></TR>
<TR><TD>34</TD> <TD>40</TD> <TD>43</TD></TR>
</TABLE>

```

```

<TABLE BORDER WIDTH="50%">
<TR><TD>16</TD> <TD>25</TD> <TD>26</TD></TR>
<TR><TD>34</TD> <TD>40</TD> <TD>43</TD></TR>
</TABLE>

```

## 9.6. Altura, posicionamento e cores

De forma análoga ao atributo WIDTH, HEIGHT=numero ou porcentagem permite especificar a altura da tabela. O suporte a esse recurso, porém, não é consistente em browsers que não suportam totalmente o HTML 4.

A tabela, se tiver uma largura menor que o espaço horizontal visível, será alinhada pela margem esquerda da página a não ser que possua o atributo ALIGN com o valor “right” (à direita) ou “center”. HSPACE e VSPACE podem ser usados para acrescentar espaço vertical e horizontal como margens externas para toda a tabela, da mesma forma como são usados por imagens.

O atributo BACKGROUND, se presente, deve receber a URL de uma imagem que irá se repetir no fundo da tabela, da linha ou da célula. BGCOLOR permite que se defina uma cor. É importante observar que a cor *default* da tabela (TABLE) das células (TD) e linhas (TR) é *transparente*, ou seja, se houver imagens ou cores de fundo na página, elas serão visíveis através da tabela. Isto é verdade apenas se a tabela ou células não tiverem atributos BACKGROUND

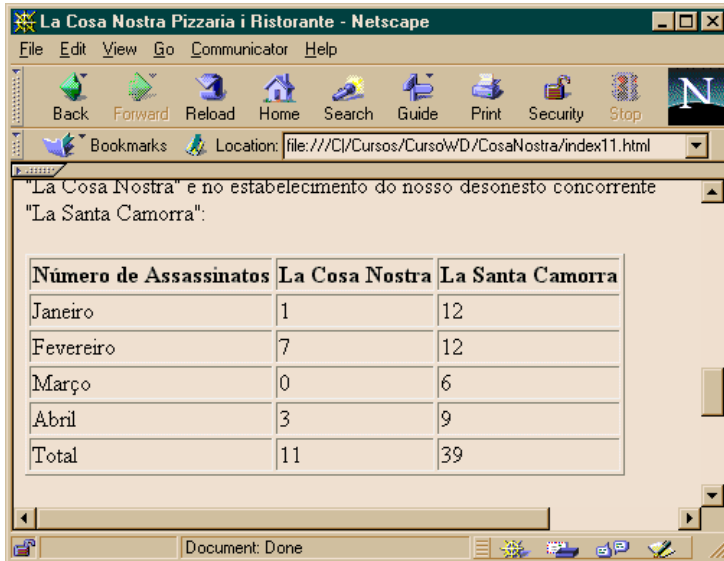
e BGCOLOR. A cor da tabela poderá não ser visível se cores opacas tiverem sido atribuídas a blocos TD e TR.

Veja agora a tabela que antes construímos com <PRE>:

```
<table border>
<tr><th>Número de Assassínatos</th>
    <th>La Cosa Nostra</th><th>La Santa Camorra</th></tr>
<tr><td>Janeiro    </td><td> 1 </td><td> 12</td></tr>
<tr><td>Fevereiro </td><td> 7 </td><td> 12</td></tr>
<tr><td>Março      </td><td> 0 </td><td> 6 </td></tr>
<tr><td>Abril       </td><td> 3 </td><td> 9 </td></tr>
<tr><td>Total      </td><td>11 </td><td> 39</td></tr>
</table>
```

A tabela fica melhor alinhada pelo centro. Use <DIV> para fazer isto.

