

# Fundamentos de HTML5 e CSS3



# Fundamentos de HTML5 e CSS3

**Maurício Samy Silva**

Novatec

Copyright © 2015 da Novatec Editora Ltda.

Todos os direitos reservados e protegidos pela Lei 9.610 de 19/02/1998.

É proibida a reprodução desta obra, mesmo parcial, por qualquer processo, sem prévia autorização, por escrito, do autor e da Editora.

Editor: Rubens Prates  
Assistente editorial: Priscila Yoshimatsu  
Capa: Carolina Kuwabata  
Revisão gramatical: Mari Kumagai  
Editoração eletrônica: Carolina Kuwabata

ISBN: 978-85-7522-708-4

Histórico de edições impressas:

Abril/2016 Primeira reimpressão

Junho/2015 Primeira edição

Novatec Editora Ltda.  
Rua Luís Antônio dos Santos 110  
02460-000 – São Paulo, SP – Brasil  
Tel.: +55 11 2959-6529  
E-mail: [novatec@novatec.com.br](mailto:novatec@novatec.com.br)  
Site: [www.novatec.com.br](http://www.novatec.com.br)  
Twitter: [twitter.com/novateceditora](https://twitter.com/novateceditora)  
Facebook: [facebook.com/novatec](https://facebook.com/novatec)  
LinkedIn: [linkedin.com/in/novatec](https://linkedin.com/in/novatec)

*Dedico este livro aos professores das escolas de formação de profissionais de desenvolvimento web.*



# Sumário

## [Agradecimentos](#)

## [Isenção de responsabilidade](#)

## [Sobre o autor](#)

## [Introdução](#)

[Para quem foi escrito este livro](#)

[Convenções tipográficas](#)

[Site do livro](#)

## [Capítulo 1 ■ Histórico, ferramentas e terminologia](#)

[1.1 Introdução](#)

[1.2 Breve relato da evolução da HTML](#)

[1.3 Desenvolvimento com HTML5](#)

[1.4 Ambiente de desenvolvimento](#)

[1.4.1 Ambiente Windows](#)

[1.5 Configurando e entendendo o Notepad++](#)

[1.5.1 Visão geral](#)

[1.5.2 Menu Visualizar](#)

[1.5.3 Menu Linguagem](#)

[1.5.4 Menu Configurações](#)

[1.5.5 Inserir ou sobrescrever](#)

[1.5.6 Outras configurações](#)

[1.5.7 Cliente FTP](#)

[1.6 Terminologia](#)

[1.6.1 Documento HTML](#)

[1.6.2 Navegador](#)

[1.6.3 Usuário](#)

[1.6.4 Dispositivo de usuário](#)

[1.6.5 Desenvolvedor web e autor web](#)

[1.6.6 Editor](#)

[1.6.7 Renderização](#)

[1.6.8 Código-fonte](#)

## [Capítulo 2 ■ Marcação HTML](#)

[2.1 Introdução](#)

- [2.2 Tag HTML](#)
- [2.3 Documento HTML mínimo](#)
- [2.4 Primeira página web](#)
- [2.5 Exercício proposto](#)
- [2.6 Modelo HTML](#)
- [2.7 Acessibilidade na web](#)

## **Capítulo 3 ■ Elementos HTML**

- [3.1 Introdução](#)
- [3.2 Elementos da HTML](#)

## **Capítulo 4 ■ Introdução às CSS**

- [4.1 Definições e conceitos CSS](#)
  - [4.1.1 Definição](#)
  - [4.1.2 Finalidade](#)
  - [4.1.3 Regra CSS](#)
- [4.2 Modelo de formatação CSS](#)
  - [4.2.1 Container](#)
  - [4.2.2 Elementos nível de bloco e boxes bloco](#)
  - [4.2.3 Elementos inline e boxes inline](#)
- [4.3 Box Model CSS](#)
  - [4.3.1 Box Model CSS modificado](#)
  - [4.3.2 Propriedades CSS para o Box Model](#)
- [4.4 Categorias de valores CSS](#)
  - [4.4.1 Palavra-chave](#)
  - [4.4.2 Número](#)
  - [4.4.3 Número não negativo](#)
  - [4.4.4 Número com unidade de medida](#)
  - [4.4.5 Número não negativo com unidade de medida](#)
  - [4.4.6 String](#)
  - [4.4.7 Notação funcional](#)
  - [4.4.8 Casos especiais](#)
- [4.5 Cores CSS](#)
- [4.6 Valor CSS](#)
- [4.7 Vinculando folhas de estilo a documentos](#)
  - [4.7.1 Estilos inline](#)
  - [4.7.2 Estilos incorporados](#)
  - [4.7.3 Estilos externos](#)

## **Capítulo 5 ■ DOM e seletores CSS**

- [5.1 DOM](#)
  - [5.1.1 Árvore do DOM](#)
- [5.2 Seletores CSS](#)
  - [5.2.1 Criando uma folha de estilo](#)

[5.2.2 Estilizando o topo do site](#)

## **Capítulo 6 ■ Posicionamento CSS**

[6.1 Introdução](#)

[6.2 Esquemas de posicionamento](#)

[6.2.1 Esquema normal ou posicionamento padrão](#)

[6.2.2 Esquema relativo](#)

[6.2.3 Esquema com float](#)

[6.2.4 Esquema absoluto](#)

[6.2.5 Esquema estático](#)

[6.2.6 Posicionamento fixo](#)

[6.2.7 Posicionamento com z-index](#)

## **Capítulo 7 ■ Construção de layout**

[7.1 Tipos de layout](#)

[7.1.1 Largura fixa](#)

[7.1.2 Líquido](#)

[7.1.3 Elástico](#)

[7.1.4 Híbrido](#)

[7.1.5 Responsivo](#)

[7.2 Centralizando o layout](#)

[7.2.1 Centralizar com uso de margens automáticas](#)

[7.3 Layout de largura fixa](#)

[7.4 Layout elástico](#)

## **Capítulo 8 ■ Estilização**

[8.1 Introdução](#)

[8.2 Bordas arredondadas](#)

[8.3 Sombras](#)

[8.3.1 Sombra em texto](#)

[8.3.2 Sombra em box](#)

[8.4 Opacidade](#)

[8.5 Gradientes](#)

[8.5.1 Gradiente linear](#)

[8.5.2 Gradiente radial](#)

[8.5.3 Gradiente repetido](#)

[8.6 Propriedade background](#)

[8.6.1 Múltiplas imagens de fundo](#)

## **Capítulo 9 ■ Formulários**

[9.1 Introdução](#)

[9.2 Elementos de formulário e seus atributos](#)

## **Apêndice A ■ Elementos da HTML5**

## **Apêndice B ■ Propriedades CSS**

B.1 Propriedades das CSS3

B1.1 Convenções

## **Apêndice C ■ Cores CSS**

## **Referências**

# Agradecimentos

Agradeço a Deus, por ter me dado forças, disposição e motivação para escrever este livro.

Sou grato aos profissionais da Novatec Editora, em particular ao editor, Rubens Prates, que, ao longo de todo o processo de criação, esteve presente guiando-me com suas dicas e informações sobre as particularidades e implicações editoriais próprias à criação de um livro.

Meu maior agradecimento é a você, leitor, por interessar-se em aprender HTML5 e CSS3, e honrar-me com a leitura deste livro.

# Isenção de responsabilidade

Todos os esforços foram feitos para assegurar o fornecimento de informações as mais precisas, completas e exatas possíveis neste livro. Contudo, tais informações são fornecidas “como estão” e sem nenhuma garantia, seja expressa, seja implícita. O autor, a editora, os distribuidores e qualquer entidade envolvida direta ou indiretamente na sua comercialização não assumirão responsabilidade alguma por eventual prejuízo ou dano, direto ou indireto, consequente aos dados contidos neste livro.

# Sobre o autor

Maurício Samy Silva é graduado em Engenharia Civil pelo Instituto Militar de Engenharia (IME). Fundador e ex-diretor-presidente da Planep Engenharia, exerceu o magistério paralelamente à Engenharia e foi, ao longo de 25 anos, professor de Geometria Descritiva e Matemática.

Sua experiência com desenvolvimento de sites iniciou-se em 1999 com o FrontPage, considerada uma ferramenta sensacional na construção de sites. Em 2002, por acaso, teve seu primeiro contato com o site do W3C e, como ficou vivamente impressionado com a proposta desse consórcio, começou a pesquisar e a estudar as Web Standards, tendo como fonte de consulta o material publicado na internet em língua inglesa. Ao contrário do que ocorre nos dias atuais, em que vários desenvolvedores escrevem em blogs pessoais sobre Web Standards, a literatura a respeito do assunto em português era, então, praticamente nula, mas, mesmo assim, quando encontrada, limitava-se ao básico do básico.

Maujor, como é conhecido o autor, mantém o site <http://maujor.com> destinado à divulgação dos Padrões Web e suas tecnologias de desenvolvimento. É um ativo frequentador de fóruns, escreve para vários portais brasileiros voltados a desenvolvedores web. É autor de artigos e de trabalhos relacionados às CSS publicados em diversos sites de desenvolvimento web.

Para conhecer todos os livros de autoria do Maujor, visite o site [www.livrosdomaujor.com.br](http://www.livrosdomaujor.com.br).

# Introdução

Este livro tem o objetivo de fornecer aos iniciantes e alunos voltados para a área de desenvolvimento web os conceitos fundamentais para a criação de sites, de interfaces gráficas e de aplicações para a web com uso da linguagem de marcação HTML5 e das funcionalidades de estilização das CSS3. Trata-se de um livro pioneiro para a área a que se destina, que mostra e exemplifica as funcionalidades dessas tecnologias e pretende fornecer ao estudante uma visão detalhada dos conceitos básicos e fundamentos da marcação HTML e estilização CSS.

Os capítulos foram estruturados de forma didática de modo que estudo possa ser feito de forma sequenciada, isto é, o entendimento do capítulo anterior é pré-requisito para a compreensão do capítulo que se segue.

## Para quem foi escrito este livro

Este livro destina-se a iniciantes e estudantes interessados na criação de sites tanto na área de design quanto na de desenvolvimento e programação, sem necessidade de conhecimento prévio de linguagem HTML e CSS, mas com conhecimentos, pelo menos básicos, de navegação na internet, criação de pastas, gravação de arquivos, download de arquivos.

O objetivo é fornecer informações detalhadas sobre os conceitos básicos do funcionamento da linguagem HTML, estudando seus elementos e atributos; além de estudar as propriedades CSS e sua vinculação à HTML. Explicações teóricas em linguagem corrente e clara, dispensando, sempre que possível, o jargão técnico avançado, são acompanhadas de exemplos práticos explicados passo a passo e complementados por arquivos HTML, disponíveis



online para consulta e para download.

Os iniciantes vão se beneficiar deste livro por principiarem seus estudos em uma fonte que aborda as mais modernas técnicas de escrita do código, ensejando uma mudança no rumo de seu estudo que vai reduzir a curva de aprendizado e acelerar tal processo. Não se intimide com conceitos ou terminologias que lhe sejam completamente desconhecidos nos primeiros capítulos. Com a sequência da leitura, as dúvidas tenderão a desaparecer naturalmente.

## Convenções tipográficas

Com a finalidade de destacar diferentes tipos de informação, adotaram-se algumas convenções tipográficas mostradas a seguir.

### ***Dica***

Texto contendo uma dica sobre o assunto tratado:



No mês de outubro de 1994, Tim Berners-Lee em parceria com a CERN, onde a web foi por ele inventada, criaram o World Wide Web Consortium (W3C), com sede no Laboratório da Ciência da Computação do Massachusetts Institute of Technology (MIT).

### ***Alerta***

Texto contendo um lembrete sobre procedimento extra em relação ao assunto tratado:



Os atributos aqui mostrados são atualmente previstos na especificação e bem suportados nos navegadores atuais, e podemos usá-los sem problemas em fase de produção.

### ***Terminologia***

Texto estabelecendo a adoção de grafia-padrão em todo o livro para termos ou frases com mais de uma terminologia, tradução ou significado:



O termo inglês *browser* é usado no jargão da internet para designar um programa capaz de ler e apresentar ao usuário os conteúdos de um documento web escrito em linguagem de marcação. Browser vem do verbo *to browse*, que significa folhear.

## **Chamada**

Uma chamada para uma seção anterior na qual o assunto em questão foi explicado com detalhes.

Por exemplo: A página foi centralizada adotando-se o método das margens automáticas, conforme mostrado em [7.2.1].

Nesse exemplo, a chamada é para a seção [7.2.1] do capítulo 7.

## **Marcação e scripts**

Nas marcações e scripts que exemplificam a teoria, transcreveram-se somente os trechos que interessam ao assunto tratado. Omitiram-se os trechos que não dizem respeito ou não são relevantes ao entendimento do assunto, para não ocupar espaço desnecessário no livro.

- **Blocos de marcação são marcados com fonte monoespaçada:**

```
<article>
  <h1>Pequenas tarefas</h1>
  <footer>Publicado em <time>05/02/2011</time>.</footer>
  <p>Trocar a lâmpada da sala da sala.</p>
</article>
```

- **Trechos de marcação que merecem destaque são marcados em negrito:**

```
<body class="home">
<div id="tudo">
  <div class="topo">
    <header class="topo_header">
      <hgroup>
        <h1><a href="/">Site do Paulo</a></h1>
        <h2>Site destinado a divulgar o trabalho de Paulo Silva</h2>
```

```

    </hgroup>
  </header>
</div> <!-- /.topo -->

```

Para explicar passo a passo cada linha de um código, este é apresentado com suas linhas numeradas e, a seguir, os comentários são referenciados ao número da linha comentada:

```

* {
  margin: 0;
  padding: 0;
}
body {
  font-size: 18px;
  font-family: arial, sans-serif;
  line-height: 1.4;
}
hgroup > h1 {
  font-size: 36px;
  margin-top: 12px;
  margin-bottom: 12px;
}
hgroup a {
  color: #c30;
  text-decoration: none;
}
h1 ~ h2 {
  font-size: 27px;
  margin-top: 14px;
  margin-bottom: 14px;
  color: #090;
}
...

```

Código comentado:

Linha	Descrição
Linha 1	Seletor universal. Seleciona (ou casa com) todos os elementos da árvore do documento. Esta regra CSS zera as margens e padding iniciais (definidas na folha de estilos do agente de usuário).
Linha 2	Seletor tipo para o elemento body. Esta regra CSS define que as características da fonte padrão do documento.

Linha	Descrição
Linha 3	Seletor filho. Esta regra CSS estiliza o elemento h1 filho do pai hgroup.
Linha 4	Seletor descendente. Esta regra CSS estiliza o elemento a descendente do elemento hgroup.
Linha 5	Seletor irmão adjacente. Esta regra CSS estiliza o elemento h2 irmão adjacente de h1.

## ***Arquivos para download***

Arquivos para download são indicados no texto conforme o exemplo a seguir:

O arquivo *seis-niveis-de-cabecalho.html*, que demonstra este exercício, está disponível para consulta online e download na pasta *capitulo2*.

## ***Destaques em geral***

Palavras ou termos cujo significado deva ser destacado são grafados em itálico. Por exemplo:

A presença do atributo `controls` faz com que o navegador renderize uma barra de controle nativa contendo botões do tipo *play* e *pause* bem como controle de volume.

## ***Variáveis***

Valores variáveis são grafados em itálico.

## **Site do livro**

O site de suporte a este livro está localizado em <http://livrosdomaujor.com.br/html5css3>. No site, incluíram-se as facilidades relacionadas a seguir:

- **Arquivo de códigos** – Os códigos dos exemplos mostrados no livro estão disponíveis tanto para consulta online quanto para download.
- **Errata** – Efetuou-se um exaustivo trabalho de revisão tipográfica. Contudo, a prática mostra que nenhum livro está isento de erros

– e com este não há de ser diferente. Uma página do site dedicada à errata encontra-se disponível para consulta.

- **Feedback** – Incentivam-se vivamente os leitores a emitir opinião sobre qualquer aspecto do livro. Serão de grande valia informações sobre qualquer erro detectado para aperfeiçoar futuras edições e atualizar a errata. Você pode se comunicar com a editora pelo email *novatec@novatec.com.br* ou diretamente com o autor pelo email *maujorcass@maujor.com*.

# CAPÍTULO 1

## Histórico, ferramentas e terminologia

### 1.1 Introdução

HTML é a sigla em inglês para *HyperText Markup Language*, que, em português, significa linguagem para marcação de hipertexto.

Hipertexto é todo texto inserido em um documento para a web e que tem como principal característica a possibilidade de se interligar a outros documentos da web com uso dos nossos já conhecidos links, presentes nas páginas dos sites que estamos acostumados a visitar. Então, todo o conteúdo textual que você vê em uma página de um site é um hipertexto, assim como imagens, vídeos, gráficos, sons e conteúdos não textuais em geral são chamados de hipermídia.

Quando a HTML foi inventada, os conteúdos eram essencialmente hipertextos, com a hipermídia surgindo posteriormente. Assim, hoje, a HTML é uma linguagem para marcação de conteúdos web em geral.

Desde a invenção da web, em 1992, por Tim Berners-Lee, a HTML evoluiu por oito versões que são:

- HTML
- HTML +
- HTML 2.0
- HTML 3.0
- HTML 3.2

- HTML 4.0
- HTML 4.01
- HTML5



Tecnicamente, o W3C considera oficialmente somente as versões HTML 2.0, HTML 3.2, HTML 4.0, HTML 4.01 e HTML5. As versões HTML e HTML+ são anteriores à criação do W3C e a versão HTML 3.0 não chegou a ser lançada oficialmente, transformando-se na versão HTML 3.2.

O World Wide Web Consortium, ou W3C, é um consórcio internacional com quase 400 membros, entre eles, empresas, órgãos governamentais e organizações independentes que têm por objetivo definir padrões destinados à criação de conteúdos para a web. O termo “Padrões Web” é usado em linguagem de desenvolvimento web para definir técnicas e práticas de criação de conteúdos web em acordo com as recomendações (ou especificações) do W3C.

O W3C tem sua sede principal distribuída em três lugares distintos: nos Laboratórios de Ciência da Computação do MIT, em Massachusetts nos Estados Unidos; no Instituto Nacional de Pesquisas de Informática e Automação, na França; e na Universidade de Keiko, no Japão; além de escritórios espalhados em várias cidades do mundo, inclusive em São Paulo no Brasil.

O termo inglês *browser* é usado no jargão da internet para designar um programa capaz de ler e apresentar ao usuário os conteúdos de um documento web escrito em linguagem de marcação. Foi traduzido para o português como *navegador*, gerando a conhecida expressão “navegar na internet”. São exemplos de navegadores o Internet Explorer, o Firefox, o Opera, o Chrome e o Safari, entre outros. Neste livro, adotaremos o termo em sua forma traduzida: *navegador*.

A língua que nós, brasileiros, entendemos e usamos para nos comunicar é o português. A linguagem que nós, criadores de conteúdos para a web, entendemos e usamos para nos comunicar

com o navegador é a HTML.

Tal como o português, a HTML tem seus termos ou palavras, suas regras de sintaxe e de formatação que devem ser seguidas para o perfeito entendimento pelo navegador. Uma vez que você saiba se comunicar com o navegador, ele saberá apresentar ao usuário a página web.

Assim, o conhecimento da HTML é o primeiro requisito para criar uma página web.

## **1.2 Breve relato da evolução da HTML**

Em outubro de 1993, Dave Raggett publicou a versão final da HTML+.

A HTML+ começa com a seguinte afirmação:

Documentos marcados com HTML+ são constituídos de títulos, parágrafos, listas, tabelas e figuras.

E continua estabelecendo:

Ao contrário da maioria das tecnologias destinadas à criação de documentos, a HTML+ não se destina a determinar a aparência; assim, nomes e tamanhos de fontes, margens, tabulações, espaçamentos entre os elementos não são funções da linguagem.

Convém salientar com muita ênfase que, desde sua criação, os idealizadores da HTML tiveram a preocupação de retirar da linguagem de marcação qualquer atribuição ou função de apresentação, ou seja, HTML destina-se exclusivamente a estruturar documentos. É nessa destinação que se fundamentam os princípios básicos do desenvolvimento seguindo os Padrões Web.

A função de apresentação dos conteúdos web criados com HTML é das CSS (Cascade Style Sheet) ou Folhas de Estilo em Cascata, conforme veremos adiante.

Em maio de 2007, o W3C tornou pública sua decisão de retomar os estudos para o desenvolvimento da HTML5, tomando como base o trabalho que já vinha sendo desenvolvido pelo WHATWG.



WHATWG é a sigla em inglês para *Web Hypertext Application Technology Working Group*, que, em português, significa Grupo de trabalho para tecnologias de hipertexto em aplicações para web. O WHATWG foi criado em 2004 por desenvolvedores da Apple, da Fundação Mozilla e do navegador Opera, e atualmente desenvolve a HTML5 em conjunto com o W3C e ambos mantêm em seus sites versões das especificações.

Em 28 de outubro de 2014, a especificação para a HTML5 atingiu o status de Recomendação do W3C. As funcionalidades da linguagem estudadas neste livro *se baseiam naquele documento* hospedado no site do W3C em <http://kwz.me/wi>.

O processo de criação de uma especificação do W3C passa pelos seguintes status (ou estágios):

- **Working Draft (WD)** – Rascunho de Trabalho – é um documento público proposto pelo W3C para ser revisto e criticado pela comunidade, por membros do W3C e por organizações técnicas em geral.
- **Candidate Recommendation (CR)** – Candidata à Recomendação – é um documento publicado pelo W3C resultante das revisões do Rascunho de Trabalho, cuja finalidade é a de servir de base para implementação, pelos fabricantes de software, em caráter experimental, das funcionalidades ali previstas.
- **Proposed Recommendation (PR)** – Proposta de Recomendação – é um documento maduro que foi amplamente revisto nos seus aspectos técnicos e de implementação, e foi enviado ao Comitê Consultivo do W3C para endosso.
- **W3C Recommendation (REC)** – Recomendação do W3C – é uma especificação ou conjunto de diretrizes finais do W3C que recebeu o endosso do Diretor e dos membros do W3C. Recomendações do W3C são semelhantes às normas publicadas por outras organizações, como o INMETRO, por

exemplo.

## 1.3 Desenvolvimento com HTML5

A especificação para a HTML5 está estruturada em 12 seções, a saber:

1. **Introdução** – Descreve público-alvo, histórico, escopo e lista notas gerais sobre o projeto da especificação. Faz uma breve introdução à HTML e apresenta regras de conformidade para autores sugerindo leituras complementares.
2. **Infraestrutura comum** – Define terminologia, regras de conformidade, classes, algoritmos, microssintaxes, partes comuns das especificações e namespaces.
3. **Semântica, estrutura e APIs para documentos HTML** – Definem as funcionalidades do DOM HTML e dos elementos HTML em geral.
4. **Elementos HTML** – Explicam o significado de cada um dos elementos HTML. São estabelecidas regras de uso dos elementos na marcação, bem como diretrizes de manipulação deles pelos agentes de usuário.
5. **Carregamento de páginas web** – Documentos HTML não aparecem do nada. Essa seção define as muitas funcionalidades relacionadas ao tratamento de páginas web pelos diferentes dispositivos.
6. **APIs para aplicações web** – Descrevem as funcionalidades básicas para desenvolvimento de scripts em aplicações HTML.
7. **Interação com o usuário** – Descreve os diferentes mecanismos de interação do usuário com um documento HTML.
8. **Sintaxe HTML** – Detalha a escrita, parseamento e serialização dos documentos em conformidade com a HTML.
9. **Sintaxe XHTML** – Detalha escrita, parseamento e serialização dos documentos em conformidade com a XHTML.

- 10. Renderização** – Define o modelo de renderização das diversas funcionalidades CSS e elementos da marcação.
- 11. Funcionalidades obsoletas** – Faz considerações sobre as funcionalidades obsoletas para a HTML5.
- 12. Considerações IANA** – Descreve a sintaxe da especificação relacionada às normas da IANA (Internet Assigned Numbers Authority, órgão responsável pela coordenação geral de protocolos para a Internet, nomes de domínio – DNS – e endereço IP).

## **1.4 Ambiente de desenvolvimento**

Já sabemos que sites são criados com uso de marcação HTML. Vejamos a seguir as ferramentas usadas para escrever marcação HTML. Tais ferramentas são conhecidas como editores HTML. Podemos escrever marcação HTML usando qualquer editor de texto, como o Word, disponível em ambiente Windows, por exemplo. Contudo é mais conveniente e apropriado usar editores de texto próprios para escrever marcação HTML e o Word definitivamente não é um editor apropriado para escrever marcação HTML.

Para acompanhar os exemplos e exercícios propostos neste livro, você vai precisar de um editor de texto simples, um programa para visualizar o resultado da marcação HTML que você escreveu e um editor de imagens simples. Tanto o editor de texto quanto o navegador estão instalados em seu computador por padrão. Será necessário um editor para tratamento de imagens caso você queira criar ou tratar imagens para inserir na marcação HTML, mas inicialmente, para o aprendizado e a prática, o editor de imagens é dispensável, e você poderá usar imagens existentes sem necessidade de editá-las.

Caso queira mostrar para o mundo os documentos web que você criou, vai precisar de uma ferramenta de publicação na internet. Ferramentas de publicação na internet são chamadas de Clientes FTP.

FTP é a sigla em inglês para File Transfer Protocol, que significa protocolo de transferência de arquivo. Como o nome sugere, trata-se de um mecanismo capaz de enviar arquivos de um computador para outro. Por exemplo: o servidor no qual ficará hospedado o site. Mostraremos essa ferramenta no item [1.5.7] adiante neste livro.

## **1.4.1 Ambiente Windows**

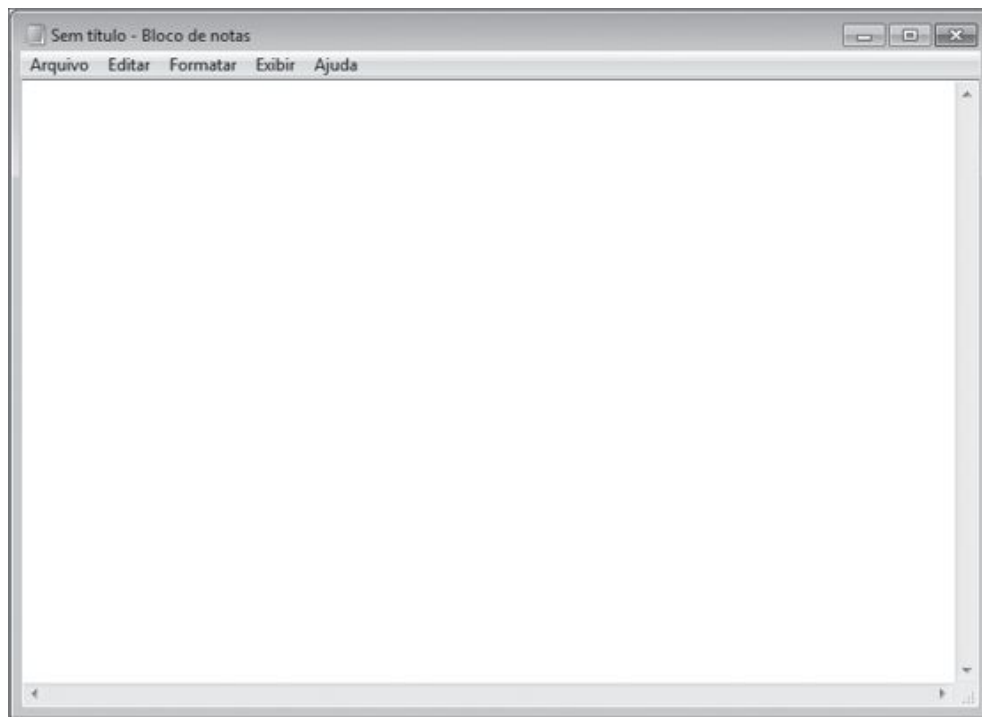
Veremos, a seguir, as ferramentas de desenvolvimento HTML nativas do ambiente Windows. Os exemplos mostrados baseiam-se no sistema operacional Windows em português.

### **1.4.1.1 Editores HTML**

#### ***Bloco de Notas***

Em ambiente Windows, o editor HTML padrão é o Bloco de Notas, mostrado na figura 1.1, e que oferece funcionalidades mínimas de edição, mas suficientes para escrever marcação HTML. Você acessa o editor seguindo o caminho, no menu do Windows, conforme mostrado a seguir:

**Iniciar > Todos os programas > Acessórios > Bloco de Notas**



*Figura 1.1 – Bloco de Notas.*

Você pode usar o Bloco de Notas para escrever suas marcações HTML e acompanhar os exemplos mostrados neste livro. E, neste caso, é uma boa ideia colocar o ícone do editor na barra de ferramentas de seu Windows, a fim de criar um atalho de acesso rápido ao editor. Na figura 1.2, mostramos o ícone-padrão do Bloco de Notas.



*Figura 1.2 – Ícone do Bloco de Notas.*

Você cria um atalho de acesso rápido ao Bloco de Notas como descrito a seguir:

1. Vá ao menu: **Iniciar > Todos os programas > Acessórios.**
2. Na interface que se apresenta, clique com o botão direito do mouse o ícone do Bloco de Notas.

3. Após clicar, no menu de contexto que se abre, escolha: Fixar na barra de tarefas.

É isso. Você está com o ícone de acesso rápido ao Bloco de Notas, fixo na barra de tarefas e pronto para uso.

### ***Notepad++***

Trata-se de um editor HTML gratuito criado para ambiente Windows e disponível para download e instalação na sua máquina. Esse é um editor com várias funcionalidades de edição e muito mais poderoso e completo do que o Bloco de Notas.

Caso você ainda não tenha um editor de texto preferido, aconselhamos a instalar na sua máquina o Notepad++, pois será esse o editor que usaremos para ilustrar os exemplos mostrados neste livro.

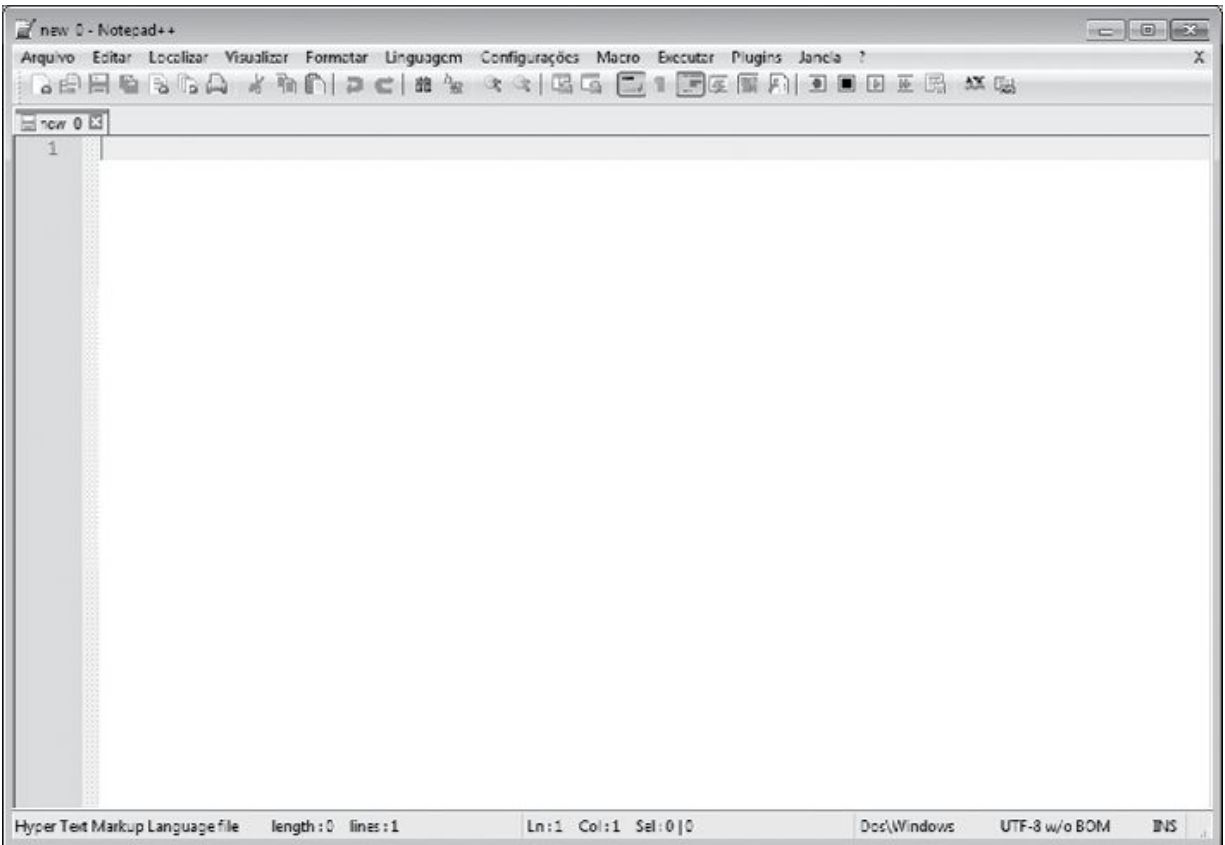
Entre no site do Notepad++ em <http://www.notepad-plus-plus.org/>, clique o link **Download** no menu de navegação do site, faça o download do editor e instale-o em sua máquina. O processo de instalação é bem fácil, bastando seguir as instruções que aparecem na tela após ter-se clicado o arquivo executável obtido com o download.

Terminado o download, na última tela que aparece, destinada a concluir o download, deixe marcada a opção de pedir que o programa seja aberto. Nessa opção, ele vai abrir e mostrar um arquivo de texto em inglês, denominado *change.log*. Feche o arquivo e estamos prontos para conhecer, caso você ainda não conheça, o editor.

Se o editor não estiver aberto, você o acessa seguindo o caminho, no menu do Windows, conforme mostrado a seguir:

**Iniciar > Todos os programas > Notepad++ > clique o ícone do editor**

Observe na figura 1.3 a interface gráfica padrão do editor.



*Figura 1.3 – Notepad++.*

É uma boa ideia colocar o ícone do editor na barra de ferramentas de seu Windows, a fim de criar um atalho de acesso rápido ao editor. Na figura 1.4, mostramos o ícone-padrão do Notepad++.



*Figura 1.4 – Ícone do Notepad.*

Você cria um atalho de acesso rápido ao Notepad++ como descrito a seguir:

1. Vá ao menu: **Iniciar > Todos os programas > Notepad++ > clique**

2. Na interface que se apresenta, clique com o botão direito do mouse o ícone do Notepad++.
3. Após clicar, no menu de contexto que se abre, escolha: Fixar na barra de tarefas.

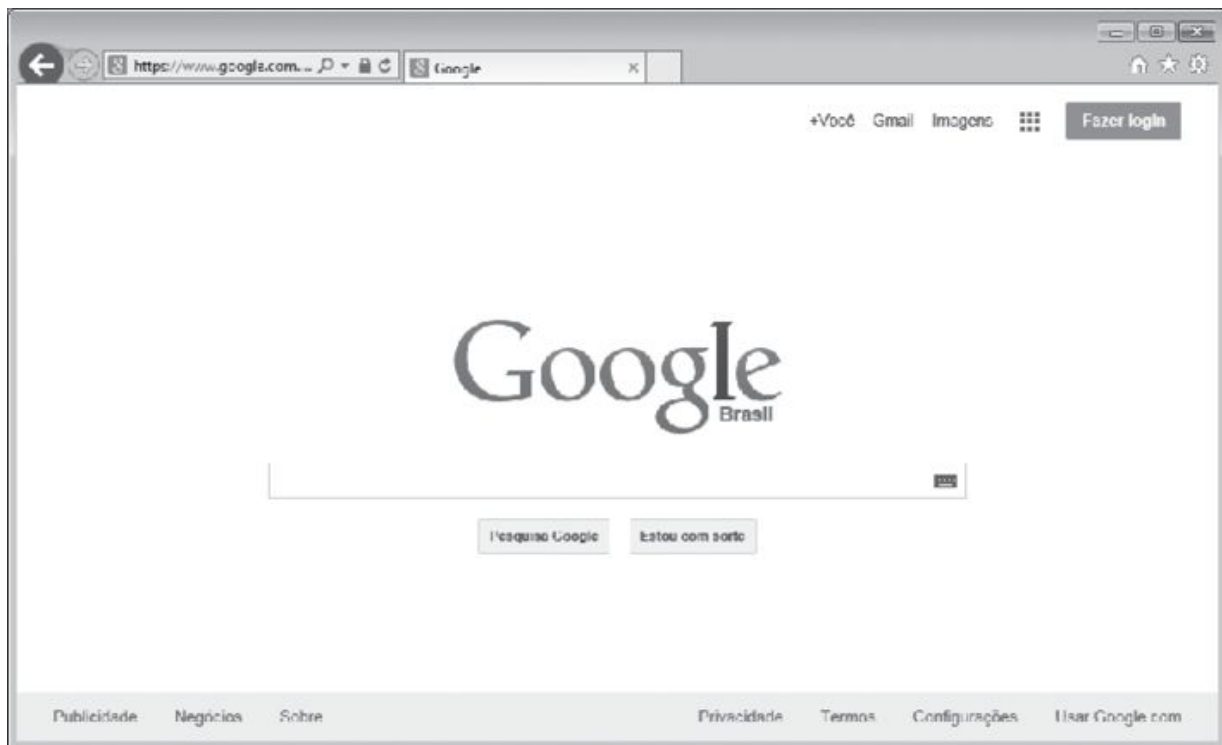
É isso. Você está com o ícone de acesso rápido ao Notepad++, fixo na barra de tarefas e pronto para uso.

#### 1.4.1.2 Navegadores

##### *Internet Explorer*

O navegador padrão do ambiente Windows é o Internet Explorer, muito provavelmente já seu conhecido, cuja interface é mostrada na figura 1.5. Vale notar que, tal como fizemos para o editor de texto, é uma boa ideia criar um atalho de acesso rápido ao navegador na barra de ferramentas. Proceda de maneira idêntica à descrita anteriormente, acionando o menu:

**Iniciar > Todos os programas > Internet Explorer**



*Figura 1.5 – Internet Explorer 11.*



## ***Chrome, Firefox e Opera***

Esses navegadores estão disponíveis na Internet para download não só para funcionamento em ambiente Windows, como também para Macintosh, Linux e outros sistemas operacionais.

Para acompanhar os exemplos e exercícios constantes deste livro, você poderá optar por qualquer um dos quatro navegadores citados anteriormente e mais o Safari em ambiente Macintosh, contudo é altamente recomendável que você instale todos os navegadores na sua máquina, com a finalidade de testar suas criações web em todos eles.

Para mostrar os exemplos constantes deste livro, vamos adotar o navegador Chrome e aconselhamos o leitor a ter, pelo menos, esse navegador instalado na sua máquina.

Os links para download e instalação dos navegadores citados são mostrados a seguir:

- Chrome – <http://kwz.me/Dc>
- Firefox – <http://kwz.me/Dy>
- Opera – <http://kwz.me/DH>
- Safari – <http://kwz.me/DE>
- Internet Explorer – <http://kwz.me/DK>

Para iniciar os estudos e desenvolver nossas primeiras páginas web, um editor HTML e um navegador são tudo o que precisamos por enquanto. Assim, para prosseguir os estudos, consideramos que você já instalou o Notepad++ e, pelo menos, a última versão do navegador Chrome na sua máquina.

Caso você já tenha prática com outros editores HTML e prefira outro navegador, permaneça com eles, mas advertimos, mais uma vez, que os exemplos e exercícios deste livro se baseiam no Notepad++ e no Chrome em ambiente Windows.

Para ambiente Macintosh, Linux ou outros, use o editor HTML equivalente ao Notepad++ ou qualquer editor de sua preferência.

## 1.5 Configurando e entendendo o Notepad++

O Notepad++ é um editor de desenvolvimento web que, além de servir para escrever marcação HTML, destina-se a servir para escrever outros tipos de marcação e também códigos em diversas linguagens de programação e criação de scripts, tais como PHP, ASP e JavaScript. Apresenta inúmeras funcionalidades e entre elas a possibilidade de se configurar o editor para que alguns aspectos de sua aparência, tais como cores e tamanhos de textos, entre outros, possam ser escolhidos de acordo com a preferência e gosto de cada um.

Mostraremos a seguir os comandos básicos disponíveis no Notepad++ com a finalidade de proporcionar ao leitor um primeiro contato com o editor. De posse dos conhecimentos aqui mostrados, o leitor ficará em condições de explorar e estudar as demais funcionalidades do editor, que são muitas.

### 1.5.1 Visão geral

Abra o Notepad++ e identifique as áreas, menus e botões mostrados na figura 1.6.

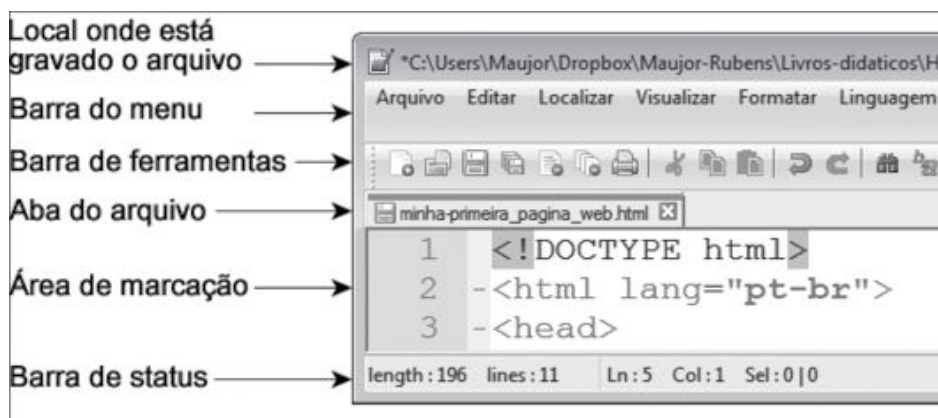


Figura 1.6 – Visão geral Notepad++.

### 1.5.2 Menu Visualizar

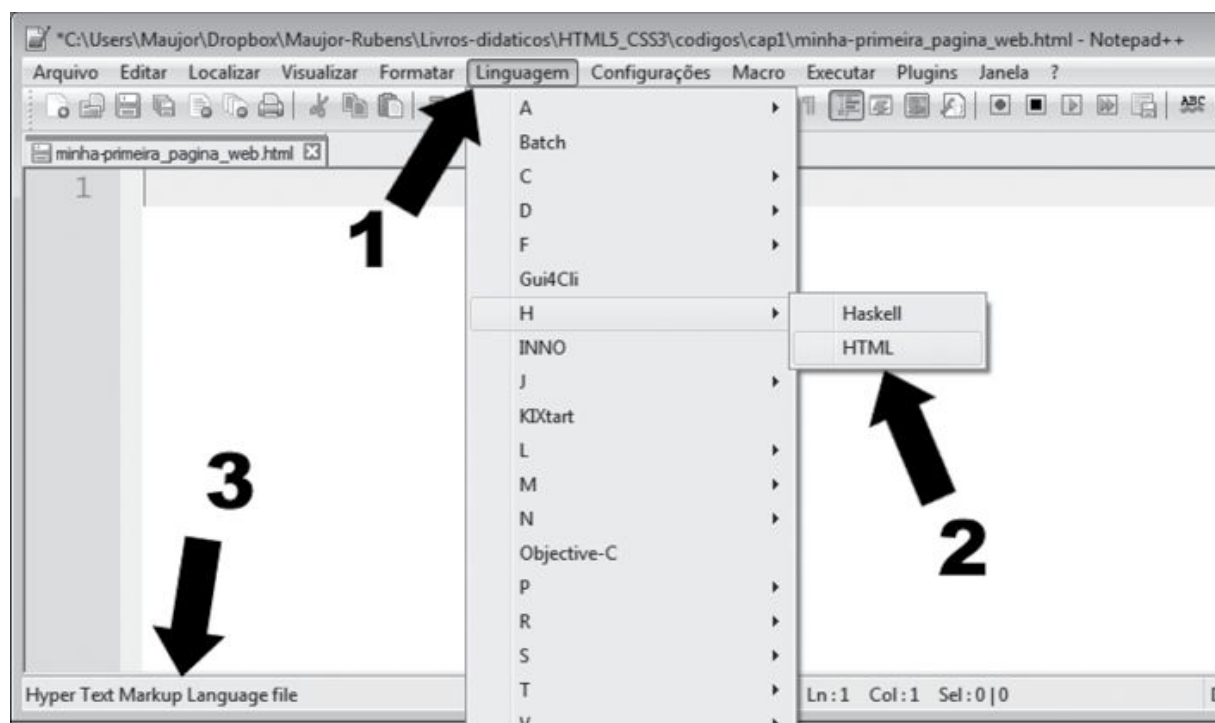
Coloque o cursor do mouse na linha 1 e digite um texto qualquer

bem longo. Observe que, quando o texto chega ao fim da linha, ele continua para a direita e aparece uma barra de rolagem horizontal na janela do editor. Isso não é conveniente para o autor, pois conteúdos da marcação criada poderão ficar fora de vista. Felizmente é possível configurar a quebra de linha evitando a barra de rolagem horizontal.

Na barra do menu, clique no menu **Visualizar**, para abrir um submenu. No submenu, escolha a opção **Quebrar linhas automaticamente**. Se você digitou o texto longo conforme descrito anteriormente, vai notar que imediatamente haverá a quebra da linha transformando-a em duas ou mais linhas, conforme o tamanho do texto digitado.

### 1.5.3 Menu Linguagem

Na barra do menu, clique no menu **Linguagem**, conforme mostrado no número 1 na figura 1.7, para abrir um painel destinado a escolher a linguagem de desenvolvimento a se escrever com o editor. A maioria das linguagens está agrupada pela letra inicial que designa a linguagem. Vá para a letra H e escolha HTML, conforme mostrado no número 2 na figura 1.7. Notar que aparece na barra de status o nome da linguagem escolhida, conforme mostrado no número 3 da figura 1.7. Pronto, seu Notepad está configurado para receber marcação HTML.



*Figura 1.7 – Menu Linguagem.*

A partir de agora, toda vez que você abrir o Notepad++ a linguagem padrão será a HTML. Você poderá alterar a linguagem a qualquer momento mudando o padrão. Dê uma olhada na letra C e verifique que ali está relacionada a linguagem CSS que também usaremos neste livro. Quando for escrever CSS, não se esqueça de alterar a linguagem de HTML para CSS, assim que abrir o Notepad++.

## 1.5.4 Menu Configurações

Na barra do menu, clique no menu Configurações, conforme mostrado no número 1 na figura 1.8, para abrir um painel destinado a escolher vários itens de configuração, conforme mostrado no número 2 na figura 1.8. Clique o primeiro item das configurações, denominado Preferências, para abrir o respectivo painel, conforme mostrado no número 3 da figura 1.8. Nesse painel existem vários itens para configurar. Vamos configurar apenas aqueles essenciais ao funcionamento do editor para escrever marcação HTML e CSS, deixando os demais com as configurações padrão existentes.

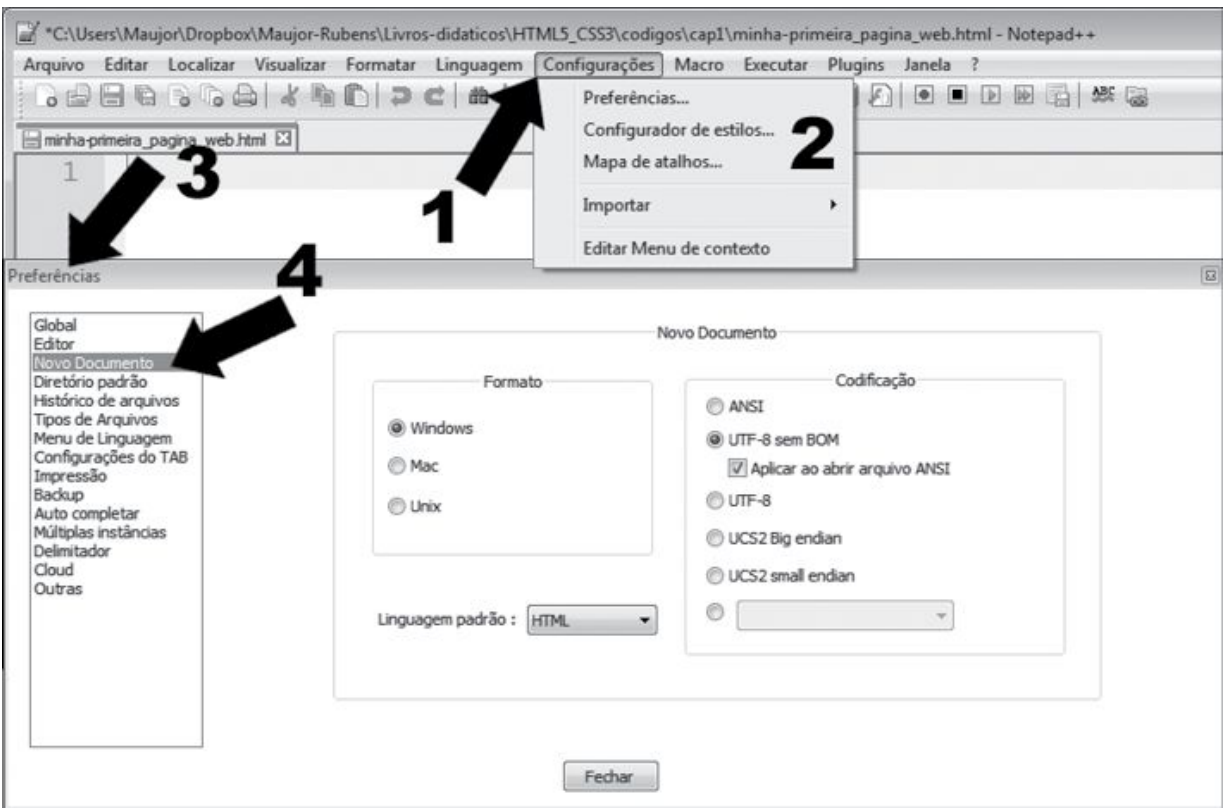


Figura 1.8 – Menu Configurações.

#### 1.5.4.1 Novo Documento

Clique o item Novo Documento, conforme mostrado no número 4 na figura 1.8, e certifique-se de que o formato *Windows* e a codificação *UTF-8* sem BOM estejam escolhidos nos botões radio e, ainda, que a linguagem *HTML* seja a escolhida no menu select existente.

Notar que aparece na barra de status a codificação de caracteres escolhida.

Leia a seguir algumas considerações sobre codificação de caracteres na web. O texto que segue não pretende ser uma explicação técnica e muito menos abordar detalhes sobre codificação de caracteres, mas tem a finalidade de fornecer ao leitor uma explicação didática do assunto, em linguagem simples, alertando-o para a importância e relevância da codificação de caracteres para criação de conteúdos web.

Diferentes idiomas usam diferentes caracteres para escrever seus

textos. Em inglês não existem letras (caracteres) acentuadas tal como em português; e em japonês, chinês e árabe, por exemplo, a escrita de textos usa caracteres completamente diferentes das letras que se usa em português ou inglês. Assim, usando um teclado para escrever textos em inglês, você não consegue digitar letras acentuadas e muito menos escrever em japonês.

Sendo a web um meio eletrônico universal, é perfeitamente possível que um internauta do Brasil, ou de qualquer outro lugar do planeta, acesse sites criados em qualquer lugar do mundo e consequentemente com seus textos formados com diferentes tipos de caractere. Nestes casos é necessário que o navegador usado seja capaz de mostrar ao internauta os caracteres com sua forma correta. Observe a figura 1.9.

Nessa figura, mostramos um trecho de uma página de um site que foi desenvolvida com uso de codificação de caracteres equivocada. O navegador não reconheceu os caracteres acentuados do texto e mostrou um sinal de interrogação no lugar deles. Conforme o tipo de erro cometido na codificação, o caractere não reconhecido será substituído por um determinado tipo de sinal, que no caso da figura foi o ponto de interrogação. Você certamente não vai querer que seu site apresente textos com caracteres de erro, tal como mostrado na figura 1.9, portanto fique atento à codificação de caracteres de suas páginas.

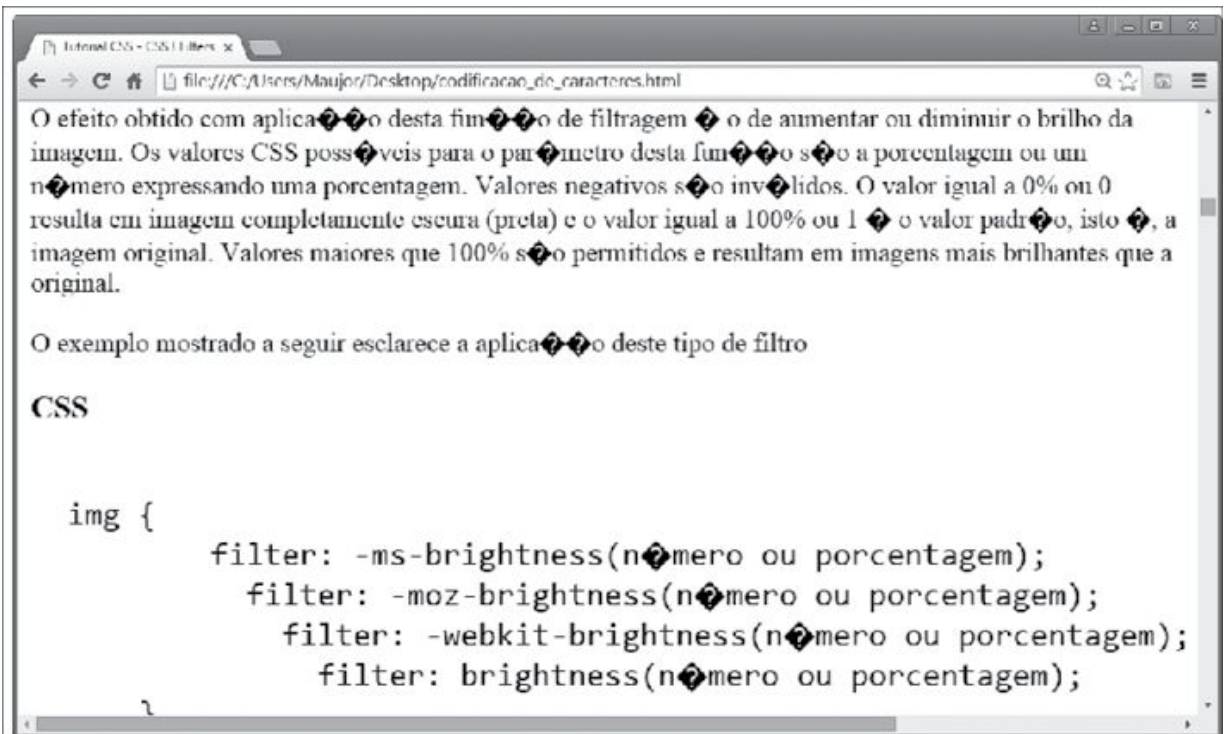


Figura 1.9 – Codificação de caracteres falha.

Para mais informações sobre codificação de caracteres para a web, consulte <http://kwz.me/DS>.