```
<div align="center"><table border> (...) </table></div>
```



Observe que, por *default*, os dados de <TH> são alinhados pelo centro, enquanto que os dados de <TD> são alinhados com a margem esquerda. Tudo isto e muito mais podem ser alterados usando atributos como ALIGN, VALIGN (alinhamento vertical) além, é claro, folhas de estilo.

Tabelas tendem a ficar muito complexas, principalmente quando usadas umas dentro das outras, o que é bastante comum

no HTML. É importante entender como funcionam as tabelas mas o ideal é criá-las através de programas gráficos como o *DreamWeaver* ou *FrontPage*. Às vezes, porém, é necessário fazer pequenas alterações por causa de diferenças entre browsers e assim, saber identificar as células de uma tabela no código HTML é essencial.

## 9.7. Alinhamento de células

Células e linhas de tabelas podem ter seu conteúdo alinhado sem usar folhas de estilo através dos atributos ALIGN e VALIGN aplicados a <TD> e <TR>. ALIGN pode assumir os valores left, right ou center. VALIGN pode ser top, bottom ou center. Com os dois é possível colocar um objeto alinhado em qualquer posição da tabela.



Qualquer <TD> também pode ter atributos HEIGHT e WIDTH para controlar a altura e largura da célula, respectivamente. Esse valor pode ser absoluto (em pixels) ou relativo à altura/largura totais da tabela.

Mas HEIGHT e WIDTH devem ser usados com cautela. Nem sempre é possível determinar essas dimensões. Blocos <NOBR> e <PRE> podem fazer a tabela crescer além de sua largura máxima estabelecida. A altura não pode ser limitada pois o texto contido na tabela sempre pode fazer com que ela tenha que crescer. Imagens tem todo o poder para destruir completamente quaisquer limitações de altura e largura definidas nas tabelas.

## 9.8. Combinação de células

Células podem ser mescladas em uma só atravessando colunas e linhas com os atributos ROWSPAN e COLSPAN aplicáveis a <TD> ou <TH>. Com esses recursos é possível construir tabelas

O seguinte exemplo utiliza COLSPAN e ROWSPAN para construir uma tabela contendo várias células mescladas:

Três colunas combinadas			Coluna 4 ocupa três linhas
Coluna 1 2 linhas	Coluna 2	Coluna 3	
Colunas 2 e 3 juntas			

```
<table border=2 cellpadding="5">
<tr>
  <td colspan="3">Três colunas combinadas</td>
  <td rowspan="3">Coluna 4<br>ocupa três<br>linhas</td>
</tr>
<tr>
  <td rowspan="2">Coluna 1<br>2 linhas</td>
  <td>Coluna 2</td>
  <td>Coluna 3</td>
</tr>
<tr>
  <td colspan="2">Colunas 2 e 3 juntas</td>
  <td></td>
</tr>
</table>
```

## 9.9. Otimização <THEAD>, <TBODY>, <COLGROUP>, <COL>

Para poder desenhar uma tabela na tela, o browser tem que carregar todo o código antes. Se a tabela for muito complexa ou muito longa, isto pode demorar. Os elementos <COL> e <COLGROUP> permite que se informe o browser com antecedência o número de colunas que a tabela terá, fazendo com que ele ganhe tempo ao desenhá-la. Esses elementos também permitem que se aplique uma formatação em uma coluna ou grupo de colunas (e não apenas nas linhas, como ocorre atualmente).

```

<table border=1>
<colgroup width=100>
  <col>
  <col align=right>
</colgroup>
  <tr><td>Dados 1.1</td><td>Dados 1.2</td><td>Dados 1.3</td></tr>
  <tr><td>Dados 2.1</td><td>Dados 2.2</td><td>Dados 2.3</td></tr>
  <tr><td>Dados 3.1</td><td>Dados 3.2</td><td>Dados 3.3</td></tr>
</table>

```

Os elementos <THEAD>, <TBODY>, e <TFOOT> servem para identificar o cabeçalho, o corpo e o rodapé de uma tabela para poder controlar sua aparência e permitir (futura-mente) a impressão de tabelas em múltiplas páginas com repetição do cabeçalho e rodapé. Se nenhum desses elementos estiver presente, toda a tabela será considerada <TBODY>. Se houver <THEAD>, a ocorrência de <TBODY> marcará o fim daquele bloco (</THEAD> é opcional nesse caso). Veja um exemplo:

```

<table border=1>
<thead valign=top align=center>
  <tr><td>Item 1</td><td>Item 2</td><td>Item 3</td></tr>
</thead>
<tfoot valign=bottom align=center>
  <tr><td>Rodapé 1</td><td> Rodapé 2</td><td> Rodapé 3</td></tr>
</tfoot>
<tbody>
  <tr><td>Dados 1.1</td><td>Dados 1.2</td><td>Dados 1.3</td></tr>
  <tr><td>Dados 2.1</td><td>Dados 2.2</td><td>Dados 2.3</td></tr>
  <tr><td>Dados 3.1</td><td>Dados 3.2</td><td>Dados 3.3</td></tr>
</tbody>
</table>

```

A utilidade de <TBODY>, <TFOOT> e <THEAD> é maior com folhas de estilo. Consulte a documentação para conhecer os atributos usados nesses elementos para controlar a formatação das tabelas.

## 9.10. *Legenda* <CAPTION>

O bloco <CAPTION> pode ser usado dentro de <TABLE>...</TABLE> para incluir uma legenda numa tabela. Pode-se usar o atributo ALIGN para posicionar a legenda acima ou ao lado da tabela.

```

<TABLE border=1>
  (...)
  <CAPTION> Tabela 1: Preços e quantidade</CAPTION>
</TABLE>

```

# 10. Elementos deprecados

## 10.1. Blocos que alteram estilo (deprecados)

O bloco `<ADDRESS>` é pouco usado atualmente. `<BLOCKQUOTE>` é usado bastante, porém somente para endentação. Isto reflete uma preocupação maior com a *aparência* da página que com sua estrutura. Com a possibilidade de uso de folhas de estilo, a formatação oferecida pelo HTML perde importância pois um estilo endentado, por exemplo, pode ser aplicado a qualquer elemento. `<BLOCKQUOTE>` então passará a ter uma importância menor. Só seria usado com a finalidade de marcar uma citação e não meramente para empurrar o texto um pouco para a direita. A função do texto marcado é o que importa em HTML e isto interessa não só a browsers mas a qualquer cliente Web, como por exemplo, os mecanismos de busca. Estes não estão interessados na formatação da página, mas na importância de cada bloco. Para um mecanismo de busca, um bloco `<H1>` é mais importante que um `<P>` ou `<H2>` independente da aparência que ele apresente no browser. Um bloco `<ADDRESS>` pode ser usado para filtrar os autores de cada página de um site, mesmo que o texto marcado por ele sequer altere a aparência visual do endereço, nome ou e-mail do autor quando mostrado na página. Mas como as folhas de estilo chegaram há pouco tempo, ainda há vestígios de elementos que foram criados com finalidades puramente visuais. Eles foram *deprecados* (uso não recomendado) pelo HTML 4.0, e podem ser substituídos por elementos novos, mas vale a pena conhecê-los pois ainda aparecem em páginas geradas por ferramentas como o *FrontPage*, *DreamWeaver* e *Netscape Composer*.

## 10.2. Bloco centrado `<CENTER>`

O Netscape 3.0 não reconhece `<DIV ALIGN=center>`. O Netscape 4 reconhece o elemento `<DIV>` mas não permite que um bloco `<DIV>` contenha outro bloco `<DIV>`. Se o objetivo de um `<DIV>` é meramente centrar um conjunto de parágrafos nele contidos, pode-se substituí-lo por um bloco `<CENTER>...</CENTER>`:

```
<CENTER>
```

```
<TABLE BORDER=1>
```

```
<TR><TD> Tabela centrada na página </TD></TR>
```

```
</TABLE>
</CENTER>
```

### 10.3. *Bloco sem quebras de linha <NOBR> e <WBR>*

Se você colocar um parágrafo dentro de um bloco <NOBR>...</NOBR>, ele não formatará como os parágrafos comuns, quebrando linhas para que todo o texto fique visível. Se o texto for grande o suficiente você terá que rolar a tela horizontalmente para vê-lo. Isto pode ser útil em listagens de programas e outros textos onde a quebra de linha automática pode prejudicar a precisão das informações.

Dentro de um bloco <NOBR>, talvez haja a necessidade de se quebrar a linha num determinado ponto. Para isto sempre pode-se usar <BR>. Em outros casos, quebrar a linha não é desejado mas se for necessário, podemos querer que a quebra ocorra em um lugar específico. Para isto tempos o elemento <WBR>, que só tem utilidade dentro de um bloco <NOBR>. <WBR> indica um ponto de quebra de linha em potencial. Se o browser precisar quebrar uma linha para torná-la visível no browser, ele o fará na posição <WBR>. Se isto não for necessário (todo o parágrafo está visível), então ele ignorará o <WBR>.

```
<NOBR>
<P>Se tiver que ser quebrado, que ocorra aqui<WBR> e não em outro
lugar.</P>
</NOBR>
```

O comportamento do <NOBR> pode ser aplicado a blocos <DIV>, parágrafos individuais, cabeçalhos <H1> a <H6> e outros blocos através de folhas de estilos. O uso de <NOBR> ainda pode ser justificado para garantir a compatibilidade com browsers *Netscape* que não implementam totalmente a especificação de folhas de estilo.

### 10.4. *Elementos de estilo (deprecados no HTML 4.0)*

Não precisamos ficar satisfeitos com o fundo cinza, letras pretas e links azuis. Podemos mudar todas as cores. Veremos futuramente como fazer tudo isto usando folhas de estilo, mas o HTML também oferece meios de realizar alterações de estilo. A especificação HTML recomenda fortemente que esses elementos não sejam usados, mas que se prefira usar folhas de estilo, mas, como o suporte a folhas de estilo não chega aos browsers versão 3.0, e você ainda pode ter clientes com tais browsers que não vão ficar satisfeitos com páginas monocromáticas, é importante conhecer esses elementos. À medida em que os browsers versão 3.0 forem entrando em extinção, deve-se gradualmente abandonar toda a formatação de cores e fontes através do HTML, pois prejudica enormemente a manutenção de um site, além de torná-lo maior e mais sujeito a erros.

## 10.5. Atributos de uma página <BODY atributos>



Os atributos podem ser mudados através do descritor <BODY>. Vários atributos permitem definir uma imagem de papel de parede, uma cor de fundo, cor para todo o texto e links. As cores são informadas usando ou um código RGB em hexadecimal ou um nome padronizado.

Para mudar a cor de fundo, utilizamos o atributo BGCOLOR.

```
<body bgcolor="white"
text="green" link="red"
vlink="black">
```

Usamos nomes de cores. Se quiséssemos usar uma cor diferente das 16 cores daquela tabela básica (veja apêndice), precisaríamos usar códigos hexadecimais. Os códigos são um conjunto de 6 algarismos hexadecimais precedidos de um "#". Pode-se, através deles, obter mais de 200 cores diferentes.

O papel de parede é uma imagem que é repetida várias vezes no fundo. Pode-se usar qualquer imagem mas ela não deve atrapalhar a leitura do texto. Deve ser informada a URL completa ou relativa:

```
<body background="fundo.jpg"
bgcolor="white" text="green" link="red" vlink="black">
```

## 10.6. Cores e fontes <FONT>

Cores de texto individuais, fontes e tamanhos podem ser alterados com um descritor *in-line* chamado <FONT>, que pode receber três atributos:

- SIZE, que informa o tamanho (variando de 1 a 7 ou de -3 a +3);
- FACE, que informa o nome exato de uma fonte ou uma lista delas separadas por vírgula (terminando em uma fonte genérica serif, sans-serif ou monospace); e
- COLOR, que informa uma cor em código hexadecimal (por exemplo #FF0080) ou pelo nome (red, blue, navy), da mesma forma como é feito em BODY.

<FONT> é totalmente dispensável forem utilizadas folhas de estilo, mas, como alguns browsers ainda não suportam folhas de estilo, pode-se usar esse elemento para garantir uma aparência melhor às páginas Web.