#### 1.5.4.2 Autocompletar

Observe no número 4 na figura 1.8, mostrada anteriormente a opção **Autocompletar**. Clique nessa opção e, na janela que se abre, marque as opções de **Autoinsérer**, conforme mostrado no número 1 da figura 1.10.

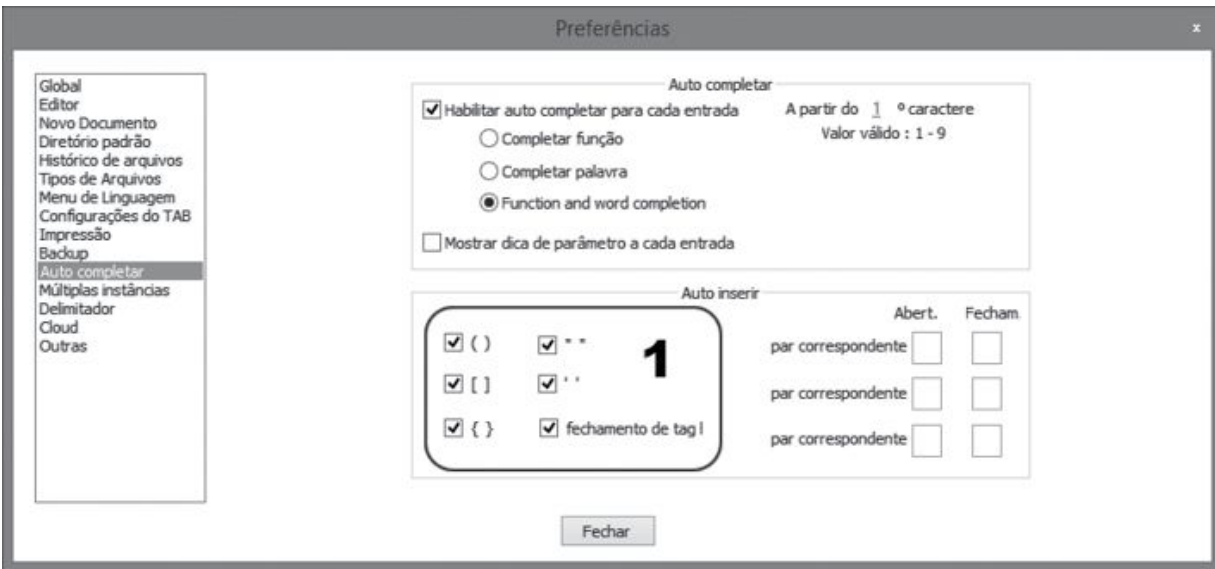


Figura 1.10 – Autocompletar.

### 1.5.4.3 Configurador de estilos

Clique o item **Configurador de estilos**, conforme mostrado no número 2 na figura 1.8, exibida anteriormente. Abre-se uma janela de configurações conforme mostrado na figura 1.11.

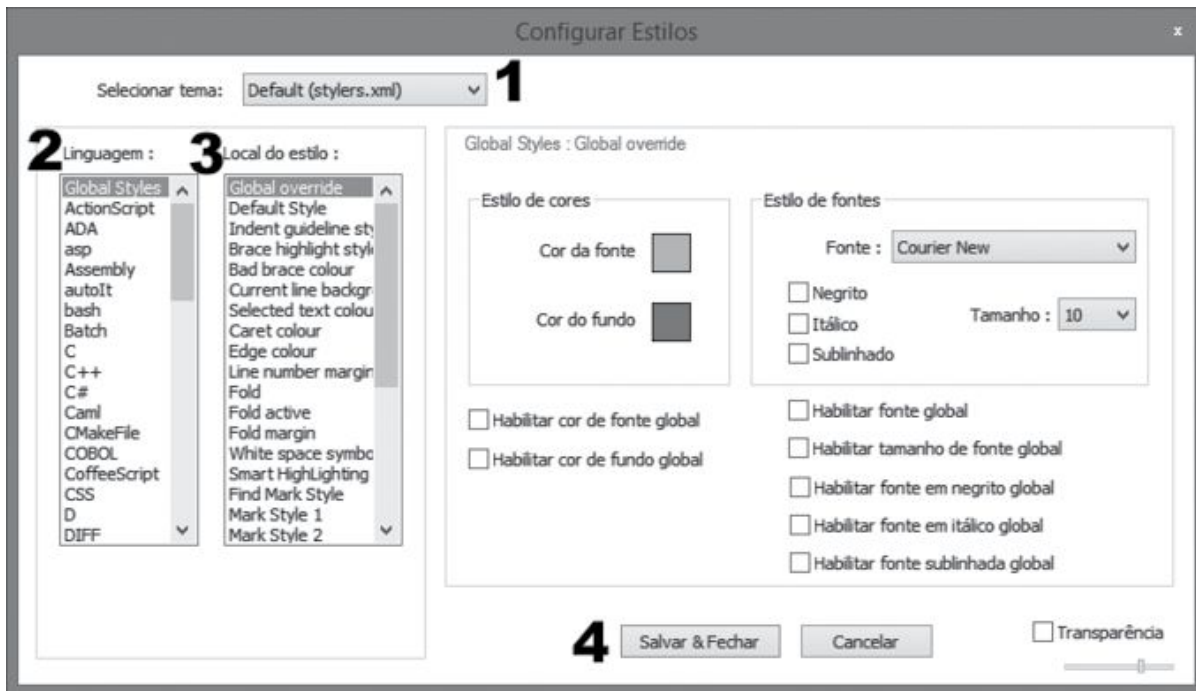


Figura 1.11 – Configurador de estilos.

A janela de configurações abre-se sobre o editor e você poderá



movimentá-la. Conforme mostrado no número 1 na figura 1.11, é possível configurar o tema do editor. O tema padrão denomina-se Default e é com fundo branco e letras pretas. Abra o menu de seleção do tema e faça algumas escolhas para ver a troca de temas. Escolha aquele que mais lhe agrada. Neste livro seguiremos com o tema padrão.

Para escolher um tipo e tamanho de fonte para a marcação, na caixa marcada com o número 3 na figura 1.11, marque a segunda opção **Default Style**. Muda-se o painel à direita e ali você poderá escolher tipo e tamanho da fonte.

### 1.5.5 Inserir ou sobrescrever

No canto inferior direito do editor há um campo denominado **INS**. Esse campo é um botão que, quando clicado, passa a se chamar **OVR**. São duas abreviaturas de palavras em inglês: INS de insert (inserir) e OVR de overwrite (sobrescrever).

Para visualizar o efeito de cada uma das opções do botão, faça o seguinte: digite algumas palavras na linha 1 e depois coloque o cursor entre duas palavras. Com o cursor ali posicionado, digite outra palavra. Ao digitar com o botão em **INS**, a palavra à direita vai sendo empurrada, e, ao digitar com o botão em **OVR**, a palavra à direita vai sendo sobrescrita.

### 1.5.6 Outras configurações

Não é do escopo deste livro apresentar um guia completo de uso e configuração do Notepad++, pois, conforme você já deve ter observado, são centenas de opções e configurações e para tal teríamos de escrever outro livro. Apresentamos aquelas mais básicas para iniciar o desenvolvimento, e, com o uso do editor, você vai descobrir por si mesmo todas as suas possibilidades. Além disso, há ainda o menu de Ajuda do próprio Notepad++.

## 1.5.7 Cliente FTP

Conforme já dissemos anteriormente, cliente FTP é um software destinado a fazer a transferência dos arquivos de um site da sua máquina para um servidor web com a finalidade de mostrar seu site para o mundo.

Existem muitos clientes FTP gratuitos que podem ser obtidos por download e que funcionam isoladamente ou mesmo como complemento de um navegador. Editores de texto mais completos também são fornecidos com um cliente FTP integrado. A boa notícia é que o Notepad++ já tem seu cliente FTP e é ele que usaremos neste livro.

Para acessar o cliente FTP do Notepad++, abra o menu **Plugins** e siga o caminho indicado a seguir:

**NppFTP > Show NppFTP Window**

Com essa ação, você abrirá o painel do cliente FTP do editor. Se você já conhece o funcionamento dessa ferramenta, não terá dificuldade em descobrir por si como ela funciona, se não, aguarde até chegarmos ao capítulo que trata da publicação de sites e lá mostraremos como usar a ferramenta.

## 1.6 Terminologia

É importante conhecer o exato significado de alguns termos de uso geral que aparecem, com frequência, no vocabulário da marcação HTML e, em particular, nas explicações constantes deste livro. A correta interpretação dos conceitos aqui relatados muitas vezes depende do conhecimento, por parte do leitor, do exato significado de um ou mais termos empregados nas conceituações. Para os iniciantes, em particular, recomendo uma acurada leitura das definições que se seguem com o objetivo de evitar que o desconhecimento ou noção errônea do significado de um termo acarrete prejuízo para o aprendizado.

Para os não iniciantes, sugiro uma leitura atenta das definições

apresentadas, pois alguns termos são amplamente empregados com significado errôneo e outros nem sequer existem nas especificações para os Padrões Web. Faça essa leitura a título de revisão de seus conceitos.

### **1.6.1 Documento HTML**

Documento HTML, documento web, página web, ou simplesmente documento, é um arquivo completo contendo toda a marcação necessária para ser lida e interpretada por um dispositivo de usuário.

Dispositivo de usuário é todo software capaz de mostrar para o usuário, de forma apropriada, o conteúdo de um documento HTML, seja transformando a marcação HTML em uma página de um site tal como a vemos na internet, com suas cores, imagens, disposição gráfica etc., seja lendo o que está escrito no conteúdo de um documento HTML para um usuário cego que navega na internet com um leitor de tela, seja inspecionando conteúdos para fornecer informações para mecanismos de busca como o Google, por exemplo.

Usa-se o verbo renderizar para designar a ação de transformação de uma marcação HTML em algo legível para o usuário, seja ele um humano, um software ou uma máquina.

### **1.6.2 Navegador**

Também conhecido como browser (em inglês), navegador é um programa destinado a visualizar documentos desenvolvidos com linguagem de marcação, ou, em sentido mais restrito, sites na internet. Diz-se que um navegador destina-se a renderizar um documento web.

Os navegadores mais usados são o Internet Explorer, o Chrome, o Firefox, o Opera e o Safari.

### **1.6.3 Usuário**

Usuário é toda pessoa, software ou máquina capaz de entender a marcação HTML ou, em sentido mais amplo, tudo que seja capaz de usar a internet ou navegar nela. Internautas, leitores de tela, robôs de busca são usuários.

### **1.6.4 Dispositivo de usuário**

Denomina-se dispositivo de usuário o software, programa, hardware ou tudo que possa ser utilizado por um usuário para ler ou entender um documento desenvolvido com linguagem de marcação, ou, em sentido mais amplo, interagir com a internet.

São exemplos de dispositivo de usuário os navegadores, as impressoras, os leitores de tela – programas especiais que são capazes de ler em voz alta, como um humano, a marcação HTML, de maneira que pessoas com restrições visuais e cegas tenham acesso aos conteúdos dos documentos –, os programas conhecidos como robôs de busca capazes de ler as informações gerais sobre um documento e, assim, catalogar seus conteúdos, como os robôs do Google, que fornecem informações sobre os sites quando o usuário faz uma busca –, entre outros.

### **1.6.5 Desenvolvedor web e autor web**

Ou simplesmente desenvolvedor e autor é todo aquele que desenvolve ou cria documentos para a web escrevendo marcação e/ou programação para a web em geral.

### **1.6.6 Editor**

Editor é um software usado pelo desenvolvedor para escrever a marcação ou a programação de um documento para a web.

Existem editores para desenvolver HTML, para PHP, para CSS e para outras tecnologias.

## 1.6.7 Renderização

Renderizar: diz-se que um navegador (ou um dispositivo de usuário qualquer) renderiza um documento web quando transforma a marcação HTML do documento em algo capaz de ser lido e entendido pelo usuário.

Quando abrimos uma página de um site, podemos vê-la na tela do computador porque o navegador renderizou a página para nós.

## 1.6.8 Código-fonte

Navegadores, quando renderizam um documento, oferecem uma funcionalidade que permite visualizar a marcação HTML criada pelo autor e que foi utilizada para desenvolver a página. Nesse contexto, emprega-se o termo código-fonte como sinônimo da marcação HTML da página.

Para ver o código-fonte (ou a marcação) de uma página qualquer da internet, o navegador fornece um menu de acesso.

Para comprovar como isso funciona, vamos abrir uma página web de um site qualquer na internet e visualizar seu código-fonte.

Com a página aberta, dê um clique com o botão direito do mouse em um lugar vazio dela e, no menu de contexto que se abre, escolha **Exibir código fonte da página**. Dependendo do navegador, o texto no menu de contexto pode variar, mas será algo relacionado às palavras “código-fonte” ou algo que as contenha.

# CAPÍTULO 2

## Marcação HTML

### 2.1 Introdução

Já sabemos que HTML é uma linguagem de marcação que o desenvolvedor (você) usa para se comunicar com o agente de usuário, que por sua vez transforma a marcação em algo que o usuário (o internauta) entenda, como uma página de um site.

Assim, marcação HTML pode ser considerada um dos idiomas (existem outros que no momento não vêm ao caso) de comunicação do desenvolvedor com o dispositivo de usuário.

Como qualquer idioma, a HTML também tem suas regras de sintaxe e escrita, e um arquivo de texto escrito em conformidade com essas regras consiste-se na chamada marcação HTML ou estrutura HTML. O produto final de uma marcação HTML é um arquivo de texto que deve ser gravado com a extensão *.html*, por exemplo: *minha-pagina-web.html*.



Existem várias outras extensões válidas para gravação de arquivos de marcação, como *.htm*, *.php* e *.asp*, cada uma cumprindo uma finalidade, mas, para os objetivos deste livro, adotaremos *.html* e faremos referência às outras extensões quando, e se, for o caso.

### 2.2 Tag HTML

Para entender a escrita na linguagem HTML, a primeira coisa que devemos saber é que os diferentes conteúdos da marcação devem ser escritos dentro de *tags*. No início do conteúdo escrevemos uma *tag de abertura* e no fim uma *tag de fechamento*. Na figura 2.1 mostramos

a marcação de um cabeçalho e a seguir um parágrafo, destacando as tags e seus conteúdos.

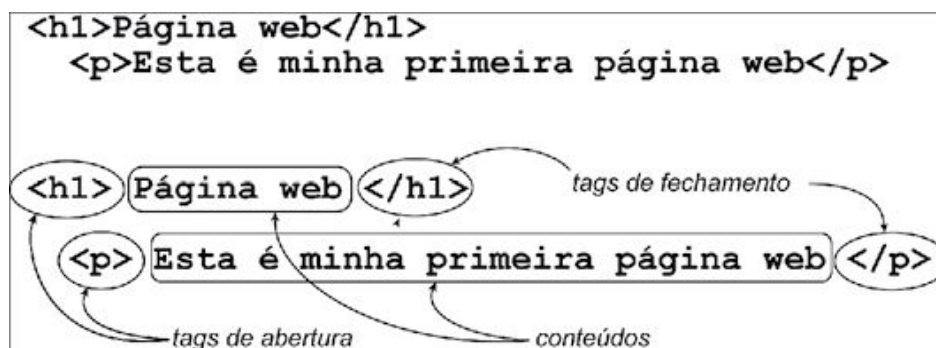


Figura 2.1 – tags HTML.

Uma tag HTML é constituída por uma ou mais letras, ou letra e número, que devem ser escritas entre os sinais < e >. A tag de fechamento é igual à tag de abertura com um sinal de barra / na frente. Existem várias tags HTML, que serão estudadas adiante neste livro, cada uma delas destinada a marcar um determinado tipo de conteúdo. No exemplo mostrado na figura 2.1, usamos as tags <h1></h1> e <p></p> que se destinam a marcar cabeçalhos de nível 1 e parágrafos, respectivamente.

Para fazer referência a um par de tag (abertura e fechamento), usamos o termo *elemento HTML* ou simplesmente *elemento*. Assim, diz-se elemento h1, elemento p ou, ainda, elemento cabeçalho nível 1, elemento parágrafo.



Existem tags denominadas *tags vazias* constituídas apenas da tag de abertura (não há tag de fechamento), tais como as tags <img> e <hr> destinadas a marcar imagens e linhas horizontais respectivamente. Estudaremos tais tags adiante neste livro.

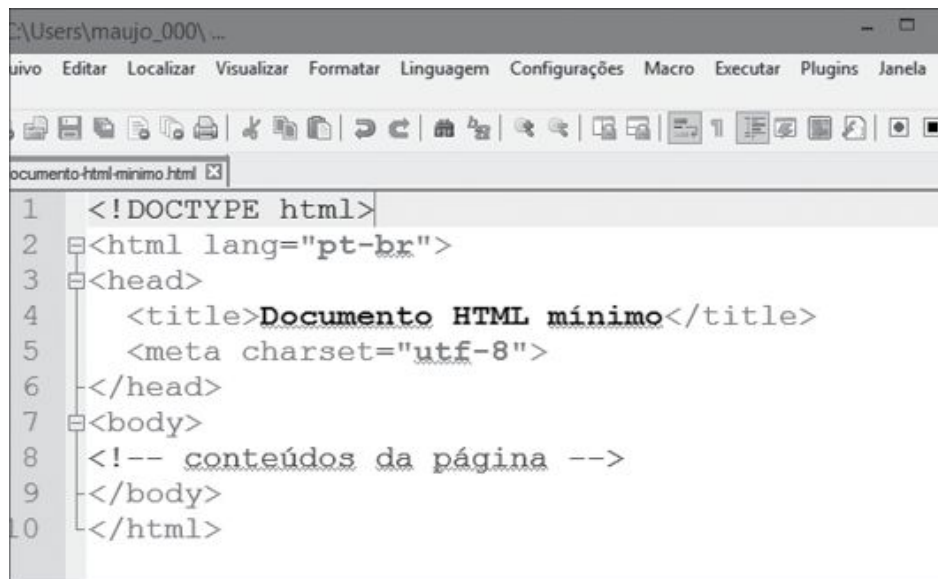
## 2.3 Documento HTML mínimo

Crie uma pasta no seu HD, com um nome *livroHTMLCSS*. Dentro dessa pasta, crie subpastas com o número do capítulo: *capitulo2*, *capitulo3* etc. Grave os arquivos constantes dos exercícios deste livro nas respectivas pastas do capítulo.

A partir daqui consideramos que você criou a pasta e subpastas conforme descrito anteriormente.

No site do livro existe uma estrutura de pastas idêntica à que foi descrita e está disponível para consulta online e também para download. Nela você encontra todos os exercícios e exemplos propostos neste livro.

Abra o Notepad++ e digite a marcação HTML conforme mostrada na figura 2.2.

A screenshot of the Notepad++ text editor. The title bar shows the file path 'C:\Users\maujo\_000\...'. The menu bar includes 'Arquivo', 'Editar', 'Localizar', 'Visualizar', 'Formatar', 'Linguagem', 'Configurações', 'Macro', 'Executar', 'Plugins', and 'Janela'. The toolbar contains various icons for file operations and editing. The active window is titled 'documento-html-minimo.html'. The code is as follows:

```
1 <!DOCTYPE html>
2 <html lang="pt-br">
3 <head>
4     <title>Documento HTML mínimo</title>
5     <meta charset="utf-8">
6 </head>
7 <body>
8     <!-- conteúdos da página -->
9 </body>
10 </html>
```

Figura 2.2 – Documento HTML mínimo.

Depois de digitar a marcação mostrada na figura 2.2, grave (salve), na pasta do capítulo 2, o arquivo com o nome *documento-html-minimo.html*.

O arquivo *documento-html-minimo.html*, que demonstra este exercício, está disponível para consulta online e download na pasta *capitulo2*.

Esta será a primeira marcação HTML criada por você para acompanhar na prática os exercícios e ensinamentos contidos neste livro. Observe a seguir a descrição (significado e finalidade) de cada linha de código do arquivo. Leia com atenção a descrição para cada linha, pois as informações ali contidas são essenciais para o entendimento do que se segue no livro.

Código comentado:



Linha	Descrição
Linha 1	<p>Toda marcação HTML que você criar deverá obrigatoriamente começar com a chamada declaração do tipo de documento (doctype). Essa linha informa ao agente de usuário (que é quem vai ler e interpretar o que está escrito a seguir) que a marcação que se segue é do tipo HTML5, que é a marcação HTML padrão e universal. A declaração de DOCTYPE não é uma tag HTML embora esteja contida entre os sinais &lt; e &gt;. Notar que escrevemos a palavra DOCTYPE em maiúscula e html em minúscula. Isso não é obrigatório: você pode usar maiúsculas e minúsculas como bem entender, inclusive para cada letra individualmente. Aconselho a adotar o formato mostrado, mas, se você preferir, adote tudo em minúscula ou tudo em maiúscula.</p> <p>A marcação HTML é insensível ao tamanho de caixa, ou seja, é indiferente usar maiúscula ou minúscula. Contudo, hoje em dia já há um consenso geral e é altamente recomendável que se excetuando o DOCTYPE, toda a marcação dos elementos da linguagem HTML sejam grafadas com letras minúsculas.</p>
Linhas 2 e 10	<p>Tags de fechamento e abertura do elemento html. Esse é o chamado elemento raiz do documento, e é quem engloba toda a marcação HTML. Notar que a declaração de DOCTYPE está fora dele e como dissemos DOCTYPE não é uma tag. Na tag de abertura desse elemento, linha 2, existe a declaração lang="pt-br". Isso é o que denominamos par atributo/valor sendo lang o atributo e pt-br o seu valor.</p> <p>Um par atributo/valor fornece informações adicionais sobre a tag na qual está inserido. Neste caso estamos informando que todo o conteúdo da marcação HTML do documento está escrito na língua (lang) portuguesa do Brasil (pt-br). Essa declaração é muito importante para fornecer informação do idioma em que a página foi escrita para tecnologias assistivas, tal como um leitor de tela que, sabendo o idioma, poderá ler a tela com a pronúncia adequada. Sempre use essa declaração na tag de abertura do elemento raiz do documento. Ou uso de aspas (" ") no valor do atributo é opcional.</p> <p>Existem centenas de atributos e valores para os diferentes elementos da HTML, cuja sintaxe e uso serão estudados neste livro.</p>
Linhas 3 e 6	<p>Tags de fechamento e abertura do elemento head. Essa é a chamada seção HEAD do documento, e, dentro dela, são admitidos vários outros elementos HTML entre eles, no mínimo os dois elementos conforme descritos a seguir.</p>

Linha	Descrição
Linha 4	Tags de fechamento e abertura do elemento title. Esse é o elemento mais importante de um documento. Ele se destina a marcar (ter como conteúdo) o título do documento. Nele o autor informa de forma resumida o conteúdo do documento. Escolha com muito critério o título de cada uma das páginas web que você desenvolver, pois ele conterá informações valiosas para que os mecanismos de busca e indexação, tal como o Google, encontrem-nas e mostrem sua página nos resultados de busca feitos pelo internauta.
Linha 5	Aqui um exemplo de tag vazia que mencionamos anteriormente. A tag meta também chamada de tag de metadado destina-se a fornecer informações adicionais sobre o documento. No nosso exemplo, mostramos essa tag com o par atributo/valor informando qual é a codificação de caracteres da marcação. Por padrão todo documento HTML5 deve ser escrito em codificação utf-8, e essa declaração é de uso obrigatório na seção HEAD do documento.
Linhas 7 e 9	Tags de fechamento e abertura do elemento body. Essa é a chamada seção BODY do documento e dentro dela são admitidos vários outros elementos HTML. Tudo o que estiver nessa seção será renderizado pelo navegador e mostrado para o usuário.
Linha 8	Nessa linha inserimos um comentário na marcação HTML. Comentar códigos é uma prática comum quando se escreve código seja ele HTML ou outro tipo qualquer. Cada linguagem de código tem sua sintaxe própria para inserção de comentários. Em HTML um comentário deve estar contido entre os sinais <!-- e --> como mostrado nessa linha. Comentários não causam nenhum efeito, eles existem apenas para serem lidos pelo desenvolvedor e são úteis para explicar certos trechos de códigos para outros desenvolvedores que eventualmente venham a alterar ou trabalhar com os códigos que nós criamos.

Na figura 2.3 mostramos a estrutura geral de um documento HTML5.

```

<!DOCTYPE html>  não é marcação HTML é declaração
<html>  elemento raiz do documento
  <head>
    seção HEAD do documento
  </head>
  <body>
    seção BODY do documento
  </body>
</html>

```

*Figura 2.3 – Estrutura HTML geral.*

A esta altura dos estudos, já criamos e salvamos um documento HTML denominado *documento-html-minimo.html*. A seguir vamos abrir o documento em um navegador. Na interface do Notepad++, com o documento nele aberto, vá para o menu Executar e, no submenu que se abre, escolha um navegador para visualizar o arquivo. Ou, se preferir, dê dois cliques no nome do arquivo para abri-lo no seu navegador padrão.

Você não verá absolutamente nada no navegador, mas não se decepcione, isso já era esperado, pois, como dissemos, o navegador renderiza (mostra) somente os conteúdos que estão dentro da seção BODY do documento, e, no nosso arquivo, essa seção está vazia, ou melhor, contém somente um comentário e já sabemos que por padrão comentário não é renderizado.

## 2.4 Primeira página web

Vamos inserir algum conteúdo na seção BODY do documento e ver o resultado. Abra o arquivo *documento-html-minimo.html* no Notepad++ e a seguir salve esse arquivo com o nome *minha-primeira-pagina-web.html*. Altere a marcação HTML mudando o conteúdo do elemento TITLE e substituindo o comentário da linha 8 por um elemento h1 e um elemento p, conforme mostrado a seguir.

```
<!DOCTYPE html>
<html lang="pt-br">
<head>
<title>Minha página web</title>
<meta charset="utf-8">
</head>
<body>
  <h1>Página web</h1>
  <p>Esta é minha primeira página web</p>
</body>
</html>
```

Após a alteração, salve o documento novamente e visualize-o em

um navegador. Agora sim! Observe que o texto do cabeçalho nível 1 aparece com letras maiores e em negrito, e o do parágrafo com letras menores e normal (não negrito). Tamanhos, tipos e cores das letras, entre outras características dos textos, podem ser facilmente alterados com uso das CSS (Cascade Style Sheets), ou folhas de estilo em cascata, como veremos adiante neste livro.

## 2.5 Exercício proposto

Na linguagem HTML existem seis níveis de cabeçalhos que são os elementos `h1`, `h2`, `h3`, `h4`, `h5`, e `h6`. Cabeçalhos são também conhecidos como títulos, assim, o elemento `h1` é o título principal e os demais níveis são os subtítulos, tal como os conhecemos em textos de jornais e revistas. Não confundir os títulos (cabeçalhos) e subtítulos constantes da seção `body` do documento com o título do documento constante da seção `head` e marcado com o elemento `title`.

Abra o arquivo *documento-html-minimo.html* no Notepad++ e a seguir salve esse arquivo com o nome *seis-niveis-de-cabecalho.html*.

Altere a marcação HTML mudando o conteúdo do elemento `title` e substituindo o comentário da linha 8 por um elemento para marcar cada um dos seis níveis de cabeçalhos e, para cada cabeçalho, crie dois ou mais elementos `p`. Invente os textos dos cabeçalhos e parágrafos ou copie os textos de uma revista ou jornal ou, melhor ainda e mais rápido, use um texto padrão “lorem ipsum”.

Uma técnica comum para fazer experiências com marcação de textos HTML, sem perder tempo com digitação, é copiar e colar trechos de um texto padrão escrito em latim e denominado “lorem ipsum”. É uma boa ideia salvar no seu HD um arquivo contendo alguns parágrafos deste texto padrão para usar quando for necessário.

Faça uma busca no Google por “lorem ipsum” e você receberá como resultado vários links com geradores daquele texto. Entre em um link, gere alguns parágrafos do texto, copie e cole no Notepad++ os parágrafos gerados e salve-os com o nome de *lorem-ipsum.txt*.

Pronto, agora você tem um arquivo com vários textos para usar quando necessário. Normalmente você obtém vários blocos de texto, mas, para suas experiências, poderá usar apenas parte ou partes dos blocos de texto.

Observe a seguir um trecho do arquivo *seis-niveis-de-cabecalho.html* que usa o “lorem ipsum”.

```
...
<h1>Lorem ipsum dolor sit</h1>
  <p>Sale errem menandri est ad, tistique mandamus partiendo vel in.</p>
<h2>Ea qui quod accumsan</h2>
  <p>Fabellas vituperata mediocritatem eam cu, sed cu viderer detraxit, dicta
numquam dolorum vel cu. Sonet detraxit cu mel, ex pri utamur feugait.</p>
  <p>Mei id illud dicta, ex ipsum movet malorum mea.</p>
<h3>Qui in illum falli nihil</h3>
  <p>Viris dictas detraxit ne nam, sapientem reformidans his no.</p>
  <p>An quo molestie nominati sadipscing, quot dolorem vis ex.</p>
  <p>Postulant dissentias necessitatibus eos ex, semper dolorem delicata usu
ea.</p>
...
```

A princípio, os textos do “lorem ipsum” podem parecer um tanto estranhos e completamente sem sentido, mas são amplamente usados e aceitos em desenvolvimento web e, embora em latim, simulam com precisão a escrita em qualquer língua no que diz respeito a número de letras em palavras e distribuição de palavras na frase. Lembre-se de que esse texto serve como simulador do texto definitivo a ser inserido em cada elemento da marcação.

O arquivo *seis-niveis-de-cabecalho.html*, que demonstra este exercício, está disponível para consulta online e download na pasta *capitulo2*.

## 2.6 Modelo HTML

Conforme observamos na renderização do arquivo *seis-niveis-de-cabecalho.html*, cada um dos cabeçalhos e dos parágrafos contidos na marcação começou em uma nova linha, isto é, ao começar e ao terminar o conteúdo daqueles elementos, houve uma quebra de linha.

As quebras de linha que forem criadas na marcação HTML não são respeitadas na renderização do documento. Também não são respeitados na renderização do documento os espaços em branco deixados entre as palavras de um texto, ou seja, deixar um espaço em branco na marcação ou deixar mais de um, a renderização será com apenas um espaço entre palavras.

Abra no Notepad++ o arquivo *seis-niveis-de-cabecalho.html* e salve-o com o nome *quebras-de-linha-na-html.html*. Altere o título do documento (na seção HEAD) para “Quebras de linha na HTML”. Altere a distribuição da marcação na seção BODY colocando uma linha da marcação imediatamente após o término da linha anterior, ou seja, elimine as quebras de linha da marcação. Aumente o número de espaços em branco entre algumas palavras dos cabeçalhos e dos parágrafos. Visualize o arquivo em um navegador e depois visualize o arquivo *seis-niveis-de-cabecalho.html*. Compare a renderização dos dois e constate que não há diferença.

O arquivo *quebras-de-linha-na-html.html*, que demonstra quebras de linha, está disponível para consulta online e download na pasta *capitulo2*.

Entre os elementos da HTML uns causam quebra de linha, conforme já constatamos no exemplo anterior, outros não causam a quebra de linha, como o elemento destinado a marcar uma palavra do texto em itálico ou negrito, como veremos adiante.

Elementos que causam quebra de linha, na renderização, são chamados de *elementos nível de bloco* e os que não causam quebra de linha são chamados *elementos inline* (em linha).

No modelo HTML, os elementos nível de bloco se estendem por toda a largura da linha, obrigando o elemento que se segue a ocupar a linha seguinte; os elementos inline ocupam apenas a largura do seu conteúdo e não quebram a linha, ou seja, é possível posicionar vários elementos inline em uma mesma linha.

A quebra de linha pelos elementos nível de bloco se dá tanto antes da tag de abertura como depois da tag de fechamento do elemento.

Esse comportamento de quebras de linhas e de espaçamentos entre palavras é próprio do modelo de renderização HTML ou simplesmente do modelo HTML.

## **2.7 Acessibilidade na web**

Um dos princípios básicos do desenvolvimento web, ou criação de marcação HTML, é fazer com que o conteúdo criado e servido seja acessível a qualquer pessoa, independentemente do dispositivo que ela esteja usando ou de suas necessidades especiais.

Isso significa que, ao projetar conteúdo web, o desenvolvedor deve garantir que o conteúdo seja acessível (possa ser entendido) para usuários com equipamentos e conexões precárias, usuários impossibilitados de usar o mouse, por pessoas idosas, por daltônicos, por pessoas sem as mãos, por cegos, enfim, por qualquer usuário que tenha qualquer tipo de restrição ou de necessidade especial.

A linguagem HTML nos fornece várias funcionalidades e mecanismos para contemplar o que é chamado de critérios de acessibilidade e que visam permitir, por exemplo, que um usuário cego tenha uma noção bem próxima de como é uma imagem colocada na tela.

Existem várias especificações do W3C para a acessibilidade na web, e o leitor, como desenvolvedor, não deve relegar a segundo plano e muito menos esquecer esse princípio de desenvolvimento.

A especificação do W3C denominada “Diretrizes de Acessibilidade para Conteúdo Web (WCAG) 2.0” foi traduzida para o português e encontra-se hospedada em <http://traducoes.w3c.br/TR/WCAG/>.

# CAPÍTULO 3

## Elementos HTML

### 3.1 Introdução

Existem cento e oito elementos previstos nas especificações do W3C para a HTML5 (considerando que cada nível de cabeçalho conta como um elemento). Neste capítulo estudaremos com detalhes os elementos mais usados em páginas web e, em capítulos posteriores, sempre que houver necessidade de uso de um elemento não descrito aqui, descreveremos esse elemento na ocasião que formos usar. No apêndice A do livro mostramos uma tabela com nome e descrição de todos os elementos da linguagem e relacionamos também os atributos que se aplicam a cada um deles. Use esse apêndice como fonte de consulta rápida para saber sobre o uso e a finalidade de cada elemento.

### 3.2 Elementos da HTML

#### ***html***

Mostramos esse elemento no capítulo anterior. Ele é o elemento raiz do documento que engloba as seções HEAD e BODY.

#### ***head***

Mostramos esse elemento no capítulo anterior. Ele se destina a marcar a seção HEAD do documento.

#### ***title***

Mostramos esse elemento no capítulo anterior. Ele se destina a marcar o título do documento.



## ***meta***

Mostramos esse elemento no capítulo anterior. Ele se destina a fornecer informações adicionais sobre o documento, tais como a codificação de caracteres, o nome do autor do documento e outras.

## ***body***

Mostramos esse elemento no capítulo anterior. Ele se destina a marcar a seção body do documento. Tudo que está dentro dessa seção será renderizado pelo agente de usuário (por exemplo: o navegador).

## ***h1-h6***

Vimos esses elementos no capítulo anterior. Eles são do tipo nível de bloco e destinam-se a marcar os textos de títulos e subtítulos dos conteúdos do documento.

## ***p***

Mostramos esse elemento no capítulo anterior. Ele é do tipo nível de bloco e se destina a marcar parágrafos.

## ***a***

Esse elemento é do tipo inline e se destina a marcar links.

Exemplo:

```
<a href="http://maujor.com">Site do Maujor</a>
```

Essa marcação cria um link para o Site do Maujor que está hospedado em *http://maujor.com*. Notar a presença do atributo href na tag de abertura do elemento A cujo valor é o endereço que será aberto quando o usuário clicar o link.

## ***ul e li***

Esses elementos são do tipo nível de bloco e se destinam a marcar listas que são chamadas de listas não ordenadas. Exemplo:

```
<ul>
```

```
<li>Carros</li>
<li>Cores</li>
<li>Frutas</li>
<li>Países</li>
</ul>
```

Essa marcação cria uma lista com quatro itens. No início de cada item da lista é renderizado, por padrão, um círculo (bolinha cheia) que se denomina bullet.

Na sintaxe para marcação de listas, o elemento ul é um container para os elementos li que marcam cada item da lista.

É possível criar listas chamadas aninhadas, nas quais um item da lista pode conter subitens e estes seus subitens e assim indefinidamente em diversos níveis de aninhamento, segundo mostrado no exemplo a seguir.

Exemplo:

```
<ul>
  <li>Carros
    <ul>
      <li>Mercedes Benz</li>
      <li>BMW</li>
      <li>Lanborghini</li>
    </ul>
  </li>
  <li>Cores</li>
  <li>Frutas
    <ul>
      <li>Abacate</li>
      <li>Laranja
        <ul>
          <li>Baía</li>
          <li>Lima</li>
          <li>Seleta</li>
        </ul>
      </li>
      <li>Mamão</li>
    </ul>
  </li>
</ul>
```

```
<li>Países</li>
</ul>
```

Para cada nível de aninhamento, até o terceiro nível, o formato do bullet muda de círculo para circunferência e para quadrado.

### ***ol e li***

Esses elementos são do tipo nível de bloco e se destinam a marcar listas que são chamadas de listas ordenadas. Notar que o container para listas ordenadas é `ol` e a marcação é semelhante à mostrada anteriormente. A diferença é que, em lugar dos bullets gráficos, as listas ordenadas são renderizadas com números ou opcionalmente com outros tipos de marcador de ordenação.

### ***dl, dt e dd***

Esses elementos são do tipo nível de bloco e se destinam a marcar listas que são chamadas de listas de definição.

Exemplo:

```
<dl>
  <dt>Características do produto</dt>
    <dd>Mais opções de cores e tamanhos</dd>
    <dd>Aumento da vida útil</dd>
  <dt>Acréscimo de itens de segurança</dt>
    <dd>Manuseio seguro por crianças acima de 3 anos</dd>
  <dt>Custos</dt>
    <dd>Redução dos custos iniciais de manutenção</dd>
    <dd>Aumento dos prazos de pagamento</dd>
    <dd>Possibilidade de inscrição em programa de leasing</dd>
</dl>
```

O container geral de uma lista de definição é o elemento `dl`.

Essa marcação cria uma lista de definição com três itens chamados de termos de definição (elementos `dt`) cada um deles com um ou mais termos descritivos (elementos `dd`).

### **Exercício:**

Antes de continuar com o estudo dos elementos da HTML, vamos

criar um arquivo com esses novos elementos que acabamos de aprender e visualizar no navegador a renderização deles.

Abra o arquivo *documento-html-minimo.html* no Notepad++ e a seguir salve esse arquivo com o nome *elementos-link-e-listas.html*. Altere a marcação HTML mudando o conteúdo do elemento `title` para “Elementos links e listas da HTML” e substituindo o comentário da linha 8 pelas marcações mostradas nos exemplos anteriores para esses elementos. Observe na figura 3.1 a renderização da marcação criada.

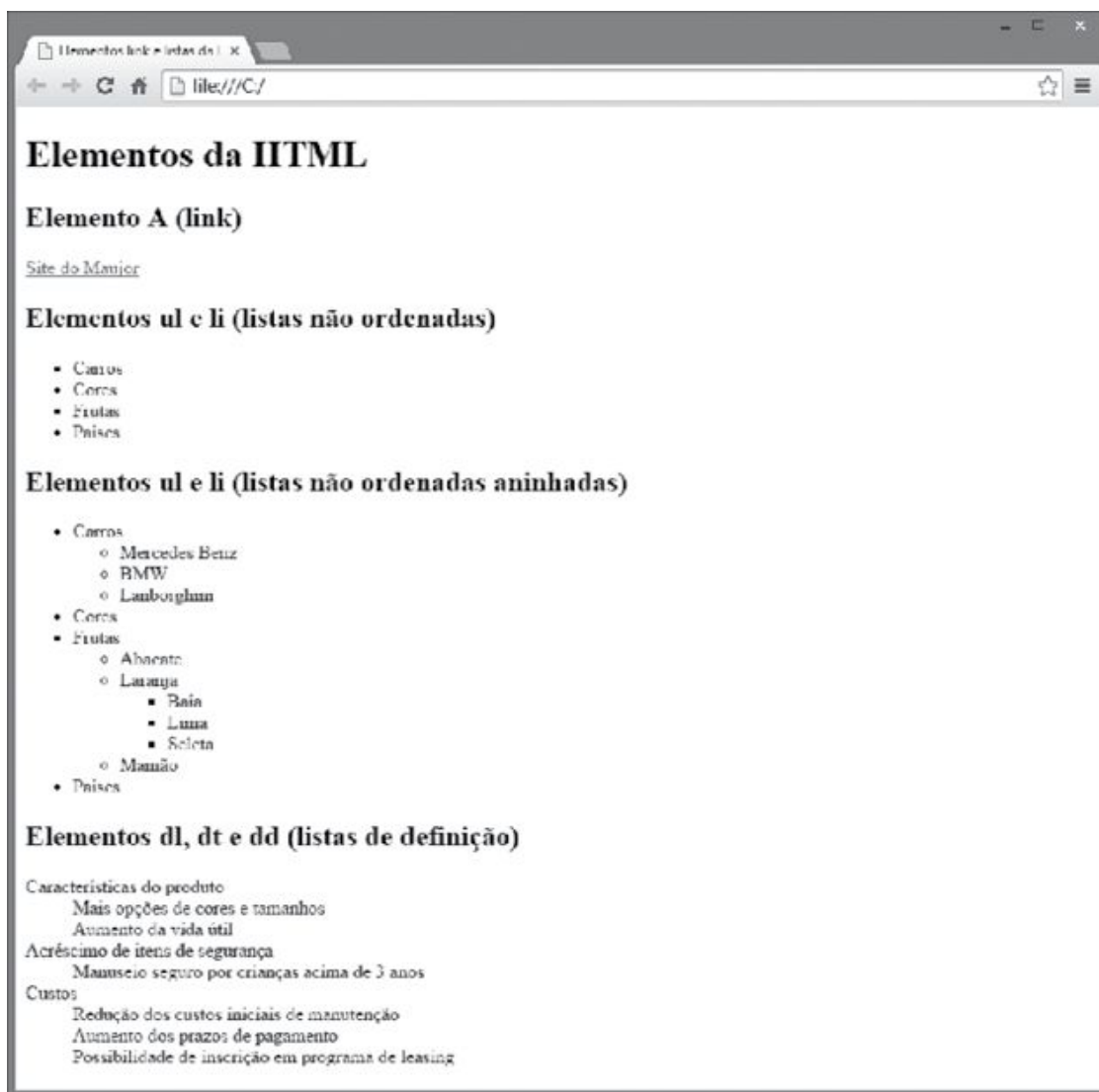


Figura 3.1 – Elementos link e listas da HTML.

O arquivo *elementos-link-e-listas.html*, que demonstra esse exercício,

está disponível para consulta online e download na pasta *capitulo3*.

## ***div***

Esse é um elemento do tipo nível de bloco e se destina a criar um container geral para outros elementos. É um elemento sem destinação específica, ou seja, diferentemente dos elementos `h1` e `p`, por exemplo, que se destinam a marcar cabeçalhos de nível 1 e parágrafos respectivamente, esse elemento pode conter (marcar) qualquer outro elemento em qualquer quantidade e até mesmo outros elementos `div`.

Elementos com essas características são chamados de elementos sem valor semântico. *Semântica* é um aspecto de alta relevância em marcação HTML e diz respeito ao correto uso do elemento para marcar o conteúdo. Por exemplo: se o conteúdo é um cabeçalho de nível 2, ele deverá ser marcado com o elemento `h2`; se é o item de uma lista não ordenada, deverá ser marcado com o elemento `li` contido em um elemento `ul`, e assim por diante.



Recomendo que a partir daqui o leitor esteja perfeitamente ciente de que marcação semântica é um aspecto fundamental e importantíssimo a ser levado em consideração quando se desenvolve e cria-se marcação HTML. Use sempre o elemento de marcação apropriado ao conteúdo que estiver criando.

### **Exemplo:**

```
<div>
  <h3>Descrição do produto</h3>
  <p>Este produto pode ser fornecido em duas versões. Etc...</p>
</div>
```

Nesse exemplo, a presença do container `div` em nada altera a renderização do cabeçalho de nível 3 e do parágrafo. Para efeito de renderização é como se o elemento `div` não estivesse presente na marcação.

Então para que serve o `div`? Conforme veremos adiante, o `div` poderá servir, entre outras inúmeras coisas, como uma espécie de

caixa com bordas e fundo colorido, para dar um formato visual atraente ao texto dentro dele. Isso deverá ser feito com uso das CSS.

### ***span***

Esse é um elemento do tipo inline e destina-se a criar um container geral para outros elementos inline. É um elemento sem destinação específica, ou seja, diferentemente dos elementos `h1` e `p`, por exemplo, que se destinam a marcar cabeçalhos de nível 1 e parágrafos respectivamente, esse elemento pode conter (marcar) qualquer outro elemento inline em qualquer quantidade e até mesmo outros elementos `span`.

Tal como o elemento `div`, esse elemento é sem valor semântico. A diferença deste elemento para o elemento `div` é que se trata de um elemento inline enquanto `div` é um elemento nível de bloco.

Exemplo:

```
<p>Texto de um parágrafo com <span>palavra</span> marcada com o  
elemento SPAN</p>
```

Nesse exemplo a presença do container `span` em nada altera a renderização do texto “palavra” contido no parágrafo. Para efeito de renderização é como se o elemento `span` não estivesse presente na marcação.

Então para que serve o `span`? Conforme veremos adiante, o `span` poderá servir, entre outras inúmeras coisas, como uma espécie de caixa para que seu conteúdo seja renderizado em uma cor diferente, ou com um tamanho de letra diferente. Isso deverá ser feito com uso das CSS.

### ***b e strong***

Estes elementos são do tipo inline e causam o mesmo efeito de renderização, mas têm valor semântico diferente. Quando aplicados a um pequeno trecho de texto ou palavra, ambos causam renderização em negrito.

Use o elemento `b` para dar o aspecto visual negrito e use o elemento `strong` para dar forte ênfase. Para melhor entender a diferença, suponha um internauta com visão normal e um internauta deficiente visual navegando com um leitor de tela ([http://pt.wikipedia.org/wiki/Leitor\\_de\\_tela](http://pt.wikipedia.org/wiki/Leitor_de_tela)). A marcação com o elemento `b` somente terá efeito (negrito) para o internauta com visão normal e a marcação com o elemento `strong` terá efeito (negrito e forte ênfase) para os dois, pois além de negrito, o leitor de tela lerá o conteúdo de `strong` com voz em forte ênfase, e o internauta deficiente visual saberá que aquele trecho foi destacado pelo autor do documento. É claro que, para marcar em negrito ou forte ênfase, deverá haver uma razão.

Exemplo:

`<p>Texto de um parágrafo com <b>palavra em negrito</b> marcada com o elemento B</p>`

`<p>Texto de um parágrafo com <strong>palavra com forte ênfase</strong> marcada com o elemento STRONG</p>`

## ***i e em***

Esses elementos são do tipo inline e causam o mesmo efeito de renderização, mas têm valor semântico diferente. Quando aplicados a um pequeno trecho de texto ou palavra, ambos causam renderização em itálico.

Use o elemento `i` para dar o aspecto visual itálico (inclinado) às letras e use o elemento `em` para dar ênfase. A justificativa de uso é semelhante àquela descrita para os elementos `b` e `strong` no item anterior.

Exemplo:

`<p>Texto de um parágrafo com <i>palavra em itálico</i> marcada com o elemento I</p>`

`<p>Texto de um parágrafo com <em>palavra com ênfase</em> marcada com o elemento EM</p>`

## ***small***

Esse é um elemento do tipo inline e se destina a marcar pequenos trechos de texto relacionados com o conteúdo do texto no qual foram marcados. Notas sobre direitos autorais no rodapé do site e pequenas notas de advertência são conteúdos para se marcar com o elemento `small`. A renderização de textos marcados com esse elemento é feita com tamanho de fonte menor do que o do texto no qual foram marcados.

Exemplo:

```
<h3>Valores das diárias</h3>
<dl>
  <dt>Quarto de solteiro
    <dd>R$320,00 <small>café da manhã incluso</small>
  <dt>Quarto de casal
    <dd>R$490,00 <small>café da manhã incluso</small>
</dl>
```

## ***code***

Esse é um elemento do tipo inline e se destina a marcar trechos de código de computador ou qualquer string que um computador reconheça. Nomes de arquivos, nome de um programa de computador, nome de um elemento HTML, trechos de código tais como os mostrados nos exemplos são conteúdos para se marcar com o elemento `code`. A renderização de textos marcados com esse elemento normalmente é feita com tipo de fonte monoespaçada, diferente da fonte do texto no qual foram marcados.

Exemplo:

```
<p>Os elementos <code>div</code> e <code>span</code> da HTML não têm
valor semântico.</p>
```

## ***hr***

Esse elemento é do tipo nível de bloco e destina-se a criar uma linha reta horizontal de 1px. Trata-se de um elemento vazio, isto é, existe apenas sua tag de abertura `<hr>`. A simples presença da tag de abertura é suficiente para criar uma linha horizontal.



Exemplo:

<p>Texto de um parágrafo</p>  
<hr> <!-- Cria uma linha horizontal entre os parágrafos -->  
<p>Texto de outro parágrafo</p>

***br***

Esse é um elemento vazio, isto é, existe apenas sua tag de abertura `<br>` e destina-se a criar uma quebra de linha. Múltiplos elementos `br` criam múltiplas quebras de linha. A simples presença da tag de abertura é suficiente para criar uma quebra de linha.

Exemplo:

<p>Texto do primeiro parágrafo</p>  
<p>Texto do segundo parágrafo</p>  
    <br><br> <!-- Cria duas quebras de linhas adicionais entre os parágrafos -->  
<p>Texto do terceiro parágrafo</p>

Dissemos anteriormente que quebras de linhas e espaçamento entre palavras do texto inseridas na marcação HTML são ignoradas quando da renderização do documento. Vimos que quebras de linhas são próprias dos elementos nível de bloco e o elemento `br` destina-se a criar quebras de linhas adicionais ou, ainda, forçar a quebra de linhas entre elementos inline.

Mas, como criar mais de um espaçamento entre palavras de um texto?

Não existe um elemento próprio para criar espaçamentos entre palavras (ou letras) de um texto, mas felizmente existe uma entidade HTML (caractere codificado) para obter tal efeito. Trata-se das entidades `&nbsp;` e `&#160;` que, quando colocadas entre palavras (ou letras) de um texto, criam um espaço em branco.

Exemplo:

```
<!-- Insere quatro espaços em branco entre de e um -->  
<p>Texto de&nbsp;&nbsp;&nbsp;&nbsp;&um parágrafo</p>
```

ou

```
<!-- Insere quatro espaços em branco entre de e um -->
```

<p>Texto de&#160;&#160;&#160;&#160;um parágrafo</p>

## ***img***

Esse é um elemento inline e vazio, isto é, existe apenas sua tag de abertura <img> e destina-se a inserir uma imagem no documento. O uso desse elemento exige no mínimo a inserção de dois atributos. Observe o exemplo a seguir e depois a explicação de como funciona esse elemento.

### **Exercício:**

Para que esse exercício funcione é necessário que você copie o arquivo *gato.jpg* para a pasta *Capitulo3* do seu HD. Esse arquivo está na pasta *Capitulo3* obtida com o download dos códigos no site do livro, ou simplesmente consulte a solução do exercício online ou os arquivos baixados do site.

Abra o arquivo *documento-html-minimo.html* no Notepad++ e a seguir salve esse arquivo com o nome *elemento-img.html*. Altere a marcação HTML mudando o conteúdo do elemento title para “Elemento img da HTML”, substituindo o comentário da linha 8 pelas marcações mostradas a seguir que esclarecem o uso desses elementos. Para que este exercício funcione, você precisa gravar a imagem do gato na pasta *capitulo3*.

```

```

Esse exemplo insere a imagem de um gato no documento. O atributo *src* recebe como valor o endereço onde se encontra a imagem do gato, e o atributo *alt* fornece uma descrição sumária da imagem. A descrição destina-se a ser lida pelos leitores de tela e outros softwares assistivos. Notar que até o leitor, mesmo sem ver a renderização, é capaz de fazer uma ideia de como seja a imagem.

O arquivo *elemento-img.html*, que demonstra esse exercício, está disponível para consulta online e download na pasta *capitulo3*.

## ***article***

Esse é um elemento do tipo nível de bloco e destina-se a marcar

conteúdos que possam ser distribuídos, reutilizados e entendidos isoladamente, como um post em um fórum, um comentário em blog, um widget interativo ou, finalmente, qualquer conteúdo independente de um documento.

Exemplo:

```
<article>
  <h1>Computador Pessoal</h1>
  <p> Um computador pessoal ou PC (do inglês Personal Computer) é um
computador de pequeno porte e baixo...</p>
  <!-- mais descrição e características do PC -->
</article>
```

No exemplo mostrado temos um artigo sobre computadores pessoais.

## ***section***

Esse é um elemento do tipo nível de bloco e destina-se a criar um container genérico para conteúdos. Se houver necessidade de um container genérico somente para fins de aplicação de regras de estilo, use o elemento `div` e não `section`. Use `section` de forma geral, quando o uso de `article` ou de outro elemento não for apropriado. Lembre-se de que a decisão pelo uso de um ou outro elemento HTML depende de sua semântica como estudamos anteriormente.

Exemplo:

```
<section>
<h1>Estação de trabalho</h1>
  <article>
    <h2>Computador</h2>
    <p>Computador é...</p>
  </article>
  <article>
    <h2>Impressora</h2>
    <p>Impressora é...</p>
  </article>
  ...
</section>
```

No exemplo mostrado, o elemento `section` foi usado como container para elementos `article`. Mas, atenção, pode haver casos em que elementos `article` contenham elementos `section`.

## ***header***

Esse é um elemento do tipo nível de bloco e destina-se a marcar conteúdos introdutórios e de auxílio à navegação. Em geral, esses elementos contêm um cabeçalho dos níveis `h1` a `h6` e podem incluir também tabela de conteúdos (índice), formulário de busca, informações complementares e logotipo.

Exemplo:

```
<article>
  <header>
    <h1>Computador Pessoal</h1>
    <p>Publicado em: 23/02/2015</p>
  </header>
  <p> Um computador pessoal ou PC (do inglês Personal Computer) é um
computador de pequeno porte e baixo...</p>
  <!-- mais descrição e características do PC -->
</article>
```

## ***footer***

Esse é um elemento do tipo nível de bloco e destina-se a marcar conteúdos de rodapé. Em geral esses elementos contêm informações sobre quem escreveu o conteúdo, links para conteúdos relacionados e informações sobre direitos autorais.

Exemplo:

```
<article>
  <header>
    <h1>Computador Pessoal</h1>
    <p>Publicado em: 23/02/2015</p>
  </header>
  <p> Um computador pessoal ou PC (do inglês Personal Computer) é um
computador de pequeno porte e baixo...</p>
  <!-- mais descrição e características do PC -->
  <footer>
```

```
<p>Copyright 2015 - Todos os direitos reservados</p>
</footer>
```

```
</article>
```

## **aside**

Esse é um elemento do tipo nível de bloco e destina-se a marcar conteúdo relacionado a outro conteúdo. Em geral, esses elementos são usados para marcar barras laterais de conteúdos, blocos de conteúdos relacionados a outro conteúdo, mas visualmente separados.