```

<li>
<font face="Palatino, serif" size=5 color="#80001F">
Pizza a moda del Capo</font>
</li>

```

A maior parte das ferramentas gráficas que criam páginas HTML usam <FONT> para ter controle sobre a aparência do texto. Alguns ignoram totalmente a estrutura (aumentando a fonte de <P> em vez de usar <H1>). Isto não só prejudica a estrutura do HTML, que também tem outras finalidades além de formatar texto em um browser gráfico, como dificulta a manutenção do site com folhas de estilo. Tudo o que é possível fazer com <FONT> se pode fazer com folhas de estilo e bem melhor. Os browsers *Netscape 4.0*, *Internet Explorer 3.0* e *4.0* suportam as propriedades de folhas de estilo que substituem totalmente a necessidade de se usar <FONT>, que é um elemento depreciado na especificação HTML 4.0.



## 10.7. Conclusão

A finalidade deste módulo foi proporcionar uma introdução à linguagem HTML. Não deixe de consultar os tutoriais e guias de referência sobre HTML para conhecer outros elementos e atributos que foram deixados de fora desta exposição. Antes de usar um determinado recurso, porém, teste nos browsers e plataformas utilizadas pelo público-alvo de seu site. Nem tudo o que é padrão é implementado nos browsers que atualmente dominam o mercado.

# 11. Testes

1. Marque apenas as alternativas verdadeiras:
  - a) Em um arquivo HTML, um bloco de texto cercado por um par de descritores HTML poderá não ter sua aparência alterada quando o arquivo for carregado em um browser.
  - b) Atributos só podem ocorrer no descritor inicial de um elemento HTML.
  - c) Atributos podem ocorrer em qualquer ordem dentro de um descritor HTML.
  - d) Descritores e atributos desconhecidos pelo browser são ignorados.
  - e) Os valores dos atributos devem ser definidos entre aspas, sempre.
  
2. O que acontecerá com o trecho de código HTML abaixo
 

```
<!-- Informação muito importante! -->
```

 quando a página no qual está contido for visualizada em um browser?
  - a) Ele aparecerá em destaque na barra de título do browser.
  - b) Ele aparecerá na página entre duas linhas horizontais.
  - c) Ele será completamente ignorado pelo browser.
  - d) Ele não aparecerá na página mas causará o aparecimento de uma linha em branco no lugar onde deveria aparecer.
  - e) O browser mostrará uma mensagem de erro.
  
3. Quais trechos de código HTML abaixo estão incorretos? Marque um ou mais.
  - a) `<p>Este é um parágrafo <b>importante</b>.</p>`
  - b) `<h1>Título do capítulo</h1 align="center">`
  - c) `<td>Voltar para <a href="casa.html">Casa</a></td>`
  - d) `<p><a href="../index.html"><b>Retornar</a></b></p>`
  - e) `<p>Parágrafo um. <p>Parágrafo dois.`
  - f) `<A HREF="arquivo.gif">Link para imagem</A>`
  - g) `<IMG SRC = "imagem.jpg" ALT = "Figura 1">`
  - h) `<p align=center>Texto alinhado pelo centro</p>`
  - i) `<p>O elemento <table> serve para construir tabelas</p>`
  - j) `<p>Copyright &copy 2000, José Severino da Silva</p>`
  
4. Marque apenas as afirmações falsas:
  - a) Espaços extras, tabulações e quebras de linha que ocorrem no texto são geralmente ignorados em páginas HTML.
  - b) Tudo o que aparece na página deve estar dentro de um bloco delimitado em HTML pelos descritores `<BODY>` e `</BODY>`.

- c) O elemento <META> serve para incluir informações sobre a página que podem ser úteis para facilitar a localização da página por mecanismos de busca.
  - d) Se uma página tiver o descritor <BASE TARGET="nova">, clicar em um de seus vínculos (*links*) poderá causar a abertura de novas janelas.
  - e) O título do documento, expresso entre <TITLE> e </TITLE> aparece no início da página HTML, antes do restante do texto ou imagens.
5. Para criar uma lista simples de itens numerados, qual conjunto de elementos HTML você utilizaria? Escolha uma alternativa.
- a) <P> e <LI>
  - b) <UL> e <LI>
  - c) <OL> e <LI>
  - d) <TD> e <LI>
  - e) <UL>, <OL> e <LI>
6. Quais trechos de código HTML abaixo sempre serão exibidos em itálico em qualquer browser? Marque uma ou mais opções.
- a) <I>Texto grifado</I>
  - b) <EM>Texto grifado</EM>
  - c) <CITE>Texto grifado</CITE>
  - d) <ADDRESS>Texto grifado</ADDRESS>
  - e) Nenhuma das alternativas acima garante que o texto aparecerá em itálico.

Para as três próximas questões a seguir, considere a seguinte estrutura de arquivos (em *itálico*) e diretórios (em **negrito**):