Existem dois contextos distintos de uso desse elemento. Quando dentro de um elemento `article`, seu conteúdo deve estar relacionado ao conteúdo daquele elemento. Quando fora do elemento `article`, seu conteúdo deve estar relacionado ao conteúdo do site ou da página, como um banner de propaganda ou links de navegação.

Exemplo:

```
<article>
  <header>
    <h1>Computador Pessoal</h1>
    <p>Publicado em: 23/02/2015</p>
  </header>
  <p> Um computador pessoal ou PC (do inglês Personal Computer) é um
computador de pequeno porte e baixo...</p>
  <!-- mais descrição e características do PC -->
  <aside> <!-- Dentro de article -->
    <!-- Conteúdo relacionado à matéria Computador Pessoal -->
  </aside>
</article>

<aside> <!-- Fora de article -->
  <!-- Conteúdo relacionado à página que contém a matéria Computador
Pessoal -->
</aside>
```

## **nav**

Esse é um elemento do tipo nível de bloco e destina-se a marcar o

mecanismo principal de navegação da página, contendo links para outras páginas ou para partes da mesma página. Pequenos grupos de links, tais como os existentes no rodapé da página apontando para política do site, termos de serviços, homepage e página de direitos autorais, não precisam ser marcados com esse elemento.

Existem dois contextos distintos de uso desse elemento. Quando dentro de um elemento `article`, seu conteúdo deve estar relacionado ao conteúdo daquele elemento. Quando fora do elemento `article`, seu conteúdo deve estar relacionado ao conteúdo do site ou da página, como um banner de propaganda ou links de navegação.

Exemplo:

```
<nav>
  <ul>
    <li><a href="/">Home</a></li>
    <li><a href="/portfolio.html">Portfólio</a></li>
    <li><a href="/artigos.html">Artigos</a></li>
    <li><a href="/contato.html">Contato</a></li>
  </ul>
</nav>
```

Nesse exemplo mostramos uma lista de links que compõem a navegação principal de um site. Notar o uso dos elementos `ul` e `li`.

## ***main***

Esse é um elemento do tipo nível de bloco e destina-se a marcar a seção do documento que contém o assunto principal da página. Não devem ser inclusos dentro desse elemento conteúdos tais como o topo da página, o rodapé da página e os links da navegação.

### **Exercício:**

Antes de continuar com o estudo, vamos criar um arquivo com alguns desses novos elementos que acabamos de aprender e visualizar no navegador a renderização deles.

Abra o arquivo *documento-html-minimo.html* no Notepad++ e a seguir salve esse arquivo com o nome *outros-elementos.html*. Altere a

marcação HTML mudando o conteúdo do elemento title para “Outros elementos da HTML” e substituindo o comentário da linha 8 pelas marcações mostradas a seguir que esclarecem o uso desses elementos.

- **HTML**

```
<h1>Elementos da HTML</h1>
  <h2>Elementos B e STRONG (negrito e forte ênfase)</h2>
    <p>Texto de um parágrafo com <b>palavra em negrito</b> marcada com
o elemento B</p>
    <p>Texto de um parágrafo com <strong>palavra com forte
ênfase</strong> marcada com o elemento STRONG</p>
  <h2>Elementos I e EM (itálico e ênfase)</h2>
    <p>Texto de um parágrafo com <i>palavra em itálico</i> marcada com o
elemento I</p>
    <p>Texto de um parágrafo com <em>palavra com ênfase</em> marcada
com o elemento EM</p>
  <h2>Elemento SMALL (advertência e notas inline)</h2>
    <h3>Valores das diárias</h3>
    <dl>
      <dt>Quarto de solteiro
        <dd>R$320,00 <small>café da manhã incluído</small>
      <dt>Quarto de casal
        <dd>R$490,00 <small>café da manhã incluído</small>
    </dl>
  <h2>Elemento CODE (trechos de código de computador)</h2>
    <p>Os elementos <code>div</code> e <code>span</code> da HTML não
têm valor semântico.</p>
  <h2>Elemento HR (linha horizontal)</h2>
    <p>Texto de um parágrafo</p>
    <hr> <!-- Cria uma linha horizontal entre os parágrafos -->
    <p>Texto de outro parágrafo</p>
  <h2>Elemento BR (quebra de linha)</h2>
    <p>Texto do primeiro parágrafo</p>
    <p>Texto do segundo parágrafo</p>
    <br><br> <!-- Cria duas quebras de linhas adicionais entre os
parágrafos -->
    <p>Texto do terceiro parágrafo</p>
```

Observe na figura 3.2 parte da renderização da marcação criada.

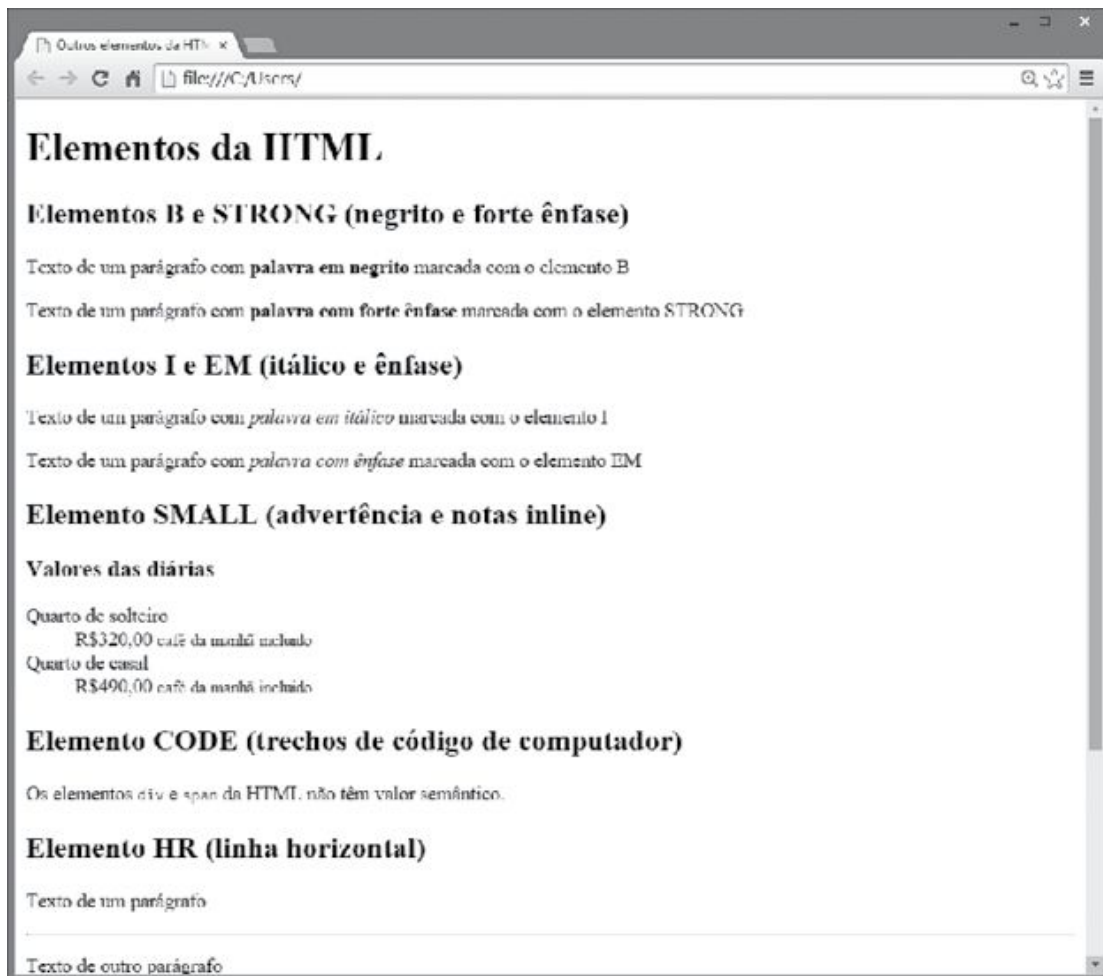


Figura 3.2 – Outros elementos da HTML.

O arquivo *outros-elementos.html*, que demonstra esse exemplo, está disponível para consulta online e download na pasta *capitulo3*.

### **audio**

Esse é um elemento do tipo nível de bloco e destina-se a inserir um som ou stream de áudio.

#### **Exercício:**

Abra o arquivo *documento-html-minimo.html* no Notepad++ e a seguir salve esse arquivo com o nome *elemento-audio.html*. Altere a marcação HTML mudando o conteúdo do elemento title para “Elemento audio da HTML5” e substituindo o comentário da linha 8 pelas marcações mostradas a seguir que esclarecem o uso desses elementos.



Para que este exercício funcione é necessário que você copie os arquivos *som.ogg*, *som.mp3* e *som.wav* para a pasta *Capitulo3* do seu HD. Esses arquivos estão na pasta *Capitulo3* obtida com o download dos códigos no site do livro, ou simplesmente consulte a solução do exercício online ou nos arquivos baixados do site.

- **HTML**

```
<audio controls>
  <source src="som.ogg" type="audio/ogg">
  <source src="som.mp3" type="audio/mpeg">
  <source src="som.wav" type="audio/wave">
  <p>Seu navegador não suporta o elemento audio da HTML5.<br>Faça <a
href="som.zip">download de som.zip</a></p>
</audio>
```

O arquivo *elemento-audio.html*, que demonstra esse exemplo, está disponível para consulta online e download na pasta *capitulo3*.

## **video**

Esse é um elemento do tipo nível de bloco e destina-se a inserir um vídeo.

### **Exercício:**

Abra o arquivo *documento-html-minimo.html* no Notepad++ e a seguir salve esse arquivo com o nome *elemento-video.html*. Altere a marcação HTML mudando o conteúdo do elemento *title* para “Elemento video da HTML5” e substituindo o comentário da linha 8 pelas marcações mostradas a seguir que esclarecem o uso desses elementos.

Para que este exercício funcione é necessário que você copie os arquivos *video.ogv*, *video.mp4* e *video.webm* para a pasta *Capitulo3* do seu HD. Esses arquivos estão na pasta *Capitulo3* obtida com o download dos códigos no site do livro, ou simplesmente consulte a solução do exercício online ou nos arquivos baixados do site.

- **HTML**

```
<video controls>
  <source src="video.ogv" type="video/ogg">
```

```
<source src="video.mp4" type="video/mp4">
<source src="video.webm" type="video/webm">
<!-- Código (X)HTML para inserção de vídeo com Flash -->
<p>Seu navegador não suporta o elemento video da HTML5.<br>
Faça <a href="video.mp4">download do vídeo</a></p>
</video>
```

O arquivo *elemento-video.html*, que demonstra esse exemplo, está disponível para consulta online e download na pasta *capitulo3*.

## **canvas**

Esse é um elemento do tipo nível de bloco e destina-se a criar uma área de desenho.

### **Exercício:**

Abra o arquivo *documento-html-minimo.html* no Notepad++ e a seguir salve esse arquivo com o nome *elemento-canvas.html*. Altere a marcação HTML mudando o conteúdo do elemento title para “Elemento canvas da HTML5”, crie uma folha de estilo incorporada ao documento e substitua o comentário da linha 8 pelas marcações mostradas a seguir que esclarecem o uso desses elementos.

### **• HTML**

```
<video controls>
  <canvas>
    <p>Conteúdo alternativo para navegadores que não suportam canvas.</p>
  </canvas>
```

O arquivo *elemento-canvas.html*, que demonstra esse exemplo, está disponível para consulta online e download na pasta *capitulo3*.

As técnicas para desenhar em canvas envolvem uma linguagem própria e JavaScript, e não serão abordadas neste livro.

# CAPÍTULO 4

## Introdução às CSS

### 4.1 Definições e conceitos CSS

#### 4.1.1 Definição

CSS é a abreviação para o termo em inglês *Cascading Style Sheet*, traduzido para o português como folha de estilo em cascata. Neste livro, adotaremos CSS como abreviação e folha de estilo em cascata para o termo por extenso.

A definição mais precisa e simples para folha de estilo encontra-se na homepage das CSS no site do W3C e diz:

“Folha de estilo em cascata é um mecanismo simples para adicionar estilos (por exemplo: fontes, cores, espaçamentos) aos documentos web.”



Já dissemos e lembramos que, em desenvolvimento web, a palavra fonte, que vem do inglês *font*, significa, genericamente, letra. Em tipografia, a palavra fonte significa um conjunto de caracteres tipográficos de determinado tamanho e estilo.

#### 4.1.2 Finalidade

A HTML foi criada para ser uma linguagem exclusivamente de marcação e estruturação de conteúdos. Isso significa que, segundo seus idealizadores, não cabe à HTML fornecer informações ao agente do usuário sobre a apresentação dos elementos. Por exemplo: cores de fontes, tamanhos de textos, posicionamentos e todo o aspecto visual de um documento não devem ser funções da HTML. Cabem às CSS todas as funções de apresentação de um

documento, e essa é sua finalidade maior. Daí a já consagrada frase que resume a dobradinha HTML + CSS:

“HTML para estruturar os conteúdos e CSS para apresentá-los.”

No mês de dezembro de 1996, as CSS1 foram lançadas como uma Recomendação oficial do W3C.

### 4.1.3 Regra CSS

Regra CSS (ou regra de estilo ou de estilização) é a unidade básica de uma folha de estilo. Nessa definição, o termo unidade básica significa a menor porção de código capaz de produzir efeito de estilização. Uma regra CSS é composta de duas partes: o *seletor* e a *declaração*. A declaração compreende uma *propriedade* e um *valor*. A sintaxe para se escrever uma regra CSS é mostrada na figura 4.1.



Figura 4.1 – Sintaxe da regra CSS.

Definição dos componentes de uma regra CSS:

- **Seletor** – É o elemento ou elementos da marcação HTML nos quais a regra CSS será aplicada.
- **Declaração** – Determina os parâmetros de estilização. Compreende a propriedade e o valor.
- **Propriedade** – Define qual será a característica do seletor a ser estilizada.
- **Valor** – É a quantificação ou a qualificação da estilização da propriedade.



A terminologia mostrada é a adotada pelo W3C e recomendo que você a adote também. Evite usar expressões em desacordo com a sintaxe oficial, tais como “atributo CSS” no lugar de *propriedade* CSS, “comando CSS” no lugar de *declaração* CSS e outras.

Uma regra CSS, escrita conforme a sintaxe mostrada, poderá conter várias declarações separadas por ponto e vírgula. Observe o exemplo a seguir:

```
p {  
  color: white;  
  background-color: black;  
  font-size: 14px;  
}
```

Nesse exemplo, identificamos os seguintes componentes da regra CSS:

- **Seletor** – É o elemento p (parágrafo). Seletor para elemento HTML é denominado *seletor tipo*.
- **Declarações** – São três – color: white; background-color: black; font-size: 14px;.
- **Propriedades** – São três – color, que define a cor dos textos, background-color, que define a cor de fundo, e font-size, que define o tamanho da fonte.
- **Valores** – São três – white, que define a cor branca, black, que define a cor preta, e 14px, que define o tamanho da fonte igual a 14px.

Na prática ocorre com frequência a necessidade de estilizar vários seletores com as mesmas declarações de estilos. Nesses casos, a sintaxe CSS prevê que os seletores podem ser agrupados. Para criar um grupamento de seletores, separamos um do outro por vírgula. Ao seletor assim formado chamamos de *grupamento de seletores*, conforme mostrado no exemplo a seguir.

```
h2, p, ul, em { font-size: 18px; }
```

A regra CSS mostrada define um tamanho de fonte igual a 18px para os conteúdos textuais dos elementos h2, p, ul e em que foram agrupados com uso de vírgula.

Existem inúmeras propriedades CSS e respectivos valores, cada propriedade e seu valor destinam-se a estilizar um aspecto do

seletor. Em lugar de descrever as propriedades e os valores em um capítulo, optamos por reunir essas informações no apêndice B deste livro, deixando para descrever as propriedades/valores à medida que forem aparecendo nos textos dos capítulos.



Sempre que surgir uma dúvida sobre uma propriedade CSS ou um valor, o leitor deverá consultar o apêndice B e lá esclarecer a dúvida.

Note que, no exemplo anterior, colocamos uma declaração CSS em cada linha. Fizemos isso apenas para facilitar a leitura do código, pois poderíamos ter colocado todas as declarações na mesma linha, sendo obrigatório apenas o uso de ponto e vírgula para separá-las. O uso de espaços em branco, entre os componentes de uma regra CSS, é facultativo e visa apenas a facilitar a leitura do código. A regra mostrada seguir, em uma linha, tem o mesmo efeito da regra mostrada anteriormente.

```
p { color:white; background-color:black; font-size:14px; } /* Menos legível, não é?
*/
```

O ponto e vírgula colocado na última declaração de uma regra de estilo, ou colocado na declaração única de uma regra, é facultativo, porém é de boa prática usá-lo, como fizemos no exemplo mostrado, pois, se no futuro tivermos que acrescentar mais declarações à regra, estaremos livres da obrigação de colocar o ponto e vírgula e, melhor ainda, não correremos o risco de interromper o funcionamento da regra CSS por termos nos esquecido dele.

Quando o valor de uma propriedade for uma palavra composta, separada por espaços, deve-se usar sinais de aspas duplas (“ ”) ou, alternativamente, de aspas simples (‘ ’), conforme mostrado na regra CSS a seguir que, usando a propriedade font-family, define a família de fontes do texto do parágrafo como sendo Times New Roman:

```
p { font-family: "Times New Roman"; }
```

ou

```
p { font-family: 'Times New Roman'; }
```

Não se usam aspas em palavras compostas separadas por hífen:

```
p { font-family: Sans-serif; }
```

A sintaxe da regra CSS *não* é sensível ao tamanho de caixa da fonte (você pode usar letras minúsculas ou maiúsculas, indiferentemente) e múltiplos espaços são tratados como espaço simples. Usar ou não espaços entre os componentes da regra CSS fica a critério do desenvolvedor. Todas as regras CSS destinadas a criar uma borda de 1px em linha contínua e na cor vermelha para o conteúdo do elemento h1, e mostradas a seguir, são válidas:

```
h1 { border: 1px solid red; }  
h1 {border:1px solid red;}  
h1{ border: 1px solid red;}  
H1 { border: 1px solid RED;}  
h1{ BORDER: 1px solid red; }
```

Tratando-se de linguagem de programação, sempre que houver mais de uma forma válida de escrever o código, o desenvolvedor deve escolher uma delas e adotá-la como seu padrão pessoal. Isso torna o código consistente e facilita a manutenção. Com as folhas de estilo aplica-se essa prática, e você pode escolher qualquer uma das formas mostradas anteriormente ou, mesmo, outras variações para escrever suas folhas de estilo. Contudo, as duas formas de uso mais difundidas são as mostradas no primeiro e no segundo itens do exemplo anterior. A primeira adota um espaço em branco junto ao sinal de chaves ({}). A segunda, um espaço somente para separar o seletor da declaração.

Um componente facultativo, mas de grande utilidade na escrita de folhas de estilo, é o sinal para inserir comentários. À semelhança de qualquer linguagem de programação, existe um sinal próprio para marcar comentários em uma folha de estilo, conforme mostrado nos exemplos a seguir:

- **Comentário em uma linha:**

```
/* Este é um comentário, em folha de estilo, em uma linha */
```

- **Bloco de comentário:**

```
/* Este é um bloco de comentário, em folha de estilo, em linhas  
diferentes contendo muitas informações  
sobre um trecho da folha de estilo */
```

Você começa um comentário com o sinal `/*` e termina com o sinal `*/`.

Um conjunto de regras CSS é denominado *folha de estilo*. O conjunto das regras CSS pode ser escrito no próprio documento no qual as regras serão aplicadas ou em um arquivo externo gravado com a extensão `.css`. Por exemplo: *principal.css*. Nesse caso diz-se que temos uma folha de estilo externa que deverá ser lincada ao documento conforme veremos adiante.

## 4.2 Modelo de formatação CSS

No item 2.6 estudamos o modelo HTML e vimos os elementos nível de bloco e elementos inline. No modelo CSS cada elemento cria um caixa, dentro da qual está inserido o conteúdo do elemento. Se o elemento for nível de bloco, a caixa tem largura igual à largura da linha (ocupa toda a largura da tela) e altura suficiente para acomodar o conteúdo. Se inline, a caixa tem a largura do conteúdo. Caixa em inglês é box, daí o termo Box Model CSS para designar o Modelo CSS de caixas.

### 4.2.1 Container

Também denominado de bloco de conteúdo, trata-se de um box retangular que contém outros boxes chamados de descendentes. A conceituação de container é importante para o entendimento dos mecanismos de cálculo de dimensões e posicionamento de boxes descendentes. Um box descendente não precisa ter necessariamente suas dimensões confinadas aos limites do container, podendo ultrapassá-lo. Esse comportamento é chamado de *overflow*.

### 4.2.2 Elementos nível de bloco e boxes bloco



Elementos nível de bloco são aqueles elementos da marcação HTML formatados visualmente como blocos de conteúdos. São exemplos: os parágrafos `p` e os cabeçalhos `h1` – `h6`.

#### 4.2.2.1 Box bloco anônimo

Considere o código a seguir:

```
<div>  
  Texto qualquer contido diretamente no DIV  
  <p>Texto de um parágrafo contido no DIV</p>  
</div>
```

Os elementos `div` e `p` são níveis de bloco. Qual será o comportamento do texto contido diretamente no `div`? Inline ou de bloco?

Quando inserirmos em um container (no nosso exemplo o `div`) um elemento nível de bloco, estaremos forçando os conteúdos inseridos diretamente no container a se comportarem como box bloco. Tais conteúdos criam os denominados *boxes blocos anônimos*.

#### 4.2.3 Elementos inline e boxes inline

Elementos inline são aqueles elementos da marcação HTML que não formam novos blocos de conteúdo. O conteúdo é distribuído em linha. São exemplos: o elemento para forte ênfase `strong` e as imagens.

##### 4.2.3.1 Box inline anônimo

Considere o código a seguir:

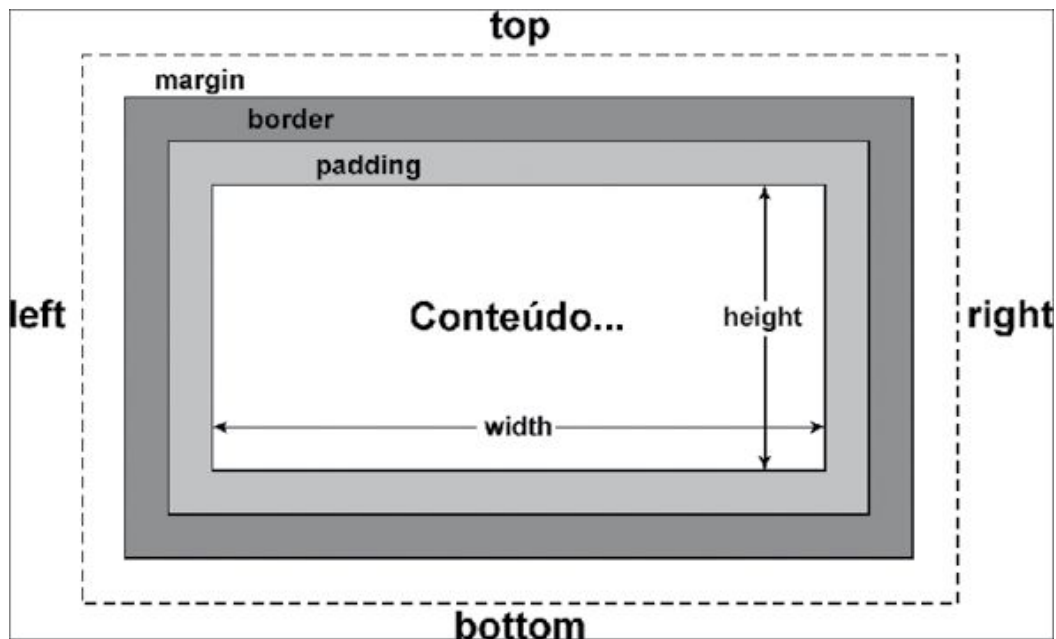
```
<p>Texto <strong>enfaticado</strong> mais texto</p>
```

O elemento `p` criou um box bloco contendo três boxes inline. O box para *enfaticado*, que foi gerado pelo elemento `strong`, e os boxes para *Texto* e *mais texto*, gerados por um elemento nível de bloco, o parágrafo. Estes dois últimos são chamados de *boxes inline anônimos*.

## 4.3 Box Model CSS

O Box Model CSS descreve os boxes criados pelos elementos da marcação HTML. O Box Model CSS é mostrado no diagrama da figura 4.2.

No diagrama destacamos uma área mais interior, denominada área dos conteúdos, cujas dimensões (largura e altura) são definidas pelas propriedades CSS `width` e `height`. Segue uma área chamada de enchimento, cuja espessura é definida pela propriedade CSS `padding`. Em volta do enchimento, temos uma borda cuja espessura, cor e tipo são definidos por `border`. Finalmente, um espaço denominado margem com espessura definida pela propriedade `margin`.



*Figura 4.2 – Diagrama do Box Model CSS.*

A área da margem é sempre transparente. As dimensões da área de conteúdos dependem de uma série de fatores, entre eles a definição explícita de dimensões, natureza do conteúdo e o tipo de conteúdo. A propriedade CSS `background` define o fundo a ser aplicado nas áreas de conteúdos, de enchimento e da borda.

Considere a marcação HTML e a estilização mostradas nos códigos a seguir.

- **HTML**

```
...  
<body>  
  <p>Lorem ipsum dolor sit amet ...</p>  
</body>  
...
```

- **CSS**

```
body {margin:0; padding:0; font-size:20px;}  
  
p {  
  width: 400px;  
  text-align: justify;  
  line-height:1;  
  background: #f6f6f6;  
  margin: 20px 80px 100px 40px;  
  border: solid #ccc;  
  border-width: 10px 30px 50px 15px;  
  padding: 35px 50px 20px 0;  
}
```

Na figura 4.3 mostramos a renderização do código conforme o do Box Model CSS.

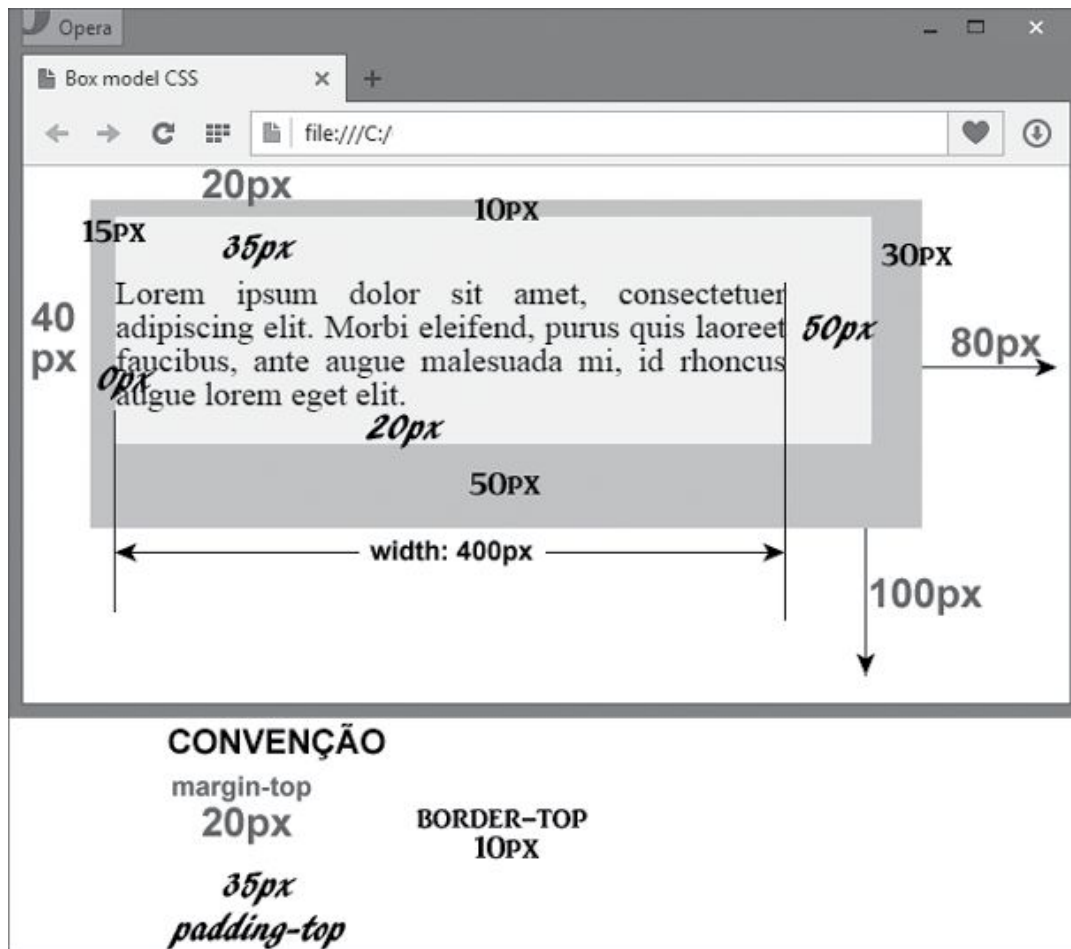


Figura 4.3 – Box Model CSS.

O arquivo *box-model-css.html*, que demonstra o box model CSS, está disponível para consulta online e download na pasta *capitulo4*.

### 4.3.1 Box Model CSS modificado

Se analisarmos detalhadamente o Box Model CSS, vamos concluir que o modelo é contraintuitivo, pois, uma vez que a propriedade *width* refere-se à largura do *conteúdo* do box, o que ocorre é que a largura total do box acaba sendo variável em relação à propriedade *width*, já que a ela são acrescidas as larguras de padding e border (que são variáveis) para se obter a largura total. Convém ressaltar que o mesmo ocorre com a altura do box.

As CSS3 criaram uma nova propriedade CSS denominada *box-sizing*, capaz de alterar esse comportamento padrão do Box Model,

fazendo com que as larguras de padding e border sejam incluídas na largura width declarada por regras CSS.

Para modificar o comportamento do Box Model para todos os boxes, inclua na folha de estilo do seu projeto a seguinte regra CSS.

```
html { box-sizing: border-box; }  
*, *:before, *:after { box-sizing: inherit; }
```

Nota: O símbolo \* é chamado de seletor universal e casa com *todos* os elementos da marcação.

A figura 4.4 mostra a renderização do mesmo box do exemplo anterior com a inclusão da regra modificadora do Box Model.

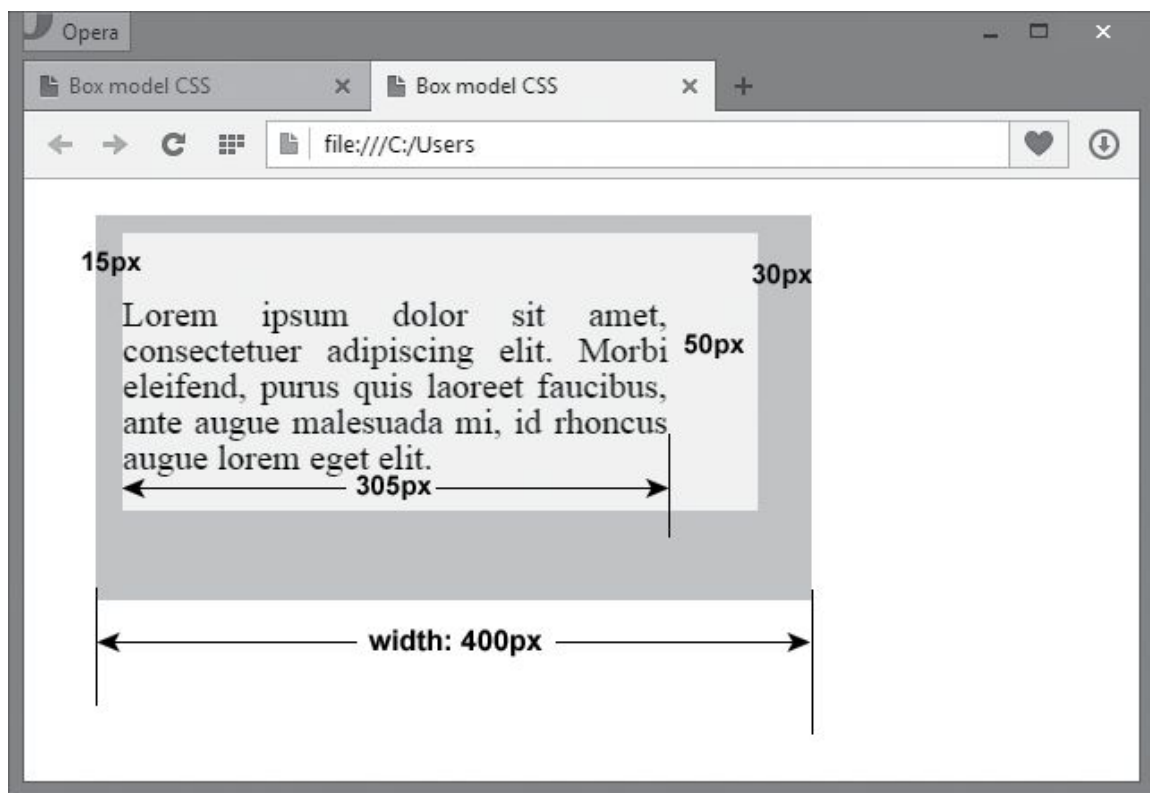


Figura 4.4 – Box Model CSS modificado.

Notar que a largura do conteúdo que no Box Model tradicional é igual ao valor declarado para width (400px), no caso do Box Model modificado passa a ser igual ao valor declarado para width (400px) menos as larguras de border (15px + 30px = 45px) e padding (0 + 50px), isto é  $400 \text{ px} - 15 \text{ px} - 30 \text{ px} - 50 \text{ px} = 305 \text{ px}$ .



Em seus projetos, a menos que haja uma razão contrária muito forte, sempre use o Box Model modificado, pois resulta em um modelo CSS muito mais intuitivo e conseqüentemente mais fácil para projetar e manter.

O arquivo *box-model-css-box-sizing.html*, que demonstra o box model CSS modificado, está disponível para consulta online e download na pasta *capitulo4*.

## 4.3.2 Propriedades CSS para o Box Model

Veremos, a seguir, as regras para aplicação das propriedades CSS `margin`, `border` e `padding`. O box criado no modelo é um quadrilátero em que cada um dos lados é identificado, conforme mostra a figura 4.2, por um termo em inglês designativo da posição do lado. Os lados superior, direito, inferior e esquerdo são identificados por `top`, `right`, `bottom` e `left`, respectivamente. Procure memorizar a ordem em que estão nomeados os lados. Para facilitar a memorização, lembre-se de que começamos com o lado superior e seguimos no sentido horário. Você constatará, logo adiante, a importância de saber essa ordem.

### 4.3.2.1 Propriedade `margin`

A propriedade `margin` define as dimensões das margens de um box. Você pode definir cada uma das quatro margens do box com valores diferentes, conforme mostrado a seguir:

```
margin-top: 20px;  
margin-right: 30px;  
margin-bottom: 5px;  
margin-left: 10px;
```

ou usar a sintaxe abreviada:

```
margin: 20px 30px 5px 10px; /* na ordem descrita anteriormente */
```

As propriedades CSS `background`, `font`, `margin`, `border`, `border-top`, `border-right`, `border-bottom`, `border-left`, `border-width`, `border-color`, `border-style`, `transition`, `transform`, `padding`, `list-style`, `border-radius` e `outline` admitem a *sintaxe abreviada*, a qual consiste em declarar uma lista de valores

separados por um espaço, conforme mostrado. Para as propriedades em que se definem valores para os quatro lados de um box é válida a sintaxe abreviada, e a ordem em que os valores são escritos na lista corresponde aos lados superior, direito, inferior e esquerdo, respectivamente.

A declaração abreviada admite ainda as seguintes variações:

- Se as quatro margens são iguais, declare um valor:

```
margin: 20px; /* margem de 20px nos quatro lados */
```

- Se as quatro margens são iguais duas a duas, declare dois valores:

```
margin: 15px 10px; /* margens superior e inferior de 15px e direita e esquerda de 10px */
```

- Declare três valores:

```
margin: 20px 10px 15px; /* margem superior de 20px margens direita e esquerda de 10px e margem inferior de 15px */
```

#### **4.3.2.2 Propriedade padding**

A propriedade padding define as dimensões do enchimento (ou espessura) entre o conteúdo e a borda. Você pode definir cada um dos quatro enchimentos do box com valores diferentes, conforme mostrado a seguir:

```
padding-top: 20px;  
padding-right: 30px;  
padding-bottom: 5px;  
padding-left: 10px;
```

ou usar a sintaxe abreviada:

```
padding: 20px 30px 5px 10px;
```

A sintaxe é idêntica àquela para a propriedade margin, e a declaração abreviada também admite as variações mostradas para tal propriedade.

#### **4.3.2.3 Propriedade border**

A propriedade `border` define a espessura, o estilo e a cor das bordas do box. Cada uma dessas três características da borda pode ser declarada separadamente para cada lado do box, conforme veremos em seguida.

### ***border-width***

Define a espessura da borda.

```
border-top-width: 2px;  
border-right-width: 3px;  
border-bottom-width: 4px  
border-left-width: 1px;
```

ou usar a sintaxe abreviada:

```
border-width: 2px 3px 4px 1px;
```

Você define a espessura da borda declarando uma medida CSS de comprimento ou usando as palavras-chave `thin`, `medium` e `thick`, obtendo as espessuras de bordas fina, média e grossa.

### ***border-color***

Define a cor da borda.

```
border-top-color: red;  
border-right-color: yellow;  
border-bottom-color: black;  
border-left-color: cyan;
```

ou usar a sintaxe abreviada:

```
border-color: red yellow black cyan;
```

Para declarar a cor da borda, você usa um dos valores CSS para cores ou o valor `transparent`.

### ***border-style***

Define o estilo da borda

```
border-top-style: solid;  
border-right-style: ridge;  
border-bottom-style: double;
```



`border-left-style: inset;`

ou usar a sintaxe abreviada:

`border-style: solid ridge double inset;`

Você pode aplicar nove estilos para bordas ou declarar o valor `none` para definir ausência de bordas. Pode parecer estranha e inútil a declaração `none`, mas, na prática, é muito usada para retirar bordas colocadas por padrão ou declaradas anteriormente em elementos específicos da marcação ou, ainda, para retirar a borda-padrão colocada em imagens que são links.

Estilo	Descrição sumária
<code>none</code>	Define espessura 0 para a borda.
<code>hidden</code>	O mesmo efeito de <code>none</code> , mas com precedência na resolução de bordas conflitantes.
<code>dotted</code>	Borda pontilhada.
<code>dashed</code>	Borda tracejada.
<code>solid</code>	Borda contínua ou sólida.
<code>double</code>	Borda constituída de duas linhas contínuas. A soma das espessuras das linhas com a do espaço que as separa e é igual ao valor de <code>border-width</code> .
<code>groove</code>	Borda com aparência entalhada.
<code>ridge</code>	Borda com aparência de ressalto.
<code>inset</code>	Borda em baixo-relevo.
<code>outset</code>	Borda em alto-relevo.

## ***border***

Define abreviadamente a borda

`border-top: 1px solid red;`

`border-right: 2px ridge yellow;`

`border-bottom: 4px double black;`

`border-left: 6px inset cyan;`

ou usar a sintaxe abreviada para bordas que sejam iguais nos quatros lados:

`border: 2px dotted white;`

Ao usar a declaração abreviada `border`, não é obrigatório declarar os três valores, portanto as regras CSS mostradas a seguir são válidas:

```
border: 5px;  
border: dotted;  
border: red;  
border: 2px double;  
border: solid red;  
border: 4px blue;
```

Os valores não declarados são interpretados pelo navegador como sendo o valor inicial da propriedade. Os valores iniciais das três propriedades são:

```
border-width: medium;  
border-style: none;  
border-color: /* o mesmo valor da propriedade color do elemento  
                em que se aplica a borda. */
```

Como o valor inicial para o estilo da borda é `none` (nenhum), concluímos que declarações abreviadas que omitem o estilo da borda, como as mostradas a seguir, isoladamente, não produzem nenhum efeito.

```
border: 5px;  
border: red;  
border: 4px blue;
```

Porém, se combinarmos aquelas declarações com outras, obteremos um método de estilização interessante. As regras CSS, a seguir, esclarecem o método proposto:

```
border: 4px blue;  
border-style: solid dotted groove double;
```

ou, ainda:

```
border: solid blue;  
border-width: 5px 8px 10px 2px;
```

As combinações de espessuras de bordas podem resultar em criações gráficas bastante interessantes. Recomendo uma visita à página demonstrativa de algumas formas construídas com a

propriedade border no endereço: <http://kwz.me/g7>.

## 4.4 Categorias de valores CSS

Para aplicar uma regra CSS, o agente de usuário (por exemplo: o navegador) identifica o valor da propriedade e renderiza o elemento, casado pelo seletor (o alvo da regra de estilo), de acordo com o valor. Observe as regras CSS que se seguem:

```
p { font-family: Arial, Sans-serif; } /* estiliza p com fonte na família
                                     especificada (valor) */
p { width: 400px; } /* estiliza p com largura 400px; */
p { font-size: 120%; } /* estiliza p com tamanho de fonte 1.2 vezes
                       o valor de referência; */
p { background-color: red; } /* estiliza p com fundo na cor vermelha */
p { height: 2em; } /* estiliza p com altura 2 vezes o valor de referência; */
```

Observe que alguns valores são absolutos (é aquele valor e pronto!) e outros relativos (dependem de um valor de referência), tais como as medidas CSS em porcentagem e em.

Os valores CSS podem ser agrupados em oito categorias, conforme descrito nos subitens a seguir.

### 4.4.1 Palavra-chave

Um valor CSS é do tipo palavra-chave quando expresso por uma string predefinida nas especificações. O exemplo típico para esse caso é quando usamos palavra-chave para definir cores, conforme mostra o exemplo a seguir:

```
p {
  color: red;
  background-color: acqua;
  border-color: teal;
}
```

Os valores red, acqua e teal são palavras-chave para cores, previstas nas especificações para as CSS3. Consulte a lista completa das palavras-chave para cores no apêndice C.

Outros exemplos de palavra-chave para expressar valores CSS são:

Palavra-chave	Utilizada
inherit	para definir uma propriedade que deverá ser herdada. (Herança CSS é uma característica das CSS que estudaremos adiante).
collapse	para definir bordas de células de tabelas que devam ser unidas.
italic	para definir fonte em itálico.
uppercase	para definir texto em caixa-alta.

## 4.4.2 Número

Um valor CSS é do tipo número quando expresso por um número inteiro ou por um número real. A especificação adota a sintaxe `<integer>` para designar números inteiros e `<number>` para designar números reais. Números inteiros são aqueles formados por um ou mais algarismos de 0 a 9, e números reais são formados por um ou mais algarismos de 0 a 9, seguidos de uma vírgula e seguidos de um ou mais algarismos de 0 a 9.

Tal como na matemática, em CSS, números podem ser precedidos dos sinais “+” e “-” para indicar o sinal positivo ou negativo do número. Esses dois caracteres, quando usados, passam a fazer parte do valor CSS.

## 4.4.3 Número não negativo

Muitas das propriedades CSS que admitem um valor do tipo número fazem restrição quanto à faixa de números admitidos. Por exemplo: há propriedades CSS, tal como a propriedade `width`, destinada a definir a largura de um elemento, que não admitem números negativos.

Nesses casos, a sintaxe prevista nas especificações é `<non-negative-integer>` para números inteiros não negativos e `<non-negative-number>` para números reais não negativos. Convém notar que para esses casos a restrição não se aplica ao número zero, que é um

número não negativo.

## 4.4.4 Número com unidade de medida

Os valores CSS, quando expressos com números seguidos por uma unidade de medida, são classificados em cinco categorias, conforme descritas nos subtítulos a seguir.

### ***Comprimento***

Comprimento refere-se às medidas horizontal e vertical. A sintaxe prevista nas especificações para designar essa categoria é `<length>`. Um valor CSS que usa uma medida de comprimento é formado por um número seguido da abrevitura para uma unidade de medida. Por exemplo: 14px, 12em, 18pt. Se o número é zero, a unidade de medida é opcional. Recomendo não usar unidade de medida nesses casos, pois não há sentido em 0px, 0em, 0cm, pois zero é zero, independentemente de unidades. A única exceção a essa grafia é na sintaxe de definição da posição do primeiro frame de uma animação CSS. Nesse caso, a definição do frame deve ser 0%, sendo obrigatório constar o sinal de porcentagem depois do valor zero.

### ***Medida relativa***

Unidade de medida relativa é aquela cujo valor é determinado em função de outro valor de uma propriedade que lhe serve de referência. Definir medidas relativas em uma folha de estilo facilita o escalonamento e possibilita servi-la para diferentes tipos de mídia, por exemplo: para uma tela de computador ou para uma tela de telefone celular.

As unidades de medidas relativas nas CSS3 são mostradas no quadro a seguir:

Unidade	Relativa
em	à font-size do elemento (ou do elemento-pai). Elemento-pai é o elemento no qual um elemento está contido.

Unidade	Relativa
ex	ao valor x-height (altura da letra x minúscula) da fonte do elemento.
px	ao dispositivo gráfico (tela, por exemplo) de renderização.
gd	ao grid definido pelo “layout-grid”, descrito no Módulo Texto da CSS3.
rem	à font-size do elemento raiz do documento (html).
vw	à largura da viewport (área de renderização).
vh	à altura da viewport.
vm	à largura ou altura da viewport (a menor das duas).
ch	à largura do número “0”, renderizado de acordo com font-size. Se não existir “0” na fonte especificada, a largura média dos caracteres deverá ser usada.

### ***Medida absoluta***

Unidade de medida absoluta é aquela cujo valor é determinado e fixo. Essas unidades são úteis para uso quando se conhece as dimensões físicas da mídia (tela, impressora etc.) para a qual a folha de estilo será servida.

As unidades de medidas absolutas nas CSS3 são mostradas no quadro a seguir:

Unidade	Descrição
in	polegada; 1 polegada = 2,54 cm
cm	centímetro
mm	milímetro
pt	ponto; 1 ponto = 1/72 polegada
pc	pica; 1 pica = 12 pontos

### ***Porcentagem***

O formato para definir um valor CSS em porcentagem é um número imediatamente seguido pelo sinal %. A sintaxe prevista nas especificações para designar essa categoria é <percentage>. Porcentagens são valores dependentes de outro valor, por exemplo: de um valor do tipo <length>.

As propriedades CSS que admitem valores em porcentagem

também definem qual o valor de referência a considerar para cálculo da porcentagem. O valor de referência pode ser o valor de outra propriedade do mesmo elemento ao qual a porcentagem foi aplicada, o de um elemento ancestral (no qual o elemento está contido) ou o valor de um contexto de formatação, como a largura de um bloco de conteúdo.

### **Ângulo**

O formato para definir um valor CSS em medida angular é um número imediatamente seguido por uma unidade de medida angular. A sintaxe prevista nas especificações para designar essa categoria é `<angle>`. As unidades de medida angular em CSS são:

Unidade	Descrição
deg	Graus
grad	Grados
rad	Radianos
turn	Volta

Valores expressos em medidas angulares são usados, por exemplo, para definir propriedades destinadas à mídia speech (mídia falada, tal como leitores de tela) ou às transformações previstas nas CSS3.

## **4.4.5 Número não negativo com unidade de medida**

Valores CSS expressos com números não negativos com unidade de medida são classificados em duas categorias, conforme descritas nos subtítulos a seguir.

### **Hora**

O formato para definir um valor CSS em medida de hora é um número imediatamente seguido por uma unidade identificadora de tempo em segundos. A sintaxe prevista nas especificações para designar essa categoria é `<time>`. As unidades de medida de tempo

em CSS são:

Unidade	Descrição
ms	Milissegundo
s	Segundo

Valores expressos em medidas de tempo são usados, por exemplo, para definir propriedades destinadas à mídia speech (mídia falada, tal como leitores de tela) ou duração de animações e transições previstas nas CSS3.

### ***Frequência***

O formato para definir um valor CSS em medida de frequência é um número imediatamente seguido por uma unidade identificadora de frequência em hertz. A sintaxe prevista nas especificações para designar essa categoria é <frequency>. As unidades de medida de frequência em CSS são:

Unidade	Descrição
Hz	Hertz
kHz	Quilohertz

Valores expressos em medidas de frequência são usados, por exemplo, para definir propriedades destinadas à mídia speech (mídia falada, tal como leitores de tela).

## **4.4.6 String**

Valores CSS expressos com strings devem ser grafados com aspas simples (') ou duplas ("). Sendo da mesma grafia, uma não pode ocorrer dentro de outra, a menos que seja escapada com uma barra invertida (\).

Uma string não pode conter uma quebra de linha, a menos que se use o caractere \A, que representa uma nova linha em CSS.

Para fins de legibilidade, é possível quebrar uma string em substrings com uso do caractere barra invertida (\).

Observe os exemplos a seguir que esclarecem essas sintaxes:



```
"Esta é uma 'string'." /* aspas simples dentro de aspas duplas */  
'Esta é uma "string".' /* aspas duplas dentro de aspas simples */  
"Esta é uma \"string\"." /* aspas duplas escapadas dentro de aspas duplas */  
'Esta é uma \'string\'' /* aspas simples escapadas dentro de aspas simples */
```

```
"Esta string está na primeira linha. \A E esta na segunda"
```

```
"Esta é uma string longa\  
que foi quebrada para\  
fins de legibilidade."
```

## 4.4.7 Notação funcional

Valores CSS podem ser expressos por uma função, e nesses casos são classificados como valores em notação funcional. Em CSS3, valores funcionais são usados para definir cores, atributos e URIs (Uniform Resource Identifier – endereço web).

A sintaxe para escrita de um valor funcional é: nome da função seguido de uma lista de argumentos entre parênteses. Considere os exemplos mostrados a seguir:

```
p { background-color: rgb(255, 0, 0); }  
img { margin-top: attr(height, px); }  
div { background-image: url(http://maujor.com/avatar.gif); }
```

Nos três exemplos mostrados, os valores das propriedades CSS, em destaque no código, são do tipo valor funcional, e as respectivas funções CSS `rgb`, `attr` e `url` retornam um valor a ser aplicado nas propriedades definidas para os seletores (elementos) `p`, `img` e `div`, respectivamente.

## 4.4.8 Casos especiais

Valores CSS que não se enquadram em nenhuma das sete categorias anteriores pertencem a uma categoria denominada “casos especiais”. Os valores CSS enquadrados nessa categoria são os valores para definição de famílias de fontes e valores para definição de cores em sintaxe hexadecimal.

Observe os exemplos a seguir que mostram o uso de valores pertencentes a essa categoria:

```
p { background-color: #f00; } ou p { background-color: #ff0000; }  
h1 { font-family: Arial, Verdana, Sans-serif; }
```

## 4.5 Cores CSS

Optamos por separar os valores CSS para definir cores em um item próprio, pois são várias as formas de definir cores em CSS. Os valores possíveis são: um valor hexadecimal, as palavras-chave `transparent`, `currentColor` e outras palavras-chave que são o nome de algumas cores (ver apêndice C), valores RGB, RGBA, HSL e HSLA.

Observe a seguir declarações CSS típicas com uso de cada um desses valores para definição da propriedade `color`:

```
color: #ff0000; /* hexadecimal minúsculas */  
color: #FF0000; /* hexadecimal maiúsculas */  
color: #f00; /* hexadecimal abreviada */  
color: pink; /* palavra-chave */  
color: rgb(255, 200, 32); /* RGB inteiros */  
color: rgb(100%, 26%, 47%); /* RGB porcentagem */  
color: rgba(200, 100, 57, 0.4); /* RGB inteiros com opacidade */  
color: rgba(90%, 86%, 37%, 0.6); /* RGB porcentagem com opacidade */  
color: hsl(120, 75%, 50%) /* HSL */  
color: hsla(120, 75%, 50%, 0.8) /* HSL com opacidade */  
color: transparent; /* palavra-chave */  
color: currentColor; /* criada nas CSS3 */
```

### ***Hexadecimal e RGB***

A representação de uma cor em notação hexadecimal começa com um sinal de tralha (#) seguido de seis números hexadecimais (os números hexadecimais são compostos de combinações de: 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C, D, E, F). A grafia dos números hexadecimais é insensível ao tamanho da caixa, isto é, letras maiúsculas e minúsculas são igualmente válidas.

Nota: Segue uma explicação técnica de como transformar uma cor hexadecimal em cor RGB. A explicação tem fins puramente

didáticos e o leitor não precisa saber como se faz a transformação. Na prática existem centenas de ferramentas online ou de complementos para o navegador que fazem a transformação. Quando necessário, use as ferramentas de transformação. Faça uma busca no Google por “HEX to RGB” ou procure uma extensão de transformação para seu navegador preferido.

O valor de uma cor hexadecimal é determinado decompondo-se o número hexadecimal que representa a cor em três grupos de dois números hexadecimais.

Cada um dos três grupos representa as cores vermelha, verde e azul, respectivamente, que entram na composição final da cor. A abreviatura em inglês para vermelho, verde, azul é RGB (red, green, blue).

Cada grupo de dois números hexadecimais é representado por 1 *byte*, que, por sua vez, é capaz de armazenar um número na faixa de 0 a 255 (em notação decimal) ou 00 a FF (em notação hexadecimal) ou ainda uma porcentagem na faixa de 0% a 100%.

As cores vermelha, verde e azul (RGB), que se combinam para formar a cor final, são também chamadas de *componentes* da cor. Cada um dos componentes da cor entra com uma *intensidade* que varia da cor preta, representada pelo decimal 0, hexadecimal 00 ou 0 em porcentagem, até a cor branca, representada pelo número decimal 255, hexadecimal FF ou 100 em porcentagem.

Note que na faixa de 0 a 255 existem 256 números. Assim, o total possível de combinações de números para formação de cores é igual a  $256 \times 256 \times 256 = 16.777.216$ , o que significa que o uso dessa representação de cores permite que se defina mais de 16 milhões de cores.

Por exemplo: a cor hexadecimal #2E8B57 é obtida pela mistura da cor vermelha em uma quantidade 2E de vermelho (R), 8B de verde (G) e 57 de azul (B).

Transformando os três números hexadecimais em decimais,

temos:



Nos cálculos mostrados a seguir, o sinal \* representa multiplicação, e \*\*, potenciação.

$2E = 2 * 16^{** 1} + 14 * 16^{** 0} = 32 + 14 = 46$  (Nota: E hexadecimal é igual a 14 decimal)

$8B = 8 * 16^{** 1} + 11 * 16^{** 0} = 128 + 11 = 139$  (Nota: B hexadecimal é igual a 11 decimal)

$57 = 5 * 16^{** 1} + 7 * 16^{** 0} = 80 + 7 = 87$

Transformando os três números decimais em porcentagem, temos:

$46 * 100 / 255 = 18\%$

$139 * 100 / 255 = 54,5\%$

$87 * 100 / 255 = 34\%$

Assim, a cor #2E8B57 é resultante de uma mistura cujo componente de cor vermelha entra com uma intensidade de 18%, de cor verde 54,5% e de cor azul 34%.

As seguintes sintaxes para declaração dessa cor são válidas (representam a mesma cor):

color: #2e8b57; /\* hexadecimal minúsculas \*/

color: rgb(46, 139, 87); /\* RGB inteiros \*/

color: rgb(18%, 54.5%, 34%); /\* RGB porcentagem \*/

As CSS admitem que a declaração de cores com uso de números hexadecimais seja abreviada quando se tratar de cores nas quais cada um dos três *componentes* seja representado por dígitos iguais, ou seja, números hexadecimais no formato #XXYYZZ. Nesses casos, a abreviatura se faz para #XYZ, ficando subentendido que cada um dos três dígitos da forma abreviada é dobrado. Observe a seguir alguns exemplos.

color: #ff9900; = color: #f90;

color: #55ddaa; = color: #5DA;

color: #FFEECC; = color: #fec;

color: #88BB43; /\* não é possível abreviar \*/

color: #333344; = color: #334;

Você pode usar maiúsculas, minúsculas e até mesmo uma combinação delas para definir cor hexadecimal, bem como adotar ou não a sintaxe abreviada. A maioria dos autores usa minúscula e adota a sintaxe abreviada, mas isso é uma questão de gosto pessoal. Adote uma maneira e siga com ela em todos os seus trabalhos de desenvolvimento.

Computadores e monitores modernos suportam 24 bits de cores e são capazes de renderizar consistentemente as mais de 16 milhões de cores possíveis de se representar com a sintaxe mostrada. Equipamentos antigos suportam 8 bits de cores e estão restritos à renderização consistente de 256 cores, apenas. O conjunto dessas 256 cores recebeu o nome de *cores seguras para a web* e é resultante da combinação de R (red), G (green) e B (blue) com intensidades de 0%, 20%, 40%, 80% e 100%.

Observe a seguir algumas declarações de *cores seguras para a web*:

```
color: #33cc66; /* tom de verde */
color: rgb(255, 204, 102); /* tom de amarelo */
color: rgb(40%, 20%, 100%); /* tom de azul */
color: #cc9966; /* marrom claro */
color: rgb(204, 153, 102); /* marrom claro */
color: rgb(80%, 60%, 40%); /* marrom claro */
```

As três últimas cores declaradas resultam em um mesmo tom marrom-claro, pois as declarações com uso de hexadecimal e RGB são equivalentes.

### ***transparent***

O valor `transparent` para cores foi criado pelas especificações para as CSS1 e era válido apenas para a propriedade `background-color`. As CSS2 estenderam a validade desse valor para a propriedade `border-color`, e as CSS3 preveem validade do valor para qualquer propriedade CSS que admita declaração de cor. Como o próprio nome sugere, esse valor destina-se a tornar transparente a propriedade cuja cor for declarada com ele.

As CSS1 e CSS2 definiram o valor inicial da cor da borda de um elemento como sendo igual à cor do próprio elemento. Por exemplo: se definirmos uma espessura e um estilo para a borda de um elemento h1 e omitirmos a cor da borda, ela será a cor definida para o elemento h1. Observe o exemplo a seguir:

```
h1 {  
  color: red;  
  border: 2px solid;  
}
```

A borda do elemento h1 será na cor vermelha.

### **Palavra-chave**

Outra forma de declarar cores é com uso de valores expressos por palavras-chave. As especificações para as CSS3 preveem três grupos de palavras-chave.

- Palavras-chave para definir cores básicas.

Esse grupo é constituído de 16 palavras-chave, a saber: aqua, black, blue, fuchsia, gray, green, lime, maroon, navy, olive, purple, red, silver, teal, white e yellow.



Consulte uma tabela mostrando as 16 cores básicas em <http://www.w3.org/TR/css3-color/#html4> (<http://kwz.me/gx>).

- Palavras-chave para definir cores estendidas.

Esse grupo é constituído de 147 palavras-chave (incluindo as 16 para cores básicas), como: brown, coral, cyan, dimgray, indigo, khaki, mintcream, mistyrose, sienna, turquoise, tomato e whitesmoke.



Consulte uma tabela mostrando as 147 no apêndice C deste livro. Para mais detalhes sobre cores estendidas, consulte [http://en.wikipedia.org/wiki/X11\\_color\\_names](http://en.wikipedia.org/wiki/X11_color_names) (<http://kwz.me/gv>).

- Palavras-chave para definir cores de acordo com o sistema operacional.

Esse grupo é constituído de 28 palavras-chave que se referem a componentes da interface gráfica do sistema operacional do usuário. Esses valores são de uso prático bastante limitado e não faremos considerações detalhadas sobre eles. Para informações, consulte <http://www.w3.org/TR/css3-color/#css-system> (<http://kwz.me/gw>).

As recomendações de acessibilidade ao conteúdo web (WCAG2.0), em relação a cores, dizem o seguinte:

1.4.1 Utilização da cor: a cor não é utilizada como o único meio visual de transmitir informações, indicar uma ação, pedir uma resposta ou distinguir um elemento visual (nível A).

O item 1.4.1 das WCAG2.0 dispensa mais explicações, portanto é errado criar uma marcação como mostrada a seguir:

<p>Se você é a favor clique no link verde, mas se é contra clique no link vermelho</p>

O trecho a seguir, a respeito de cores, também foi transcrito das WCAG2.0:

1.4.3 Contraste (mínimo): a apresentação visual de texto e imagens de texto tem uma relação de contraste de, no mínimo, 4.5:1, exceto para o seguinte: (nível AA).

Texto ampliado: texto ampliado e as imagens compostas por texto ampliado têm uma relação de contraste de, no mínimo, 3:1.

Texto em plano secundário: o texto ou as imagens de texto que fazem parte de um componente de interface de usuário inativo, que são meramente decorativos, que não estão visíveis para ninguém ou que são parte de uma imagem que inclui outro conteúdo visual significativo, não têm requisito de contraste.

Logotipos: o texto que faz parte de um logotipo ou uma marca comercial não tem requisito de contraste.

Assim, ao definir cores para o plano de frente e de fundo, certifique-se de que o contraste entre as cores está dentro dos limites mínimos estabelecidos pela WCAG2.0.

Existem ferramentas para verificar o contraste de cores. Uma ferramenta online para verificar o contraste de cores em uma página web pode ser encontrada em <http://www.accesskeys.org/tools/color->

*contrast.html* (<http://kwz.me/gz>). Opcionalmente, você pode fazer download da ferramenta para análise de contraste de cores hospedada em <http://www.visionaustralia.org.au/info.aspx?page=628> (<http://kwz.me/gA>) e que se destina à instalação em sistema operacional Windows.

### ***currentColor***

Esse valor declara que a cor a ser aplicada a uma determinada propriedade CSS em um elemento é igual à cor declarada para a propriedade `color` de seu elemento ancestral mais próximo. O exemplo a seguir esclarece a aplicação desse valor, considerando elementos `p` como elementos-filho de `div`.

```
div { color: red; }  
p { border: 1px solid currentColor; }
```

Todos os parágrafos contidos no `div` terão uma borda na cor vermelha (`red`). O uso desse valor para declarar cor não é muito comum, mas dependendo do contexto poderá tornar-se bastante útil. Note que, no exemplo mostrado, basta alterar a cor declarada para o `div` para que seja alterada a cor da borda de todos os parágrafos nele contidos.

### ***HSL***

As CSS3 criaram o valor HSL para definir cores. A declaração de cores com uso de HSL (hue, saturation, lightness), não existia nas CSS2.1. Ela permite que você declare as cores com uso de três parâmetros:

Hue = tom; saturation = saturação; e lightness = luminosidade.

A sintaxe geral para a declaração com uso desse valor é conforme a mostrada no exemplo a seguir:

```
color: hsl(120, 75%, 50%);
```

O primeiro valor é para o tom (hue) da cor. O seu valor é um número que representa a medida de um ângulo (varia de 0 grau a 360 graus) apontando para um tom de cor na roda de cores.



Observe os valores do ângulo e os respectivos tons de cor:

Ângulo	Tons de cor
0	vermelho
60	amarelo
120	verde
180	ciano
240	azul
300	púrpura
360	vermelho

Exemplos de declaração de cor com uso de valores HSL:

```
color: hsl(0, 100%, 50%) /* cor vermelha */  
color: hsl(120, 100%, 50%) /* cor verde */  
color: hsl(120, 100%, 25%) /* cor verde-escura */  
color: hsl(120, 100%, 75%) /* cor verde-clara */  
color: hsl(120, 75%, 75%) /* cor verde pastel */
```

Como você já deve ter notado, o uso de HSL para declarar cores é semelhante ao uso de RGB. A vantagem do uso de HSL sobre RGB é que HSL proporciona uma maneira mais simples de se obter variações de uma mesma cor.



O segundo (saturation) e o terceiro (lightness) parâmetros dessa declaração são expressos em porcentagem, e a sintaxe para a declaração determina que o sinal de porcentagem é obrigatório mesmo se o valor for 0 (zero). Essa exigência pode parecer absurda, uma vez que zero é zero independentemente da unidade de medida, mas, nesse caso, é assim que a sintaxe exige.

## HSLA

As CSS3 criaram uma extensão do valor HSL denominada HSLA para definição de cores. Esse valor acrescenta um quarto parâmetro à declaração (hue, saturation, lightness, alpha). O quarto parâmetro, denominado alpha, define a opacidade (ou transparência) da cor.

A sintaxe geral para a declaração com uso desse valor é conforme mostrada no exemplo a seguir:

```
color: hsla(120, 75%, 50%, 0.6);
```

Os três primeiros parâmetros têm o mesmo significado de quando se usa a declaração HSL, e o quarto é um valor compreendido entre 0 e 1. O valor 0 representa transparência total, e o valor 1, opacidade total. Assim, um valor igual a 0.6 significa 60% opaco ou 40% transparente.

Para verificar o resultado da aplicação de cores com uso desse valor, visite a página web em <http://www.maujor.com/tutorial/css3-modulo-para-cores.php> (<http://kwz.me/gF>). Nela, sob o título “Declaração com HSLA”, você encontra um quadro interativo, no qual poderá inserir diferentes valores para HSLA e verificar o resultado.

## **RGBA**

As CSS3 criaram uma extensão do valor RGB denominada RGBA para definição de cores. Esse valor acrescenta um quarto parâmetro à declaração (red, green, blue, alpha). O quarto parâmetro, denominado alpha, define a opacidade (ou transparência) da cor.

A sintaxe geral para a declaração com uso desse valor é conforme a mostrada no exemplo a seguir:

```
color: rgba(255, 204, 102, 0.4);  
color: rgba(40%, 20%, 100%, 0.7);
```

Os três primeiros parâmetros têm o mesmo significado de quando se usa a declaração RGB, e o quarto é um valor compreendido entre 0 e 1. O valor 0 representa transparência total, e o valor 1, opacidade total. Assim, um valor igual a 0.7 significa 70% opaco ou 30% transparente.

Para verificar o resultado da aplicação de cores com uso desse valor, visite a página web em <http://www.maujor.com/tutorial/css3-modulo-para-cores.php> (<http://kwz.me/gF>). Nela, sob o título “Declaração com RGBA”, você encontra um quadro interativo, no qual poderá inserir diferentes valores para RGBA e verificar o resultado.

## **Cores do SO**

As CSS2 preveem um mecanismo capaz de definir cores com base no sistema operacional do usuário. Esse mecanismo é usado para estilizar elementos e controles utilizados em Interfaces de Usuário (UI – User Interface) de forma que assumam uma cor semelhante às cores usadas em elementos e controles de interface do sistema operacional do usuário. Por exemplo: os valores `ButtonFace` e `ButtonText` destinam-se a simular as cores usadas na face e no texto dos botões da interface do Windows ou do Mac, por exemplo, caso o visitante da página esteja usando Windows ou Mac.

As CSS2 previram 28 valores para simular as cores do sistema operacional. A título de informação, citamos alguns desses valores: `ActiveCaption`, `Highlight`, `InfoText`, `Scroll`, `ThreeDShadow` etc.

Por exemplo: para estilizar a cor de face de um botão com a mesma cor de face dos botões do sistema operacional do usuário, usamos a seguinte regra CSS:

```
button { background-color: ButtonFace; }
```

Esse mecanismo foi colocado em desuso pelas CSS3. Por questões de retrocompatibilidade, os navegadores deverão continuar oferecendo suporte para ele; contudo, em novos projetos, use a alternativa prevista nas CSS3.

Para substituir a definição de cores com base no sistema operacional do usuário, que foi colocada em desuso, as CSS3 criaram uma propriedade nova denominada `appearance`, que se destina a definir a aparência de um elemento com base no sistema operacional utilizado pelo usuário.

Os valores possíveis para essa propriedade são: `icon`, `window`, `desktop`, `workspace`, `document`, `tooltip`, `status-bar`, `dialog`, `message-box`, `button`, `caption`, `small-caption`, `push-button`, `hyperlink`, `radio-button`, `checkbox`, `menu-item`, `tab`, `menu`, `menubar`, `pull-down-menu`, `pop-up-menu`, `list-menu`, `radio-group`, `checkbox-group`, `outline-tree`, `range`, `field`, `combo-box`, `signature`, `password`.



Essa propriedade de estilização e de uso bastante específico e restrito e consta deste livro apenas a título de informação.

## 4.6 Valor CSS

Vimos anteriormente os oito grupamentos para classificação de valores CSS. É importante também conhecer o conceito de valor CSS para efeito de aplicação da regra CSS no elemento. Observe o exemplo a seguir:

```
p { font-size: 120%; }
```

O valor 120% para a propriedade font-size usada nessa regra CSS enquadra-se no grupamento de valores denominado “Número com unidade de medida”, conforme vimos anteriormente. Contudo, como o navegador aplica um tamanho de fonte igual a 120%? Qual o valor em pixels? 120% do quê?

Para aplicar valores CSS, o navegador precisa, em certos casos, efetuar cálculos e, em outros, “tirar algumas conclusões” para chegar ao valor a aplicar. Ao longo do processo de investigação, o navegador passa por etapas, e em cada etapa chega a um tipo de valor. Veremos a seguir esses tipos de valor.

Observe as regras CSS que se seguem, já mostradas anteriormente:

```
p { font-family: Arial, Sans-serif; } /* estiliza p com fonte na família  
                                     especificada (valor) */  
p { width: 400px; } /* estiliza p com largura 400px; */  
p { font-size: 120%; } /* estiliza p com tamanho de fonte 1.2  
                       vezes o valor de referência; */  
p { background-color: red; } /* estiliza p com fundo na cor vermelha */  
p { height: 2em; } /* estiliza p com altura 2 vezes o  
                  valor de referência; */
```

Observe que alguns valores são absolutos (é aquele valor e pronto!) e outros relativos (dependem de outros valores), tais como as medidas CSS em porcentagem e em. Para aplicar valores CSS, os mecanismos das CSS consideram cinco tipos de valor que

veremos a seguir.

A todas as propriedades CSS é atribuído, por padrão, um valor denominado valor inicial. O valor inicial de cada uma das propriedades CSS é definido por uma folha de estilo nativa do agente de usuário (por exemplo: navegador). Infelizmente, não há padronização para o valor inicial das propriedades CSS, e cada navegador implementa essa funcionalidade à sua maneira. Esse comportamento pode trazer inconsistência de renderização em diferentes navegadores.

Felizmente, todos os navegadores adotam o mesmo valor para muitas das propriedades CSS. As inconsistências, em sua maioria, estão relacionadas à definição de valores iniciais para `margin` e `padding`. Observe a seguir o valor inicial de algumas propriedades CSS:

```
border: none;  
color: black;  
background: transparent;  
font-family: serif;  
font-size: 16px;
```

Todas as propriedades CSS admitem, como valor, a palavra-chave `initial` para forçar a adoção do valor inicial da propriedade. Observe o exemplo a seguir, baseado no exemplo mostrado anteriormente:

- **HTML**

```
<div>  
  <p>Parágrafo com <em>palavra</em> marcada com ênfase</p>  
</div>
```

- **CSS**

```
div {  
  color: red;  
  border: 1px solid blue;  
}  
p { color: initial; }
```

Nesse caso, forçamos o elemento `p` a ser estilizado com a cor

inicial preta, anulando o efeito herança.

Em abril de 2007, Eric Meyer publicou em seu blog uma matéria comentando as inconsistências de renderização entre navegadores em razão da não padronização de valores iniciais, para as diferentes propriedades CSS. Propôs então uma solução que consistia em criar uma folha de estilos padrão que, em resumo, se destinava a padronizar os valores iniciais. Eric expôs ainda razões para preservar determinados valores iniciais e fez algumas outras considerações. Se você estiver interessado em aprofundar-se no assunto, a matéria encontra-se em <http://kwz.me/rb>.

Como consequência, Eric criou uma folha de estilos que vem sendo atualizada regularmente e hoje se encontra em sua versão 2.0, lançada em 26 de janeiro de 2011. A folha de estilo, mundialmente conhecida e usada, foi denominada “CSS Reset” e lançada para uso livre e gratuito sob o rótulo de domínio público. Inúmeros frameworks, ferramentas de desenvolvimento e desenvolvedores em geral usam a folha de estilos criada por Eric como ponto de partida para os valores iniciais das propriedades CSS. Para obter uma cópia atualizada das CSS Reset de Eric Meyer, visite <http://kwz.me/rt>.

Posteriormente foram criadas, por diferentes desenvolvedores e companhias, outras folhas de estilo CSS Reset, e até mesmo a Yahoo criou a sua. Nos dias atuais, além daquelas criadas por Eric Meyer, vêm sendo largamente usadas as CSS Reset criadas por Nicolas Galangher denominadas *normalize.css* que podem ser obtidas em <http://necolas.github.io/normalize.css/> (<http://kwz.me/7a>). Nos exemplos deste livro usaremos as CSS Reset de Nicolas Gallagher.

É comum encontrarmos em fóruns e matérias publicadas em blogs a indicação do uso de uma regra CSS para zerar os valores de *margin* e *padding* de todos os elementos da marcação com uso do seletor universal, como mostrado a seguir:

```
* {  
    margin: 0;
```

```
padding: 0;  
}
```

Eric Meyer adverte que, embora esse seja um bom ponto de partida, é preciso estar ciente de que todos os elementos da marcação terão aquelas duas propriedades zeradas e isso nem sempre ocorre na prática, pois, para elementos de formulário, por exemplo, dependendo do navegador, a regra será ignorada ou, ainda, poderá haver alteração na apresentação do elemento, perdendo-se nesses casos a consistência.

## 4.7 Vinculando folhas de estilo a documentos

Depois que você acabou de escrever sua folha de estilo, precisa informar ao documento onde ele deve buscá-la. Ou seja, você precisa de um método capaz de vincular a folha de estilo ao documento ao qual ela será aplicada.

As folhas de estilo podem ser escritas no próprio documento HTML ao qual serão aplicadas ou ser arquivos externos independentes, gravados com a extensão *.css*, por exemplo, um arquivo chamado de *main.css*, e ligados ao documento.

### 4.7.1 Estilos inline

O método direto e simples de aplicar estilos a um elemento da marcação é com o emprego do atributo *style* da HTML. Você escreve as regras de estilo diretamente dentro da tag de abertura do elemento a estilizar, conforme mostra o exemplo a seguir:

```
<p style="width: 200px; color: white; background: red; font-size: 1.8em;">  
<!-- Parágrafo com aplicação de estilos inline -->  
</p>
```

Esse método dificulta a manutenção e retira um dos maiores poderes da folha de estilo, que é o controle centralizado da apresentação. Toda vez que for preciso alterar a apresentação, será necessário percorrer todo o código de marcação do documento ou centenas de documentos, se o site for grande, à procura das regras

de estilo inline.

## 4.7.2 Estilos incorporados

Outro método de escrever a folha de estilos no próprio documento HTML é com o emprego do elemento `style`. Você escreve as regras de estilo dentro das tags `<style></style>`, declaradas na seção HEAD do documento, como mostrado no exemplo a seguir:

```
<head>
...
<style rel="stylesheet" type="text/css" media="all">
  body {
    margin: 0;
    padding: 0;
    font-size: 80%;
    color: black;
    background: white;
  }
</style>
</head>
```

A vantagem desse método sobre o anterior é que, agora, localizamos com mais facilidade a folha de estilos, mas a desvantagem é que, ao colocar a folha de estilos dentro do próprio documento, as regras de estilização serão aplicadas somente ao documento. Não seria sensato vincular uma mesma folha de estilo a vários documentos empregando esse método. Toda vez que for preciso alterar a apresentação, será necessário abrir o documento ou centenas de documentos, se o site for grande, e fazer a mesma alteração de estilo em todos eles. Uma boa escolha para uso desse método seria para o caso de aplicação de estilos específicos a um, e somente um, documento do site.

O elemento `style` deve estar contido na seção HEAD do documento. Convém ressaltar que em marcação HTML5 o uso de atributos no elemento `style` é facultativo. O atributo `type` informa qual tipo de dado está sendo enviado, e o atributo `media` informa a qual tipo de mídia devem ser aplicados os estilos. Os valores do atributo



media e a mídia a que se destinam são listados a seguir:

Valor	Mídia	Nota
screen	Telas de monitores	
tty	Teletipo e similares	
tv	Dispositivos tipo televisão	
projection	Projetores	
handheld	Dispositivos portáteis	
print	Impressoras e visualização no modo impressão	
braille	Dispositivos táteis	
aural	Sintetizadores de voz	em desuso pela CSS3
all	Todos os tipos de mídia	
speech	Sintetizadores de voz	criada pela CSS3
embossed	Impressoras braille	criada pela CSS3

## 4.7.3 Estilos externos

Folha de estilo externa é aquela que não foi escrita no documento HTML. Trata-se de um arquivo de texto contendo as regras de estilo e os comentários CSS. Um arquivo de folha de estilos deve ser gravado com a extensão .css e pode ser vinculado a um documento HTML de duas maneiras distintas, conforme explicado em seguida.

### 4.7.3.1 Folhas de estilo lincadas

Você vincula uma folha de estilo externa a um documento empregando o elemento link. O elemento link deve estar contido na seção HEAD do documento e tem por finalidade associar outros documentos ao documento no qual ele está contido. Veja, a seguir, o uso de link para associar (ou vincular) uma folha de estilo ao documento:

```
<head>
...
<link rel="stylesheet" type="text/css" href="estilos.css" media="all">
...
```

</head>

O atributo `href` aponta para o endereço no qual se encontra o arquivo da folha de estilo.

#### 4.7.3.2 Folhas de estilo importadas

Nesse método, você vincula uma folha de estilo externa a outra folha de estilo externa usando a diretiva `@import` dentro da folha de estilo.

O exemplo a seguir mostra uma folha de estilo externa na qual importamos uma folha de estilo denominada *main.css*:

```
@import "main.css"
body {
    margin: 0;
    font: 62.5% Arial, Sans-serif;
}
... mais regras de estilo ...
```

É válido importar mais de uma folha de estilo para dentro de uma folha de estilo.

A declaração de importação de uma folha de estilo dentro de outra deve ocupar a primeira linha da folha, exceto no caso em que se use uma declaração de codificação de caracteres da folha de estilo. A diretiva `@charset` destina-se a declarar a codificação de caracteres de uma folha de estilo e deve ocupar a primeira linha na folha de estilo. Observe o exemplo a seguir, que demonstra as duas diretivas inseridas em uma folha de estilo:

```
@charset "utf-8"
@import "main.css"
body {
    margin: 0;
    font: 62.5% Arial, Sans-serif;
}
... mais regras de estilo ...
```

A diretiva `@import` deve preceder todas as demais regras de estilo para o documento. Havendo necessidade de vincular outras folhas de estilo ao documento, elas deverão ser importadas uma após

outra.

O método de vincular folhas de estilo externas permite que se apliquem regras de estilo comuns a todos os documentos de um site. A grande vantagem do método é que o autor controla a apresentação do site em um arquivo central. A alteração de uma cor ou do tamanho de fonte na folha de estilo imediatamente se reflete no site inteiro, quer ele tenha dez ou 10 mil páginas.

# CAPÍTULO 5

## DOM e seletores CSS

### 5.1 DOM

DOM é a sigla em inglês para *Document Object Model* e, que, em português, significa Modelo de objeto de documento.

Em descrição simples, o DOM é uma convenção que, no contexto da marcação HTML, destina-se a representar os componentes da marcação como objetos capazes de serem estilizados, acessados por programas e scripts e manipulados de variadas formas.

O DOM é constituído de nós denominados nós do DOM. Os elementos da marcação, tais como os elementos `h1` e `p`, são nós do tipo elemento; os atributos da marcação HTML são nós de tipo atributo; os conteúdos textuais de um elemento são nós do tipo texto; os comentários são nós do tipo comentário; e o próprio documento como um todo é um nó do tipo documento.

#### 5.1.1 Árvore do DOM

Árvore do DOM, comumente chamada de árvore do documento é uma representação gráfica da marcação do documento, mostrando o inter-relacionamento entre os nós do DOM. O termo árvore para designar essa representação foi adotado pela semelhança existente entre uma árvore do documento com a conhecida árvore genealógica.

No DOM o relacionamento entre os componentes da marcação, descrito na árvore do documento, usa também denominações adotadas em árvores genealógicas, tais como, elemento-pai, elementos irmãos, elementos descendentes, elementos ancestrais e

assim por diante.

Paulo Silva concluiu o curso de HTML5 e CSS3 há uns meses e, após ter desenvolvido alguns sites para amigos e familiares, resolveu que chegara a hora de criar um site para divulgar seu trabalho, oferecer seus serviços para o mundo e compartilhar o que aprendeu no curso.

Munido de uma folha de papel e lápis, Paulo fez uma lista das páginas que colocaria no seu site. Depois de algumas tentativas, decidiu que o site começaria com cinco páginas e, se fosse o caso, seria ampliado no futuro sem necessidade retirar as cinco páginas e seus conteúdos. As cinco páginas e o nome de seus arquivos seriam os seguintes:

- **Homepage do site** (página de apresentação do site)

*index.html*

- **Portfólio** (página de apresentação dos trabalhos do Paulo)

*portfólio.html*

- **Artigos HTML** (página com artigos sobre HTML escritos pelo Paulo)

*artigos-html.html*

- **Artigos CSS** (página com artigos sobre CSS escritos pelo Paulo)

*artigos-css.html*

- **Contato** (página com dados para entrar em contato com o Paulo)

*contato.html*

Paulo rascunhou como seria o layout das páginas do site. No rascunho mostrou um topo do site contendo logotipo, título e a barra de navegação, a seguir outras funcionalidades, tais como banners de propaganda, links para redes sociais e caixas de busca, uma coluna com o conteúdo principal da página e um rodapé.

A partir deste capítulo do livro, sempre que for conveniente, vamos usar o site do Paulo para exemplificar os conceitos e ensinamentos

contidos nos capítulos.

Abra o Notepad++ e digite a marcação HTML mostrada a seguir e destinada a criar o topo do site do Paulo conforme descrito anteriormente. Salve o arquivo na pasta do capítulo 5 com o nome *topo-do-site-do-paulo-1.html*.

```
<!DOCTYPE html>
<html lang="pt-br">
<head>
<title>Criação de sites com HTML e CSS | Paulo Silva</title>
<meta charset="utf-8">
</head>
<body class="home">
  <header class="topo">
    <hgroup>
      <h1><a href="/">Site do Paulo</a></h1>
      <h2>Site destinado a divulgar o trabalho de Paulo Silva na criação de
sites</h2>
    </hgroup>
    <nav>
      <ul>
        <li><a class="corrente" href="home.html">home</a></li>
        <li><a href="portfolio.html">portfólio</a></li>
        <li><a href="artigos-html.html">artigos HTML</a></li>
        <li><a href="artigos-css.html">artigos CSS</a></li>
        <li><a href="contato.html">contact</a></li>
        <div class="clear"></div> /* Capitulo 6 - "limpar" floats */
      </ul>
    </nav>
  </header>
  <!-- mais conteúdo -->
  <footer class="rodape">
    <!-- conteúdo do rodapé do site -->
  </footer>
</body>
</html>
```

Vamos fazer uso da marcação mostrada para exemplificar os conceitos de DOM de árvore do documento, de nós, de seletores CSS e da sintaxe da árvore e dos seletores.

Observe que *diretamente* dentro da seção body do documento (<body></body>) existem os elementos header (<header></header>) e footer (<footer></footer>). *Diretamente* dentro do elemento header (<header></header>) existem os elementos hgroup (<hgroup></hgroup>) e nav (<nav></nav>) e assim por diante com os demais elementos.

- Elementos que estão diretamente dentro de um elemento qualquer F são chamados de *elementos-filho* de F, e o elemento F é chamado de *elemento-pai*.
- Elementos que estão dentro de um elemento qualquer F, mas não diretamente, são chamados de *elementos-descendentes* de F, e o elemento F é chamado de *elemento-ancestral*.
- Elementos que são elementos-filho de um mesmo elemento-pai são chamados de *elementos-irmãos*.
- Elementos E e F que na marcação HTML um vem logo depois do outro são chamados de *elementos-adjacentes*.
- Se elementos-adjacentes são filhos do mesmo elemento-pai diz-se que são *elementos-irmãos-adjacentes*.



Para efeito de sintaxe e terminologia HTML e CSS, a classificação de elementos-descendentes não se aplica a elementos-filho, embora sejam descendentes do seu elemento-pai. Eles são simplesmente elementos-filho.

Notar que tudo se passa como se houvesse um grau de parentesco familiar entre os elementos do DOM (ou da marcação) dando origem à árvore do documento cujo diagrama, para a marcação mostrada anteriormente, está na figura 5.1.

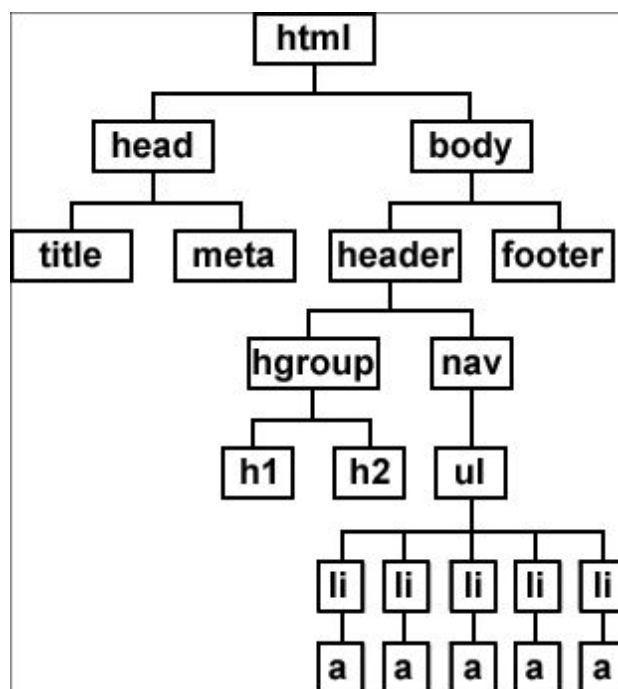


Figura 5.1 – Árvore do documento.

Da árvore do documento é fácil verificar os graus de parentesco conforme alguns exemplos descritos a seguir.

Elemento	Descrição
html	Elemento-pai dos elementos head e body e elemento-ancestral ou simplesmente ancestral de todos os elementos da marcação.
hgroup e nav	Elementos-filho do elemento header, portanto elementos-irmãos.
a	Elementos-descendentes do elemento ul.
h1	Elemento-irmão-adjacente do elemento h2.
header	Não é elemento-irmão do elemento meta.
li	Não é elemento-descendente do elemento hgroup, mas é elemento descendente dos elementos nav, header, body e html.

## 5.2 Seletores CSS

Seletor CSS é a parte componente de uma regra CSS, ver [4.1.3], que se destina a selecionar um conteúdo do DOM a ser estilizado com as declarações de estilos contidas na regra. Dito de outra forma: seletor CSS é o responsável por escolher a parte do DOM que será estilizada. O estudo e entendimento dos seletores CSS é



um fundamento básico e o mais importante aspecto das CSS para que o desenvolvedor torne-se um expert na criação e aplicação das regras de estilos.

Um seletor CSS representa uma estrutura que descreve quais são os elementos a serem selecionados na árvore do documento. A sintaxe para descrever um seletor pode ser desde uma simples letra representativa de um elemento do DOM até uma estrutura complexa descrevendo o grau de parentesco ou inter-relacionamento ou, ainda, as características específicas de nós do DOM.

Observe na tabela 5.1 a sintaxe para os seletores das CSS, uma breve descrição da parte da árvore do documento que é selecionada e o tipo do seletor. Faça uma leitura da tabela, procurando familiarizar-se com cada seletor e sua finalidade, mas não se preocupe em decorar a tabela mostrada, pois, com a sequência dos estudos, você vai assimilando naturalmente o formato dos seletores. Havendo dúvida, consulte a tabela.

*Tabela 5.1 – Seletores das CSS*

<b>Seletor</b>	<b>Seleciona</b>	<b>Tipo</b>
*	Qualquer elemento	Seletor universal
E	Qualquer elemento de nome E	Seletor tipo
E[foo]	Elemento E com atributo “foo”	Seletor de atributo
E[foo=“bar”]	Elemento E cujo valor do atributo “foo” é “bar”	Seletor de atributo
E[foo~=“bar”]	Elemento E cujo valor do atributo “foo” é exatamente “bar” ou contém o nome “bar” entre espaços em branco	Seletor de atributo
E[foo^=“bar”]	Elemento E cujo valor do atributo “foo” começa com a string “bar”	Seletor de atributo
E[foo\$=“bar”]	Elemento E cujo valor do atributo “foo” termina com a string “bar”	Seletor de atributo
E[foo*=“bar”]	Elemento E cujo valor do atributo “foo” contém a substring “bar”	Seletor de atributo
E[foo =“val”]	Elemento E cujo valor do atributo “foo” é igual a “val” ou começa com “val” seguido de hífen	Seletor de atributo
E:root	Elemento E raiz do documento	Pseudoclassee estrutural

Seletor	Seleciona	Tipo
E:nth-child(n)	Elemento E filho de ordem n do seu elemento-pai	Pseudoclassee estrutural
E:nth-last-child(n)	Elemento E filho de ordem n do seu elemento-pai, contando a ordem a partir do último filho	Pseudoclassee estrutural
E:nth-of-type(n)	Irmão do mesmo tipo e de ordem n do elemento E	Pseudoclassee estrutural
E:nth-last-of-type(n)	Irmão do mesmo tipo e de ordem n do elemento E, contando a ordem a partir do último irmão	Pseudoclassee estrutural
E:first-child	Elemento E primeiro-filho do seu elemento-pai	Pseudoclassee estrutural
E:last-child	Elemento E último-filho do seu elemento-pai	Pseudoclassee estrutural
E:first-of-type	Elemento E primeiro irmão do mesmo tipo de E	Pseudoclassee estrutural
E:last-of-type	Elemento E último irmão do mesmo tipo de E	Pseudoclassee estrutural
E:only-child	Elemento E filho único de seu elemento-pai	Pseudoclassee estrutural
E:only-of-type	Elemento E irmão único do mesmo tipo de E	Pseudoclassee estrutural
E:empty	Elemento E sem elementos-filho (inclusive nós de texto)	Pseudoclassee estrutural
E:link	Elemento E marcador de hyperlink em estado não visitado	Pseudoclassee link
E:visited	Elemento E marcador de hyperlink em estado visitado	Pseudoclassee link
E:active	Elemento E marcador de hyperlink quando ativado	Pseudoclassee link
E:hover	Elemento E marcador de hyperlink com ponteiro sobre ele	Pseudoclassee link
E:focus	Elemento E sobre determinadas ações do usuário	Pseudoclassee ação do usuário
E:target	Elemento E alvo de um URI	Pseudoclassee alvo
E:lang(fr)	Elemento E no idioma especificado (por exemplo: "fr")	Pseudoclassee :lang()
E:enabled	Elemento E habilitado	

Seletor	Seleciona	Tipo
E:disabled	Elemento E desabilitado	Pseudoclasse estado do elemento E
E:checked	Elemento E “ticado” (por exemplo: radio-button ou checkbox)	Pseudoclasse estado do elemento E
E::first-line	Primeira linha de um elemento E	Pseudoelemento ::first-line
E::first-letter	Primeira letra de um elemento E	Pseudoelemento ::first-letter
E::before	Conteúdo gerado para ser posicionado antes do elemento E	Pseudoelemento ::before
E::after	Conteúdo gerado para ser posicionado depois do elemento E	Pseudoelemento ::after
E.warning	Elemento E ao qual foi atribuído o atributo classe com valor “warning”	Seletor classe
E#myid	Elemento E ao qual foi atribuído o atributo id com valor “myid”	Seletor ID
E:not(s)	Elemento E não casado por um seletor simples	Pseudoclasse negação
E F	Elemento F descendente do elemento E	Caractere de combinação espaço em branco
E > F	Elemento F filho do elemento E	Caractere de combinação filho
E + F	Elemento F imediatamente após o elemento E (E e F são irmãos)	Caractere de combinação irmão adjacente
E ~ F	Elementos F após o elemento E (E e F devem ser irmãos)	Caractere de combinação irmão geral

## 5.2.1 Criando uma folha de estilo

Vamos criar uma folha de estilo a ser incorporada, ver [4.3.2], ao arquivo criado anteriormente denominado *topo-do-site-do-paulo-1.html*. Abra esse arquivo e salve-o com o nome *topo-do-site-do-paulo-2.html*.

Nesse novo arquivo, que contém o topo do site do Paulo, vamos incorporar a folha de estilos mostrada a seguir. Os números que aparecem no código são referência para a explicação de cada seletor (código comentado após a folha de estilo) e não fazem parte da folha de estilo.

```
* {
  margin: 0;
  padding: 0;
}
body {
  font-size: 18px;
  font-family: arial, sans-serif;
  line-height: 1.4;
}
hgroup > h1 {
  font-size: 36px;
  margin-top: 12px;
  margin-bottom: 12px;
}
hgroup a {
  color: #c30;
  text-decoration: none;
}
h1 ~ h2 {
  font-size: 27px;
  margin-top: 14px;
  margin-bottom: 14px;
  color: #090;
}
nav > ul {
  margin-left: 50px;
}
nav > ul > li:first-child > a {
  color: #c60;
  text-decoration: none;
}
nav > ul > li:first-child > a:hover {
  color: #555;
  text-decoration: underline;
```

```

    }
    nav > ul > li:last-child {
        font-size: 26px;
        list-style-type: square;
    }
    nav > ul > li:nth-child(3) {
        background: #ccc;
    }

```

Código comentado:

Linha	Descrição
Linha 1	Seletor universal. Seleciona (ou casa com) todos os elementos da árvore do documento. Essa regra CSS zera as margens e padding iniciais (definidas na folha de estilo do agente de usuário).
Linha 2	Seletor tipo para o elemento body. Essa regra CSS define as características da fonte padrão do documento.
Linha 3	Seletor filho. Essa regra CSS estiliza o elemento h1 filho do pai hgroup.
Linha 4	Seletor descendente. Essa regra CSS estiliza o elemento a descendente do elemento hgroup.
Linha 5	Seletor irmão adjacente. Essa regra CSS estiliza o elemento h2 irmão adjacente de h1.
Linha 6	Seletor filho. Essa regra CSS estiliza o elemento ul filho do pai nav.
Linha 7	Seletor composto. Essa regra CSS estiliza o elemento a filho do primeiro elemento li que é filho de ul que por sua vez é filho de nav.
Linha 8	Seletor composto. Essa regra CSS estiliza o elemento a da linha 7 quando o usuário passa o mouse sobre ele.
Linha 9	Seletor composto. Essa regra CSS estiliza o elemento li último filho do elemento ul que por sua vez é filho de nav.
Linha 10	Seletor composto. Essa regra CSS estiliza o terceiro elemento li filho do elemento ul que por sua vez é filho de nav.

Nos arquivos disponíveis para download, o leitor encontrará um arquivo denominado *topo-do-site-do-paulo-2-interativo.html*. Esse arquivo é interativo. Abra o arquivo no navegador e observe que, na área em fundo preto, encontra-se a folha de estilo que acabamos de estudar, e ao lado a renderização da marcação mostrada no exemplo.

A folha de estilo é editável no próprio navegador, ou seja, o leitor poderá alterar as regras de estilo no navegador e até mesmo criar novas regras para outros seletores da marcação e verificar imediatamente, na renderização da página, os efeitos das regras modificadas ou acrescentadas. Use esse arquivo para estudar e fazer experiências com estilização.

## 5.2.2 Estilizando o topo do site

No Notepad++ abra o arquivo *topo-do-site-do-paulo-1.html* e salve esse arquivo na pasta do capítulo 5 com o nome *topo-do-site-do-paulo-3a.html*. Vamos criar uma folha de estilo incorporada ao documento como fizemos no exemplo anterior, definindo regras CSS para estilizar o topo do site conforme mostrado na figura 5.2.

Em lugar de apresentar e comentar a folha de estilo final para obter a estilização do topo, conforme mostrado na figura anterior, optamos por criar um passo a passo de estilização apresentando-a em etapas.



Figura 5.2 – Topo do site do Paulo estilizado.

### 1ª etapa

Conforme dito anteriormente, acabamos de salvar o arquivo *topo-do-*

*site-do-paulo-3a.html* e estamos com o Notepad++ aberto nesse arquivo. Nele vamos fazer as alterações para esta 1ª etapa.

Criamos um elemento `div`, container geral para toda a marcação HTML, e a ele definimos o atributo classe denominado `tudo`. Definimos para o elemento `body` a classe denominada `home`, conforme mostrado a seguir.

```
...
<body class="home">
  <div class="tudo">
    <!-- marcação HTML do topo -->
  </div> <!-- /.tudo -->
</body>
</html>
```

Esse arquivo será a homepage do site. Assim, ao atribuir uma classe denominada `home` para o elemento `body`, criamos um seletor CSS capaz de estilizar qualquer elemento constante da página sem interferir nas outras páginas. Faremos o mesmo com as demais páginas do site atribuindo um nome de classe personalizado para cada página.

Ao criar um container geral para a página `div.tudo`, estamos na verdade criando uma caixa com todo o conteúdo renderizado, capaz de ser estilizada e movimentada com CSS, por exemplo: cor de fundo e centralização da página na tela. Notar na figura 5.2 que a renderização é centralizada na tela e a cor de fundo da página é branca com o resto da tela cinza.

Já dissemos que, para fins de estudo, é melhor que a folha de estilo seja incorporada à página. Quando a estilização final estiver concluída, retira-se a folha de estilos da página, cria-se um arquivo externo com ela, do tipo *nome.css*, e substitui-se a folha incorporada por uma folha de estilo externa, lincada ao documento.

Convém ressaltar que essa é uma prática válida para o desenvolvimento em fase de produção real. É melhor desenvolver com folha de estilo incorporada e passar para externa quando o desenvolvimento estiver concluído.

Na seção head do documento, linkamos para a folha de estilo externa CSS Reset de Nicolas Gallagher (ver [4.6]), denominada *normalize.css*, e criamos o container para a folha de estilo a ser incorporada ao documento que conterá as regras de estilo para estilizar o topo conforme mostrado a seguir.

```
<!DOCTYPE html>
<html lang="pt-br">
<head>
<title>Criação de sites com HTML e CSS | Paulo Silva</title>
<meta charset="utf-8">
<!--[if lt IE 9]>
    <script src="../../html5shiv-master/src/html5shiv.js"></script>
<![endif]-->
<link rel="stylesheet" href='../normalize.css'></link> <!-- CSS Reset de Nicolas
    Gallagher -->
<style rel="stylesheet">
    <!-- regras de estilos entram aqui -->
</style>
</head>
...
```

Código comentado:

Linha	Descrição
Linhas 6 a 7	Os sinais <!-- [if lt IE9]> <![endif]--> marcam um comentário condicional. Todo o conteúdo HTML dentro desses sinais é reconhecido apenas pelos navegadores Internet Explorer e em consequência ignorado pelos demais navegadores. Trata-se de uma invenção da Microsoft válida somente para seus navegadores Internet Explorer. Na abertura do comentário é colocada uma condicional que no caso do nosso exemplo é if lt IE9, que é lido como “se menor do que o IE9”, ou seja, o HTML que segue é para ser executado somente se o navegador for anterior ao IE9 (IE8, IE7, IE6, ...). Mais sobre comentários condicionais em <a href="http://www.maujor.com/tutorial/condcom.php">http://www.maujor.com/tutorial/condcom.php</a> ( <a href="http://kwz.me/DV">http://kwz.me/DV</a> ).
	A marcação HTML dentro do comentário condicional é a tag <script> </script> com o atributo src apontando para o endereço onde se encontra um script JavaScript. Esse script destina-se a fazer com que os navegadores IE9 e anteriores reconheçam os novos elementos da HTML5, pois eles, ao contrário dos demais navegadores, não dão suporte para os novos elementos. A seguir mostraremos onde fazer o download desse script.



Linha	Descrição
Linha 8	Aqui importamos uma folha de estilo do tipo CSS Reset conforme mostrado a seguir.
Linha 9 a 10	Esse é o container para as regras de estilo incorporadas ao documento e destinadas a estilizar o topo conforme veremos na sequência deste exemplo.

Faça o download do script *html5shiv.js* em <http://kwz.me/wS>, salve o arquivo zip na pasta *livroHTMLCSS* que contém as subpastas dos capítulos (ver [2.3]). Descompacte o arquivo e você vai obter vários arquivos e subpastas com informações sobre o script que se encontra nas subpastas *html5shiv-master/src*. Uma vez que todos os arquivos dos exemplos deste livro, a partir deste capítulo, deverão linkar para o script *html5shiv.js*, consideramos que esse arquivo foi salvo e linkado aos arquivos dos exemplos, conforme indicado.

Faça o download da folha de estilo *normalize.css* em <http://kwz.me/7a>, salve-a na pasta *livroHTMLCSS* que contém as subpastas dos capítulos (ver [2.3]) e crie o link conforme mostrado no código anterior. Uma vez que todos os arquivos dos exemplos deste livro, a partir deste capítulo, deverão linkar para a folha de estilo *normalize.css*, consideramos que esse arquivo foi salvo e linkado aos arquivos dos exemplos, conforme indicado.

## 2ª etapa

Conforme dito anteriormente, acabamos de salvar o arquivo *topo-do-site-do-paulo-3a.html* e estamos com o Notepad++ aberto nele.

Digite as regras de estilo no container para a folha de estilo incorporada, conforme mostrado a seguir.

1. `<style rel="stylesheet">`
2. `html { box-sizing: border-box; }`
3. `*, *:before, *:after { inherit; }`  
`}`
4. `body {`  
`font: 18px/1.4 arial, sans-serif;`  
`background: #ddd;`  
`color: #333;`

```

    }
5. .todo {
    width: 100%;
    max-width: 960px;
    margin: 0 auto;
    background: #fff;
    }
6. .topo { width:100%; }
7. .topo h1 { font-size: 36px; }
8. .topo h2 { font-size: 27px; }
9. </style>

```

Código comentado:

Linha	Descrição
Linhas 1 a 9	Container para a folha de estilo incorporada.
Linhas 2 e 3	Definem o Box Model CSS modificado.
Linha 4	Define características da fonte, a cor de fundo (#ddd cinza) e de frente (#333 preta) para o elemento body (todo o conteúdo a ser renderizado).
Linha 5	Estiliza a página que deverá ter uma largura width igual à largura da tela, até um valor máximo max-width igual a 960px, ou seja, para telas com largura menor do que 960px, a largura da página será igual à largura da tela. Para telas maiores, a largura será de 960px. Ao visualizar a renderização da página em um navegador, redimensione a tela para observar na prática esse comportamento. A cor de fundo da página será #fff (branca) e ela será centrada na tela (margin: 0 auto;)
Linha 6	O container do topo (header.topo) terá largura igual à largura do seu elemento-pai (div.todo)
Linhas 7 e 8	Definem o tamanho da fonte dos elementos cabeçalho contidos no topo.

Observe na figura 5.3 a renderização do arquivo *topo-do-site-do-paulo-3a.html* depois da inclusão das regras de estilo mostradas.

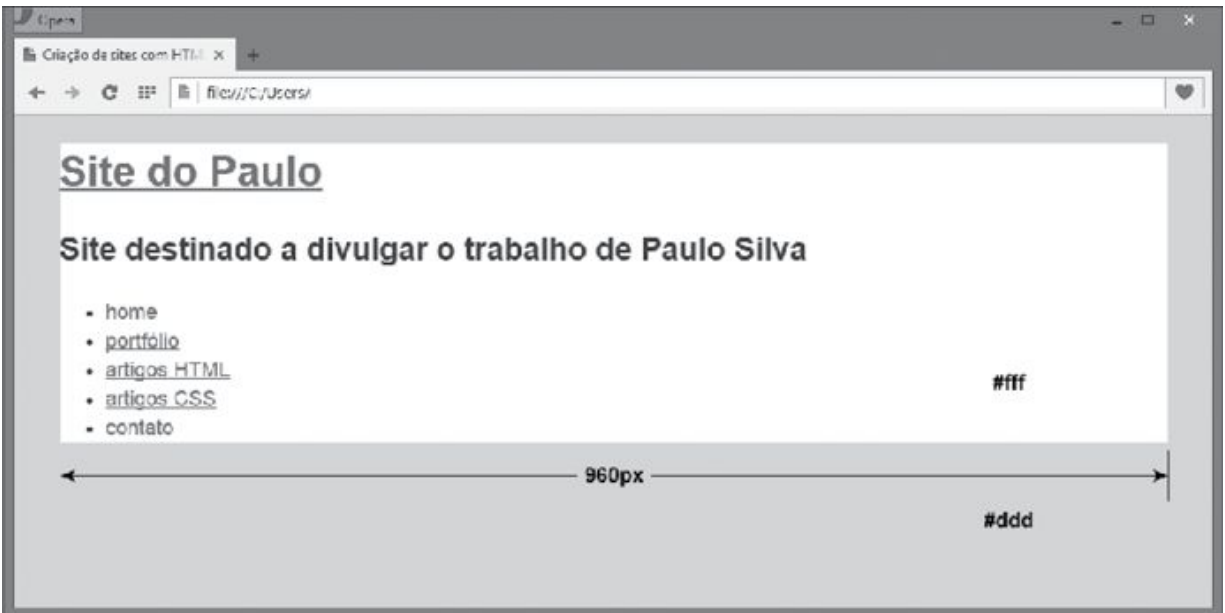


Figura 5.3 – Estilização do topo do site do Paulo, 2ª etapa.

### 3ª etapa

Abra o arquivo *topo-do-site-do-paulo-3a.html* no Notepad++ e salve-o com o nome *topo-do-site-do-paulo-3b.html*. Nessa etapa, vamos prosseguir com a estilização do topo do site do Paulo acrescentando as regras CSS às existentes que foram criadas na etapa anterior. Essas regras são mostradas em destaque a seguir.

```
<style rel="stylesheet">
...
.topo {
    width:100%;
    line-height: 1;
3. padding: 1px 20px 10px 20px;
}
.topo h1 {
    font-size: 36px;
4. margin: 4px 0 2px 0;
5. text-align: right;
}
6. .topo a {
    color: #900;
    text-decoration: none;
}
```

```

        .topo h2 {
            font-size: 20px;
7. margin: 2px 0 4px 0;
8. text-align: right;
        }
9. </style>

```

Código comentado:

Linha	Descrição
Linhas 1 a 9	Folha de estilo incorporada ao documento. As declarações CSS acrescentadas estão em destaque e numeradas.
Linha 2	A declaração line-height controla o espaçamento entre linhas, definindo um box container para cada linha. Declarar 1 para essa propriedade faz com que o container dos elementos h1 e h2 do topo tenha altura igual à altura do tamanho da fonte (font-size) e, em consequência, o espaçamento entre h1 e h2 seja reduzido.
Linha 3	Ajusta o padding para o container do topo. O valor 1px para padding-top destina-se a evitar o comportamento <i>margin collapse</i> . Substitua 1px por 0 na folha de estilo e compare os resultados de renderização para visualizar esse comportamento.
Linha 4	Ajusta as margens do elemento h1 contido no topo.
Linha 5	Alinha à direita o elemento h1 contido no topo.
Linha 6	Cria regra CSS para os links do topo (mecanismo de navegação), definindo para eles uma cor (#900), e retira o sublinhado padrão.
Linha 7	Ajusta as margens do elemento h2 contido no topo.
Linha 8	Alinha à direita o elemento h2 contido no topo.

Observe na figura 5.4 a renderização do arquivo *topo-do-site-do-paulo-3b.html* depois da inclusão das regras de estilo mostradas.



Figura 5.4 – Estilização do topo do site do Paulo, 3ª etapa.

#### 4ª etapa

Abra o arquivo *topo-do-site-do-paulo-3b.html* no Notepad++ e salve-o com o nome *topo-do-site-do-paulo-3c.html*. Nesta etapa vamos prosseguir com a estilização do topo do site do Paulo acrescentando as regras CSS às existentes que foram criadas na etapa anterior. Essas regras destinam-se a estilizar a navegação e são mostradas em destaque a seguir.

```
<style rel="stylesheet">
...
nav > ul {
    width: 100%;
    background: #900;
    margin: 0;
    padding: 0;
}
.topo ul li {
    list-style: none;
    display: inline;
}
.topo ul li a {
    float: left;
    text-decoration: none;
```

```

text-transform:uppercase;
text-align: center;
border-right: 1px solid #fff;
padding: 10px 20px;
color: #fff;
background:#900;
font-size: 14px;
font-weight: bold;
}
.topo ul li:last-child a { border: none; }
.topo ul li a:hover { color: #900; background: #ccc; }
</style>

```

Observe na figura 5.5 a renderização do arquivo *topo-do-site-do-paulo-3c.html* depois da inclusão das regras de estilo mostradas.



Figura 5.5 – Estilização do topo do site do Paulo, 4ª etapa.

## 5ª etapa

Abra o arquivo *topo-do-site-do-paulo-3c.html* no Notepad++ e salve-o com o nome *topo-do-site-do-paulo-3final.html*. Nesta etapa vamos terminar a estilização do topo do site do Paulo acrescentando as regras CSS às existentes que foram criadas na etapa anterior. Essas regras destinam-se a estilizar e a dar os ajustes finais à navegação. A folha de estilo final é mostrada a seguir com os

acréscimos e modificações desta etapa em destaque.

```
<style rel="stylesheet">
html { box-sizing: border-box }

*, *:before, *:after { inherit; }
.clear { clear: both; }
body {
    font: 18px/1.4 arial, sans-serif;
    background: #ddd;
    color: #333;
}
.tudo {
    width: 100%;
    max-width: 960px;
    margin: 0 auto;
    background: #fff;
}
.topo {
    width: 100%;
    line-height: 1;
    padding: 1px 20px 10px 20px;
}
.topo h1 {
    font-size: 36px;
    margin: 4px 0 2px 0;
    text-align: right;
}
.topo a {
    color: #900;
    text-decoration: none;
}
.topo h2 {
    font-size: 20px;
    margin: 2px 0 4px 0;
    text-align: right;
}
nav > ul {
    margin: 5px 0 0 0;
    padding: 0;
    width: 100%;
```

```

    background: #900;
}
nav {
    border-top: 1px solid #900;
    width: 100%;
    width: calc(100% + 40px);
    margin-left: -20px;
}
.topo ul li {
    list-style: none;
    display: inline;
}
.topo ul li a {
    float: left;
    text-decoration: none;
    text-transform: uppercase;
    text-align: center;
    border-right: 1px solid #fff;
    padding: 10px 20px;
    color: #fff;
    background: #900;
    font-size: 14px;
    font-weight: bold;
}
.topo ul li: last-child a { border: none; }
.topo ul li a: hover, .home .corrente {
    color: #900;
    background: rgb(225, 185, 185);
}
</style>

```

Observe na figura 5.6 a renderização do arquivo *topo-do-site-do-paulo-3final.html* depois da inclusão das regras de estilo mostradas e do elemento `div.clear` na marcação.