```

/        index.html
|        documentos
|            |        mar.html
|            |        ondas.jpg
|        imagens
|            |        barco.gif

```

7. Qual das alternativas abaixo contém o código HTML para que o arquivo *index.html* mostre a imagem *barco.gif* e contenha um *link* para o arquivo *mar.html*? Marque uma alternativa.
- a)  <a href="mar.html">Mar</a>
  - b)   
<a href="../documentos/mar.html">Mar</a>
  - c)   
<a href="documentos/mar.html"></a>Mar
  - d)   
<a href="documentos/mar.html">Mar</a>
  - e)  <a href="../mar.html">Mar</a>
8. Qual das alternativas abaixo contém o código HTML para que o arquivo *mar.html* mostre respectivamente as imagens *ondas.jpg* e *barco.gif*? Marque uma alternativa.
- a)  e 



- b) `` e ``  
 c) `` e ``  
 d) `` e ``  
 e) `` e ``
9. Suponha que o arquivo `index.html` possua uma seção ancorada por um elemento `<a name="secao2"></a>`. Qual das alternativas abaixo contém o código HTML para que o arquivo `mar.html` contenha um link que aponte para essa âncora do arquivo `index.html`? Marque uma alternativa.
- a) `<a name="/index.html#secao2">Seção 2 do Índice</a>`  
 b) `<a href="index.html#secao2">Seção 2 do Índice</a>`  
 c) `<a href="../index.html">Seção 2 do Índice</a>`  
 d) `<a href="#secao2">Seção 2 do Índice</a>`  
 e) `<a href="/index.html#secao2">Seção 2 do Índice</a>`

10. Qual dos trechos de código HTML abaixo desenhará a seguinte tabela?

a) 

```
<table border="1" width="80">
  <tr> <td colspan="2">1</td> </tr>
  <tr> <td rowspan="2">4</td>
      <td>5</td>
      <td rowspan="2">2</td> </tr>
  <tr> <td colspan="2">3</td> </tr>
</table>
```

1		2
4	5	2
	3	

b) 

```
<table border="1" width="80">
  <tr> <td colspan="2">1</td>
      <td rowspan="2">2</td>
      <td rowspan="2">4</td>
      <td>5</td>
      <td colspan="2">3</td> </tr>
</table>
```

c) 

```
<table border="1" width="80">
  <tr> <td colspan="2">1</td>
      <td rowspan="2">2</td> </tr>
  <tr> <td rowspan="2">4</td>
      <td rowspan="1">5</td> </tr>
  <tr> <td colspan="2">3</td> </tr>
</table>
```

d) 

```
<table border="1" width="80">
  <tr> <td rowspan="2">1</td>
      <td colspan="2">2</td> </tr>
  <tr> <td colspan="2">4</td>
      <td>5</td> </tr>
  <tr> <td rowspan="2">3</td> </tr>
</table>
```

e) Nenhuma das alternativas anteriores