*Figura 5.6 – Estilização do topo do site do Paulo, 5ª etapa.*

Encerramos este capítulo com a estilização do topo do site do Paulo. No próximo capítulo estudaremos as técnicas de posicionamento de boxes HTML com uso das CSS. Conhecer essas técnicas é pré-requisito para criação de layouts. Após o estudo das técnicas, continuaremos com a marcação HTML do site do Paulo e a construção do layout.

# CAPÍTULO 6

## Posicionamento CSS

### 6.1 Introdução

As CSS preveem várias funcionalidades destinadas a manipular e definir o posicionamento dos boxes da marcação HTML em uma página. Distribuir boxes na página significa criar o layout da página, ou seja, definir como os diferentes boxes se distribuem visualmente na tela do usuário.

Por padrão, os elementos HTML nível de bloco distribuem-se verticalmente um após o outro na ordem em que aparecem na marcação HTML e os elementos inline posicionam-se em linha na horizontal.

Os mecanismos de posicionamento CSS permitem ao autor alterar o comportamento padrão, não só alterando a ordem como também posicionando elementos nível de bloco um ao lado do outro. Tais alterações são feitas com regras CSS definindo valores para as propriedades CSS destinadas a manipular a posição dos boxes da página.

No capítulo anterior estilizamos o topo do site do Paulo e visualizamos sua renderização no topo da tela do usuário. É perfeitamente possível, com uso de algumas declarações CSS, fazer com que o topo do site seja renderizado no final da tela ou, ainda, alterar a navegação de horizontal para vertical e posicioná-la no lado direito.

### 6.2 Esquemas de posicionamento

As CSS preveem três esquemas de posicionamento: normal flow

(fluxo normal), float (flutuado) e absolute (absoluto), conforme descritos a seguir.

- **Fluxo normal** – É o posicionamento padrão dos boxes, na vertical ou horizontal, conforme descrito anteriormente.
- **Flutuado** – Quando o box *continua* no fluxo normal e é posicionado à esquerda ou à direita, tal como se estivesse “flutuando” no conteúdo (texto) do box seguinte, que se desenvolve ao lado do box flutuado.
- **Absoluto** – Quando o box *é retirado* do fluxo normal e posicionado segundo um sistema de coordenadas.

## 6.2.1 Esquema normal ou posicionamento padrão

Nesse esquema o posicionamento de um box pode ser feito declarando-se margens para o box. As margens controlam as distâncias verticais e horizontais entre boxes.

### 6.2.1.1 Elementos nível de bloco

A distância vertical entre elementos nível de bloco que se seguem na marcação HTML (fluxo do documento) é determinada pela propriedade *margin*. Não tendo sido definida por regras de estilo uma margem vertical para o bloco, será tomada a margem inicial padrão, própria da folha de estilo nativa do navegador. Margens verticais entre blocos que se seguem sempre se sobrepõem (*collapse*), ou seja, em lugar de somar a margem inferior de um bloco com a margem superior do bloco que lhe segue para obter a margem final, esta será tomada como sendo a maior entre ambas, conforme mostrado no esquema da figura 6.1.

O comportamento denominado *border collapse* (sobreposição de bordas) mostrado na figura 6.1 supõe que ambas as margens sejam positivas. Caso ambas sejam negativas, a borda final será a de maior valor absoluto, por exemplo: para margens iguais a -40px e -20px será tomado o valor -40px. Se de sinais contrários, a borda

final será a soma algébrica, por exemplo: para margens iguais a -40px e 30px será tomado o valor -10px.

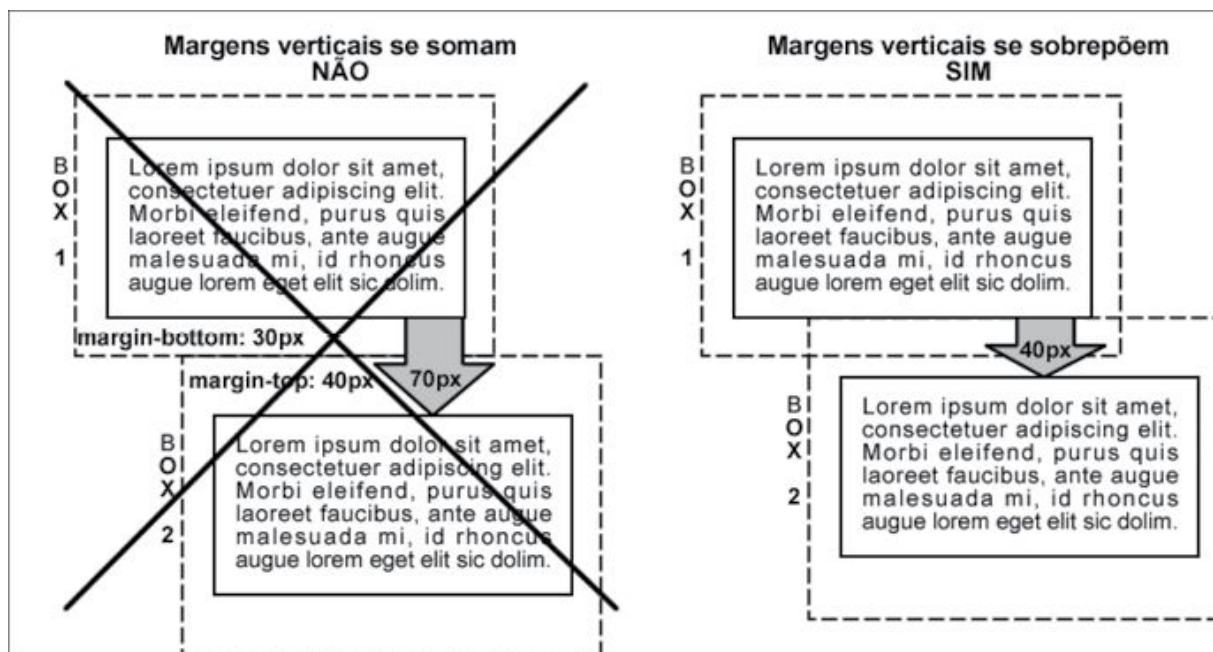


Figura 6.1 – Sobreposição de margens.

### 6.2.1.2 Elementos inline

Para elementos inline, a formatação dá-se em linha, na horizontal, e dentro do bloco que contém o box inline. Elementos inline admitem somente margens horizontais, ou seja, margin-left e margin-right. Margens verticais, ou seja, margin-bottom e margin-top são ignoradas.

Na marcação que se segue identificamos os seguintes boxes: um bloco container formado pelo elemento parágrafo `p` e três boxes inline, a saber: *Este parágrafo contém* e *elementos inline*, que são denominados boxes inline anônimos, e *três* que é um box inline contido no elemento inline `strong`.

```
<p>Este parágrafo contém <strong>três</strong> elementos inline</p>
```

Considere o parágrafo mostrado anteriormente inserido entre dois outros parágrafos e com a classe especial nele declarada, conforme a marcação HTML mostrada a seguir.

```
<p>Este é o primeiro parágrafo.</p>
```

```
<p class="especial">Este parágrafo contém <strong>três</strong> elementos
```

inline.</p>

<p>Este é o terceiro parágrafo.</p>

Vamos aplicar uma margem no box inline strong do segundo parágrafo e uma borda em todos os parágrafos somente para facilitar a visualização do exemplo, conforme mostrado nas regras de estilo a seguir.

```
p { border: 1px solid black; }
.especial > strong {
    margin-top: 3800px; /* será ignorada */
    margin-bottom: 5400px; /* será ignorada */
    margin-left: 150px; /* será aplicada */
    margin-right: 280px; /* será aplicada */
}
```

Na figura 6.2 mostramos o efeito de declarar margens para elementos inline.

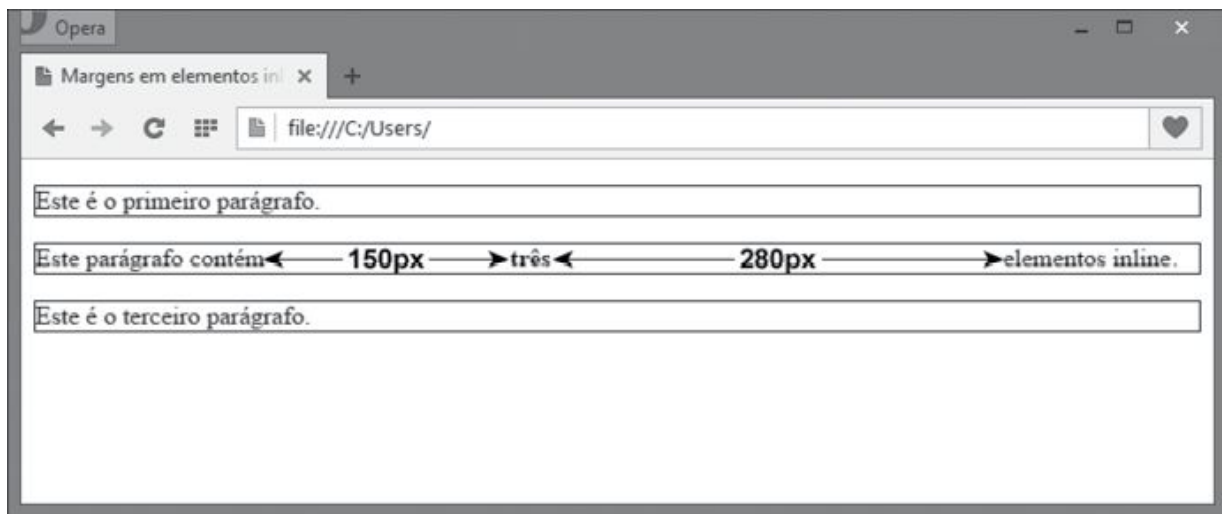


Figura 6.2 – Margens em boxes inline.

O arquivo deste exemplo (*margens-elementos-inline.html*) está disponível para consulta online e download na pasta *capitulo6*.

## 6.2.2 Esquema relativo

O esquema de posicionamento relativo é regido pela propriedade CSS `position` e seu valor `relative`. Esse posicionamento não retira o elemento posicionado do fluxo do documento, preservando o

espaço que ele ocupava antes de ser posicionado.

### 6.2.2.1 Posicionamento relativo

A declaração CSS `position: relative` sozinha não causa nenhum efeito no posicionamento de um box. Quando usada em conjunto com as propriedades `left`, `top`, `right` e `bottom`, movimenta o bloco da sua posição inicial a uma distância definida pelos valores declarados nessas propriedades.

As propriedades `left` e `right` definem o quanto o bloco deve ser deslocado para a direita ou à esquerda, respectivamente, e as propriedades `top` e `bottom` definem o deslocamento para baixo e para cima, respectivamente.

Para exemplificar o posicionamento relativo, considere três `div` (`div.um`, `div.dois` e `div.tres`) em sequência na marcação HTML e uma margem entre eles. Para o segundo `div` (`div.dois`) foi declarado um posicionamento relativo com definição das propriedades `left` e `top` e margem superior e inferior, conforme mostrado no código a seguir.

- **HTML**

```
<div class="um">Este é o elemento DIV 1</div>  
<div class="dois">Este é o elemento DIV 2</div>  
<div class="tres">Este é o elemento DIV 3</div>
```

- **CSS**

```
div {  
    width: 170px  
    height: 40px;  
    border: 1px solid black;  
}  
  
div.dois {  
    position: relative;  
    left: 60px;  
    top: 15px;  
    margin: 20px 0;  
}
```

Na figura 6.3 é mostrado à esquerda a renderização padrão dos

três elementos div e à direita a renderização com a declaração de posicionamento relativo.

O arquivo desse exemplo (*posicionamento-relativo.html*) está disponível para consulta online e download na pasta *capitulo6*.

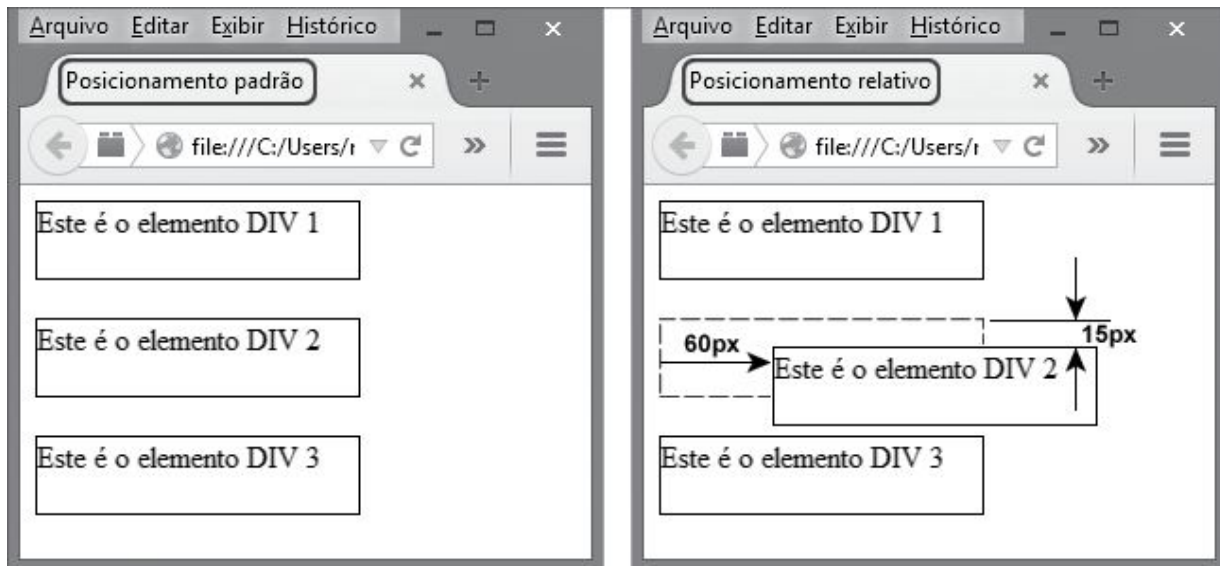


Figura 6.3 – Posicionamento relativo.

Notar que o bloco deslocado (div.dois) com a declaração `position: relative` sai da posição padrão, mas o espaço que ele ocupava naquela posição é preservado.

Elementos inline também podem ser deslocados da sua posição padrão com o uso dessa declaração. Seguem a marcação e a regra de estilo que exemplificam esse caso.

- **HTML**

```
<h2>Neste cabeçalho <span>estas palavras</span> foram deslocadas de sua  
posição padrão.</h2>
```

- **CSS**

```
span {  
    position: relative;  
    left: 60px;  
    top: 40px;  
    border: 1px solid black;  
}
```

A figura 6.4 mostra o efeito do deslocamento de um elemento inline conforme os códigos anteriores.

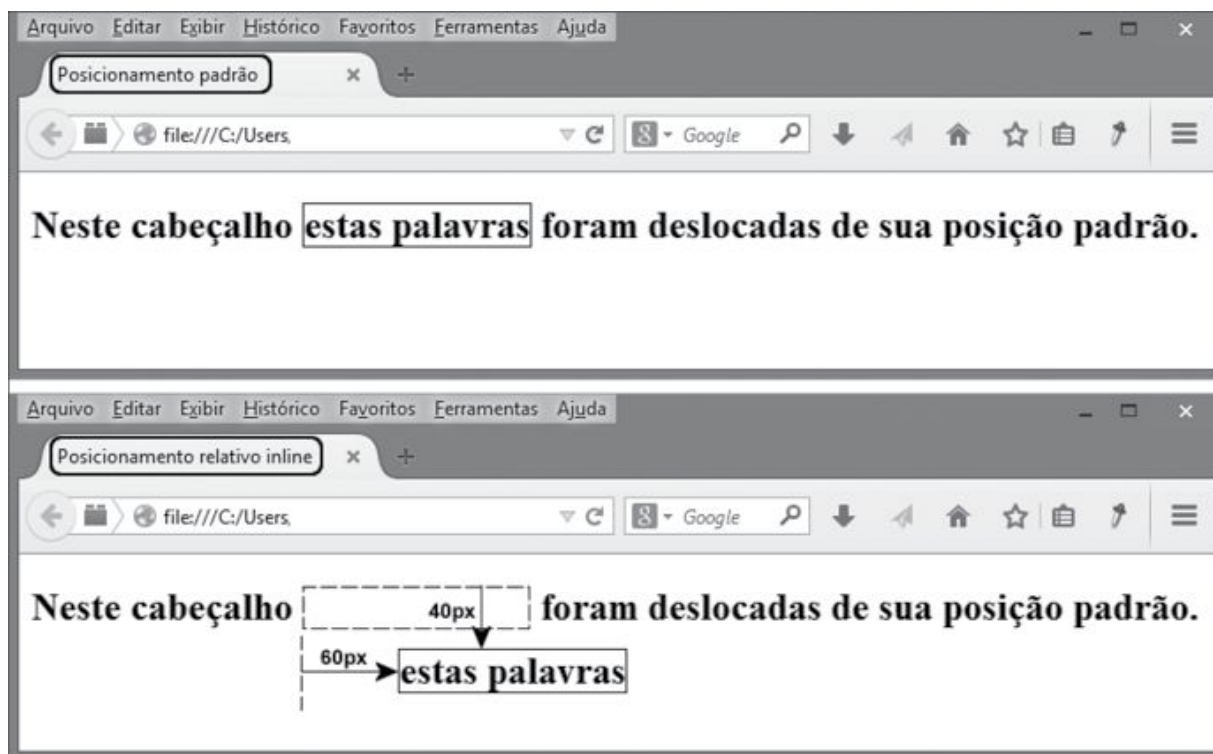


Figura 6.4 – Posicionamento relativo inline.

O arquivo desse exemplo (*posicionamento-relativo-inline.html*) está disponível para consulta online e download na pasta *capitulo6*.

Notar que o elemento inline deslocado (`span`) com a declaração `position: relative` sai da posição padrão, mas o espaço que ele ocupava naquela posição é preservado.

### 6.2.3 Esquema com float

O esquema de posicionamento com float ou posicionamento *flutuado* é definido pela propriedade CSS `float` e seus valores `left`, `right`, `none` e `inherit` (herdado). Outra propriedade CSS usada nesse esquema de posicionamento é a propriedade `clear` e seus valores `none`, `left`, `right`, `both` e `inherit` (herdado).

Segundo as regras do posicionamento flutuado, o box é retirado de sua posição no fluxo do documento e flutuado para a direita ou



para a esquerda. Ao contrário do posicionamento relativo estudado anteriormente, nesse esquema de posicionamento, o espaço original ocupado pelo box não será deixado livre, mas ocupado pelo elemento que se segue no fluxo do documento.

### 6.2.3.1 Flutuando elementos nível de bloco

Considere vários blocos dentro de um mesmo elemento container. O primeiro bloco flutuado desloca-se lateralmente, para a esquerda (valor: left) ou para a direita (valor: right), até tocar a borda lateral (esquerda ou direita) e superior do elemento container.

O segundo bloco flutuado desloca-se igualmente até tocar a borda do primeiro, e assim sucessivamente, enquanto houver espaço horizontal no elemento container. Faltando espaço, o próximo bloco ocupará uma linha abaixo, e assim por diante.

Para exemplificar, considere a marcação HTML para seis elementos div e um container e as regras de estilo conforme mostradas a seguir.

- **HTML**

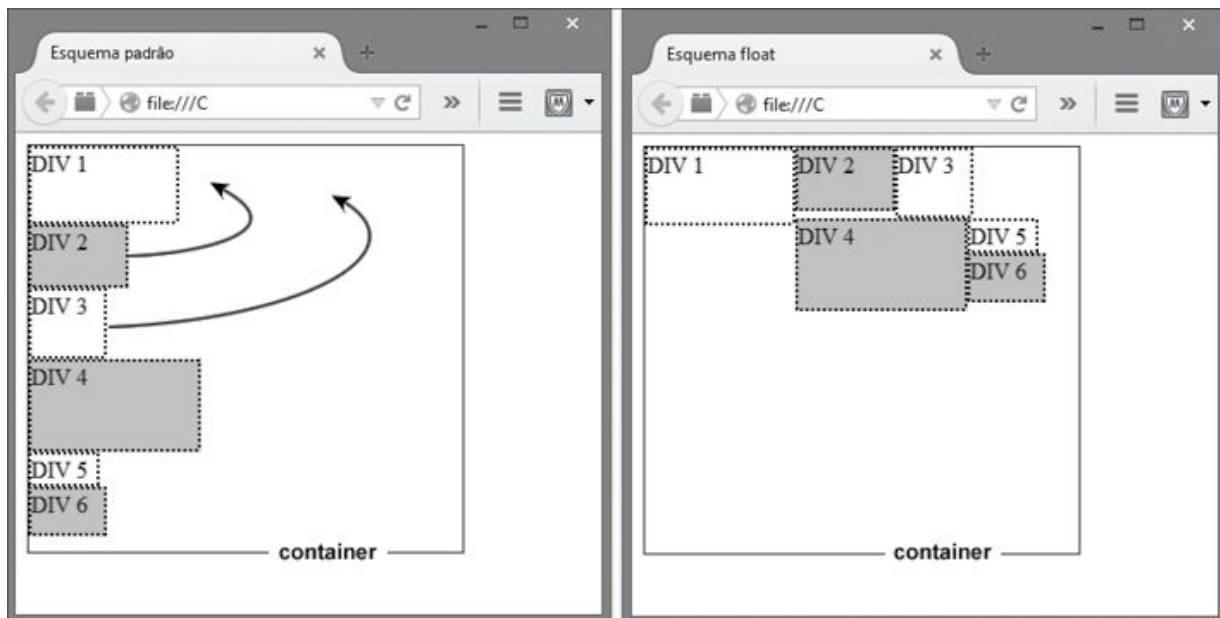
```
<div class="container">
  <div class="um">DIV 1</div>
  <div class="dois">DIV 2</div>
  <div class="tres">DIV 3</div>
  <div class="quatro">DIV 4</div>
  <div class="cinco">DIV 5</div>
  <div class="seis">DIV 6</div>
</div>
```

- **CSS**

```
.container {
  border: 1px solid black;
  width: 300px;
  height: 280px;
}
div {border: 2px dotted black;}
.container > div:nth-child(2n) {background: #ccc;}
.um {width: 100px; height: 50px;}
```

```
.dois {width: 65px; height: 40px;}  
.tres {width: 50px; height: 45px;}  
.quatro {width: 115px; height: 60px; }  
.cinco {width: 45px; height: 20px;}  
.seis {width: 50px; height: 30px;}
```

A figura 6.5 apresenta uma ilustração da regra para flutuar blocos. Na figura, vê-se à esquerda, a renderização dos seis blocos na posição padrão e à direita o resultado de declarar `float: left` para os blocos.



*Figura 6.5 – Posicionamento float.*

Observe, na parte direita da figura, que o espaço horizontal à direita do DIV3 não foi suficiente para acomodar o DIV4. Assim, o DIV4 passou para a linha seguinte, mas (importante notar) não encostou na borda esquerda do container, pois a altura do DIV1 não permitiu.

Para melhor entender o posicionamento flutuado, imagine que os elementos div estão cheios de gás que os fazem flutuar seguindo uma trajetória conforme mostrada nas duas setas na parte esquerda da figura, isto é, eles sobem até encostar em um “teto” (parte superior do container ou inferior de um div já flutuado) e depois se deslocam para a esquerda (no caso de `float: left`) até encostar em

uma “parede” (parte esquerda do container ou esquerda de um elemento div já flutuado).

O arquivo desse exemplo (*posicionamento-float.html*) está disponível para consulta online e download na pasta *capitulo6*.

Faça duas cópias desse arquivo e, em uma delas, retire a regra para flutuar à esquerda e observe a renderização padrão dos elementos div, e na outra flutue à direita, substituindo, na regra para flutuar os elementos div, o valor `left` pelo valor `right`. Tal substituição permitirá observar a flutuação à direita.

Uma aplicação prática de uso desse tipo de posicionamento para uma sequência de blocos, conforme mostrado anteriormente, é na construção de uma barra de navegação. Barras de navegação, em geral, são marcadas com o elemento HTML lista, nas quais cada item é um link:

#### • HTML

```
<ul>
  <li><a href="home.html">Home</a></li>
  <li><a href="quemsomos.html">Quem Somos</a></li>
  <li><a href="portfolio.html">Portfólio</a></li>
  <li><a href="contato.html">Contato</a></li>
</ul>
```

Em uma lista não ordenada, como a mostrada anteriormente, o elemento `ul` é o container para os elementos `li`, que são os blocos a serem flutuados. Se definirmos uma largura para o elemento `ul` que seja suficiente para acomodar a soma das larguras dos elementos `li` e flutuarmos à esquerda tais elementos, obteremos uma lista na horizontal.

A declaração `clear` com seus valores: `none`, `left`, `right`, `both` e `inherit` (herdado) tem a finalidade de “limpar” o que está abaixo de boxes flutuados, como veremos adiante.

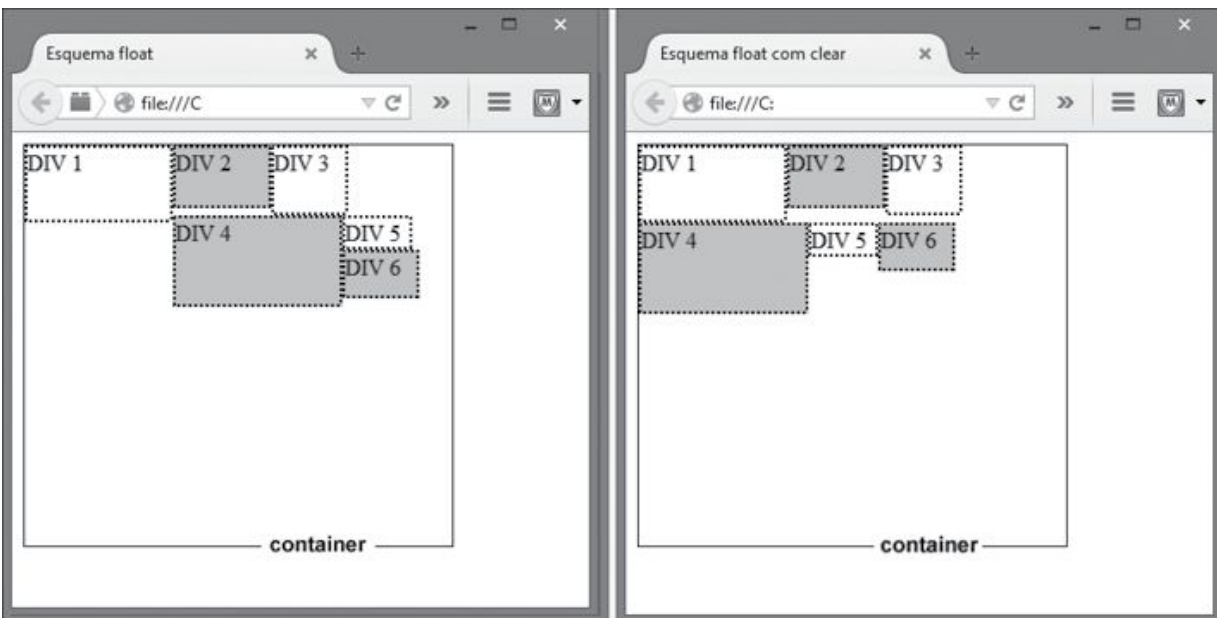
Se a flutuação for à esquerda, use `clear: left`; se à direita, `clear: right`. Ou, faça como a maioria dos desenvolvedores e use sempre `clear: both` que se aplica a ambos os casos de flutuação e vai funcionar se

no futuro você resolver inverter o esquema de flutuação.

Na figura 6.6 mostramos o mesmo caso anterior, de flutuação de seis elementos div. À esquerda da figura, o comportamento sem uso da declaração `clear: both;` e, à direita da figura, com uso da declaração `clear: both` para um elemento nível de bloco logo após o DIV3. Na marcação, o elemento logo após o DIV3 é o DIV4. Assim, nesse caso, declaramos `clear: both` para o DIV4.

O arquivo desse exemplo (*posicionamento-float-clear.html*) está disponível para consulta online e download na pasta *capitulo6*.

Outro comportamento que deve ser considerado quando usamos o posicionamento flutuado diz respeito ao elemento container dos boxes flutuados. Se não for especificada uma altura, o container de um ou mais elementos, por padrão, expande-se para poder conter os elementos dentro dele.



*Figura 6.6 – Posicionamento float com clear.*

Uma vez que elementos flutuados são retirados do fluxo normal do documento, tudo se passa como se eles deixassem de forçar a expansão da altura do container, que, se não tiver sido especificada (a altura), torna-se zero.

Nos dois exemplos mostrados anteriormente, notar que a soma

total das alturas dos seis elementos div é igual a 50px + 40px + 45px + 60px + 20px + 30px = 245px e que foi declarada uma altura igual a 280px para o container. Portanto há uma folga de 35px embaixo do container como se pode observar na parte esquerda das figuras dos exemplos. Notar ainda que, após a flutuação dos elementos div, o container manteve-se com sua altura declarada igual a 280px.

Vamos retirar a declaração de altura do container, flutuar os elementos div e observar o comportamento da altura do container antes e depois de declarar a propriedade clear.

Na figura 6.7 é mostrado o comportamento da altura do container de elementos flutuados, quando ela, altura, não é especificada. Observar, na parte esquerda da figura, o comportamento padrão do container, assumindo uma altura total igual a zero, pois os elementos div flutuados foram retirados do fluxo normal do documento. Observe à direita da figura o efeito de declarar clear para um elemento colocado logo após o DIV6. A declaração clear além de “limpar” elementos flutuados também faz com que o container dos elementos flutuados expanda para contê-los.

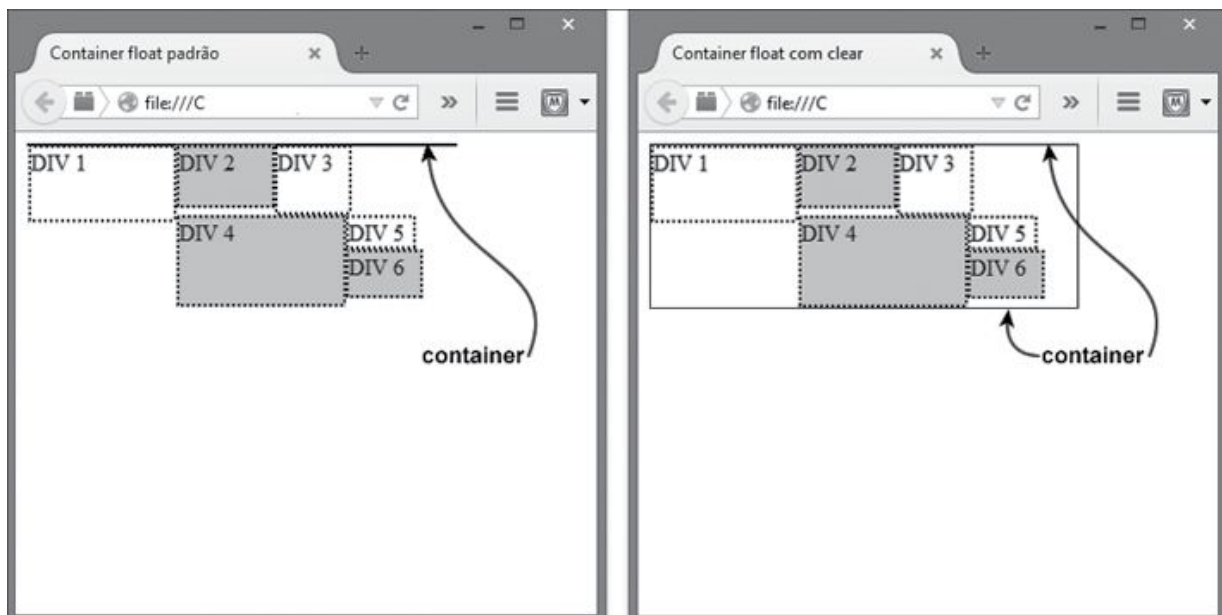


Figura 6.7 – Altura do container de boxes flutuados.

O arquivo desse exemplo (*container-de-float-clear.html*) está disponível para consulta online e download na pasta *capitulo6*.

A técnica para “limpar” elementos flutuados produz dois efeitos: faz com que seja restabelecido o fluxo normal dos elementos, isto é, cada elemento nível de bloco vai sendo renderizado na sequência em que se encontra na marcação; e faz com que o container dos elementos flutuados se expanda para contê-los.

Como regra geral, toda vez que se usa posicionamento com flutuação deve-se “limpar” os elementos flutuados.

A técnica mais antiga para “limpar” elementos flutuados consiste em marcar um elemento vazio (sem conteúdos) logo após a marcação dos elementos flutuados e declarar para eles `clear: both`. Essa será a solução que adotaremos neste livro, mas existem outras soluções, consideradas melhores, sem uso de marcação adicional. Ver: <http://kwz.me/Ge>.

Para a técnica adotada neste livro, crie um `div` vazio e atribua a ele a classe `clear`.

Nas CSS declare `clear: both` para a classe `clear`, como mostrado a seguir.

- **HTML**

```
<div class="clear"></div>
```

- **CSS**

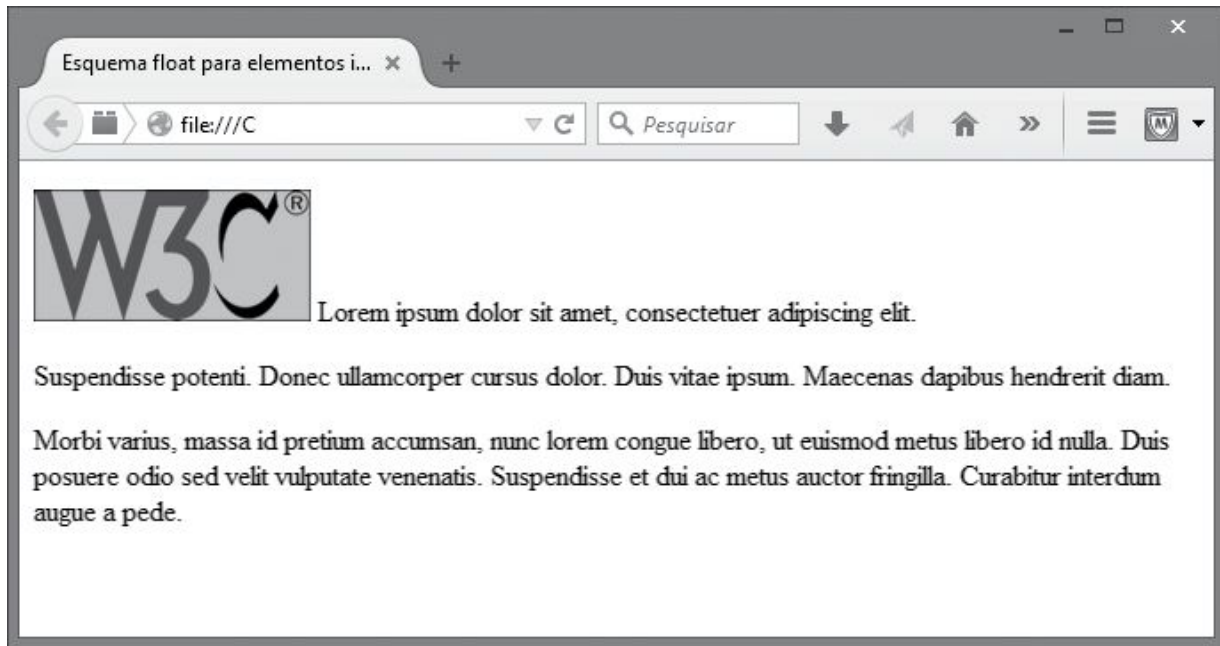
```
.clear { clear: both; }
```

### 6.2.3.2 Flutuando elementos inline

Vamos tomar como exemplo o caso de uma imagem dentro de um parágrafo. O elemento container é o parágrafo, e o elemento inline a flutuar é a imagem. Considere dois parágrafos no fluxo do documento e a imagem no primeiro parágrafo, conforme a seguinte marcação:

```
<p>
Lorem ipsum dolor sit amet, consectetur adipiscing elit.</p>
<p>Suspendisse potenti. ...</p>
<p>Morbi varius, ...</p>
```

A figura 6.8 mostra a renderização padrão da marcação.



*Figura 6.8 – Imagem inline em parágrafo.*

Notar que o texto do primeiro parágrafo inicia-se no canto inferior esquerdo da imagem e os demais parágrafos seguem no fluxo normal do documento.

O arquivo desse exemplo (*posicionamento-float-elementos-inline.html*) está disponível para consulta online e download na pasta *capitulo6*.

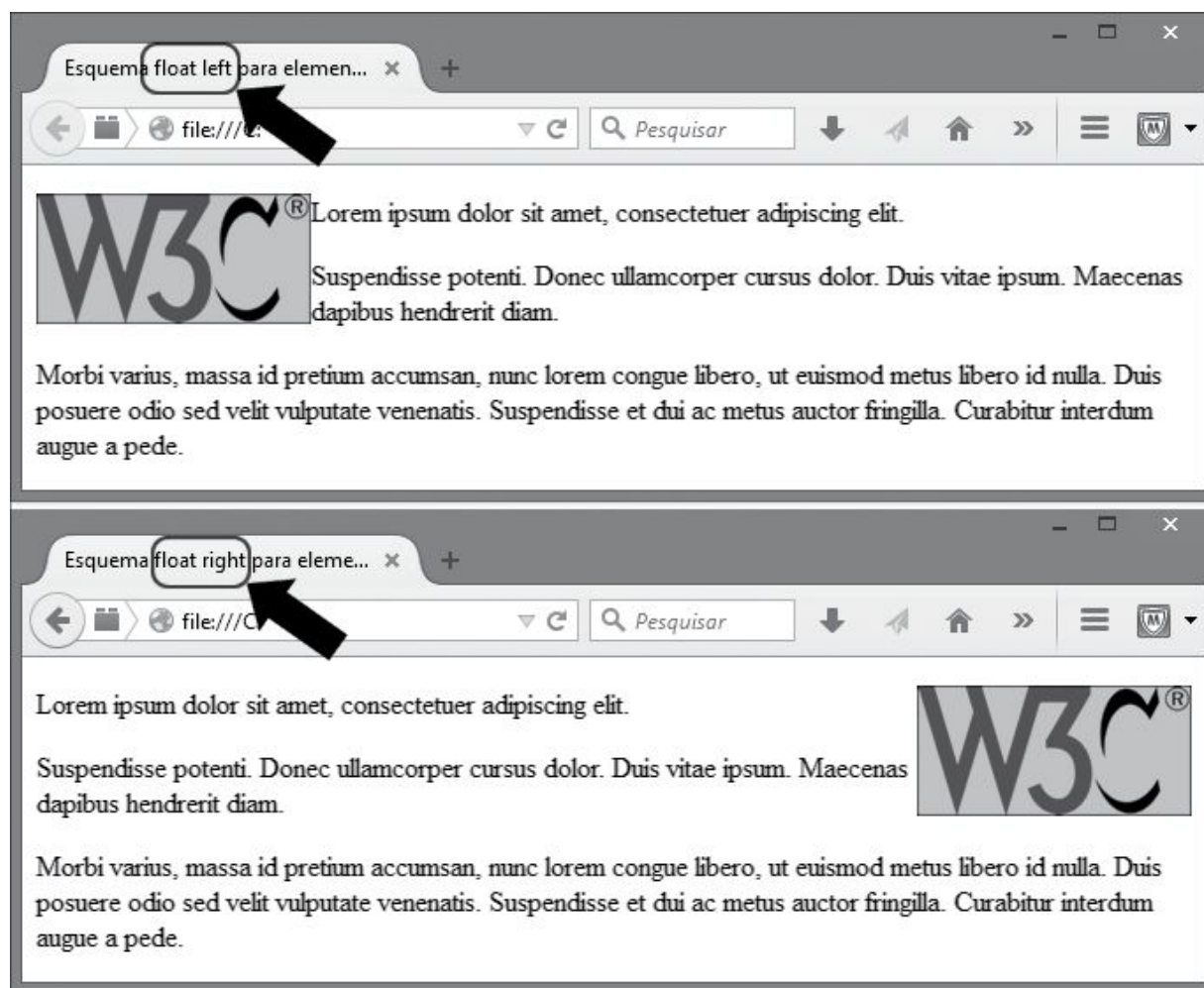
Para flutuar a imagem à esquerda ou à direita, aplicamos a seguinte regra CSS:

```
img {float: left;}
```

ou

```
img {float: right;}
```

Obtendo o efeito de flutuar à esquerda e à direita, conforme mostrado na figura 6.9.



*Figura 6.9 – Imagem float left e right.*

Observar que, para elementos inline, o comportamento é diferente daquele de boxes flutuados. A imagem flutuada foi deslocada para uma posição até tocar a borda do seu elemento container (no exemplo um parágrafo), e tanto o texto conteúdo do container quanto o elemento bloco que se segue no fluxo do documento (o segundo parágrafo) ocuparam uma posição em volta da imagem. Se a altura da imagem fosse maior, o terceiro parágrafo também se posicionaria ao lado da figura, e assim por diante para todos os elementos que se seguem no fluxo do documento. Na prática, tal comportamento é usado para alinhar imagens à esquerda ou à direita de um texto.

Os arquivos desse exemplo (*posicionamento-float-left-elementos-inline.html* e *posicionamento-float-right-elementos-inline.html*) estão



disponíveis para download, encontram-se na pasta *capitulo6*.

Já sabemos que elementos flutuados são retirados do fluxo normal do documento, e foi por essa razão que o segundo parágrafo “subiu” e se posicionou ao lado da imagem. Sabemos também que para evitar a “subida” do segundo parágrafo, basta “limpar” o elemento flutuado acrescentando a seguinte marcação.

- **HTML**

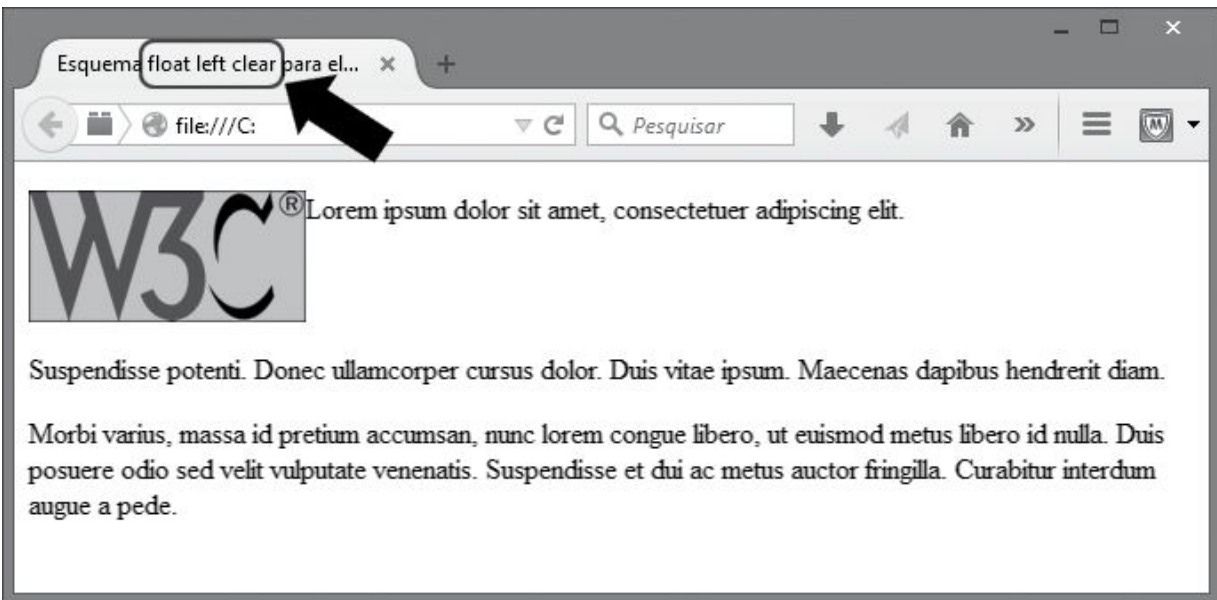
```
<p>
Lorem ipsum dolor sit amet, consectetur adipiscing elit.</p>
<div class="clear"></div>
<p>Suspendisse potenti. ...</p>
<p>Morbi varius, ...</p>
```

- **CSS**

```
.clear { clear: both; }
```

A figura 6.10 mostra o efeito de “limpar” o elemento flutuado.

O arquivo desse exemplo (*posicionamento-float-left-clear-elementos-inline.html*) está disponível para consulta online e download na pasta *capitulo6*.



*Figura 6.10 – Imagem float left com clear.*

## 6.2.4 Esquema absoluto

O esquema de posicionamento absoluto é regido pela propriedade CSS `position` e seus valores `absolute` e `fixed`. Ao contrário do posicionamento com o valor `relative`, conforme estudado em [6.2.2.1], tal posicionamento retira o elemento posicionado do fluxo do documento, fazendo com que o elemento que se segue ocupe seu lugar.

### 6.2.4.1 Posicionamento absoluto

A declaração CSS `position: absolute;` sozinha não causa qualquer efeito no posicionamento do box, mas faz com que o elemento seguinte na marcação se desloque para a posição que ele ocupava antes de ser posicionado. Um box deslocado pode ou não ser obscurecido pelo box com o qual compartilhe a mesma posição. Na seção 6.3 veremos a propriedade CSS `z-index` que rege o posicionamento de boxes sobrepostos.

A declaração `position: absolute;` quando usada em conjunto com as propriedades `left`, `top`, `right` e `bottom`, movimenta o bloco da sua posição inicial de uma distância definida pelos valores declarados em tais propriedades. Medidas de comprimento CSS [4.4] são empregadas para definir as coordenadas do deslocamento.

As propriedades `left` e `right` definem o quanto o bloco deve ser deslocado para a direita ou à esquerda, respectivamente, e as propriedades `top` e `bottom` definem o deslocamento para baixo e para cima, respectivamente. Contudo, ao contrário do que ocorre com o posicionamento relativo, no qual a referência para o deslocamento (origem das coordenadas) é o próprio box a ser posicionado, quando se trata de posicionamento absoluto, a referência não é sempre a mesma e depende do que chamamos de *contexto de posicionamento*.

Quem estabelece o contexto de posicionamento é o desenvolvedor, e a regra para criar um contexto de posicionamento é a descrita a seguir.

Ao posicionar um box de forma absoluta, o contexto de posicionamento (origem das coordenadas) é o elemento ancestral mais próximo para o qual se tenha declarado a propriedade position com valor relative, absolute ou fixed. Se não houver elemento ancestral posicionado, o contexto de posicionamento é a viewport (em geral, a janela do navegador), e, nesse caso, chamamos esse contexto de contexto padrão.

Para exemplificar os diferentes casos de posicionamento absoluto, usaremos a seguinte marcação HTML:

```
<p>Lorem ipsum dolor sit amet...</p>
<div class="container">
  <p>Morbi varius, massa ...</p>
  <p class="tres">> Suspendisse potenti. Donec...</p>
  <p>Curabitur henderit ...</p>
</div>
```

O código marca quatro elementos p sendo que o segundo, terceiro e quarto estão contidos em um elemento div com a classe container. Na folha de estilo definimos bordas para os elementos da marcação e uma cor de fundo para fins de clareza nas figuras, mostrando os posicionamentos. A folha de estilo inicial é:

```
div {
  width:360px;
  padding:10px 20px;
  border: 2px solid black;
}
p { border:2px dotted black; padding: 10px 20px; }
.tres { background: #ddd; }
```

A figura 6.11 mostra a renderização padrão da marcação.

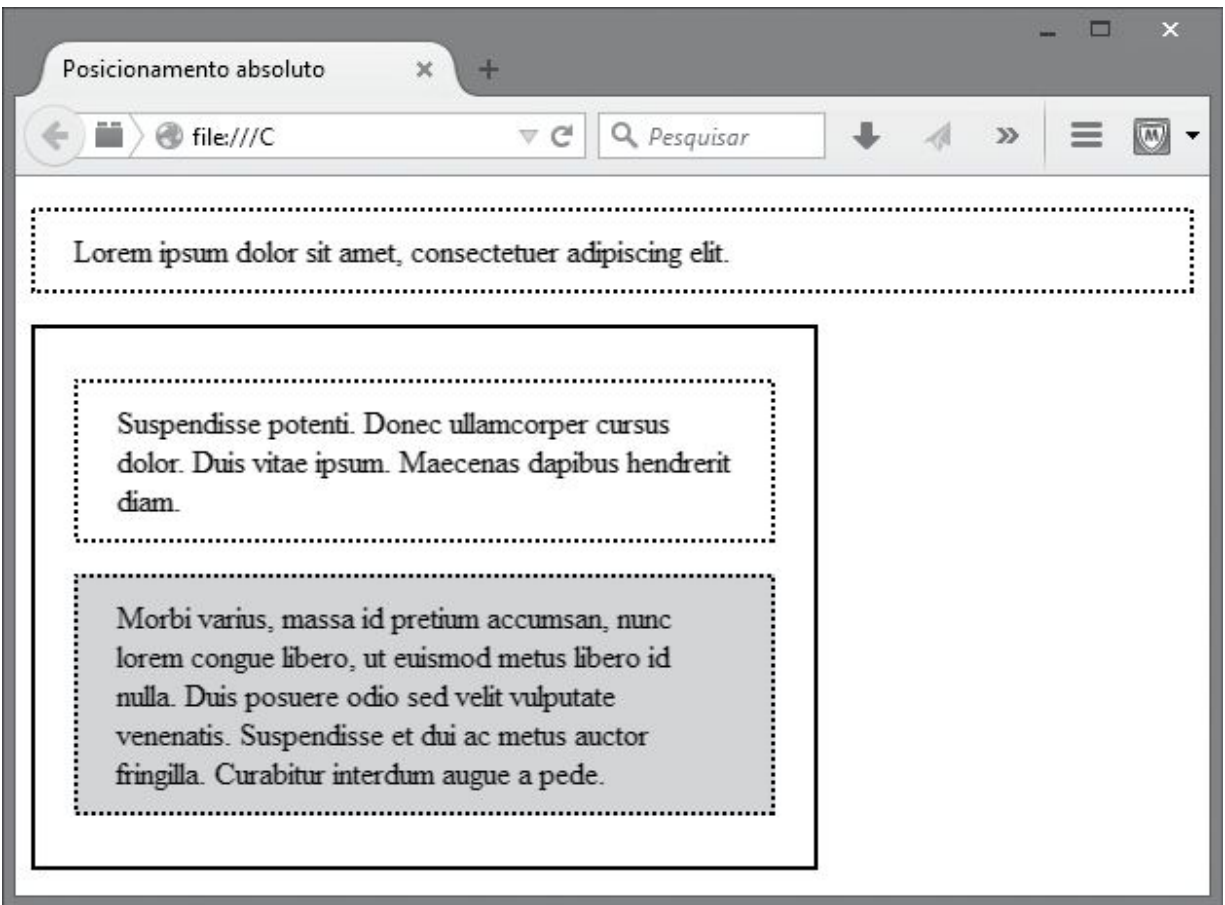


Figura 6.11 – Renderização padrão.

O arquivo desse exemplo (*posicionamento-absoluto.html*) está disponível para consulta online e download na pasta *capitulo6*.

Vamos, simplesmente, declarar posicionamento absoluto para o parágrafo p.tres acrescentando na folha de estilo o seguinte.

```
.tres {  
    position: absolute;  
    background: #ddd;  
}
```

A figura 6.12 mostra o efeito de declarar posição absoluta sem definir coordenadas.

Observar que a simples declaração de `position: absolute;` sem definir as coordenadas do deslocamento, fez com que o parágrafo três permanecesse no mesmo lugar, liberando o espaço que ocupava e causando a subida do parágrafo quatro que o segue no fluxo do

documento. No contexto do documento, o parágrafo três obscureceu a parte do parágrafo quatro que ficou atrás dele, no eixo z. Notar ainda que, ao sair do fluxo normal do documento, o parágrafo três “libertou-se” do seu container e assumiu uma largura igual a 100%.

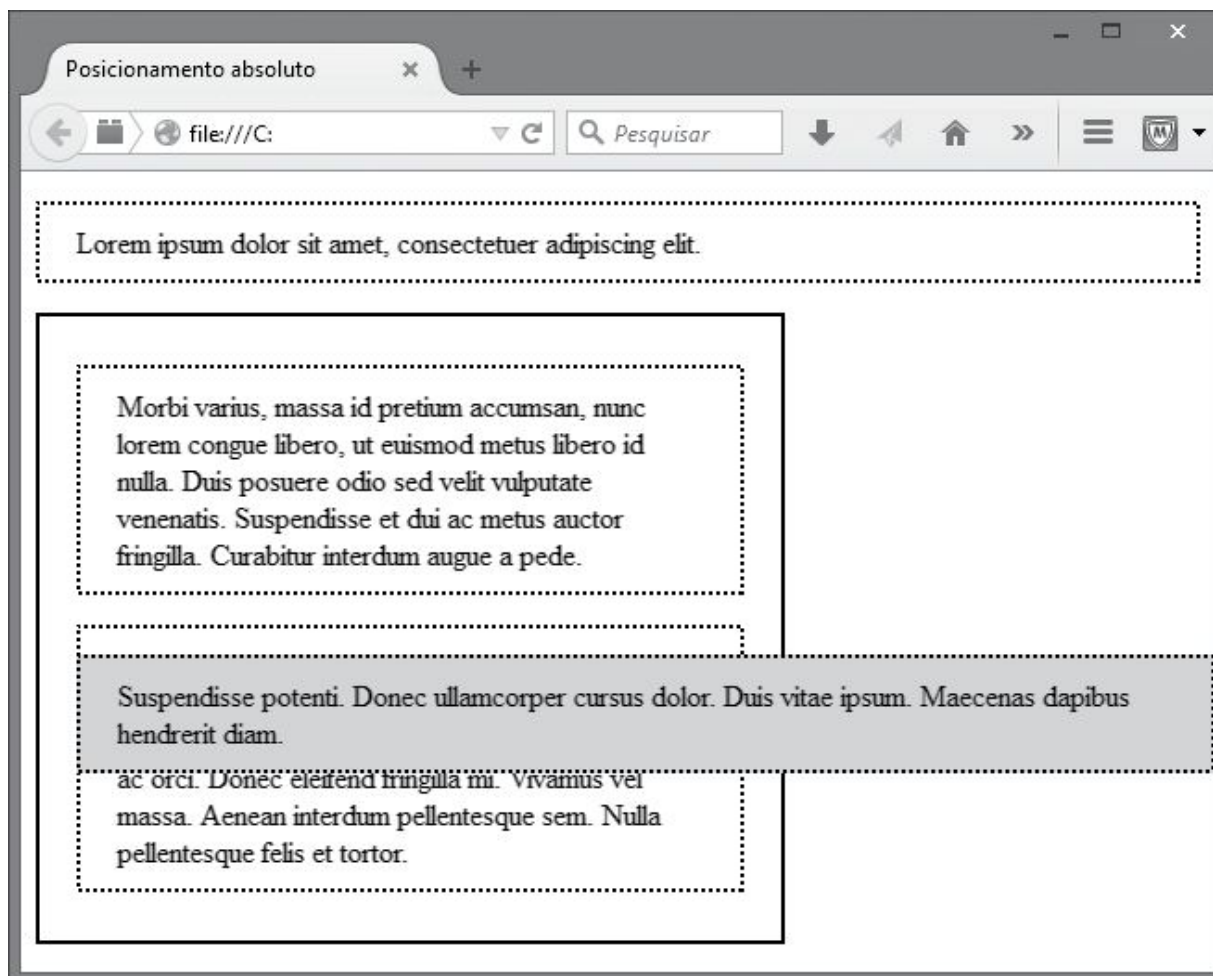


Figura 6.12 – Posicionamento absoluto sem coordenadas.

O arquivo desse exemplo (*posicionamento-absoluto-sem-coordenadas.html*) está disponível para consulta online e download na pasta *capitulo6*.

Vamos definir coordenadas para o posicionamento do parágrafo três, acrescentando as seguintes declarações na folha de estilo:

```
p.tres {  
  position: absolute;  
  background: #ddd;  
  left: 90px;
```

```
top: 125px;  
}
```

A origem padrão das coordenadas para o posicionamento absoluto (contexto padrão) é o canto superior esquerdo da área de renderização na janela do navegador (viewport). Contudo, essa origem pode mudar, dependendo do conceito conhecido como *contexto de posicionamento* conforme descrito anteriormente. A regra CSS mostrada anteriormente para o parágrafo três está definida no contexto de posicionamento padrão, e o resultado do deslocamento do parágrafo após aplicada a nova folha de estilo é apresentado na figura 6.13.

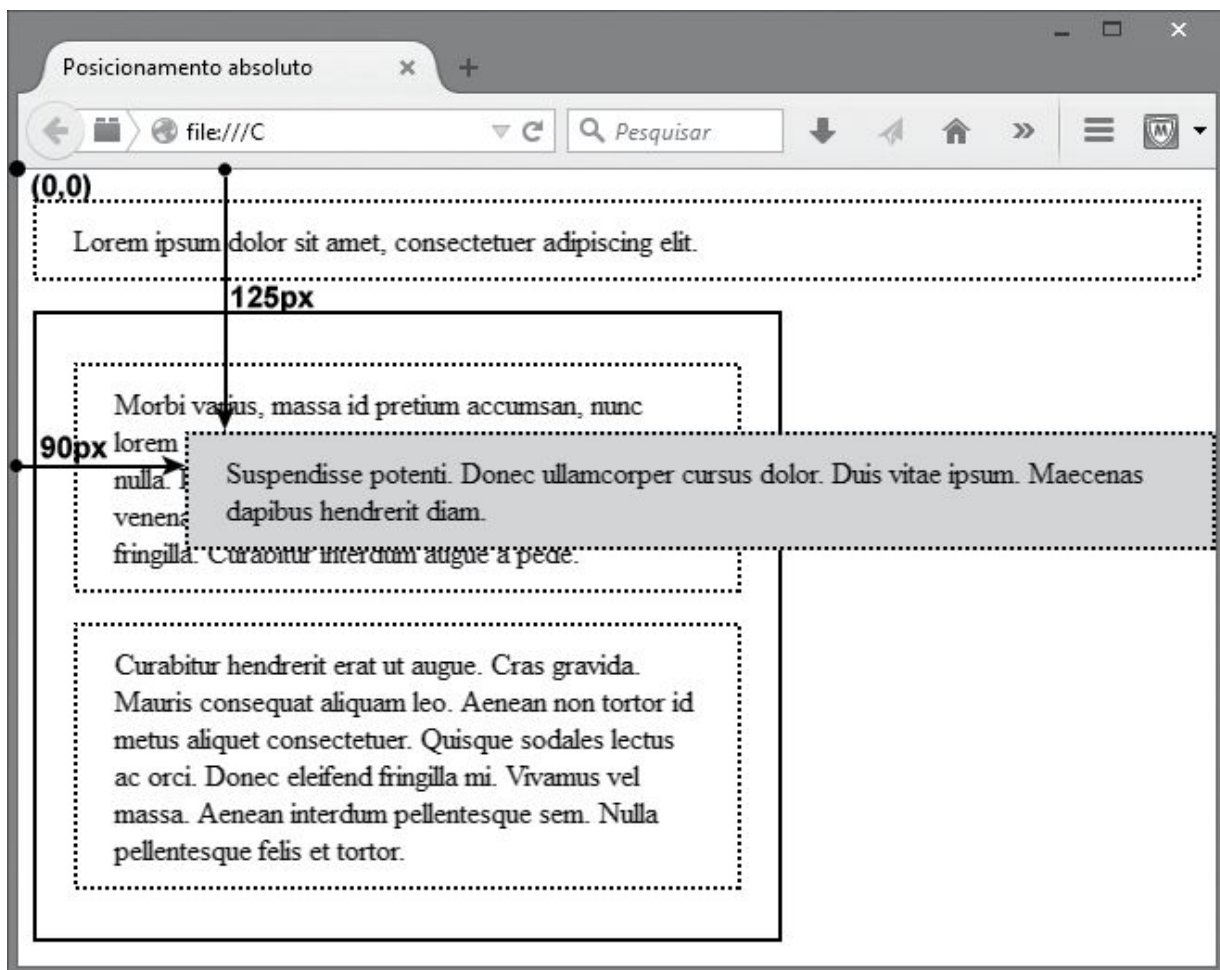


Figura 6.13 – Posicionamento absoluto com coordenadas.

O arquivo desse exemplo ([posicionamento-absoluto-com-coordenadas.html](#)) está disponível para consulta online e download na

*capítulo 6.*

O *contexto de posicionamento* é que estabelece a origem do sistema de coordenadas para posicionamento absoluto. A regra que rege o posicionamento estabelece que elementos posicionados com a declaração `position: absolute` serão deslocados, tomando como base para determinação das coordenadas o ancestral mais próximo posicionado, ou seja, no qual tenham sido declarados os valores `relative`, `absolute` ou `fixed` para a propriedade `position`. Não havendo ancestral posicionado, o contexto de posicionamento será a `viewport` (janela do navegador).

Para esclarecer o conceito de contexto de posicionamento, vamos supor o mesmo cenário do exemplo anterior e, adicionalmente, posicionar o elemento `div.container` que contém o parágrafo três. Vamos declarar `position: relative` para aquele `div` sem declarar coordenadas, mantendo-a em seu lugar no fluxo do documento. Assim fazendo, estabeleceremos um novo contexto de posicionamento para o parágrafo três, pois agora o elemento-pai do parágrafo foi posicionado, e a origem do sistema de coordenadas será tomada a partir dele. A regra CSS a acrescentar na folha de estilo é a seguinte:

```
div.container {position: relative;}
```

Na figura 6.14 é mostrado o efeito de se mudar o contexto de posicionamento.

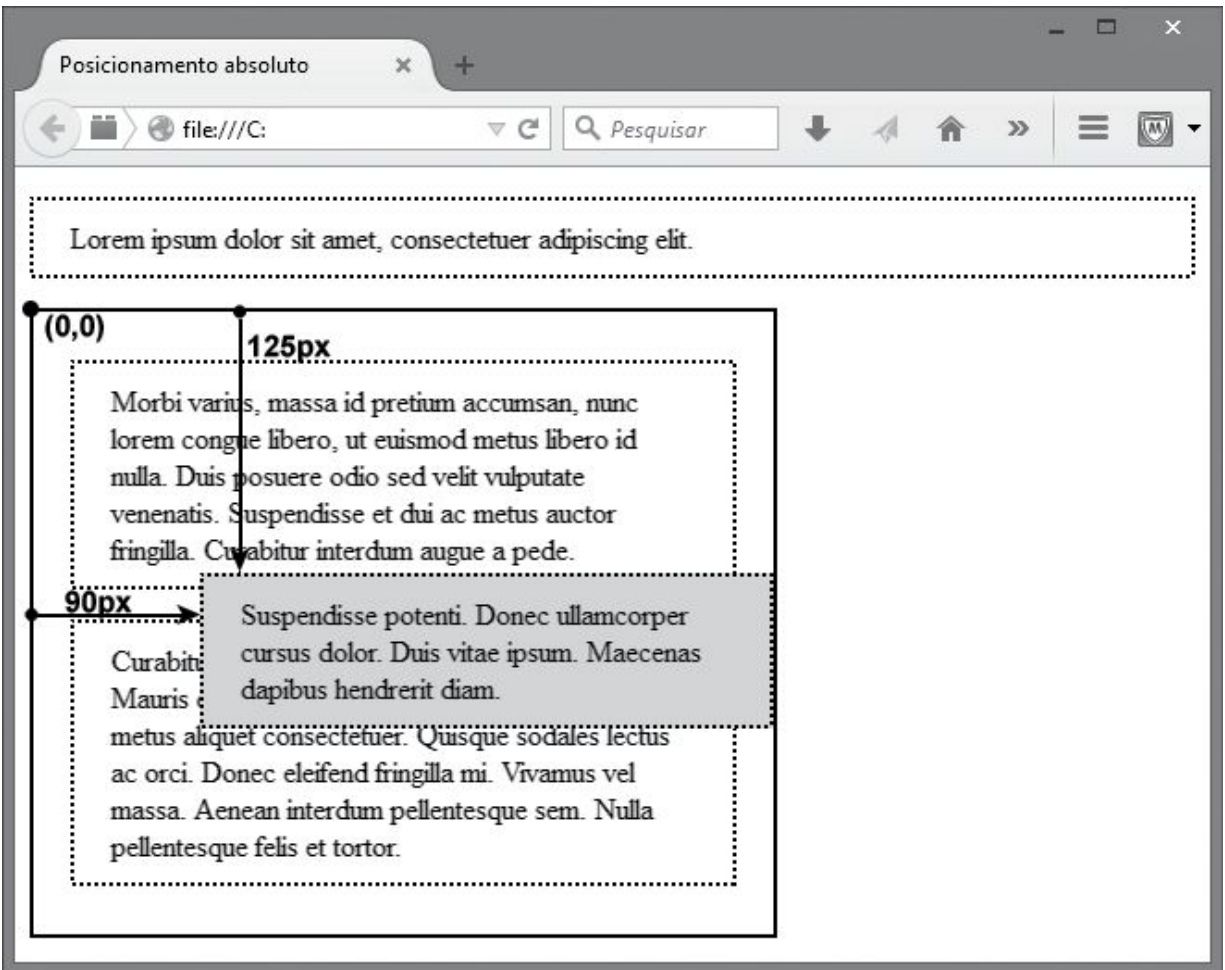


Figura 6.14 – Posicionamento absoluto em contexto.

Observe que neste caso a referência para a contagem das coordenadas é o `div.container`, e não mais a viewport, pois foi estabelecido um novo contexto de posicionamento com uso da declaração `position: relative` para aquele container.

O arquivo desse exemplo (*posicionamento-absoluto-em-contexto.html*) está disponível para consulta online e download na pasta *capitulo6*.

Elementos posicionados de forma absoluta podem ser colocados em qualquer lugar, inclusive fora dos limites de seu elemento-pai. Podemos, também, concluir que, quando posicionamos um elemento de forma absoluta, este cria um contexto de posicionamento para os seus elementos-filho.



Boxes inline, quando posicionados de forma absoluta, comportam-se como



elementos nível de bloco.

## **6.2.5 Esquema estático**

Os demais valores possíveis para a propriedade CSS `position` são: `static`, `fixed` e `inherit` (herdado). O valor `static` é o valor inicial ou padrão de posicionamento. Os boxes permanecem como no fluxo normal do documento. O uso dessa declaração é feito com a finalidade de sobrescrever um posicionamento declarado anteriormente. Os valores de posicionamento `left`, `top`, `right` e `bottom` não se aplicam para esse posicionamento.

## **6.2.6 Posicionamento fixo**

O posicionamento fixo faz-se com a declaração `position: fixed`; e pode ser considerado um caso especial de posicionamento absoluto. A diferença é que, nesse tipo de posicionamento, o box posicionado permanece fixo em relação a uma referência. O contexto de posicionamento é sempre a área de renderização na janela do navegador (viewport). Em mídia visual, o box fica fixo em relação à viewport e não se movimenta quando há rolagem da página. Em mídia impressa, o box será impresso em cada uma das folhas.

## **6.2.7 Posicionamento com z-index**

A propriedade CSS `z-index` rege o empilhamento de boxes segundo o eixo `z`. O eixo `z` é perpendicular à tela do monitor e tudo se passa como em um sistema tridimensional com coordenadas horizontal, vertical e coordenada `z` ou de profundidade. A propriedade `z-index` determina qual box fica à frente.

Quando retiramos um ou mais boxes do fluxo normal do documento com uso da propriedade `position`, pode ocorrer sobreposição de boxes conforme vimos nos exemplos de posicionamento absoluto mostrados nas figuras 6.12, 6.13 e 6.14 anteriores, quando um parágrafo foi colocado sobre outro. Havendo

sobreposição de boxes posicionados, entra em cena a propriedade z-index, a qual regula a ordem de empilhamento no eixo z, ou seja, qual elemento está mais próximo do usuário e como eles se distribuem em profundidade.

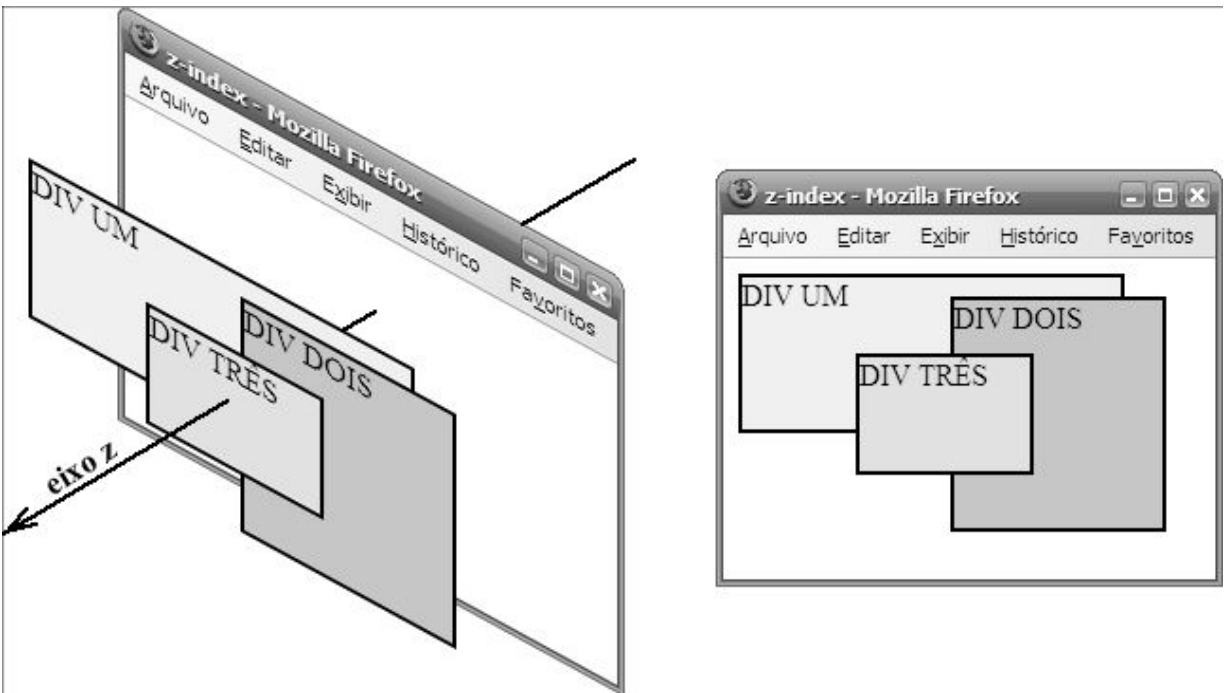
Considere a marcação a seguir, na qual três elementos div se seguem no fluxo do documento:

```
<div class="um">DIV UM</div>  
<div class="dois">DIV DOIS</div>  
<div class="tres">DIV TRÊS</div>
```

E a folha de estilo posicionando os três div.

```
div {  
    border: 2px solid #000;  
    position: absolute;  
}  
div.um {  
    width: 200px;  
    height: 80px;  
    background: #cff;  
}  
div.dois {  
    width: 110px;  
    height: 120px;  
    left: 120px;  
    top: 20px;  
    background: #6f9;  
}  
div.tres {  
    width: 90px;  
    height: 60px;  
    left: 70px;  
    top: 50px;  
    background: #fcf;  
}
```

Na figura 6.15 mostramos à esquerda uma vista tridimensional da renderização dos elementos div posicionados e sobrepostos, na qual acrescentamos o eixo z de profundidade, e à direita uma visão bidimensional da renderização.



*Figura 6.15 – Eixo z de profundidade.*

Quando não há declaração explícita de z-index, cabe ao agente de usuário determinar a ordem de empilhamento. A maioria dos navegadores adota como critério posicionar mais à frente os elementos que aparecem por último na marcação. Assim, conforme mostrado na figura 6.15, o elemento div mais próximo do usuário é o três, sendo, em consequência, posicionado mais à frente, seguindo-se o dois e, por último, o um.

Considere os seguintes acréscimos na folha de estilo do exemplo anterior:

```
div {  
    border: 2px solid #000;  
    position: absolute;  
}  
div.um {  
    width: 200px;  
    height: 80px;  
    background: #cff;  
    z-index: 3;  
}  
div.dois {
```

```
width: 110px;
height: 120px;
left: 120px;
top: 20px;
background: #6f9;
z-index: 2;
}
div.tres {
width: 90px;
height: 60px;
left: 70px;
top: 50px;
background: #fcf;
z-index: 1;
}
```

Definimos coordenadas z para cada um dos elementos-filho, estabelecendo a coordenada mais alta para o elemento div um e mais baixa para o elemento div três.

A figura 6.16 mostra o efeito causado pela nova folha de estilo.

A propriedade z-index controla o posicionamento dos elementos sobrepostos no sentido da profundidade. Notar que, com a declaração dessa propriedade, invertemos a ordem-padrão de renderização mostrada no exemplo anterior. Podemos estabelecer qualquer ordem de posicionamento para atender às necessidades do layout.



*Figura 6.16 – Declaração z-index.*

Os valores possíveis para a propriedade z-index são: auto, integer e inherit (valor herdado). O valor inicial ou default é auto. O valor integer (inteiro) é um número inteiro positivo ou negativo.



Alguns agentes de usuário tratam valores negativos de z-index de forma não prevista nas Recomendações do W3C para as CSS, ou seja, em lugar de posicionar os elementos definidos com aqueles valores na ordem natural de crescimento do eixo z, colocam esses elementos atrás da tela, fazendo com que desapareçam da vista do usuário.

Não existe obrigatoriedade quanto à sequência de números a adotar para declarar os valores inteiros de z-index. A única regra é: quanto maior o valor, mais próximo do usuário o box é posicionado. Assim, uma sequência de valores 1, 2, 3, 4, 5 tem o mesmo efeito de 36, 120, 204, 999.

#### **6.2.7.1 Contexto de empilhamento**

Conforme vimos na seção anterior, o posicionamento em profundidade deu-se em relação à tela do monitor do usuário. Os elementos foram dispostos tendo como fundo ou base a viewport.

No entanto, o comportamento de renderização para posicionamento em profundidade não é tão simples assim, havendo outros fatores a considerar. As Recomendações do W3C para as

CSS preveem um *contexto de empilhamento* segundo o eixo dos z. No exemplo anterior, os três elementos div foram posicionados de forma absoluta, tendo como origem das coordenadas horizontal e vertical o canto superior da área de renderização do navegador.

Você se lembra do conceito *contexto de posicionamento*? Caso se lembre dele, siga em frente. Se não, faça uma revisão deste conceito voltando à seção 6.2.4.1 deste capítulo.

Todo elemento posicionado com o valor *relative*, *absolute* ou *fixed* cria um *contexto de posicionamento*, estabelecendo uma origem para contagem das coordenadas de posição horizontal e vertical. De modo similar, um elemento posicionado da forma citada anteriormente cria um *contexto de empilhamento*, estabelecendo uma origem para contagem da coordenada z de profundidade de seus elementos-filho. Para exemplificar esse conceito, vamos considerar a seguinte marcação:

```
<div id="a">  
  DIV A  
    <div class="a-um">DIV A-um</div>  
    <div class="a-dois">DIV A-dois</div>  
    <div class="a-tres">DIV A-tres</div>  
</div>  
  
<div id="b">  
  DIV B  
    <div class="b-um">DIV B-um</div>  
    <div class="b-dois">DIV B-dois</div>  
</div>
```

Na marcação, identificamos o div A que contém como elementos-filho os div a-um, div a-dois e div a-tres, e, em seguida, no fluxo do documento, o div B com os div b-um e div b-dois.

E a folha de estilo com aplicação de bordas, cor de fundo e dimensões para fins de clareza do exemplo:

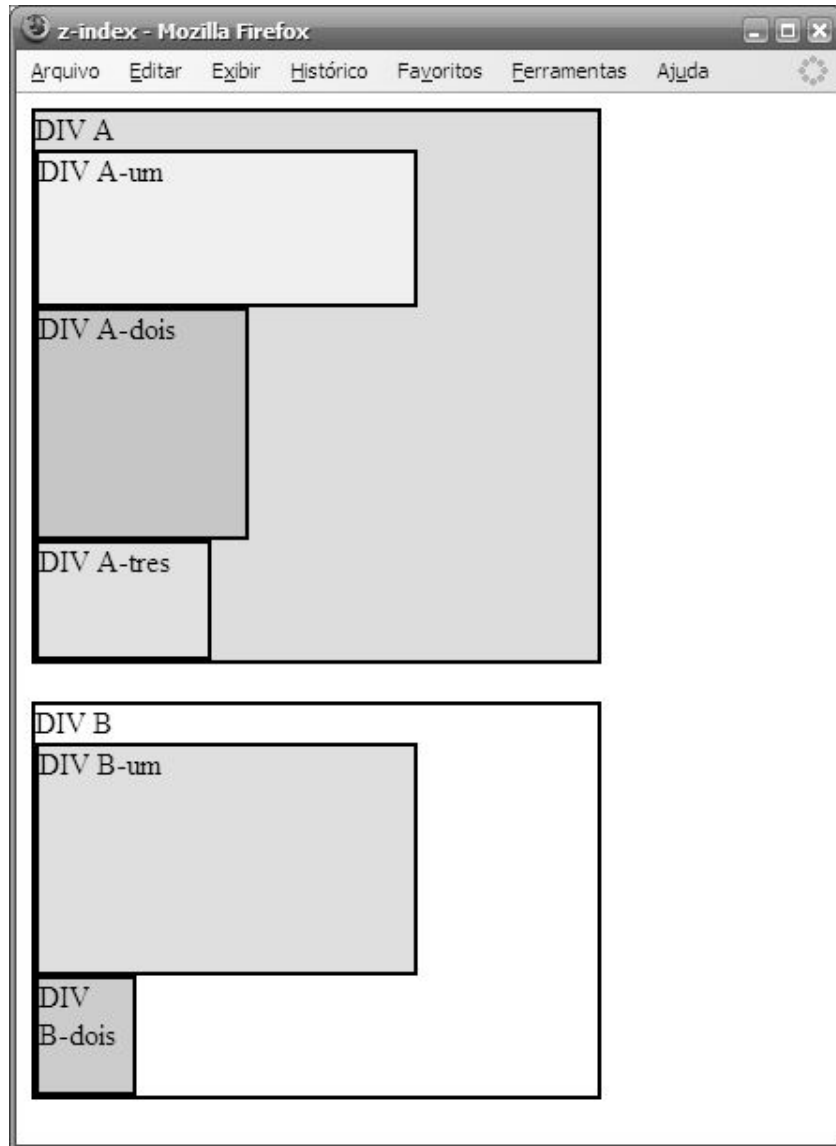
```
div {  
  border: 2px solid #000;  
}
```

```
div#a, div#b {  
    width:300px;  
    margin-bottom:20px;  
}  
div#a {  
    background: #ddd;  
}  
div.a-um {  
    width: 200px;  
    height: 80px;  
    background: #cff;  
}  
div.a-dois {  
    top:70px;  
    width: 110px;  
    height: 120px;  
    background: #6f9;  
}  
div.a-tres {  
    top:90px;  
    width: 90px;  
    height: 60px;  
    background: #fcf;  
}  
div.b-um {  
    width: 200px;  
    height: 120px;  
    background: #ffd1d1;  
}  
div.b-dois {  
    width: 50px;  
    height: 60px;  
    background: #ccc;  
}
```

A renderização da marcação com aplicação da folha de estilo é mostrada na figura 6.17.

Vamos definir posicionamento absoluto para os cinco elementos-filho dos div A e div B, fazendo com que eles se sobreponham. Os div A e div B não serão posicionados de modo a não estabelecer um

novo contexto de empilhamento para seus elementos-filho. Assim, o contexto de empilhamento será o canto superior da área de renderização. Aplicaremos a folha de estilo anterior com as alterações definindo posicionamento, conforme mostram regras CSS a seguir:



*Figura 6.17 – Marcação básica.*

```
div {border: 2px solid #000;}  
div#a, div#b {  
    width:300px;  
    margin-bottom:20px;  
}
```



```
div#a {background: #ffc;}
```

```
div.a-um {  
    position:absolute;  
    left: 65px;  
    top:20px;  
    width: 200px;  
    height: 80px;  
    background: #cff;  
}
```

```
div.a-dois {  
    position:absolute;  
    left: 120px;  
    top:70px;  
    width: 110px;  
    height: 120px;  
    background: #6f9;  
}
```

```
div.a-tres {  
    position:absolute;  
    left: 165px;  
    top:90px;  
    width: 90px;  
    height: 60px;  
    background: #fcf;  
}
```

```
div.b-um {  
    position:absolute;  
    left: 145px;  
    top:130px;  
    width: 200px;  
    height: 120px;  
    background: #ffd1d1;  
}
```

```
div.b-dois {  
    position:absolute;  
    left: 215px;  
    top:170px;  
    width: 50px;  
    height: 60px;  
    background: #ccc;  
}
```

A figura 6.18 apresenta a renderização da marcação com a folha de estilo aplicada.

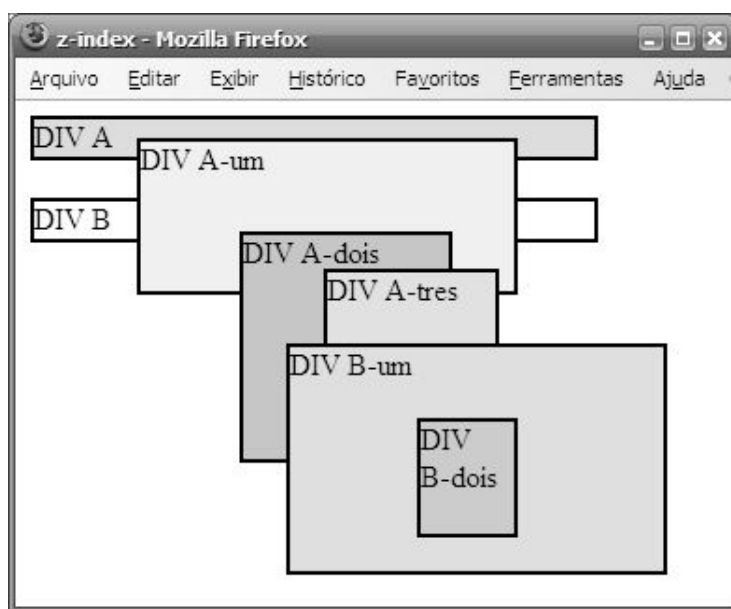
Convém ressaltar os seguintes comportamentos nesse contexto-padrão de empilhamento:

- Os div A e div B encolheram até uma altura suficiente para conter apenas seus nomes.

Isso porque seus elementos-filho foram posicionados de forma absoluta, liberando o espaço que ocupavam no fluxo do documento.

- A ordem de empilhamento obedece à ordem em que os elementos-filho posicionados aparecem na marcação.

Isso porque nenhum dos elementos-filho tem um elemento ancestral posicionado.



*Figura 6.18 – Contexto de empilhamento-padrão.*

Vamos alterar o contexto de empilhamento no nosso exemplo. Observe os acréscimos que faremos na folha de estilo:

```
div {border: 2px solid #000;}  
div#a, div#b {  
    width: 300px;  
    margin-bottom: 20px;
```

```

    }
div#a {
    position: relative;
    left: 10px;
    top: 15px;
    z-index: 25;
    width: 350px;
    height: 200px;
    background: #ddd;
}
div#b {
    position: relative;
    top: -230px;

z-index: 20;
    height: 200px;
}
...igual a anterior...
div.b-dois {
    position: absolute;
    left: 215px;
    top: 170px;
    width: 50px;
    height: 60px;
    background: #ccc;
}

```

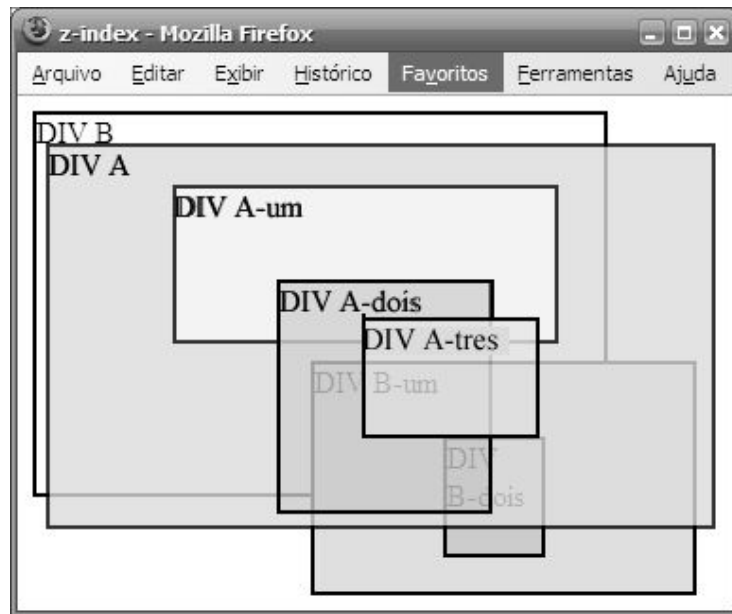
Observe as seguintes alterações e sua finalidade:

- Os div A e div B foram posicionados de forma relativa e, em consequência, cada um deles estabeleceu um contexto de empilhamento. Para os div a-um, div a-dois e div a-tres, a origem do sistema de coordenadas horizontal, vertical e de profundidade passa a ser o div A, e não mais a viewport como no exemplo anterior. O mesmo acontece para as div b-um e div b-tres, que agora têm como base do sistema o div B. Dito de outra maneira: tanto o div A quanto o div B estabeleceram um novo contexto de empilhamento para seus elementos-filho.
- Para os div A e div B definimos largura, altura e coordenadas

horizontal e vertical com o propósito de tornar mais clara a renderização.

- Definimos z-index para os div A e div B. E é aqui que o conceito de contexto de empilhamento fica claro. Ao definirmos um valor z-index para o div A maior do que para o div B, invertemos o comportamento de sobreposição-padrão. Forçamos um elemento anterior no código de marcação a ser colocado à frente de um seu posterior.
- Agora, o div A foi colocado à frente do div B e todos os seus elementos-filho ficarão sempre à frente dos elementos-filho do div B independentemente do valor de z-index. Ou seja, z-index funciona independentemente para os elementos-filho dos div A e div B.

Na figura 6.19 é mostrada a renderização da marcação com a nova folha de estilo a ela aplicada.



*Figura 6.19 – Contexto de empilhamento z-index.*

Vamos examinar a aplicação de z-index nos elementos-filho dos div A e div B. Considere os seguintes acréscimos na folha de estilo anterior:

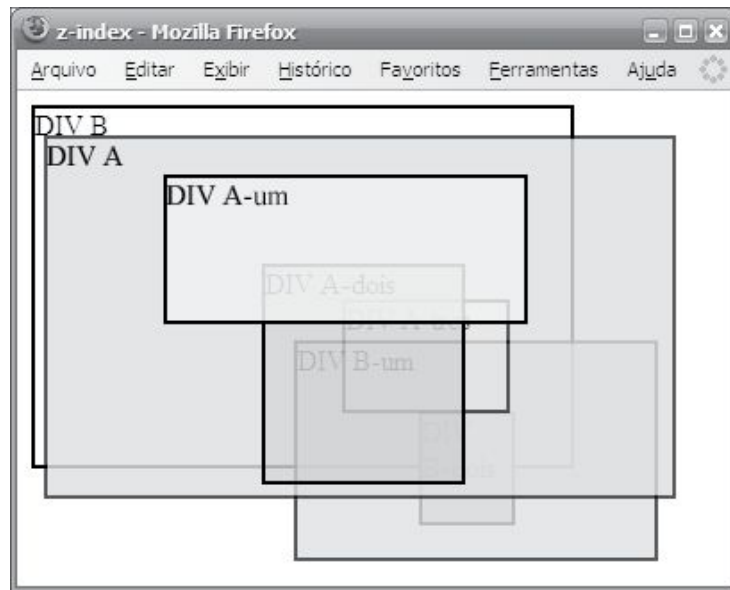
```
div {border: 2px solid #000;}
```

```

div#a {
    ...
    z-index: 25;
    ...
}
div#b {
    ...
    z-index: 20;
    ...
}
div.a-um {
    ...
    z-index: 10;
}
div.a-dois {
    ...
    z-index: 5;
}
div.a-tres {
    ...
    z-index: 2;
}
div.b-um {
    ...
    z-index: 9500;
}
div.b-dois {
    ...
    z-index: 1000;
}

```

Observar, na figura 6.20, que nesse novo cenário a atribuição de z-index para os elementos-filho contidos no div A e div B alterou a ordem de empilhamento. Contudo, o fato de ter atribuído um valor z-index muito alto para aquelas contidas no div B não fez com que elas se posicionassem à frente do div A.



*Figura 6.20 – Empilhamento z-index.*

## **CAPÍTULO 7**

# **Construção de layout**

Para a construção de layout com CSS é fundamental um sólido conhecimento das técnicas de posicionamento estudadas no capítulo 6. O foco deste capítulo, por sua vez, é estudo de layouts de largura fixa, líquidos, elásticos e híbridos. Entendendo como funciona cada um dos tipos de layout possíveis de se construir, você estará capacitado a escolher e desenvolver de forma consistente o que for mais apropriado para o seu site.

### **7.1 Tipos de layout**

Os layouts podem ser classificados segundo dois critérios. Critério do número de colunas e critério da largura da página.

Quanto ao número de colunas, os layouts podem ser de uma, duas, três ou, em casos muito especiais, multicolunares.

Quanto à largura das páginas, os layouts podem ser de largura fixa, líquidos, elásticos, híbridos ou responsivos. Neste livro estudaremos os dois tipos de layout mais usados que são os de largura fixa e os líquidos. Os demais tipos de layout serão apenas definidos conforme a seguir.

#### **7.1.1 Largura fixa**

A página e suas colunas componentes têm uma largura fixa, não escalável, normalmente definida em pixel. Assim, para qualquer que seja o tamanho da janela do navegador ou resolução, as larguras não variam.

Monitores com resolução menor do que a largura total adotada

apresentam parcialmente a página no sentido horizontal, e o navegador coloca uma barra de rolagem também horizontal para acessar a lateral fora de visão.

Monitores com resolução maior do que a largura escolhida apresentam espaços horizontais vazios na tela, em volta da página, que, para resoluções maiores, resulta em mau aproveitamento do espaço disponível. As larguras normalmente empregadas em sites que usam essa técnica são valores em torno de 760px para contemplar resoluções de 800 x 600 ou por volta de 1000px para resoluções de 1024 x 760. Ressalvados alguns cuidados a respeito de legibilidade, que veremos adiante, nada impede que você adote 500 ou 1260px ou outro valor qualquer para a largura das páginas do seu site.

A vantagem deste tipo de layout é o perfeito controle que o desenvolvedor tem sobre as dimensões dos componentes do layout, além da garantia de renderização igual em dimensões, independentemente do tamanho de janela adotado pelo usuário.

As desvantagens são o mau aproveitamento do espaço horizontal em altas resoluções e a necessidade de rolagem horizontal em resoluções abaixo da largura escolhida.

## 7.1.2 Líquido

As larguras são definidas em porcentagem. Neste tipo, o layout acomoda-se em largura a qualquer tamanho de janela ou resolução do monitor, otimizando o aproveitamento do espaço horizontal disponível.



Layout líquido é também conhecido como layout fluido. Adotaremos neste livro a designação layout líquido.

As desvantagens desse tipo de layout estão relacionadas diretamente ao comprimento das linhas de texto. Em tamanhos de janela menores, especialmente nos layouts multicolunares, as larguras das colunas podem ser reduzidas a comprimentos mínimos



dificultando a leitura. A solução para esse caso consiste em definir, para a página, uma largura mínima em pixel, cessando o efeito líquido e forçando o aparecimento de barra de rolagem horizontal, a partir daquela largura.

Por outro lado, em tamanhos de janela e resoluções maiores, os comprimentos das linhas de texto podem se tornar tão extensos que dificultam a leitura. Algumas soluções clássicas são adotadas para tais casos e cabe ao desenvolvedor escolher uma delas ou, mesmo, uma combinação de ambas, baseando-se em particularidades do seu layout.

A primeira solução consiste em definir uma largura total para a página, menor que a largura disponível, por exemplo: ocupar 85% da largura da janela, centrar o layout e obter como resultado margens laterais à esquerda e à direita da página igual a 7,5% do espaço horizontal total. Isso diminui a taxa de crescimento da largura do layout e pode minimizar o efeito de linhas muito extensas.

Outra solução consiste em definir paddings e margens em porcentagens, o que também diminui a taxa de crescimento da largura do layout.

E, finalmente, a opção de definir para a página uma largura máxima em pixel, cessando o efeito líquido a partir daí. Essa é a solução adotada na maioria dos casos.

### **7.1.3 Elástico**

Esse tipo de layout pode ser uma opção para o layout líquido. Como vimos, naquele tipo de layout, o comprimento total das linhas de texto podem se tornar tão reduzido ou tão extenso que afete seriamente a legibilidade.

Em lugar de definir larguras com base na largura da janela, no tipo de layout elástico, definimos com base no tamanho da fonte. A unidade de medida CSS que se baseia em tamanho de fonte é o `em`, e é essa a unidade usada para definir larguras em layout elástico.

Tal tipo de layout, por escalar com o tamanho de fonte, é particularmente útil em questões de acessibilidade para alguns grupos de pessoas com necessidades especiais de visão.

A desvantagem desse tipo de layout é que nem sempre pode ser possível o aproveitamento máximo do espaço horizontal disponível, além do possível aparecimento de barra de rolagem horizontal a partir de determinado ponto de crescimento do tamanho da fonte, mesmo nas resoluções mais altas em uso. Definir um comprimento máximo para a largura total da página pode ser uma solução.

### **7.1.4 Híbrido**

Nessa solução, a largura da página, de colunas e dos demais elementos componentes do layout são definidas em unidades *ems* e adicionalmente definimos para tais componentes uma largura máxima em porcentagem. Há, ainda, a opção de inverter o conceito definido, como larguras do layout em porcentagem e larguras máxima e mínima em unidades *ems*.

### **7.1.5 Responsivo**

Nessa solução, a largura da página, de colunas e dos demais elementos componentes do layout são definidas em unidades *ems* ou porcentagem e adicionalmente definimos para tais componentes uma largura máxima. Em um layout responsivo usamos uma funcionalidade das CSS3 denominada media query, que permite ao desenvolvedor servir o layout de acordo com a largura da viewport, ou seja, para dispositivos móveis ou com pequenas larguras de viewport, o layout servido será de uma coluna e a partir daí, quando a largura da viewport do dispositivo do usuário atingir determinada largura, o layout assume uma configuração de duas colunas, três colunas ou mesmo de várias colunas.

A construção de layout responsivo não é do escopo deste livro. Se você estiver interessado no estudo desse tipo de layout, consulte o

livro Web Design Responsivo. Para mais informações, visite o site do livro em <http://livrosdomaujor.com.br>.

## **7.2 Centralizando o layout**

Com o lançamento de monitores com resoluções cada vez maiores e a imposição de se definir uma largura máxima ou fixa para as páginas, visando resolver o problema de legibilidade em linhas de texto muito extensas, centralizar as páginas do site é a forma de se distribuir pelos dois lados da tela os espaços horizontais não aproveitados, minimizando ou, pelo menos, equalizando o efeito de áreas vazias na tela.

Assim, a prática de centralizar horizontalmente a página tem sido amplamente empregada – e trata-se de um efeito simples de se conseguir com uso de regras CSS. Existem dois métodos para centralizar: declarar margens automáticas para a página, com uso da propriedade `margin`, ou posicionar de forma relativa no centro da tela e usar declaração de margem negativa para obter o efeito final. Adiante, detalharemos o primeiro deles que é o mais usado.

### **7.2.1 Centralizar com uso de margens automáticas**

Esse método baseia-se no cálculo que o agente de usuário (por exemplo: o navegador) faz para encontrar o valor `auto` quando este for declarado para a propriedade `margin`. As Recomendações do W3C para as CSS preveem que, quando o valor `auto` for declarado para as margens esquerda e direita de um elemento com largura explicitamente especificada, o agente de usuário deverá calcular as margens horizontais subtraindo a largura total disponível (largura do elemento-pai) da largura declarada para o elemento e dividir o resultado da subtração por dois, tomando o valor encontrado como margens esquerda e direita. Na prática, o resultado da declaração `auto`, nessas condições, é o de centralizar o elemento dentro do seu elemento-pai. Para ilustrar esse comportamento, considere a

seguinte marcação e folha de estilo:

- **HTML**

```
<body>
  <div id="tudo">
    <h1>Cabeçalho nível 1</h1>
    <p>Lorem ipsum ...</p>
    <p>Ut sollicitudin ...</p>
    <h2>Cabeçalho nível 2</h2>
    <p>Duis posuere ...</p>
    <p>Curabitur hendrerit ...</p>
    <p>Quisque sodales ...</p>
  </div>
</body>
```

A marcação simula os conteúdos de uma página contida dentro de um div, identificada com o par atributo/valor igual a id="tudo". O elemento-pai de uma página é o elemento body, isto é, a área de renderização na tela do usuário. Por sua vez, div#tudo engloba tudo o que está contido em body. Assim, centralizando aquele div, estaremos centralizando a página.

Vamos aplicar a seguinte folha de estilo na nossa página de exemplo:

- **CSS**

```
#tudo {
  width: 60%;
  margin: 0 auto;
  border: 1px solid #000;
}
```

Na figura 7.1 é mostrada a página centralizada com a aplicação da folha de estilo exibida.

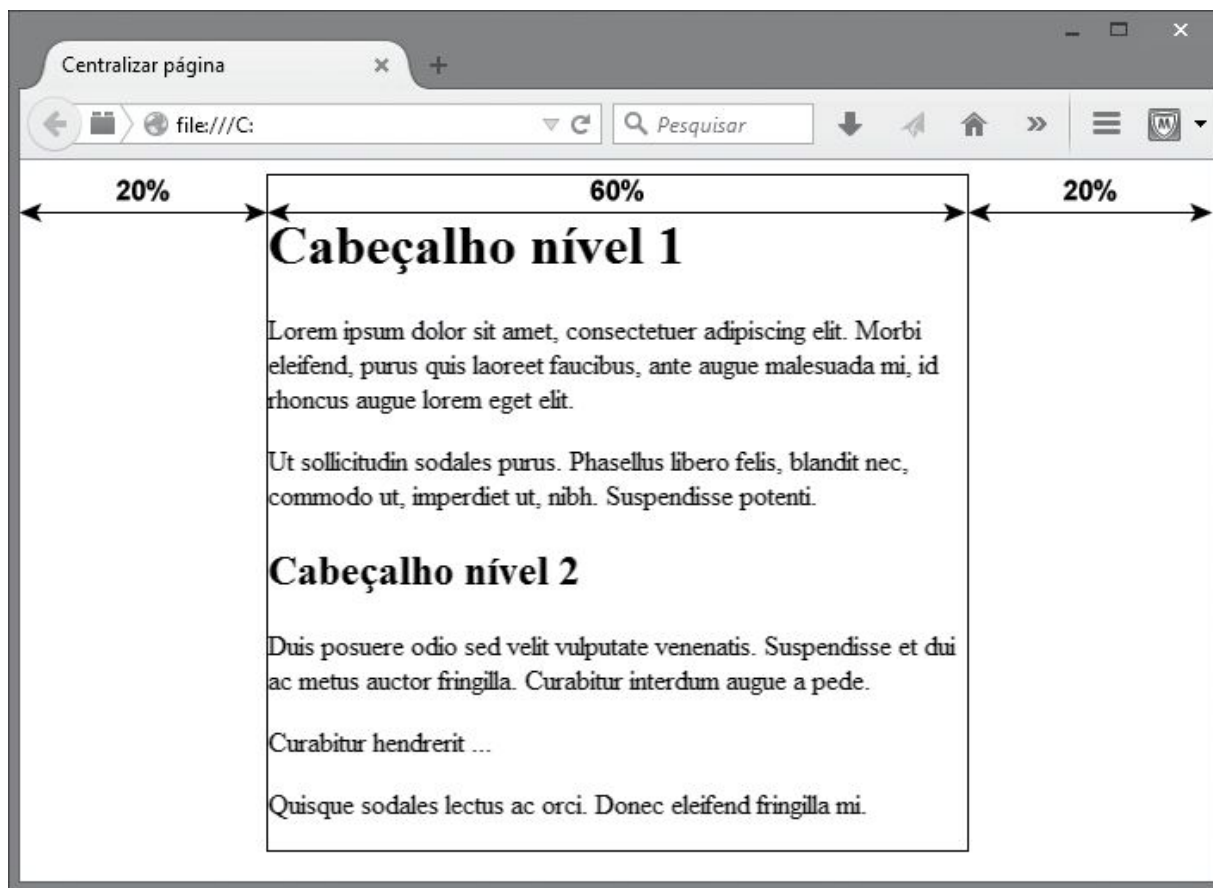


Figura 7.1 – Centralizar com margem.

O arquivo desse exemplo (*centralizar-com-margem-automatica.html*) está disponível para consulta online e download na pasta *capitulo7*.

## 7.3 Layout de largura fixa

Estudaremos as técnicas de construção de layouts baseados em posicionamento flutuado. A escolha desse tipo de posicionamento para construir layouts está consagrada, principalmente pela flexibilidade e versatilidade que oferece.

A estrutura típica de um layout de duas colunas com largura fixa, e que adotaremos como exemplo neste livro, consiste de uma área para o topo do site contendo logotipo e identificação do site, uma coluna de navegação e outra coluna de conteúdo principal, as quais chamaremos de topo, auxiliar e principal, respectivamente, e um fechamento que denominaremos rodapé.

Toda a estrutura descrita estará contida em um elemento-pai, container geral do layout, que, em nossos exemplos, será um div com nome identificador tudo. A marcação HTML é conforme a mostrada a seguir.

- **HTML**

```
...
<body class="home">
  <div id="tudo">
    <div class="topo">
      <header class="topo_header">
        <hgroup>
          <h1><a href="/">Site do Paulo</a></h1>
          <h2>Site destinado a divulgar o trabalho de Paulo Silva</h2>
        </hgroup>
      </header>
    </div> <!-- /.topo -->
    <div class="principal">
      <main>
        <h1>Título UM</h1>
        <section class="principal_secaoum">
          <h2>Subtítulo um</h2>
          <p>Lorem ipsum dolor...</p>
          <h2>Subtítulo dois</h2>
          <p>Curabitur hendrerit...</p>
        </section>
      </main>
    </div> <!-- /.principal -->
    <div class="auxiliar">
      <nav class="auxiliar_navegacao">
        <h2>Menu</h2>
        <ul>
          <li><a class="corrente" href="../capitulo7/home.html">home
            </a></li>
          <li><a href="../capitulo7/portfolio.html">portfólio</a></li>
          <li><a href="../capitulo7/artigos-html.html">artigos HTML
            </a></li>
          <li><a href="../capitulo7/artigos-css.html">artigos CSS
            </a></li>
          <li><a href="../capitulo7/contato.html">contato</a></li>
        </ul>
      </nav>
    </div>
  </div>
</body>
```

```

    </ul>
</nav>
<aside class="auxiliar_parceiros">
  <ul class="auxiliar_parceiros_textos">
    <li><a href="#">Parceiro Um</a></li>
    <!-- mais três itens -->
  </ul>
  <div class="auxiliar_parceiros_imagens">
    <figure><a href="#"></a></figure>
    <!-- mais quatro itens -->
  </div>
</aside>
</div> <!-- /.auxiliar -->
<div class="rodape">
  <footer>
    <small>Site do Paulo - 2015 &copy; Todos os direitos reservados
    </small>
  </footer>
</div> <!-- /.rodape -->
</div> <!-- /.tudo -->
</body>
</html>

```

A marcação sem estilização é renderizada no navegador conforme mostra a figura 7.2.

O arquivo desse exemplo (*layout-2colunas-fixo-1.html*) está disponível para consulta online e download na pasta *capitulo7*.

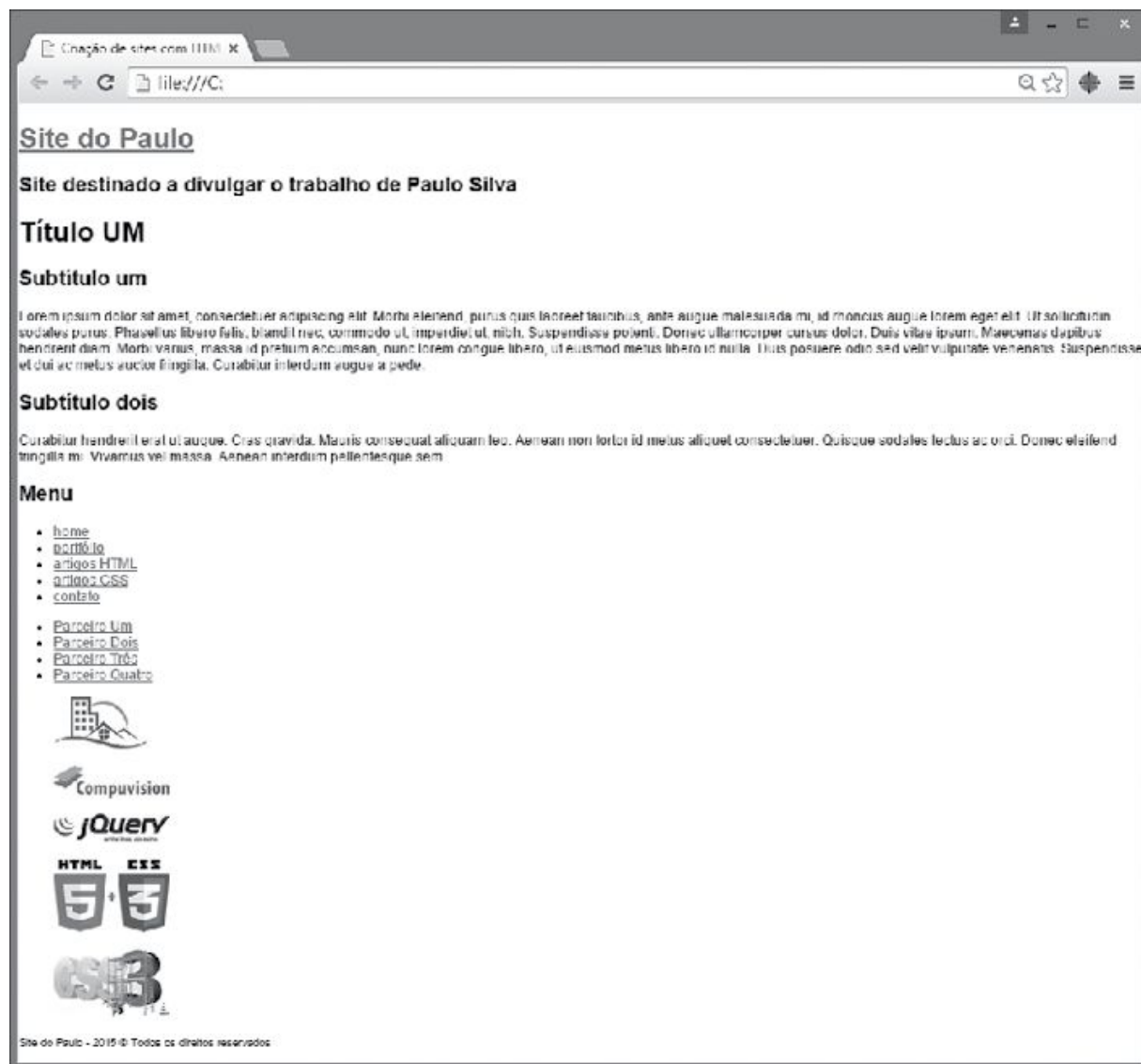


Figura 7.2 – Layout sem estilização.

Analisando a marcação proposta para construção do nosso layout podemos notar que, por não haver compromisso da marcação com a apresentação, colocamos o conteúdo principal do site no início da marcação e a navegação no fim. Isso incrementa a acessibilidade ao site, expondo não só aos usuários com necessidades especiais, mas também aos mecanismos de busca e outros agentes de usuário, logo no início, o assunto principal tratado na página, facilitando o acesso e a indexação do documento.

Vamos construir nosso layout seguindo as seguintes premissas:

- Página centralizada na janela do navegador.



- Largura total da página igual a 960px.
- Coluna principal à direita com 720px e navegação à esquerda com 220px, resultando em um espaço de 20px entre elas.
- Altura do rodapé igual a 25px.

Existem dois métodos para posicionar as colunas com float. Flutuar ambas as colunas à direita e definir uma margem entre elas para separá-las em 20px ou flutuar uma para cada lado, deixando que a diferença da soma de suas larguras para a largura total resulte nos 20px de separação. Notar o seguinte cálculo:  $960\text{px} - (720\text{px} + 220\text{px}) = 20\text{px}$ . Adotaremos a segunda solução. A folha de estilo para cumprir as premissas do layout é a mostrada a seguir.

## • CSS

```
...
<link rel="stylesheet" href='../normalize.css'></link>
<style rel="stylesheet">
*, *:before, *:after { /* regras para alterar o box Model padrão */
    -webkit-box-sizing: border-box;
    -moz-box-sizing: border-box;
    box-sizing: border-box;
}
.clear { clear: both; } /* classe para "limpar" elementos flutuados */
body { font: 18px/1.4 arial, sans-serif; color: #333;}
#tudo { width: 960px; margin: 0 auto; } /* centraliza a página de 960px
de largura */
.topo_header h1 { font-size: 36px; text-align: right; margin: 4px 0 2px 0;}
.topo_header a { color: #900; text-decoration: none; }
.topo_header h2 { font-size: 20px; text-align: right; margin: 2px 0 4px 0;}
.principal { width: 720px; float: right; } /* flutua à direita */
.auxiliar { width: 220px; float: left; } /* flutua à esquerda */
.rodape { clear: both; height: 25px; } /* "limpa" as colunas flutuadas e
define altura do rodapé */
```

Observe na folha de estilo proposta:

- O link para a folha de estilo *normalize.css* conforme [4.6]
- A página foi centralizada adotando-se o método das margens automáticas, conforme mostrado em [7.2.1].

- As regras de estilos para o topo do site são conforme as mostradas em [5.2.3 – 5ª etapa].

A renderização do layout com a aplicação da folha de estilo é conforme mostra a figura 7.3.



Figura 7.3 – Estilização inicial do layout de duas colunas.

O arquivo desse exemplo (*layout-2colunas-fixo-2.html*) está disponível para consulta online e download na pasta *capitulo7*.

A técnica de construção e a estrutura geral do layout estão prontas. Vamos a seguir criar regras de estilos para ajustar e alterar a apresentação do topo e do rodapé, bem como definir uma cor de fundo cinza para o elemento `body` e branca para a área das colunas do site. Considere as seguintes alterações na folha de estilo do

exemplo anterior, conforme mostradas em destaque a seguir.

- **CSS**

```
body {  
    font: 18px/1.4 arial, sans-serif;  
    color: #333;  
    background: #ccc; /* cor de fundo cinza */  
}  
#tudo {  
    width: 960px;  
    margin: 0 auto;  
    background: #fff; /* cor de fundo branca */  
}  
h1, h2, h3, h4, h5, h6 { color: #036; }  
.topo_header {  
    background: #036; /* cor de fundo azul para o topo */  
    padding: 5px 20px;  
}  
.topo_header a {  
    color: #900;  
    color: #fff;  
    text-decoration: none;  
}  
.topo_header h2 {  
    font-size: 20px;  
    text-align: right;  
    color: #fff;  
}  
.rodape {  
    clear: both;  
    height: 25px;  
    background: #036; /* cor de fundo azul para o rodapé */  
    color: #fff;  
    padding: 5px 0;  
    text-align: center;  
}
```

A figura 7.4 mostra a estilização da cor de fundo de body e da área de conteúdos, e a estilização do topo e do rodapé depois de alterada a folha de estilo.



Figura 7.4 – Estilização do topo e rodapé.

O arquivo desse exemplo (*layout-2colunas-fixo-3.html*) está disponível para consulta online e download na pasta *capitulo7*.

O próximo passo é a estilização do menu do site. Vamos criar um menu do tipo botões com efeito rollover. Esse efeito consiste em fazer com que a estilização do botão mude quando o usuário colocar o ponteiro do mouse sobre ele e volte ao estado inicial quando o ponteiro do mouse for retirado de sobre ele. No nosso exemplo vamos alterar a cor de fundo do botão para obter o efeito rollover.

Observe as regras CSS para estilizar o menu.

## • CSS

```
.auxiliar { ...; padding-left: 15px; }
.principal { ...; padding-right: 15px; }
/* as declarações anteriores estabelecem margens para os conteúdos das
colunas */
/* Novas regras CSS */
a { text-decoration: none; } /* retira o sublinhado do link */
.auxiliar_navegacao h2 { text-align: center; } /* centra o título do menu */
ul { list-style: none; padding:0; margin:0; } /* ver comentários a seguir */
.auxiliar_navegacao a {
    background: #ff6e19;
    color: #fff;
    display: block;
    margin-bottom: 1px;
    line-height: 2;
    padding-left: 10px;
    text-transform: capitalize;
}
.auxiliar_navegacao a:hover {
    background: #036; /* efeito rollover */
}
```

Comentários sobre a folha de estilo proposta:

- As declarações de estilo para o elemento `ul` visam retirar o marcador (bolinhas à esquerda) dos itens da lista e zerar padding e margin padrão das listas. No arquivo do exemplo existente no site do livro retire uma a uma as declarações dessa regra CSS (para o elemento `ul`) e você constatará o efeito delas.



Para todos os exemplos mostrados neste livro, quando você ficar em dúvida sobre o efeito de uma declaração CSS, abra o arquivo do exemplo, retire a declaração, salve o arquivo e observe a alteração na renderização, ou, se você sabe usar a ferramenta de desenvolvimento do navegador, edite as CSS e faça as alterações nela. Zeno Rocha publicou uma série de vídeos explicando como usar essa ferramenta em <http://kwz.me/lm>.

A figura 7.5 mostra como ficou a estilização do menu do site.

O arquivo desse exemplo (*layout-2colunas-fixo-4.html*) está disponível para consulta online e download na pasta *capitulo7*.

Vamos a seguir estilizar a área dos parceiros. Notar que essa área está dividida em duas outras áreas, uma contendo links de textos e outra contendo links de imagens dos parceiros.



Figura 7.5 – Estilização do menu.

Observe as regras CSS para estilizar a área de links de textos.

## • CSS

```
.auxiliar_parceiros_textos {  
    margin: 30px 0; /* Cria espaçamento acima e abaixo */  
    text-align: center; /* Centra os textos dos links */  
}  
.auxiliar_parceiros_textos li {  
    margin-bottom: 3px; /* Cria espaçamento entre links */  
    background: #eee; /* Cor de fundo dos links */  
}  
.auxiliar_parceiros_textos a {  
    padding: 8px 0;  
    color: #333; /* cor dos textos dos links */  
    display: block; /* toda a área do link será clicável */  
}
```

Comentários sobre a folha de estilo proposta:

- Declarar `display: block` para links a dentro de item de lista li faz com que não somente o texto, mas toda a área em volta dele seja clicável.

A figura 7.6 mostra como ficou a estilização da área parceiros.



*Figura 7.6 – Estilização da área de parceiros.*

O arquivo desse exemplo (*layout-2colunas-fixo-5.html*) está disponível para consulta online e download na pasta *capitulo7*.

## 7.4 Layout elástico

Nesse tipo de layout as larguras são definidas em porcentagem. O layout acomoda-se em largura a qualquer tamanho de janela do navegador ou resolução do monitor, otimizando o aproveitamento do espaço horizontal disponível.

Para larguras de janela maiores ou menores que um determinado tamanho, definimos uma largura máxima e uma largura mínima de modo que o layout passe a ser de largura fixa quando a largura da janela estiver fora da faixa entre mínima e máxima, estabelecida

pelo desenvolvedor.

Para o nosso exemplo, vamos estabelecer como ponto de partida os valores de 1200px para largura mínima e 760px para largura máxima, contudo convém notar que a determinação desses valores depende de vários fatores inerentes a cada layout. O desenvolvedor deverá fazer experiências em diferentes tamanhos de janela e considerar diferentes resoluções de monitor para estabelecer essas larguras.

Você pode adotar os valores que bem entender nos seus projetos. Para exemplificar esse tipo de layout, usaremos a marcação HTML e estilização do exercício anterior, considerando uma largura máxima de 1200px, sendo 960px para a coluna principal, 220px para a coluna auxiliar e 20px de distância entre elas.

Observe a tabela 7.1 de correspondência das dimensões fixas com porcentagens:

*Tabela 7.1 – Larguras fixas e porcentagens*

Referência	Pixel	% calculada
Largura total	1200px	100%
Coluna principal	960px	80%
Coluna auxiliar	220px	18.3333%
Espaçamento colunas	20px	1.6666%

As alterações a serem marcadas na folha de estilo anterior (largura fixa) são mostradas em destaque a seguir.

## • CSS

```
#tudo {  
    width: 100%;  
    max-width: 1200px;  
    min-width: 760px;  
    margin: 0 auto;  
    background: #fff;  
}  
.principal {  
    width: 80%;
```



```
float: right;
padding-right: 15px;
}
.auxiliar {
width: 18.3333%;
float: left;
padding-left: 15px;
}
```

O arquivo *layout-2colunas-liquido1.html*, desse exemplo está disponível para consulta online e download na pasta *capitulo7*.

Abra o arquivo no navegador e movimente a janela para diferentes resoluções observando o comportamento do layout. Retire as declarações de `max-width` e `min-width` do `div#tudo` e observe o comportamento do layout. Faça experiências com outros valores de porcentagem.

# CAPÍTULO 8

## Estilização

### 8.1 Introdução

Neste capítulo, estudaremos algumas técnicas de estilização com uso das CSS3. Veremos a aplicação de bordas arredondadas, sombras, opacidade, gradientes, cores e imagens de fundo.

### 8.2 Bordas arredondadas

As propriedades CSS para estabelecer bordas arredondadas são listadas e definidas a seguir.

- `border-top-left-radius` – Raio da borda superior esquerda do box.
- `border-top-right-radius` – Raio da borda superior direita do box.
- `border-bottom-right-radius` – Raio da borda inferior direita do box.
- `border-bottom-left-radius` – Raio da borda inferior esquerda do box.
- `border-radius` – Sintaxe abreviada para definir o raio da borda nos quatro cantos do box.

Os valores CSS para essas propriedades são as medidas CSS de comprimento, como px, em, pt etc. e porcentagem.

Para demonstrar a aplicação de diferentes tipos de borda arredondada nos exemplos constantes do site do livro, usaremos boxes retangulares simples. O arquivo *bordas-arredondadas.html*, que demonstra os exemplos de bordas arredondadas, está disponível para consulta online e download na pasta *capitulo8*.

Podemos declarar um ou dois valores para a curvatura de cada um dos cantos de um box individualmente, obtendo efeitos

diferentes, como mostra o exemplo a seguir.

- **CSS**

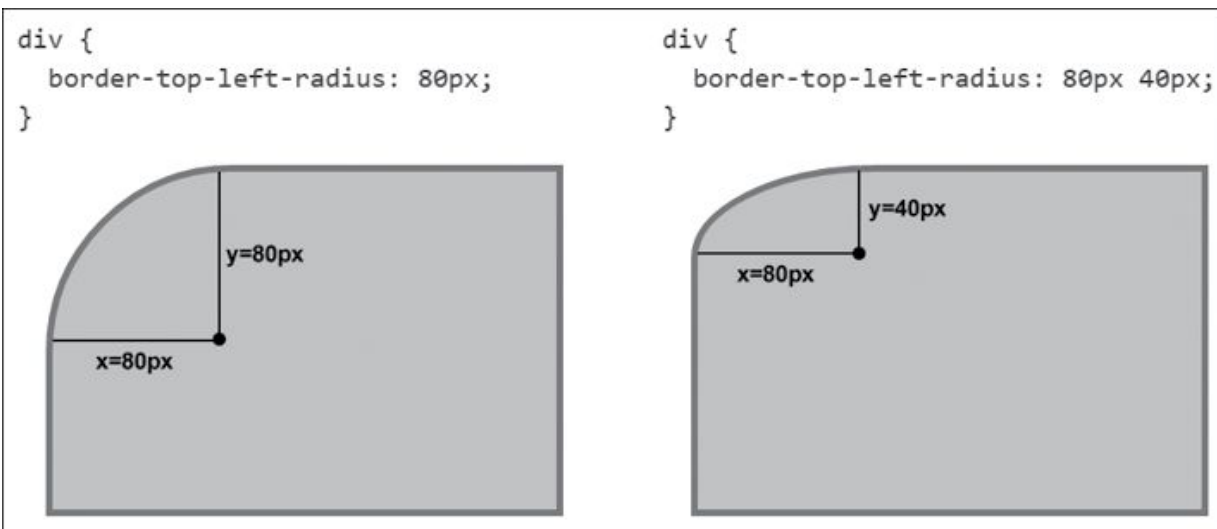
```
border-top-left-radius: 80px;
```

ou

```
border-top-left-radius: 80px 40px;
```

Declarar um só valor produz curvaturas iguais nos sentidos x e y, e declarar dois valores produz curvatura igual ao primeiro valor declarado no sentido x e ao segundo valor declarado no sentido y.

A figura 8.1 mostra a aplicação das bordas arredondadas com as regras CSS anteriores.



*Figura 8.1 – Bordas arredondadas.*

O arquivo desse exemplo (*bordas-arredondadas.html*) está disponível para consulta online e download na pasta *capitulo8*.

Para aplicar bordas arredondadas aos quatro cantos do box, podemos ainda usar a declaração abreviada `border-radius` que admite um, dois, três ou quatro valores, conforme descrito a seguir.

- `border-radius: raio;` – Os quatro cantos serão arredondados igualmente com raio de curvatura com valor igual a raio.
- `border-radius: raio1 raio2;` – Os cantos top-left e bottom-right serão arredondados igualmente com raio de curvatura com valor igual a

raio1; e os cantos top-right e bottom-left serão arredondados igualmente com raio de curvatura com valor igual a raio2.

- border-radius: raio1 raio2 raio3; – O canto top-left será arredondado com raio de curvatura com valor igual a raio1, os cantos top-right e bottom-left serão arredondados igualmente com raio de curvatura com valor igual a raio2 e o canto bottom-right será arredondado com raio de curvatura com valor igual a raio4.
- border-radius: raio1 raio2 raio3 raio4; – Os raios de curvatura declarados serão aplicados na seguinte ordem: top-left, top-right, bottom-right e bottom-left.

O arquivo *bordas-arredondadas.html*, que demonstra a aplicação de bordas arredondadas, conforme mostrado, está disponível para consulta online e download na pasta *capitulo8*.

A declaração simplificada border-radius também admite a sintaxe para declarar valores para curvatura da borda segundo os eixos x e y. Nesse caso devemos separar as declarações para os eixos x e y com uma barra (/). Observe o exemplo a seguir.

```
border-radius: 80px 20px 40px 10px / 40px 60px 10px 5px;
```

Essa sintaxe admite que se declare de um a quatro valores para qualquer uma das duas curvaturas x e y. Por exemplo:

#### • CSS

```
border-radius: 80px / 40px 60px 10px 5px;  
border-radius: 80px 20px 40px / 40px 60px;  
border-radius: 80px 20px / 10px;
```

## 8.3 Sombras

Com declarações CSS é possível aplicar sombras em textos e em boxes. Vamos estudar esses dois casos separadamente.

### 8.3.1 Sombra em texto

A propriedade CSS para declarar sombra em texto é text-shadow. A

sintaxe para definir sombra em texto consiste em um valor CSS que é opcional para *cor*, seguido de *duas unidades de medida CSS*, obrigatórias, que definem os deslocamentos da sombra na horizontal e vertical respectivamente, e opcionalmente mais uma *unidade de medida CSS* que define o efeito blur (borrado) da cor. Todos os valores devem ser separados por espaço em branco.

A ordem de declaração dos valores é a seguinte: cor, deslocamento horizontal, deslocamento vertical e blur. Nessa ordem o valor CSS da cor pode estar no início ou no fim dos valores declarados.

Os valores para deslocamento da sombra podem ser positivos ou negativos. Valores positivos indicam deslocamento horizontal para a direita e vertical para baixo, e negativos indicam deslocamento nos sentidos inversos.

Não sendo declarado valor para a cor, a sombra será na mesma cor do texto.

É possível declarar simultaneamente quatro posicionamentos de sombra para um mesmo texto, usando as quatro possíveis combinações de sinais (positivo e negativo) para as duas unidades de medida CSS obrigatórias para definir deslocamentos. Nesse caso as declarações dos valores da sombra devem ser separadas por vírgula.

Seguem exemplos de sintaxe da regra CSS para aplicar sombra em texto.

- **CSS**

```
h1 { text-shadow: red 5px 5px; }
h1 { text-shadow: red 5px 5px 5px; }
h1 { text-shadow: red -5px 5px; }
h1 { text-shadow: red 5px -5px; }
h1 { text-shadow: red -5px -5px; }
h1 {
  text-shadow: red 3px 3px 5px,
               blue -6px -6px,
               green -25px 15px 5px;
```

}

O arquivo *sombra-em-texto.html*, que demonstra a aplicação de sombras em textos, está disponível para consulta online e download na pasta *capitulo8*.

### 8.3.2 Sombra em box

A propriedade CSS para declarar sombra em box é `box-shadow`. A sintaxe para definir sombra em box consiste da palavra-chave `inset`, que é opcional, seguida de duas unidades de medida CSS, obrigatórias, que definem os deslocamentos da sombra na horizontal e vertical respectivamente; e opcionalmente mais uma unidade de medida CSS que define o efeito blur (borrado) da cor; e ainda opcionalmente mais uma unidade de medida CSS que define uma expansão (valor positivo) ou uma contração (valor negativo) da sombra. Todos os valores devem ser separados por espaço em branco.

A palavra-chave opcional `inset`, quando declarada, faz com que a sombra seja interna ao box.

Os valores para deslocamento da sombra podem ser positivos ou negativos. Valores positivos indicam deslocamento horizontal para a direita e vertical para baixo, e negativos indicam deslocamento nos sentidos inversos.

Não sendo declarado valor para a cor, a sombra será na mesma cor do texto do box.

É possível declarar simultaneamente quatro posicionamentos de sombra para um mesmo box, usando as quatro possíveis combinações de sinais (positivo e negativo) para as duas unidades de medida CSS obrigatórias para definir deslocamentos. Nesse caso as declarações dos valores da sombra devem ser separadas por vírgula.

Seguem exemplos de sintaxe da regra CSS para aplicar sombra em texto.

## • CSS

```
div { text-shadow: red 5px 5px; }  
div { box-shadow: #f90 5px 5px 5px; }  
div { box-shadow: #f90 5px 5px 5px 15px; }  
div { box-shadow: inset #f90 -10px 10px 15px; }  
div {  
    box-shadow: #f90 10px 0 8px -4px,  
                #f90 -10px 0 8px -4px;  
}  
div {  
    box-shadow: #f90 0 5px 4px -2px,  
                #f90 0 -5px 4px -2px;  
}  
div { box-shadow: #f90 0 0 10px 5px; }
```

O arquivo *sombra-em-box.html*, que demonstra a aplicação de sombras em textos, está disponível para consulta online e download na pasta *capitulo8*.

## 8.4 Opacidade

Essa é uma propriedade CSS que se destina a definir a opacidade (ou transparência) de um elemento e seus elementos descendentes e bordas. O valor possível para essa propriedade é um número compreendido entre 0 e 1. O valor 0 representa transparência total, e o valor 1, opacidade total. Assim, um valor igual a 0.9 significa 90% opaco ou 10% transparente.

Observe a seguir declarações CSS típicas para a propriedade *opacity*:

```
opacity: 0.7;  
opacity: .4;  
opacity: 0;
```

Aplicar opacidade em um elemento faz com que todos os elementos nele contidos herdem o valor da opacidade a ele aplicada. Considere o texto de um parágrafo inserido em um *div*, como mostrado a seguir:

```
<div>
  <p class="um">Texto do parágrafo um</p>
  <p class="dois">Texto do parágrafo dois</p>
</div>
```

e as seguintes regras CSS:

```
div {
  opacity: 0.5;
  color: black;
}
p.um { opacity: 0.5 }
p.dois { opacity: 1 }
```

O texto dos dois parágrafos herdará a opacidade 0.5, definida para o elemento div. A regra de estilo que define a opacidade para o p.um faz com que o texto tenha opacidade  $0.5 * 0.5 = 0.25$ . A regra de estilo que define a opacidade para o p.dois não produz nenhum efeito, pois não é possível aumentar a opacidade herdada, somente diminuí-la, como foi feito com p.um.

O uso de cores RGBA ou HSLA para aplicar opacidade, ao contrário do uso de opacity, não faz com que os elementos descendentes herdem a opacidade. Observe o exemplo anterior, mas agora usando RGBA:

```
<div>
  <p class="um">Texto do parágrafo um</p>
  <p class="dois">Texto do parágrafo dois</p>
</div>
```

e as regras CSS:

```
div { color: rgba(0, 0, 0, 0.5); }
p.um { color: rgba(0, 0, 0, 0.5); }
p.dois { color: rgba(0, 0, 0, 1); }
```

Nesse exemplo, diferentemente do anterior, p.um terá opacidade 0.5 e p.dois será totalmente opaco. Isso porque aplicar opacidade com uso RGBA (ou HSLA) não implica herança CSS.

Observe a seguir um exemplo prático de aplicação de opacidade.

- **CSS**



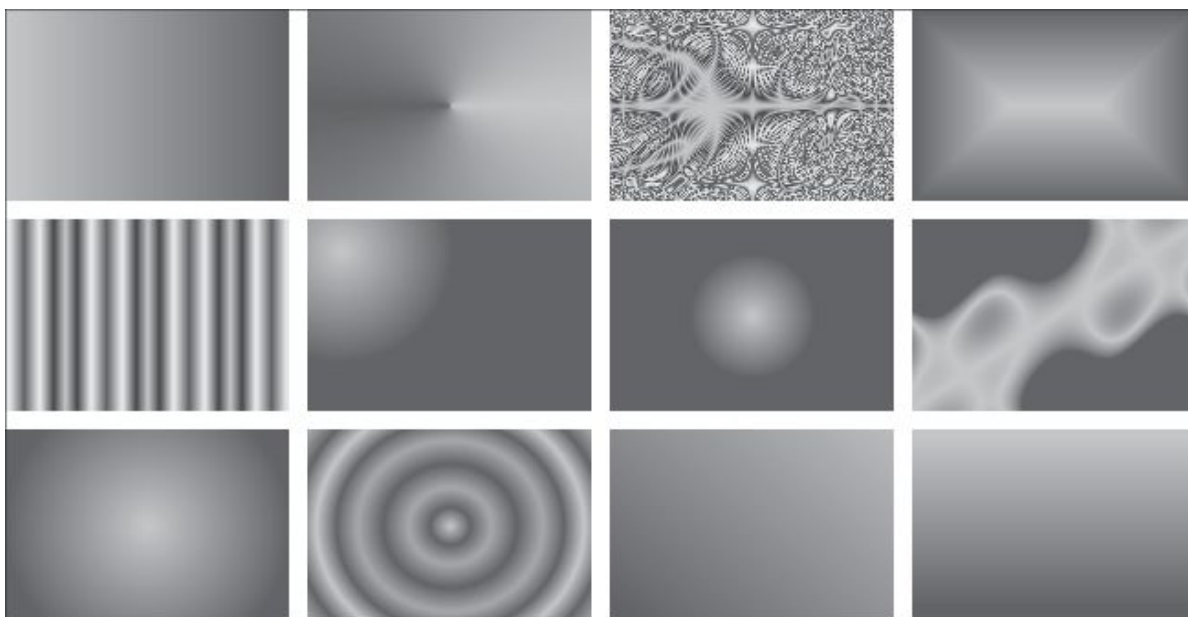
```
div {  
    opacity: /* ver observação a seguir */  
    background-image: url(logo.png);  
    background-size: contain;  
    background-position: center bottom;  
    background-repeat: no-repeat;  
}
```

Observação: nesse exemplo criamos três containeres e neles inserimos um conteúdo textual e uma imagem do tipo PNG transparente. Para o primeiro container, não definimos opacidade (é o mesmo que definir opacidade igual a 1), no segundo definimos opacidade igual a 0.6 e no terceiro 0.4.

O arquivo *opacidade.html*, que demonstra a aplicação de transparência em containeres e seus conteúdos, conforme o exemplo mostrado, está disponível para consulta online e download na pasta *capitulo8*.

## 8.5 Gradientes

Com declarações CSS é possível criar gradientes de cores para serem usados como imagens de fundo. Define-se gradiente CSS como sendo uma imagem que mostra uma transição suave entre duas ou mais cores. A transição das cores pode ser segundo um eixo (uma linha reta) dando origem a um gradiente linear, ou segundo círculos concêntricos criando um gradiente radial. Vamos estudar esses dois tipos de gradiente. Observe na figura 8.2 alguns tipos de gradiente.



*Figura 8.2 – Exemplos de gradientes.*



Convém ressaltar que as funcionalidades CSS para definição de gradientes possibilitam que se crie transição abrupta de cor em lugar da transição suave padrão.

## 8.5.1 Gradiente linear

Gradiente linear é aquele em que a transição de uma cor para outra é feita ao longo de um eixo que determina a direção da transição.

A sintaxe para criar um gradiente linear a ser aplicado como valor para uma propriedade CSS que admita imagem como valor, por exemplo: `background-image` ou `list-style-image`, usa uma função CSS denominada `linear-gradient()` cujos parâmetros são conforme os descritos a seguir.

`linear-gradient(direção, cor1 stop, cor2 stop, cor3 stop,..., corn stop)`

Notar que os parâmetros são separados por vírgula.

- O parâmetro *direção* define o eixo do gradiente linear que pode ser declarado com uso de uma medida de ângulo CSS ou da palavra inglesa *to* (significa em direção a) seguida de um lado ou de um canto.

- O parâmetro *cor* define as cores do gradiente e devem ser declaradas duas ou mais cores.
- O parâmetro *stop* define uma distância medida sobre o eixo do gradiente em que será feita a transição de cores. Esse parâmetro será estudado com detalhes adiante.

A sintaxe e renderização de gradientes lineares são conforme o exemplo mostrado nas figuras a seguir.

No exemplo mostrado na figura 8.3, criou-se um gradiente linear com transição da cor black para white segundo um eixo na direção 45 graus. A origem da contagem do ângulo CSS (direção 0 grau) é a vertical em cima, valores positivos do ângulo são no sentido horário e negativos anti-horário.

No exemplo mostrado na figura 8.4, criou-se gradientes lineares com transição da cor black para white segundo um eixo definido pela palavra *to* (em direção a). Notar que, depois da palavra *to*, define-se um dos quatro lados do box (top, right, bottom, left).

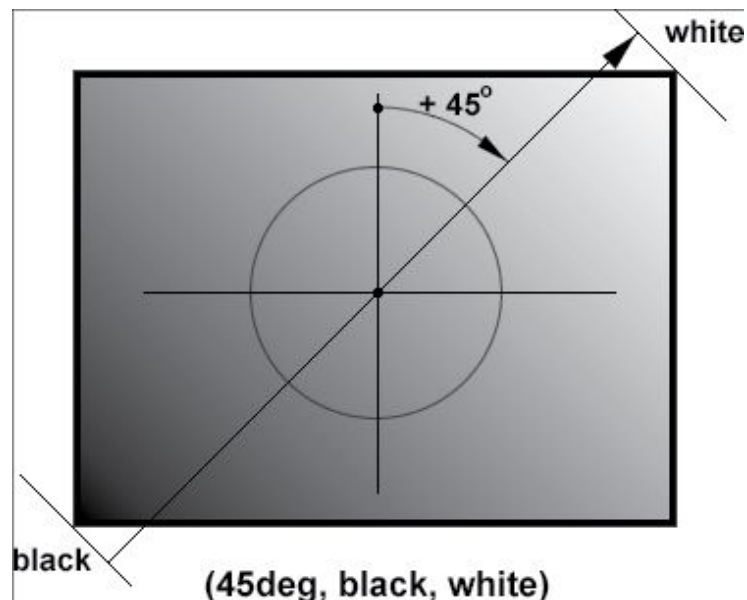
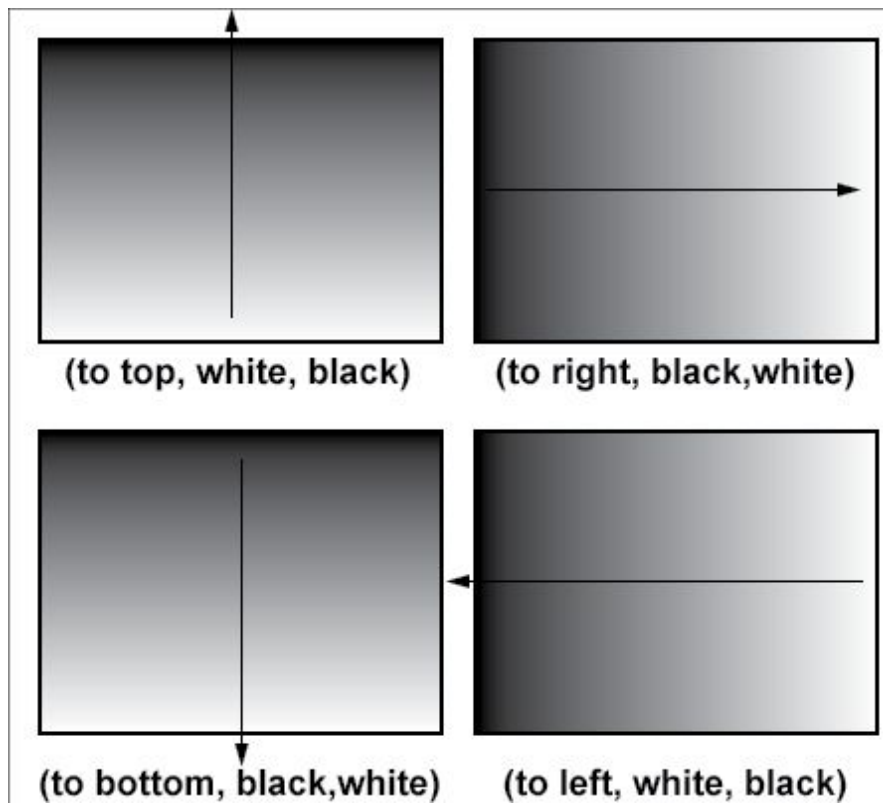


Figura 8.3 – Definir eixo com uso de ângulo.



*Figura 8.4 – Definir eixo com uso de direção a um lado.*

No exemplo mostrado na figura 8.5, criou-se gradientes lineares com transição da cor black para white segundo um eixo definido pela palavra *to* (em direção a). Notar que, depois da palavra *to*, define-se um dos quatro cantos do box (top left, top right, bottom right, bottom left).

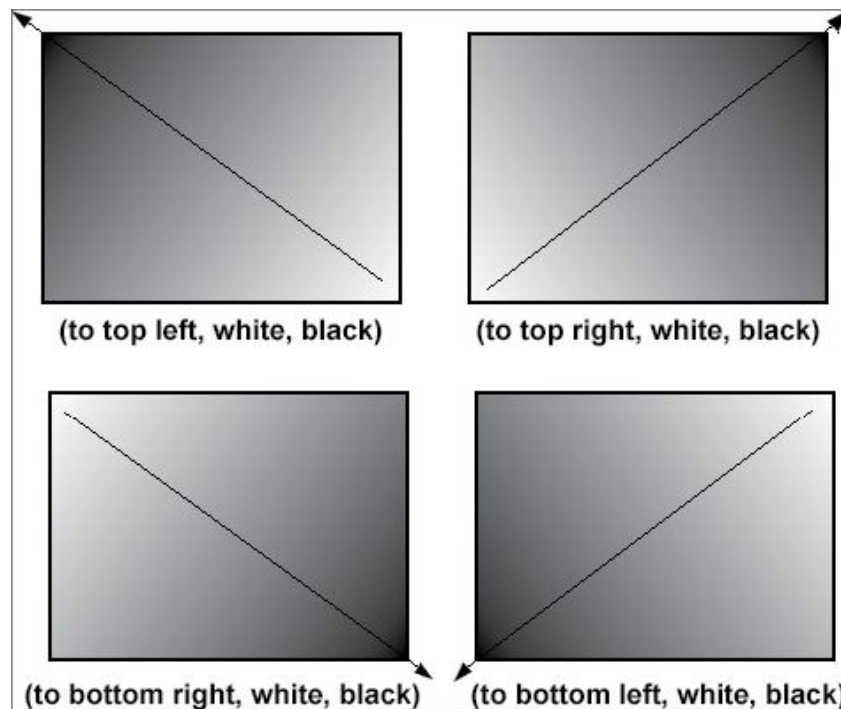


Figura 8.5 – Definir eixo com uso de direção a um canto.



Se não for declarado o parâmetro *direção*, o valor padrão adotado será to bottom que é o mesmo que 180deg.

E, para finalizar, vamos examinar o parâmetro *stop* da função `linear-gradient()`.

Como você deve ter notado nas imagens anteriores, a transição entre duas cores se faz de modo que cada uma delas e sua respectiva área de transição se desenvolva ocupando a metade do container. Se definirmos um gradient linear de três cores, a área ocupada por cada uma delas será de 1/3 do container; para gradientes de quatro cores será 1/4 do container, e assim por diante.

O parâmetro *stop* permite que se interfira na distribuição das cores. O valor desse parâmetro é uma unidade de comprimento CSS ou porcentagem definindo a distância da origem do gradiente em que a cor deve sofrer a transição.

Observe os exemplos mostrados a seguir e note que, para a primeira e quarta regras CSS, o gradiente foi declarado sem uso do parâmetro *stop*, e para as demais, com uso do parâmetro.

- **CSS**

```
background-image: linear-gradient(to left, black, white) /* sem stop */  
background-image: linear-gradient(to right, white, black 20%) /* com stop */  
background-image: linear-gradient(to right, white, black 180px) /* com stop */  
background-image: linear-gradient (to left, black, white, black, white) /* sem  
stop */  
background-image: linear-gradient(to right, black 33%, pink 33%, pink 67%,  
gray 67%) /* com stop */
```

A figura 8.6 mostra a renderização do gradiente para as cinco regras CSS do exemplo. Notar o efeito resultante para cada regra.

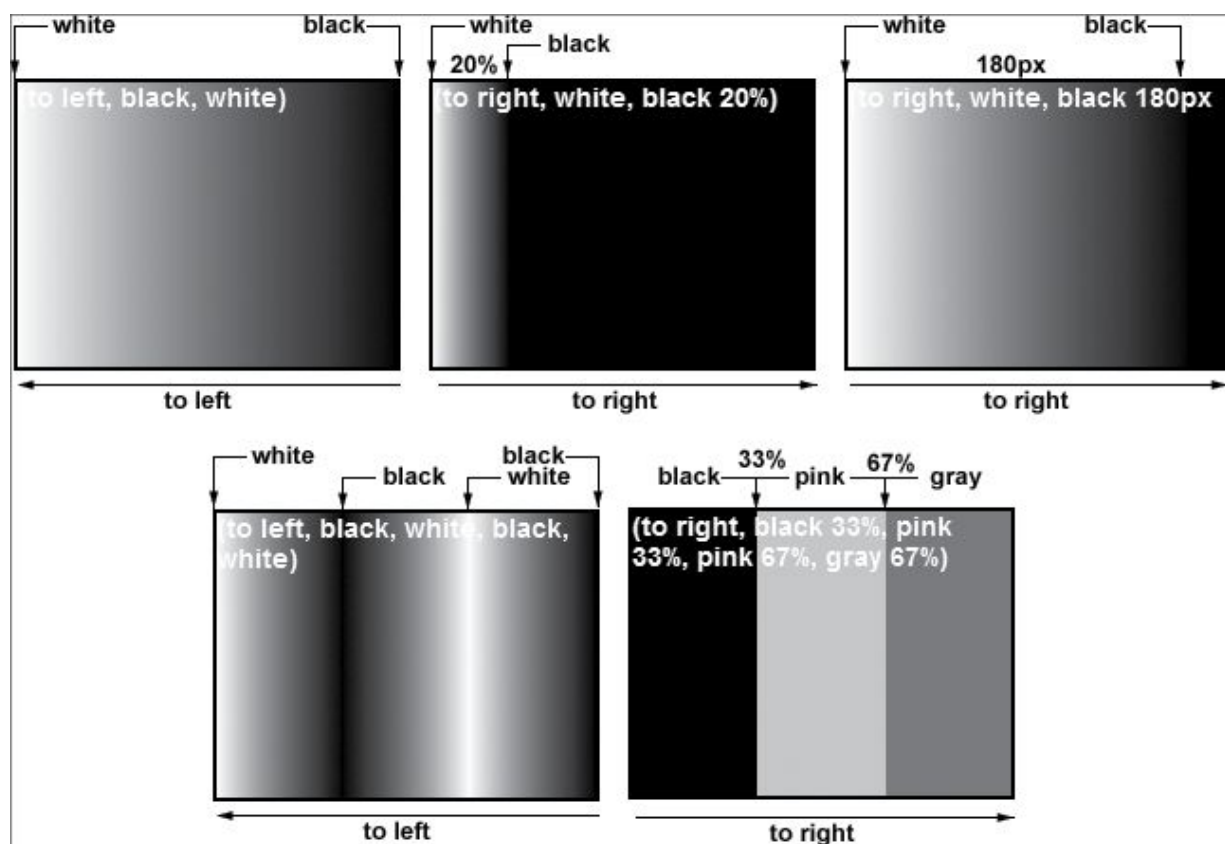


Figura 8.6 – Parâmetro stop para gradientes.

## 8.5.2 Gradiente radial

Gradiente radial é aquele em que a transição de uma cor para outra se faz segundo linhas radiais, a partir de um ponto chamado centro do gradiente. A forma das linhas radiais pode ser circular ou elíptica, criando-se dois tipos de gradiente radial.

Observe na figura 8.7 algumas formas de gradientes radiais.

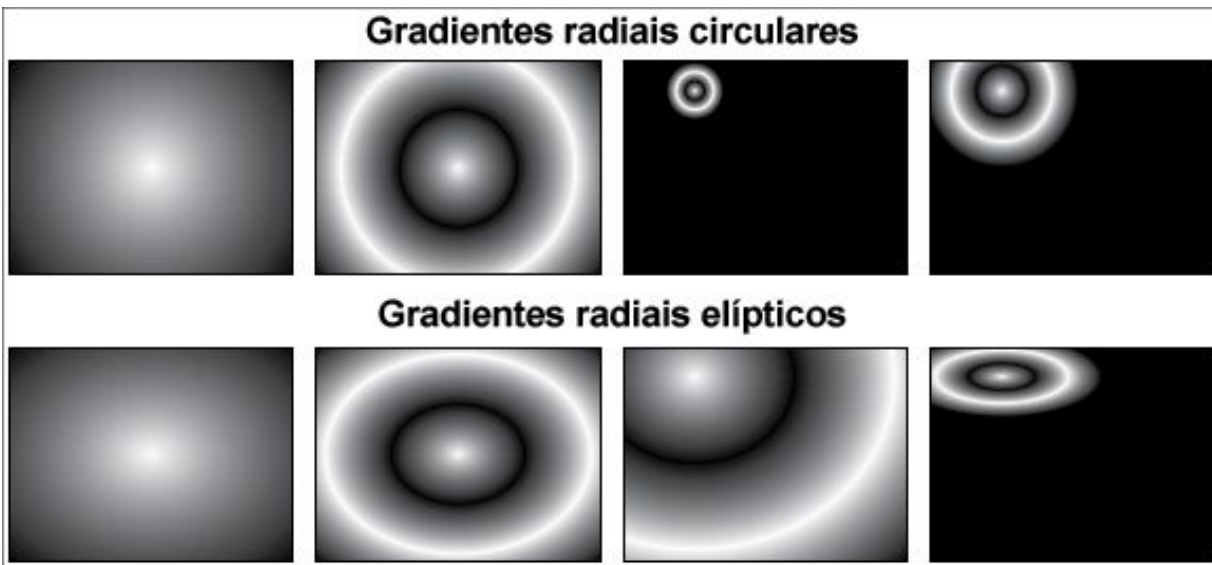


Figura 8.7 – Exemplos de gradientes radiais.

A sintaxe para criar um gradiente radial a ser aplicado como valor para uma propriedade CSS que admita imagem como valor, por exemplo: `background-image` ou `list-style-image`, usa uma função CSS denominada `radial-gradient()` cujos parâmetros são conforme os descritos a seguir.

`radial-gradient (coordenadas do centro, forma tamanho, cor1 stop, cor2 stop, cor3 stop,..., corn stop)`

Notar que os parâmetros são separados por vírgula.

O parâmetro *coordenadas do centro* define as coordenadas do centro do gradiente e admite os mesmos valores usados para definir a propriedade `background-position` (ver apêndice B). Se esse parâmetro for omitido, o valor padrão será `center`, definindo o centro do gradiente no centro do box no qual ele será aplicado.

O parâmetro *forma* admite os valores `circle` e `ellipse` definindo o gradiente circular ou elíptico respectivamente, e o parâmetro *tamanho* define como o gradiente se desenvolve em relação aos limites do box em que é aplicado.

Os valores possíveis são: `closest-side`, `farthest-side`, `closest-corner`, `farthest-corner`, `contain` e `cover`. Alternativamente podemos definir um

valor explícito com fornecimento de coordenadas. O valor padrão é `cover`.

Veamos a seguir o significado de cada um desses valores.

- `closest-side` – Se a forma do gradiente for `circle`, ele deverá estender-se até o lado do container mais próximo do centro. Se a forma do gradiente for `ellipse`, ele deverá estender-se até os lados horizontal e vertical do container, mais próximos do centro.
- `farthest-side` – Se a forma do gradiente for `circle`, ele deverá estender-se até o lado do container mais afastado do centro. Se a forma do gradiente for `ellipse`, ele deverá estender-se até os lados horizontal e vertical do container, mais afastados do centro.
- `closest-corner` – Se a forma do gradiente for `circle`, ele deverá estender-se até o canto do container mais próximo do centro. Se a forma do gradiente for `ellipse`, ele deverá estender-se até o canto do container mais próximo do centro, mas mantendo o mesmo `aspect-ratio`.
- `farthest-corner` – Se a forma do gradiente for `circle`, ele deverá estender-se até o canto do container mais afastado do centro. Se a forma do gradiente for `ellipse`, ela deverá estender-se até o canto do container mais afastado do centro, mas mantendo o mesmo `aspect-ratio`.
- `contain` – Esse valor produz o mesmo efeito de `closest-side`.
- `cover` – Esse valor produz o mesmo efeito de `farthest-corner`.

Os exemplos a seguir mostram a aplicação de gradientes radiais como imagem de fundo de um elemento `div` com uso dos parâmetros.

```
.um {  
  background-image: -moz-radial-gradient(40px 60px, circle closest-side,  
    white, black, white);  
  background-image: -webkit-radial-gradient(40px 60px, circle closest-side,  
    white, black, white);  
  background-image: radial-gradient(40px 60px, circle closest-side, white,  
    black, white);  
}
```



```

    }
.dois {
    background-image: -moz-radial-gradient(40px 60px, circle farthest-side,
        white, black, white);
    background-image: -webkit-radial-gradient(40px 60px, circle farthest-side,
        white, black, white);
    background-image: radial-gradient(40px 60px, circle farthest-side, white,
        black, white);
    }
.tres {
    background-image: -moz-radial-gradient(40px 60px, circle closest-corner,
        white, black, white);
    background-image: -webkit-radial-gradient(40px 60px, circle closest-corner,
        white, black, white);
    background-image: radial-gradient(40px 60px, circle closest-corner, white,
        black, white);
    }
.quatro {
    background-image: -moz-radial-gradient(40px 60px, circle farthest-corner,
        white, black, white);
    background-image: -webkit-radial-gradient(40px 60px, circle farthest-corner,
        white, black, white);
    background-image: radial-gradient(40px 60px, circle farthest-corner,
        white, black, white);
    }
.cinco {
    background-image: -moz-radial-gradient(40px 60px, circle contain, white,
        black, white);
    background-image: -webkit-radial-gradient(40px 60px, circle contain,
        white, black, white);
    background-image: radial-gradient(40px 60px, circle contain, white,
        black, white);
    }
.seis {
    background-image: -moz-radial-gradient(40px 60px, circle cover, white,
        black, white);
    background-image: -webkit-radial-gradient(40px 60px, circle cover, white,
        black, white);
    background-image: radial-gradient(40px 60px, circle cover, white, black,
        white);
    }

```

```
.sete {
  background-image: -moz-radial-gradient(40px 60px, 100px 60px, white,
    black, white);
  background-image: -webkit-radial-gradient(40px 60px, 100px 60px, white,
    black, white);
  background-image: radial-gradient(40px 60px, circle 100px 60px, white,
    black, white);
}
```

Observe na figura 8.8 o efeito de aplicação das regras CSS mostradas.

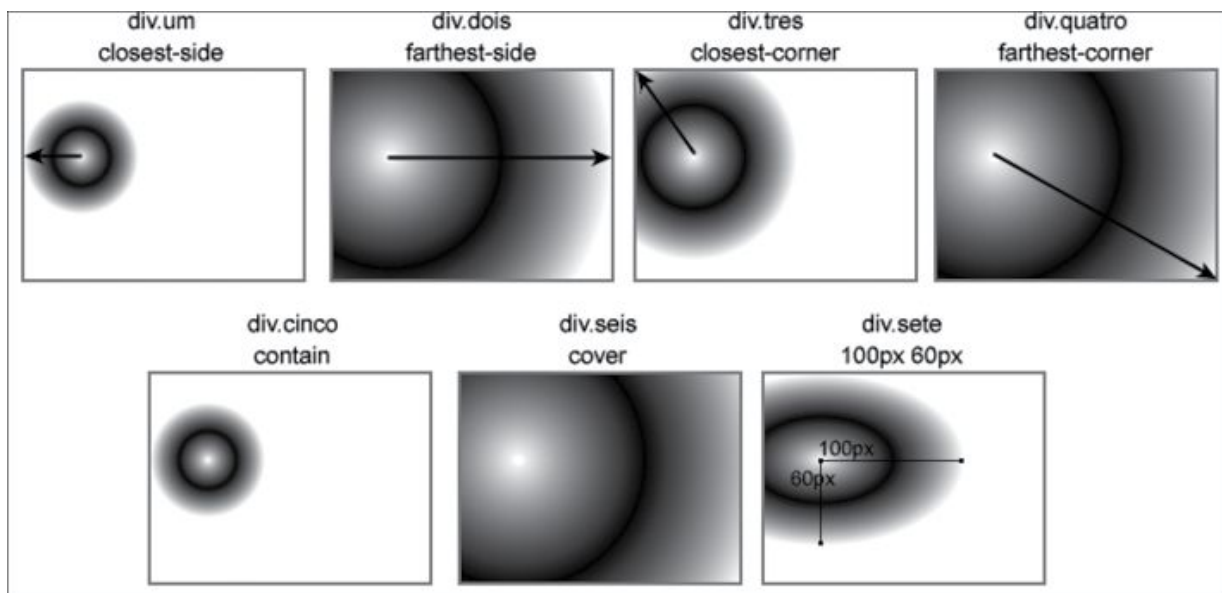


Figura 8.8 – Efeito do parâmetro tamanho.

O arquivo *gradiente-radial-parametro-tamanho.html*, que demonstra a aplicação de gradientes radiais, conforme o exemplo mostrado, está disponível para consulta online e download na pasta *capitulo8*.

### 8.5.3 Gradiente repetido

Existe uma funcionalidade para criar gradientes que permite definir um gradiente e fazer com que ele se repita indefinidamente. A repetição de gradientes só é possível naqueles para os quais se definiu o parâmetro *stop*, que pode ser aplicado tanto a gradientes lineares como a gradientes radiais.

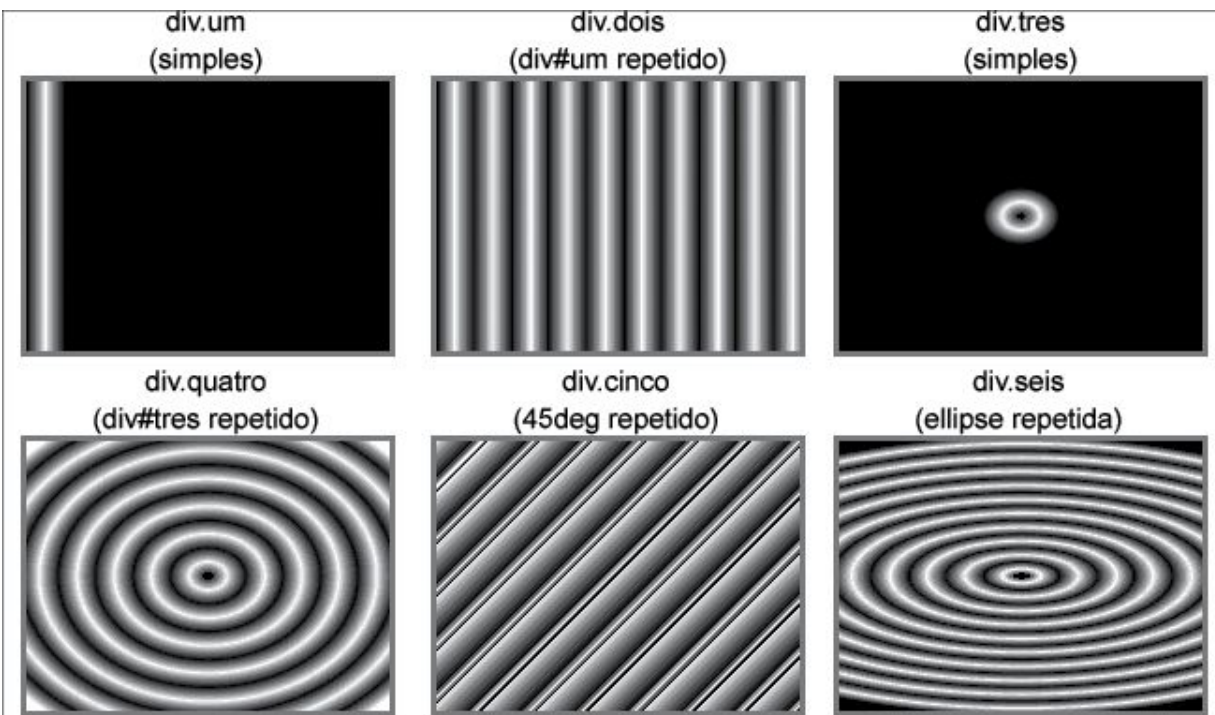
O exemplo a seguir mostra a aplicação de gradiente repetido em

vários boxes. Nos boxes `div#um` e `div#tres`, aplicou-se um gradiente simples, e nos `div#dois` e `div#quatro`, o mesmo gradiente com repetição. Nos `div#cinco` e `div#seis`, aplicaram-se gradientes com repetição linear e radial, respectivamente.

- **CSS**

```
.um {background: linear-gradient(to right, black, white 10px, black 20px);}
.dois {background: repeating-linear-gradient(to right, black, white 10px, black 20px);}
.tres {background: radial-gradient(50% 50% ellipse, black, white 10px, black 20px);}
.quatro {background: repeating-radial-gradient(50% 50% ellipse, black, white 10px, black 20px);}
.cinco {background: repeating-linear-gradient(-45deg, black, white 2px, black 5px, white 20px);}
.seis {background: repeating-radial-gradient(100% 50% ellipse, black, white 10px, black 20px);}
```

Observe, na figura 8.9, o efeito da aplicação dos gradientes com repetição, conforme mostrado no código anterior.



*Figura 8.9 – Gradientes repetidos.*

O arquivo *gradiente-repetido.html*, que demonstra a aplicação de

gradientes repetidos, conforme o exemplo mostrado, está disponível para consulta online e download na pasta *capítulo8*.

## 8.6 Propriedade background

A propriedade CSS background é a forma abreviada de declarar várias propriedades destinadas a estilizar o fundo de um box. Tais propriedades estão listadas e descritas a seguir.

- background-color – Define uma cor de fundo para o box.
- background-image – Define uma imagem de fundo para o box.
- background-repeat – Define como se dá a repetição da imagem de fundo.
- background-attachment – Define se a imagem permanece fixa ou rola com o conteúdo do box.
- background-position – Define a posição da imagem no box.
- background-clip – Define a área do box na qual será aplicada a imagem.
- background-origin – Define a posição de origem da imagem no box.
- background-size – Define as dimensões da imagem no box.
- background – Forma abreviada de declarar uma ou mais propriedades anteriores.

Estudaremos a seguir cada uma dessas propriedades.

A sintaxe para declarar valor para cada uma das propriedades listadas anteriormente é conforme mostrada a seguir.

### ***background-color***

Essa propriedade define uma cor de fundo para o box e é descrita a seguir. Os valores possíveis para essa propriedade são: valor de cor CSS, como #f00, red e rgb(255, 0, 0) – ver [4.5]. O valor padrão é a cor transparent.

```
background-color: #ffc9d4;
```

```
background-color: blue;
```

```
background-color: rgba(255, 120, 150, 0.7);  
background-color: hsla(0, 100%, 50%, 0.6);  
background-color: rgb(20%, 40%, 38%);
```

## ***background-image***

Essa propriedade define uma imagem de fundo para o box e é descrita a seguir.

```
background-image: url("/imagens/imagem.jpg");
```



Observe que usamos aspas duplas no endereço da imagem. Alternativamente podemos usar aspas simples ou simplesmente não usar aspas.

```
background-image: linear-gradient(...);  
background-image: radial-gradient(...);
```



Por padrão, a área de um box, na qual é aplicada a cor e a imagem de fundo, estende-se até a área das bordas inclusive. Aplicando- uma cor de fundo em um box com uma borda larga e tracejada, visualiza-se facilmente a cor aplicada preenchendo os espaços entre o tracejado da borda.

## ***background-repeat***

Essa propriedade define como a imagem de fundo se repete no box, e é descrita a seguir.

- `background-repeat: repeat;` – A imagem repete-se na horizontal e na vertical – esse é o valor padrão.
- `background-repeat: no-repeat;` – A imagem não se repete.
- `background-repeat: repeat-x;` – A imagem repete-se na horizontal.
- `background-repeat: repeat-y;` – A imagem repete-se na vertical.
- `background-repeat: space;` – A imagem toca as quatro bordas internas do box, e são espaçadas de modo a se distribuírem igualmente.
- `background-repeat: round;` – A imagem toca as quatro bordas internas do box, e são redimensionadas de modo a preencherem o fundo tocando umas nas outras.

## ***background-attachment***

Essa propriedade define o comportamento, se fixo ou não, da imagem de fundo no box, e é descrita a seguir.

- background-attachment: scroll; – A imagem não permanece fixa em relação à viewport (janela do navegador) e rola com o conteúdo – esse é o valor padrão.
- background-attachment: fixed; – A imagem permanece fixa em relação à viewport (janela do navegador) e não rola com o conteúdo.
- background-attachment: local; – A imagem permanece fixa em relação à viewport (janela do navegador), mas, quando aplicada ao box de um elemento, rola com o conteúdo se o elemento tiver um mecanismo de rolagem definido por overflow: scroll.

## ***background-position***

Essa propriedade define as coordenadas x e y da imagem de fundo – valores de medida CSS de comprimento, porcentagem ou as palavras-chave top, right, bottom e left. O valor padrão é 0 0 ou left top. Pode-se declarar apenas um valor para a coordenada, que será considerada a coordenada horizontal, e, nesse caso, a coordenada vertical será considerada center.

O ponto de referência, tanto para o início da contagem das coordenadas no box como na imagem a ser inserida como fundo, é o canto superior esquerdo do box e da imagem respectivamente; contudo, se as coordenadas forem definidas com uso de porcentagem, o ponto de referência na imagem é aquele cujas coordenadas na imagem são iguais às porcentagens declaradas, por exemplo: para coordenadas 50% 50%, a referência na imagem para posicioná-la é o centro dela mesma. Observe a seguir alguns exemplos.

```
background-position: 40px 50px;  
background-position: 100px;  
background-position: 40% 20%;  
background-position: right top;
```

background-position: center;

### ***background-clip***

Essa propriedade define a área de preenchimento da imagem de fundo no box, e é descrita a seguir.

- background-clip: border-box; – A imagem ocupa até a área das bordas do box, inclusive, se houver uma – esse é o comportamento padrão.
- background-clip: padding-box; – A imagem ocupa até a área de padding do box, inclusive, se houver uma e não ocupa a área de bordas.
- background-clip: content-box; – A imagem ocupa até a área de conteúdo do boxe não ocupa as áreas de padding e bordas se houver.

### ***background-origin***

Essa propriedade define a origem de preenchimento da imagem de fundo no box e é descrita a seguir.

- background-origin: border-box; – A imagem tem por origem o canto superior esquerdo do box, considerando as bordas do box, se houver – esse é o comportamento padrão.
- background-origin: padding-box; – A imagem tem por origem o canto superior esquerdo do box sem considerar as bordas do box, se houver.
- background-origin: content-box; – A imagem tem por origem o canto superior esquerdo do box, sem considerar o padding e as bordas do box, se houver.

### ***background-size***

Essa propriedade define as dimensões da imagem de fundo do box, e é descrita a seguir. Os valores possíveis são uma medida de comprimento CSS, a porcentagem e as palavras-chave: auto, contain

e cover, sendo auto o valor padrão, que significa as dimensões originais da imagem.

- background-size: 150px 75px; – A imagem terá as dimensões de 150px x 75px.
- background-size: 250px; – A imagem terá largura igual a 200px e uma altura tal que o aspect ratio (relação entre largura e altura) seja preservado.
- background-size: 80% 10%; – A imagem terá as dimensões de 80% da largura do box e 10% da altura do box.
- background-size: 60%; – A imagem terá largura igual a 60% da largura do box e uma altura tal que o aspect ratio (relação entre largura e altura) seja preservado.
- background-size: auto; – A imagem terá suas dimensões originais – esse é o valor padrão.
- background-size: contain; – A imagem terá as dimensões de modo que sua maior dimensão ocupe toda a extensão do box e a menor dimensão seja tal que o aspect ratio (relação entre largura e altura) seja preservado.
- background-size: cover; – A imagem terá as dimensões de modo que sua menor dimensão ocupe toda a extensão do box e a maior dimensão seja tal que o aspect ratio (relação entre largura e altura) seja preservado.

## ***background***

Essa propriedade é a maneira abreviada de declarar todas as oito propriedades para definição de fundo. A sintaxe para essa propriedade é mostrada a seguir.

```
seletor {  
    background: background-image background-position / background-size  
                background-repeat  
                background-attachment background-origin background-clip background-color;  
}
```



Há um arquivo interativo, *propriedade-background.html*, que demonstra a aplicação e os efeitos causados pela propriedade `background`, conforme as explicações anteriores. O arquivo está disponível para consulta online e download na pasta *capitulo8*.

## 8.6.1 Múltiplas imagens de fundo

Podemos definir quantas imagens de fundo quisermos em um mesmo box, posicionando e dimensionando cada uma delas individualmente.

Cada uma das imagens de fundo definidas para um mesmo box cria um layer com uma coordenada *z* (como se fosse um *z-index*), definindo sua visibilidade. Os layers são posicionados em uma pilha de layers colocados uns sobre outros fazendo com que os layers com maior coordenada *z* se sobreponham àqueles com menor coordenada *z*.

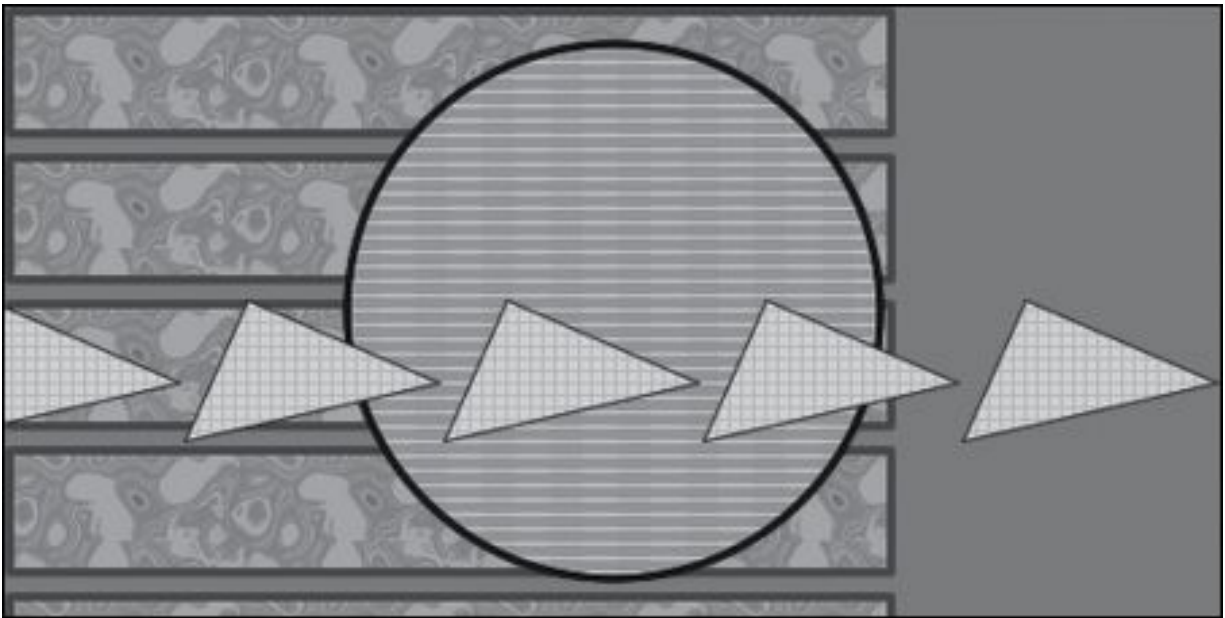
A ordem de empilhamento, ou seja, a coordenada *z* é definida pela ordem em que a imagem é declarada dentro da regra CSS. A primeira imagem declarada recebe a maior coordenada *z*, a segunda recebe a segunda maior coordenada, e assim por diante, até a última imagem declarada que recebe a menor coordenada *z*. A sintaxe para declarar múltiplas imagens de fundo consiste em separar os valores das propriedades com uma vírgula.

Considere a seguir a regra CSS destinada a definir três imagens como fundo de um elemento `div`.

### • CSS

```
div {  
    background-image: url(triangulo.png), url(circulo.png), url(retangulo.png);  
    background-repeat: repeat-x, no-repeat, repeat-y;  
    background-position: right 120px, center, 0 0;  
    background-color: #f0c;  
}
```

Na figura 8.10 mostramos a renderização das três imagens (triângulo, círculo e retângulo).



*Figura 8.10 – Múltiplas imagens de fundo.*

# CAPÍTULO 9

## Formulários

### 9.1 Introdução

Neste capítulo estudaremos algumas técnicas de criação e estilização de formulários. Veremos os elementos HTML5 para criar containeres, rótulos e campos de formulários. Mostraremos os atributos especiais (aqueles que merecem destaque) para os elementos de formulários e suas finalidades. Para a lista completa dos atributos para os elementos de formulários, consulte o apêndice A deste livro.

### 9.2 Elementos de formulário e seus atributos

Apresentamos a seguir uma lista dos elementos da HTML5 que se destinam a marcar formulários, uma breve descrição de cada um deles, seus respectivos atributos especiais e a sintaxe de marcação.

#### **form**

Esse elemento destina-se a ser o container geral de um formulário. Seus principais atributos são:

- **action** – Aponta para o URL em que se encontra uma página contendo o script encarregado de processar os dados do formulário. A HTML não processa dados de formulário. Para processar os dados, é necessário criar um script com uso de uma linguagem de programação como PHP ou ASP, por exemplo.
- **method** – Define o método de envio dos dados do formulário.

Sintaxe:

```
<form action="http://site.com/scripts/formulario.php" method="post">  
  <!-- marcação do formulário -->  
</form>
```

## **fieldset e legend**

Em formulários complexos e/ou extensos pode ser interessante agrupar campos do formulário por assuntos, como campos para coletar dados pessoais, campos para coletar preferências, campos para coletar habilidades etc. O elemento `fieldset` destina-se a servir de container para agrupar campos do formulário, e o elemento `legend` destina-se a criar uma descrição ao grupamento de campos contido no `fieldset`.

Sintaxe:

```
<form action="http://site.com/scripts/formulario.php" method="post">  
  <fieldset>  
    <legend>Dados pessoais</legend>  
    <!-- campos para coleta de dados pessoais -->  
  </fieldset>  
  <fieldset>  
    <legend>Suas preferências</legend>  
    <!-- campos para coleta de preferências-->  
  </fieldset>  
  <!-- mais marcação do formulário -->  
</form>
```

## **label**

Esse elemento destina-se a marcar um rótulo (texto com o nome do campo) para os campos do formulário. Os rótulos podem se implícitos ou explícitos conforme eles sejam ou não container do campo. Seu principal atributo é:

- `for` – Usado quando o rótulo é explícito e destina-se a associar o rótulo com seu campo.

Sintaxe para rótulo implícito: O elemento `label` é container do campo para o qual seu texto é o rótulo.

```
<label>Nome:  
  <input type="text" name="nome"> <!-- campo para coletar o nome -->  
</label>
```

Sintaxe para rótulo explícito: O elemento `label` recebe o atributo `for` com o valor igual ao valor do atributo `id` do campo. Esse valor recebe o nome que o desenvolvedor quiser (no exemplo escolhemos *nome*), lembrando apenas que o valor de um `id` deve ser único na página.

```
<label for="nome">Nome:</label>  
  <input type="text" id="nome" name="nome"> <!-- campo para coletar o nome -  
  -->
```

O arquivo *fieldset-legend-label.html*, que mostra a sintaxe de marcação e a renderização dos elementos `fieldset`, `legend`, `label` e `input`, está disponível para consulta online e download na pasta *capitulo9*.

## input

Esse elemento destina-se a coletar dados textuais, marcar preferências, fazer seleção, enviar dados, coletar data, hora, email, telefone, números etc. É o elemento mais versátil dos formulários e cada uma de suas finalidades é definida com o atributo `type`. Existem dezenove diferentes tipos desse elemento. Apresentamos a seguir cada um dos valores do atributo `type` para esse elemento e descrevemos sua finalidade.

- `text` – Define um campo para coleta de textos. Esse é o tipo de campo padrão. Se um navegador não oferecer suporte para um determinado tipo de campo, ele considera o campo desse tipo.
- `hidden` – Define um campo que não é renderizado (mostrado para o usuário) e destina-se à coleta de dados sem conhecimento do usuário, como o número de IP da máquina do usuário.
- `search` – Define um campo para busca.
- `tel` – Define um campo para coleta de número de telefone.

- url – Define um campo para coleta de endereço na web.
- email – Define um campo para coleta de email.
- password – Define um campo para coleta de senha.
- date – Define um campo para coleta de data.
- time – Define um campo para coleta de hora.
- number – Define um campo para coleta de números.
- range – Define um campo para coleta de números compreendidos em determinado intervalo.
- color – Define um campo para coleta de cores.
- checkbox – Define um campo para coleta de mais de uma opção.
- radio – Define um campo para coleta de uma só opção.
- file – Define um campo para coleta de arquivo.
- submit – Define um campo para enviar o formulário.
- image – Transforma uma imagem em um botão.
- reset – Botão para limpar campos de formulários previamente preenchidos.
- button – Cria um botão.

O atributo booleano `required`, quando definido para qualquer campo de entrada de dados de um formulário, faz com que o navegador valide o campo antes de enviar o formulário. Se o campo for deixado em branco, ou, para alguns tipos de `input`, se o dado entrado pelo usuário estiver em desacordo com o formato de dado esperado, o navegador cria uma caixa de alerta ao usuário, informando que o campo não foi validado e solicitando nova entrada corrigida.

Atributo booleano é aquele cuja simples presença no elemento causa o efeito previsto. Não há necessidade de declarar um valor para ele, a menos que se queira que a marcação HTML seja compatível com XML. Nesse caso, o valor do atributo booleano é igual ao seu nome.

O arquivo *input.html*, que mostra a sintaxe para os diferentes tipos

de input, está disponível para consulta online e download na pasta *capitulo9*.

Abra o arquivo em um navegador, verifique a renderização de cada tipo de campo e comprove a validação do formulário, preenchendo os campos e clicando os botões para enviar o formulário.

- **HTML**

```
<form action="" method="">
  <p><label>Nome: <input type="text" name="nome" required></label></p>
  <p><label>Hidden: <input type="hidden" name="ip"></label></p>
  <p><label>Busca: <input type="search" name="q"></label></p>
  <p><label>Telefone: <input type="tel" name="tel" required></label></p>
  <p><label>URL: <input type="url" name="url" required></label></p>
  <p><label>Email: <input type="email" name="email" required></label></p>
  <p><label>Senha: <input type="password" name="pass" required></label>
</p>
  <p><label>Data: <input type="date" name="data" required></label></p>
  <p><label>Hora: <input type="time" name="hora" required></label></p>
  <p><label>Número: <input type="number" name="numero" required>
</label></p>
  <p><label>Intervalo: <input type="range" name="faixa" required></label>
</p>
  <p><label>Cor: <input type="color" name="cor" required></label></p>
  <p><label><input type="checkbox" name="abc"> Vermelho</label></p>
  <p><label><input type="radio" name="cde"> Sim</label></p>
  <p><label>Arquivo: <input type="file" name="arquivo" required></label>
</p>
  <p><input type="submit" value="ENVIAR"></p>
  <p><input type="image" src="botao.png" value="MEU BOTÃO"></p>
  <p><input type="reset" value="LIMPAR"></p>
  <p><input type="button" value="OK"></p>
</form>
```

O atributo name, presente em campos de entrada de dados em formulários, destina-se a servir como referência para coleta do dado entrado no campo pelos scripts de processamento.

Apresentamos a seguir os atributos comuns ao elemento input e descrevemos suas finalidades.

- `maxlength` e `minlength` – Destinam-se a definir o número máximo e mínimo, respectivamente, de caracteres que podem ser digitados pelo usuário em um campo de formulário.
- `size` – destina-se a definir o número de caracteres a ser visualizado no campo. Como consequência, esses caracteres determinam a largura do campo `input`, sendo 20 o valor padrão. Contudo, considerando que a largura de um caractere depende do próprio caractere (por exemplo: `i` tem largura diferente de `m`) e do tipo de caractere (monoespaçado tem largura maior do que serif), a definição do atributo `size` acaba por se tornar de pouco ou nenhum valor prático. Assim, é preferível que se defina a largura de um `input` com uso da propriedade `width` das CSS, evitando-se usar `size`, tendo em vista não só as inconsistências próprias desse atributo, como também o fato de esse ser um atributo de apresentação. E, como sabemos, apresentação é com as CSS.
- `value` – Destina-se a definir um valor inicial (padrão) para o campo.
- `readonly` – Destina-se a fazer com que o campo seja somente para leitura, não sendo possível a entrada de dados. Esse é um atributo booleano, isto é, não é necessário atribuir um valor a ele.
- `disabled` – Destina-se a desabilitar o campo, não sendo possível a entrada de dados. Esse é um atributo booleano, isto é, não é necessário atribuir um valor a ele.
- `autofocus` – Destina-se a dar o foco ao campo quando a página é carregada. Esse é um atributo booleano, isto é, não é necessário atribuir um valor a ele.
- `required` – Destina-se a informar ao navegador que esse é um campo de preenchimento obrigatório, ou seja, deve ser validado pelo navegador antes do envio dos dados do formulário. Em resposta a esse atributo, normalmente o navegador coloca, automaticamente, uma caixa de mensagem contendo um alerta ao usuário caso o campo não seja validado. Esse é um atributo booleano, isto é, não é necessário atribuir um valor a ele.



- **pattern** – Destina-se a definir uma expressão regular que retorna as possíveis combinações de caracteres que serão admitidas como valor a ser entrado no campo, ou seja, uma expressão regular contra a qual o dado entrado no campo deve ser casado. O estudo de expressões regulares está fora do escopo deste livro. Caso o leitor não saiba o que é expressão regular, comece lendo um artigo hospedado em <http://kwz.me/lp>. Por exemplo: A expressão regular `[A-Z]{5}` casa com qualquer combinação de cinco letras maiúsculas.
- **min e max** – Destina-se a definir valores máximo e mínimo permitidos como dado entrado em um campo. Define-se em campos input dos tipos `type="number"`, `type="date"`, `type="time"` e `type="range"`.
- **step** – Destina-se a definir qual o valor do intervalo de crescimento dos valores de dentro de um campo. Por exemplo: em um campo com `min="10"` e `max="100"`, definindo-se `step="10"` faz com que os valores possíveis de serem entrados no campos estejam entre 10 e 100 em intervalos de 10, ou seja, 10, 20, 30, 40, 50, e assim até 100.
- **placeholder** – Destina-se a criar um valor inicial dentro do campo que servirá como dica de preenchimento.

O arquivo *atributos.html*, que mostra a renderização e o efeito desses atributos, está disponível para consulta online e download na pasta *capitulo9*.

## **button**

Esse elemento destina-se a criar um botão.

A sintaxe de marcação desse elemento é mostrada a seguir.

```
<button type="button">Meu botão</button>
```

Os valores do atributo `type` são listados a seguir.

- **button** – Define um botão de uso geral e não precisa estar necessariamente em um formulário. Pode conter um link, pois

admite o atributo href.

- submit – Define um botão para envio de dados de um formulário.
- reset – Define um botão para limpeza de dados dos campos de um formulário.

## **select**

Esse elemento destina-se a criar uma caixa do tipo dropdown contendo opções de seleção.

## **option**

Esse elemento destina-se a marcar as opções de seleção de um elemento select.

## **optgroup**

Esse elemento destina-se a agrupar opções em uma caixa de seleção do tipo select.

A sintaxe de marcação dos elementos select, option e optgroup é mostrada a seguir.

```
<select>
  <optgroup label="Frutas">
    <option>Abacaxi</option>
    <option>Banana</option>
    <option>Laranja</option>
  <optgroup>
    <optgroup label="Cores">
      <option>Vermelha</option>
      <option>Verde</option>
      <option>Azul</option>
    </optgroup>
    <optgroup label="Bairros">
      <option>Leme</option>
      <option>Urca</option>
    </optgroup>
</select>
```

## **datalist**

Esse elemento destina-se a criar uma lista de opções para escolher um valor que vai preencher um campo. Difere do select, pois, enquanto select destina-se a selecionar diretamente uma opção, datalist destina-se a selecionar uma opção para preencher um campo.

A sintaxe de marcação desse elemento é mostrada a seguir. Nela o valor selecionado preencherá um campo do tipo input.

```
<label>Sexo
  <input name="sexo" list="sexos">
    <datalist id="sexos">
      <option value="Feminino">
      <option value="Masculino">
    </datalist>
</label>
```

Notar, conforme consta nesse exemplo, que, para atrelar um elemento datalist a um campo input, é necessário definir um atributo id para datalist com o mesmo valor do atributo list do input, que no caso do exemplo mostrado foi "sexos".

## **textarea**

Esse elemento destina-se a criar uma área de entrada de texto.

A sintaxe de marcação desse elemento é mostrada a seguir.

```
<label for="msg">Sua mensagem:</label>
<textarea id="msg" rows="10" cols="30"></textarea>
```

Os principais atributos para esse elemento são listados a seguir.

- **rows** – Define o número de linhas a ser mostrado na área de entrada de dados.
- **cols** – Define o número de caracteres a ser mostrado na área de entrada de dados.

Esses atributos definem a largura e a altura da área de texto. Por tratar-se de atributo que controla apresentação, dê preferência para o uso das propriedades width e height das CSS.

O arquivo *campos-de-formulario.html*, que demonstra a sintaxe e renderização dos elementos mostrados, está disponível para consulta online e download na pasta *capitulo9*.

# APÊNDICE A

## Elementos da HTML5

Esta tabela foi extraída do site do W3C e relaciona os elementos previstos na especificação da HTML5. Fornece, ainda, uma breve descrição dos elementos, além de relacionar os atributos a eles aplicáveis.

Incluímos na tabela mais cinco elementos que estão previstos na HTML5 Nightly (a próxima edição da HTML). Ressaltamos esses casos colocando uma observação na coluna Descrição.

Elemento	Descrição	Atributos
A	Hyperlink	globais; href; target; download; rel; hreflang; type
abbr	Abreviação	globais
address	Informações de contato para uma página ou seção	globais
área	Área em um mapa de imagem com ou sem hyperlink	globais; alt; coords; shape; href; target; download; rel; hreflang; type
article	Conteúdo passível de sindicalização (distribuição independente) ou conteúdo reusável	globais
aside	Conteúdo tangencialmente relacionado a outro conteúdo	globais

Elemento	Descrição	Atributos
áudio	Player de áudio	globais; src; crossorigin; preload; autoplay; mediagroup; loop; muted; controls
B	Marca palavras-chave	globais
base	URL base para hyperlinks	globais; href; target
BDI	Marca de direção de texto	globais
bdo	Formata direção de texto	globais
blockquote	Citação longa	globais; cite
body	Corpo do documento	globais; onafterprint; onbeforeprint; onbeforeunload; onhashchange; onlanguagechange; onmessage; onoffline; ononline; onpagehide; onpageshow; onpopstate; onstorage; onunload
BR	Quebra de linha	globais
Button	Controle tipo botão	globais; autofocus; disabled; form; formaction; formenctype; formmethod; formnovalidate; formtarget; menu; name; type; value
canvas	Área de bitmap para scripts	globais; width; height
caption	Legenda de tabela	globais
cite	Título de um trabalho	globais
code	Código máquina	globais
col	Coluna de tabela	globais; span
colgroup	Grupamento de colunas de uma tabela	globais; span
data	Dados para leitura por máquina	globais; value

Elemento	Descrição	Atributos
datalist	Container para opções de um controle tipo combo box	globais
dd	Conteúdo para correspondente elemento(s) dt	globais
Del	Conteúdo removido do documento	globais; cite; datetime
details	Controle para apresentar detalhes (HTML5 Nightly)	globais; open
dfn	Instância de definição	globais
dialog	Caixa de diálogo (HTML5 Nightly)	globais; open
div	Container genérico	globais
dl	Container para listas de definição	globais
dt	Legenda conteúdos em elemento(s) dd	globais
em	Forte ênfase	globais
embed	Plugin	globais; src; type; width; height; any*
fieldset	Grupamento de controles de formulário	globais; disabled; form; name
figcaption	Legenda de figuras	globais
figure	Figura com legenda opcional	globais

Elemento	Descrição	Atributos
footer	Rodapé para uma página ou seção	globais
form	Formulário	globais; accept-charset; action; autocomplete; enctype; method; name; novalidate; target
h1, h2, h3, h4, h5, h6	Cabeçalho de uma seção	globais
head	Container para metadados do documento	globais
header	Conteúdo introdutório ou de auxílio de navegação para uma página ou seção	globais
hgroup	Grupamento de cabeçalhos (ver nota)	globais
hr	Quebra temática de linha	globais
HTML	Elemento raiz	globais; manifest
I	Alteração de entonação	globais
iframe	Contexto de navegador aninhado	globais; src; srcdoc; name; sandbox; seamless; allowfullscreen; width; height
img	Imagem	globais; alt; src; srcset; crossorigin; usemap; ismap; width; height
input	Controle de formulário	globais; accept; alt; autocomplete; autofocus; checked; dirname; disabled; form; formaction; formenctype; formmethod; formnovalidate; formtarget; height; inputmode; list; max; maxlength; min; minlength; multiple; name; pattern; placeholder; readonly; required; size; src; step; type; value; width
ins	Acréscimo no documento	globais; cite; datetime



Elemento	Descrição	Atributos
kbd	Entrada de usuário	globais
keygen	Controle de formulário gerador de chave criptográfica	globais; autofocus; challenge; disabled; form; keytype; name
label	Legenda para controle de formulário	globais; form; for
legend	Legenda para fieldset	globais
li	Item de uma lista	globais; value*
link	Metadado link	globais; href; crossorigin; rel; media; hreflang; type; sizes
main	Conteúdo principal do documento	globais
map	Mapa de imagem	globais; name
Mark	Destaque	globais
menu	Menu de comandos (HTML5 Nightly)	globais; type; label
menuitem	Menu de comandos (HTML5 Nightly)	globais; type; label; icon; disabled; checked; radiogroup; default; command
meta	Metadado texto	globais; name; http-equiv; content; charset
meter	Medidor	globais; value; min; max; low; high; optimum
nav	Seção contendo links de navegação	globais
noscript	Conteúdo alternativo para script	globais

Elemento	Descrição	Atributos
object	Imagem, contexto de navegador aninhado ou plugin	globais; data; type; typemustmatch; name; usemap; form; width; height
ol	Lista ordenada	globais; reversed; start; type
optgroup	Grupamento de opções em um controle list box	globais; disabled; label
option	Opção em um controle list box ou combo box	globais; disabled; label; selected; value
output	Container para um valor calculado	globais; for; form; name
P	Parágrafo	globais
param	Parâmetro para o elemento object	globais; name; value
pré	Bloco de texto pré-formatado	globais
progress	Barra de progresso	globais; value; max;
Q	Citação curta	globais; cite
Rb	Base ruby texto	globais
rp	Parênteses para anotação ruby em texto	globais
RT	Anotação ruby em texto	globais
rtc	Container de texto para anotação ruby	globais
ruby	Anotação ruby	globais
S	Texto incorreto	globais
samp	Saída genérica de computador	globais

Elemento	Descrição	Atributos
script	Script incorporado no documento	globais; src; async; defer; type; charset; crossorigin
section	Documento ou seção genérica de uma aplicação.	globais
select	Controle do tipo list box	globais; autocomplete; autofocus; disabled; form; multiple; name; required; size
small	Comentário à parte	globais
source	Mídia source para vídeo ou áudio	globais; src; type
span	Container genérico	globais
strong	Importância	globais
style	Informa sobre estilização incorporada no documento	globais; media; type; scoped
sub	Subscrito	globais
summary	Legenda para o elemento details (HTML5 Nightly)	globais
sup	Sobrescrito	globais
table	Tabela	globais; border; sortable
tbody	Grupo de linhas em uma tabela	globais
td	Célula de tabela	globais; colspan; rowspan; headers
template	Template	globais
textarea	Campo de texto multilinhas	globais; autofocus; cols; dirname; disabled; form; inputmode; maxlength; minlength; name; placeholder; readonly; required; rows; wrap
tfoot	Grupo de linhas de rodapé de uma tabela	globais

Elemento	Descrição	Atributos
th	Célula de cabeçalho de uma tabela	globais; colspan; rowspan; headers; scope; sorted; abbr
thead	Grupo de células de cabeçalho de uma tabela	globais
time	Data e/ou data-hora	globais; datetime
title	Título do documento	globais
TR	Linha de tabela	globais
track	Trilha de texto	globais; default; kind; label; src; srclang
U	Palavra-chave	globais
ul	Container para lista	globais
var	Variável	globais
video	Player de vídeo	globais; src; crossorigin; poster; preload; autoplay; mediagroup; loop; muted; controls; width; height
wbr	Quebra de linha	globais



Nota: o elemento hgroup, embora ainda não incluso na próxima edição da HTML, tem grandes chances de constar da especificação.

# APÊNDICE B

## Propriedades CSS

Neste apêndice, mostraremos as propriedades CSS3 descrevendo sumariamente suas finalidades, relacionando os valores válidos para cada uma delas, seu valor inicial, o comportamento para herança CSS, a base para cálculo de valores relativos, a mídia à qual se aplicam e a versão CSS na qual foram criadas.

### B.1 Propriedades das CSS3

Como vimos no capítulo 1, cada nova versão das CSS não somente amplia o conjunto de funcionalidades já existentes, mas também mantém intactas as características e a forma de emprego das funcionalidades constantes das versões anteriores. Veremos a seguir uma descrição das propriedades das CSS3 que presumivelmente já atingiram um estágio estável e deixaremos de fora várias propriedades que ainda se encontram em fase de estudos nas especificações.

Contudo, não há garantias de que algumas das propriedades mostradas permanecerão como estão até o lançamento da especificação final de cada um dos módulos das CSS3.

Mostraremos não somente as novas propriedades, mas também as constantes das versões anteriores que continuam válidas e não mudarão. Na descrição de cada propriedade, assinalamos a versão na qual ela foi criada.

Muitas das novas propriedades criadas pelas CSS3 já estão implementadas em navegadores modernos e podem ser usadas com toda segurança. Algumas delas, quando não suportadas por

esse ou aquele navegador, não causam perda de conteúdo ou danos ao layout. A prática e, sobretudo, a experimentação dirão ao desenvolvedor quais propriedades CSS3 usar e quais não usar.



Para facilitar o trabalho de investigação dos leitores deste livro, vamos criar vários exemplos de uso das novas propriedades e disponibilizá-los no site do livro. Tais exemplos não somente servirão para ilustrar a teoria, mas também para valer de fonte de consulta para propriedades já suportadas.

## B1.1 Convenções

As convenções adotadas para apresentar as propriedades CSS são conforme as descrições a seguir:

### Valores:

- Palavras-chave (por exemplo: auto, disc etc.).
- Tipo de dado: entre “<” e “>” (por exemplo: <length>, <percentage> etc.).
- Valores seguidos devem ocorrer na ordem em que aparecem.
- Barra vertical (|): separa duas ou mais alternativas na ordem em que devem ocorrer.
- Barra vertical dupla (||): separa duas ou mais alternativas – uma ou mais alternativas devem ocorrer em qualquer ordem.
- Duplo “e” comercial (&&): separa dois ou mais componentes – todos devem ocorrer em qualquer ordem.
- Colchetes ([ ]): indicam grupamento.

### Modificadores:

- Asterisco (\*): valor deve ocorrer zero ou mais vezes.
- Sinal de adição (+): valor deve ocorrer uma ou mais vezes.
- Interrogação (?): a ocorrência do valor é opcional.
- Par de números entre chaves ({A,B}): valor deve ocorrer no mínimo A vezes e no máximo B vezes.

## alignment-adjust

Essa propriedade destina-se a alinhar com precisão elementos inline.

<b>Valor inicial</b>	auto
<b>Aplica-se</b>	a elementos inline
<b>Herdada</b>	não
<b>Referência para porcentagem</b>	propriedade line-height do elemento
<b>Mídia</b>	visual
<b>Criada na versão</b>	CSS3

### Valores

Os valores possíveis para alignment-adjust são:

baseline | before-edge | text-before-edge | middle | central | after-edge | text-after-edge | ideographic | alphabetic | hanging | mathematical | <percentage> | <length>

## alignment-baseline

Essa propriedade destina-se a definir o alinhamento de um elemento inline com relação ao seu elemento-pai. Ou seja, qual ponto das linhas de base do elemento-pai servirá de referência para o elemento inline.

<b>Valor inicial</b>	auto
<b>Aplica-se</b>	a elementos inline
<b>Herdada</b>	não
<b>Referência para porcentagem</b>	—
<b>Mídia</b>	visual
<b>Criada na versão</b>	CSS3

### Valores

Os valores possíveis para alignment-baseline são:

baseline | use-script | before-edge | text-before-edge | middle | central | after-edge | text-after-edge | ideographic | alphabetic | hanging | mathematical

## animation-name

Essa propriedade destina-se a definir uma lista de nomes ou o nome da animação a ser aplicada ao elemento.

Valor inicial	nenhum
Aplica-se	a elementos nível de bloco e inline
Herdada	não
Referência para porcentagem	–
Mídia	visual
Criada na versão	CSS3

### Valores

Os valores possíveis para animation-name são:

none | IDENT [, none | IDENT]\*

IDENT = nome escolhido para a animação

## animation-duration

Essa propriedade destina-se a definir o tempo de duração de um ciclo da animação.

Valor inicial	0
Aplica-se	a elementos nível de bloco e inline
Herdada	não
Referência para porcentagem	–
Mídia	visual
Criada na versão	CSS3

### Valores

Os valores possíveis para animation-duration são:

<time> [, <time>]\*

## animation-delay

Essa propriedade destina-se a definir o tempo de espera para início da animação.

Valor inicial	0
---------------	---



<b>Aplica-se</b>	a elementos nível de bloco e inline
<b>Herdada</b>	não
<b>Referência para porcentagem</b>	–
<b>Mídia</b>	visual
<b>Criada na versão</b>	CSS3

## Valores

Os valores possíveis para animation-delay são:

<time> [, <time>]\*

Valores negativos são admitidos e significam que a animação inicia imediatamente no ponto em que ela estaria se já tivesse sido transcorrida a quantidade de tempo especificada como valor negativo.

## animation-direction

Essa propriedade destina-se a definir o sentido da animação.

<b>Valor inicial</b>	normal
<b>Aplica-se</b>	a elementos nível de bloco e inline
<b>Herdada</b>	não
<b>Referência para porcentagem</b>	–
<b>Mídia</b>	visual
<b>Criada na versão</b>	CSS3

## Valores

Os valores possíveis para animation-direction são:

normal | alternate | [, normal | alternate]\*

## animation-timing-function

Essa propriedade destina-se a definir a maneira como a animação se processa ao longo de um ciclo.

<b>Valor inicial</b>	ease
<b>Aplica-se</b>	a elementos nível de bloco e inline
<b>Herdada</b>	não

<b>Referência para porcentagem</b>	–
<b>Mídia</b>	visual
<b>Criada na versão</b>	CSS3

## Valores

Os valores possíveis para animation-timing-function são:

ease | linear | ease-in | ease-out | ease-in-out | cubic-bezier(<number>, <number>, <number>, <number>) [, ease | linear | ease-in | ease-out | ease-in-out | cubic-bezier(<number>, <number>, <number>, <number>)]\*

## animation-iteration-count

Essa propriedade destina-se a definir o número de vezes que o ciclo da animação se repete.

<b>Valor inicial</b>	1
<b>Aplica-se</b>	a elementos nível de bloco e inline
<b>Herdada</b>	não
<b>Referência para porcentagem</b>	–
<b>Mídia</b>	visual
<b>Criada na versão</b>	CSS3

## Valores

Os valores possíveis para animation-iteration-count são:

infinite | <number> [, infinite | <number>]\*

## animation-play-state

Essa propriedade destina-se a definir o estado da animação, ou seja, se está em pausa ou em movimento.

<b>Valor inicial</b>	running
<b>Aplica-se</b>	a elementos nível de bloco e inline
<b>Herdada</b>	não
<b>Referência para porcentagem</b>	–
<b>Mídia</b>	visual
<b>Criada na versão</b>	CSS3

## Valores

Os valores possíveis para animation-play-state são:

running | paused [, running | paused]\*

Essa funcionalidade pode ser obtida com uso de outros métodos, e, por essa razão, poderá ser removida das especificações.

## animation

Essa propriedade é a forma abreviada de declarar seis propriedades da animação, conforme relacionadas a seguir, no subtítulo Valores.

<b>Valor inicial</b>	ver propriedades individuais
<b>Aplica-se</b>	a elementos nível de bloco e inline
<b>Herdada</b>	não
<b>Referência para porcentagem</b>	–
<b>Mídia</b>	visual
<b>Criada na versão</b>	CSS3

## Valores

Os valores possíveis para animation são:

[<animation-name> || <animation-duration> || <animation-timing-function> || <animation-delay> || <animation-iteration-count> || <animation-direction>] [, [<animation-name> || <animation-duration> || <animation-timing-function> || <animation-delay> || <animation-iteration-count> || <animation-direction>] ]\*

## appearance

Usa-se essa propriedade para fazer com que um elemento tenha aparência semelhante à de um elemento de interface standard na plataforma usada.

<b>Valor inicial</b>	normal
<b>Aplica-se</b>	a todos os elementos
<b>Herdada</b>	não
<b>Referência para porcentagem</b>	–
<b>Mídia</b>	visual, interativa

Criada na versão	CSS3
------------------	------

## Valores

normal | <appearance> | inherit

Alguns valores possíveis para appearance são: button, push-button, hyperlink, radio-button, checkbox, pop-up-menu, list-menu, radio-group, checkbox-group, field, password.

## azimuth

Essa propriedade destina-se à mídia falada e é empregada para um azimute para o som.

Valor inicial	center
Aplica-se	a todos os elementos
Herdada	sim
Referência para porcentagem	–
Mídia	falada
Criada na versão	CSS2

## Valores

Os valores possíveis para azimuth são:

<angle> | [[ left-side | far-left | left | center-left | center | center-right | right | far-right | right-side ] || behind ] | leftwards | rightwards | inherit

## backface-visibility

Essa propriedade destina-se a definir se a parte posterior de um elemento será ou não visível quando a ele se aplica uma transformação 3D.

Valor inicial	visible
Aplica-se	a elementos nível de bloco e inline
Herdada	não
Referência para porcentagem	–
Mídia	visual
Criada na versão	CSS3

## Valores

Os valores possíveis para backface-visibility são:

visible | hidden

## background-image

Essa propriedade destina-se a definir uma ou mais imagens de fundo para um elemento.

Valor inicial	none
Aplica-se	a todos os elementos
Herdada	não
Referência para porcentagem	–
Mídia	visual
Criada na versão	CSS1

## Valores

<bg-image> [<bg-image>]\*

Os valores possíveis para bg-image são:

URI | none

## background-position

Essa propriedade destina-se a definir a posição inicial da imagem de fundo para um elemento.

Valor inicial	0 0
Aplica-se	a todos os elementos
Herdada	não
Referência para porcentagem	dimensão da área de posicionamento do fundo menos dimensão da imagem
Mídia	visual
Criada na versão	CSS1

## Valores

<bg-position> [,<bg-position>]\*

Os valores possíveis para bg-position são:

[[ top | bottom ] | [ <percentage> | <length> | left | center | right ] [ <percentage> | <length> | top | center | bottom ]? | [ center | [ left | right ] [ <percentage> | <length> ]? ] && [ center | [ top | bottom ] [ <percentage> | <length> ]? ]]

## background-size

Essa propriedade destina-se a definir as dimensões da imagem de fundo para um elemento.

<b>Valor inicial</b>	auto
<b>Aplica-se</b>	a todos os elementos
<b>Herdada</b>	não
<b>Referência para porcentagem</b>	dimensão da área de posicionamento do fundo
<b>Mídia</b>	visual
<b>Criada na versão</b>	CSS3

### Valores

<bg-size> [,<bg-size>]\*

Os valores possíveis para bg-size são:

[ <length> | <percentage> | auto ]{1,2} | cover | contain

## background-origin

Essa propriedade destina-se a definir as dimensões da imagem de fundo para um elemento.

<b>Valor inicial</b>	padding-box
<b>Aplica-se</b>	a todos os elementos
<b>Herdada</b>	não
<b>Referência para porcentagem</b>	–
<b>Mídia</b>	visual
<b>Criada na versão</b>	CSS3

### Valores

<box> [,<box>]\*

Os valores possíveis para box são:

padding-box | border-box | content-box

## background-clip

Essa propriedade destina-se a definir a área, no elemento, na qual a imagem de fundo será inserida.

Valor inicial	border-box
Aplica-se	a todos os elementos
Herdada	não
Referência para porcentagem	–
Mídia	visual
Criada na versão	CSS3

### Valores

<box> [, <box>]\*

Os valores possíveis para box são:

padding-box | border-box | content-box

## background-attachment

Essa propriedade destina-se a definir se a imagem de fundo será fixa em relação à viewport, se rolará com o elemento ou rolará com seu conteúdo.

Valor inicial	scroll
Aplica-se	a todos os elementos
Herdada	não
Referência para porcentagem	–
Mídia	visual
Criada na versão	CSS1

### Valores

<attachment> [, <attachment>]\*

Os valores possíveis para attachment são:

scroll | fixed | local

## background-repeat

Essa propriedade destina-se a definir como será a repetição da

imagem de fundo após ela ter sido dimensionada (background-size) e posicionada (background-position).

<b>Valor inicial</b>	repeat
<b>Aplica-se</b>	a todos os elementos
<b>Herdada</b>	não
<b>Referência para porcentagem</b>	–
<b>Mídia</b>	visual
<b>Criada na versão</b>	CSS1

## Valores

<repeat-style> [,<repeat-style>]\*

Os valores possíveis para repeat-style são:

repeat-x | repeat-y | repeat | space | round | no-repeat | repeat

## background-color

Essa propriedade destina-se a definir a cor de fundo para um elemento. A imagem de fundo, se houver uma, será posicionada sobre a cor de fundo.

<b>Valor inicial</b>	transparent
<b>Aplica-se</b>	a todos os elementos
<b>Herdada</b>	não
<b>Referência para porcentagem</b>	–
<b>Mídia</b>	visual
<b>Criada na versão</b>	CSS1

## Valores

<color>

Os valores possíveis para color são:

<keyword> | <numeric-values>

## background

Essa propriedade é a forma abreviada de declarar as propriedades para background.

--	--



<b>Valor inicial</b>	ver propriedades individuais
<b>Aplica-se</b>	a todos os elementos
<b>Herdada</b>	não
<b>Referência para porcentagem</b>	ver propriedades individuais
<b>Mídia</b>	visual
<b>Criada na versão</b>	CSS1

## Valores

[ <bg-layer>, ]\* <final-bg-layer>

<bg-layer> = <bg-image> || <bg-position> [ / <bg-size> ]? || <repeat-style> || <attachment> || <box>{1,2}

<final-bg-layer> = <bg-image> || <bg-position> [ / <bg-size> ]? || <repeat-style> || <attachment> || <box>{1,2} || <'background-color'>

## border

Essa propriedade é a forma abreviada de declarar as propriedades para as quatro bordas de um elemento.

<b>Valor inicial</b>	ver propriedades individuais
<b>Aplica-se</b>	a todos os elementos
<b>Herdada</b>	não
<b>Referência para porcentagem</b>	ver propriedades individuais
<b>Mídia</b>	visual
<b>Criada na versão</b>	CSS1

## Valores

<border-width> || <border-style> || <color>

<border-width> = <lenght> | thin | medium | thick

<border-style> = none | hidden | dotted | dashed | solid | double | groove | ridge | inset | outset

<color> = <keyword> | <numeric-values>

## border-top

Essa propriedade é a forma abreviada de declarar as propriedades

para a borda superior de um elemento.

<b>Valor inicial</b>	ver propriedades individuais
<b>Aplica-se</b>	a todos os elementos
<b>Herdada</b>	não
<b>Referência para porcentagem</b>	–
<b>Mídia</b>	visual
<b>Criada na versão</b>	CSS1

## Valores

<border-width> || <border-style> || <color>

<border-width> = <lenght> | thin | medium | thick

<border-style> = none | hidden | dotted | dashed | solid | double | groove | ridge  
| inset | outset

<color> = <keyword> | <numeric-values>

## border-right

Essa propriedade é a forma abreviada de declarar as propriedades para a borda direita de um elemento.

<b>Valor inicial</b>	ver propriedades individuais
<b>Aplica-se</b>	a todos os elementos
<b>Herdada</b>	não
<b>Referência para porcentagem</b>	–
<b>Mídia</b>	visual
<b>Criada na versão</b>	CSS1

## Valores

<border-width> || <border-style> || <color>

<border-width> = <lenght> | thin | medium | thick

<border-style> = none | hidden | dotted | dashed | solid | double | groove | ridge  
| inset | outset

<color> = <keyword> | <numeric-values>

## border-bottom

Essa propriedade é a forma abreviada de declarar as propriedades para a borda inferior de um elemento.

<b>Valor inicial</b>	ver propriedades individuais
<b>Aplica-se</b>	a todos os elementos
<b>Herdada</b>	não
<b>Referência para porcentagem</b>	–
<b>Mídia</b>	visual
<b>Criada na versão</b>	CSS1

### Valores

<border-width> || <border-style> || <color>

<border-width> = <lenght> | thin | medium | thick

<border-style> = none | hidden | dotted | dashed | solid | double | groove | ridge  
| inset | outset

<color> = <keyword> | <numeric-values>

## border-left

Essa propriedade é a forma abreviada de declarar as propriedades para a borda esquerda de um elemento.

<b>Valor inicial</b>	ver propriedades individuais
<b>Aplica-se</b>	a todos os elementos
<b>Herdada</b>	não
<b>Referência para porcentagem</b>	–
<b>Mídia</b>	visual
<b>Criada na versão</b>	CSS1

### Valores

<border-width> || <border-style> || <color>

Os valores possíveis para border-width são:

<border-width> = <lenght> | thin | medium | thick

<border-style> = none | hidden | dotted | dashed | solid | double | groove | ridge

| inset | outset

<color> = <keyword> | <numeric-values>

## **border-width**

Essa propriedade destina-se a definir a espessura para as quatro bordas de um elemento.

<b>Valor inicial</b>	ver propriedades individuais
<b>Aplica-se</b>	a todos os elementos
<b>Herdada</b>	não
<b>Referência para porcentagem</b>	largura do elemento-pai
<b>Mídia</b>	visual
<b>Criada na versão</b>	CSS1

### **Valores**

Os valores possíveis para <border-width> são:

<border-width>{1,4} = <lenght> | thin | medium | thick

## **border-top-width**

Essa propriedade destina-se a definir a espessura para a borda superior de um elemento.

<b>Valor inicial</b>	medium
<b>Aplica-se</b>	a todos os elementos
<b>Herdada</b>	não
<b>Referência para porcentagem</b>	largura do elemento-pai
<b>Mídia</b>	visual
<b>Criada na versão</b>	CSS1

### **Valores**

Os valores possíveis para border-top-width são:

<lenght> | thin | medium | thick

## **border-right-width**

Essa propriedade destina-se a definir a espessura para a borda

direita de um elemento.

<b>Valor inicial</b>	medium
<b>Aplica-se</b>	a todos os elementos
<b>Herdada</b>	não
<b>Referência para porcentagem</b>	largura do elemento-pai
<b>Mídia</b>	visual
<b>Criada na versão</b>	CSS1

### **Valores**

Os valores possíveis para border-right-width são:

<length> | thin | medium | thick

### **border-bottom-width**

Essa propriedade destina-se a definir a espessura para a borda inferior de um elemento.

<b>Valor inicial</b>	medium
<b>Aplica-se</b>	a todos os elementos
<b>Herdada</b>	não
<b>Referência para porcentagem</b>	largura do elemento-pai
<b>Mídia</b>	visual
<b>Criada na versão</b>	CSS1

### **Valores**

Os valores possíveis para border-bottom-width são:

<length> | thin | medium | thick

### **border-left-width**

Essa propriedade destina-se a definir a espessura para a borda esquerda de um elemento.

<b>Valor inicial</b>	medium
<b>Aplica-se</b>	a todos os elementos
<b>Herdada</b>	não
<b>Referência para porcentagem</b>	largura do elemento-pai

<b>Mídia</b>	visual
<b>Criada na versão</b>	CSS1

## Valores

Os valores possíveis para border-left-width são:

<length> | thin | medium | thick

## border-style

Essa propriedade destina-se a definir o estilo para as quatro bordas de um elemento, exceto quando há uma imagem para o estilo da borda.

<b>Valor inicial</b>	ver propriedades individuais
<b>Aplica-se</b>	a todos os elementos
<b>Herdada</b>	não
<b>Referência para porcentagem</b>	–
<b>Mídia</b>	visual
<b>Criada na versão</b>	CSS1

## Valores

Os valores possíveis para <border-style> são:

<border-style>{1,4} = none | hidden | dotted | dashed | solid | double | groove | ridge | inset | outset

## border-top-style

Essa propriedade destina-se a definir o estilo para a borda superior de um elemento.

<b>Valor inicial</b>	none
<b>Aplica-se</b>	a todos os elementos
<b>Herdada</b>	não
<b>Referência para porcentagem</b>	–
<b>Mídia</b>	visual
<b>Criada na versão</b>	CSS1

## Valores

Os valores possíveis para <border-top-style> são:

<border-top-style>{1,4} = none | hidden | dotted | dashed | solid | double | groove | ridge | inset | outset

## **border-right-style**

Essa propriedade destina-se a definir o estilo para a borda direita de um elemento.

<b>Valor inicial</b>	none
<b>Aplica-se</b>	a todos os elementos
<b>Herdada</b>	não
<b>Referência para porcentagem</b>	–
<b>Mídia</b>	visual
<b>Criada na versão</b>	CSS1

### **Valores**

Os valores possíveis para <border-right-style> são:

<border-right-style>{1,4} = none | hidden | dotted | dashed | solid | double | groove | ridge | inset | outset

## **border-bottom-style**

Essa propriedade destina-se a definir o estilo para a borda inferior de um elemento.

<b>Valor inicial</b>	none
<b>Aplica-se</b>	a todos os elementos
<b>Herdada</b>	não
<b>Referência para porcentagem</b>	–
<b>Mídia</b>	visual
<b>Criada na versão</b>	CSS1

### **Valores**

Os valores possíveis para <border-bottom-style> são:

<border-bottom-style>{1,4} = none | hidden | dotted | dashed | solid | double | groove | ridge | inset | outset

## border-left-style

Essa propriedade destina-se a definir o estilo para a borda esquerda de um elemento.

Valor inicial	none
Aplica-se	a todos os elementos
Herdada	não
Referência para porcentagem	–
Mídia	visual
Criada na versão	CSS1

### Valores

Os valores possíveis para <border-left-style> são:

<border-left-style>{1,4} = none | hidden | dotted | dashed | solid | double | groove | ridge | inset | outset

## border-color

Essa propriedade destina-se a definir a cor para as quatro bordas de um elemento.

Valor inicial	ver propriedades individuais
Aplica-se	a todos os elementos
Herdada	não
Referência para porcentagem	–
Mídia	visual
Criada na versão	CSS1

### Valores

Os valores possíveis para <color> são:

<color> {1,4} = <keyword> | <numeric-values>

## border-top-color

Essa propriedade destina-se a definir a cor para a borda superior de um elemento.

Valor inicial	cor definida para o elemento



<b>Aplica-se</b>	a todos os elementos
<b>Herdada</b>	não
<b>Referência para porcentagem</b>	–
<b>Mídia</b>	visual
<b>Criada na versão</b>	CSS1

## Valores

Os valores possíveis para border-top-color são:

<color>{1,4} = <keyword> | <numeric-values>

## border-right-color

Essa propriedade destina-se a definir a cor para a borda direita de um elemento.

<b>Valor inicial</b>	cor definida para o elemento
<b>Aplica-se</b>	a todos os elementos
<b>Herdada</b>	não
<b>Referência para porcentagem</b>	–
<b>Mídia</b>	visual
<b>Criada na versão</b>	CSS1

## Valores

Os valores possíveis para border-right-color são:

<color>{1,4} = <keyword> | <numeric-values>

## border-bottom-color

Essa propriedade destina-se a definir a cor para a borda inferior de um elemento.

<b>Valor inicial</b>	cor definida para o elemento
<b>Aplica-se</b>	a todos os elementos
<b>Herdada</b>	não
<b>Referência para porcentagem</b>	–
<b>Mídia</b>	visual
<b>Criada na versão</b>	CSS1

## Valores

Os valores possíveis para border-bottom-color são:

<color>{1,4} = <keyword> | <numeric-values>

## border-left-color

Essa propriedade destina-se a definir a cor para a borda esquerda de um elemento.

<b>Valor inicial</b>	cor definida para o elemento
<b>Aplica-se</b>	a todos os elementos
<b>Herdada</b>	não
<b>Referência para porcentagem</b>	–
<b>Mídia</b>	visual
<b>Criada na versão</b>	CSS1

## Valores

Os valores possíveis para border-left-color são:

<keyword> | <numeric-values>

## border-image

Essa propriedade destina-se a definir como uma imagem será usada para criar as bordas de um elemento.

<b>Valor inicial</b>	ver propriedades individuais
<b>Aplica-se</b>	ver propriedades individuais
<b>Herdada</b>	não
<b>Referência para porcentagem</b>	–
<b>Mídia</b>	visual
<b>Criada na versão</b>	CSS3

## Valores

Os valores possíveis para border-image são:

<'border-image-source'> || <'border-image-slice'> [ / <'border-image-width'>? [ / <'border-image-outset'> ]? ]? || <'border-image-repeat'>

## border-image-source

Essa propriedade destina-se a definir a imagem que será usada para criar as bordas de um elemento.

<b>Valor inicial</b>	ver propriedades individuais
<b>Aplica-se</b>	ver propriedades individuais
<b>Herdada</b>	não
<b>Referência para porcentagem</b>	–
<b>Mídia</b>	visual
<b>Criada na versão</b>	CSS3

### Valores

Os valores possíveis para border-image-source são:

none | URI

## border-image-slice

Essa propriedade destina-se a definir como a imagem será recortada em nove regiões para ser usada na criação das bordas de um elemento.

<b>Valor inicial</b>	100%
<b>Aplica-se</b>	a todos os elementos, exceto elementos internos de tabelas, quando as bordas se sobrepõem (collapse)
<b>Herdada</b>	não
<b>Referência para porcentagem</b>	dimensões da imagem
<b>Mídia</b>	visual
<b>Criada na versão</b>	CSS3

### Valores

Os valores possíveis para border-image-slice são:

[ <number> | <percentage> ]{1,4} && fill?

## border-image-width

Essa propriedade destina-se a definir como a área da borda será recortada em nove regiões para ser usada na criação das bordas de

um elemento.

<b>Valor inicial</b>	1
<b>Aplica-se</b>	a todos os elementos, exceto elementos internos de tabelas, quando as bordas se sobrepõem (collapse)
<b>Herdada</b>	não
<b>Referência para porcentagem</b>	dimensões da área da imagem
<b>Mídia</b>	visual
<b>Criada na versão</b>	CSS3

### **Valores**

Os valores possíveis para border-image-width são:

[ <length> | <percentage> | <number> | auto ]{1,4}

### **border-image-outset**

Essa propriedade destina-se a definir quanto a área da borda se estenderá pelos quatro lados.

<b>Valor inicial</b>	0
<b>Aplica-se</b>	a todos os elementos, exceto elementos internos de tabelas, quando as bordas se sobrepõem (collapse)
<b>Herdada</b>	não
<b>Referência para porcentagem</b>	–
<b>Mídia</b>	visual
<b>Criada na versão</b>	CSS3

### **Valores**

Os valores possíveis para border-image-outset são:

[ <length> | <number> ]{1,4}

### **border-image-repeat**

Essa propriedade destina-se a definir quanto a área da borda se estenderá pelos quatro lados.

<b>Valor inicial</b>	stretch

<b>Aplica-se</b>	a todos os elementos, exceto elementos internos de tabelas, quando as bordas se sobrepõem (collapse)
<b>Herdada</b>	não
<b>Referência para porcentagem</b>	–
<b>Mídia</b>	visual
<b>Criada na versão</b>	CSS3

## Valores

Os valores possíveis para border-image-repeat são:

[ stretch | repeat | round ]{1,2}

## border-collapse

Essa propriedade destina-se a definir a apresentação das bordas das células de uma tabela como justaposta.

<b>Valor inicial</b>	separate
<b>Aplica-se</b>	a elementos table e inline-table
<b>Herdada</b>	sim
<b>Referência para porcentagem</b>	–
<b>Mídia</b>	visual
<b>Criada na versão</b>	CSS2

## Valores

Os valores possíveis para border-collapse são:

collapse | separate | inherit

## border-spacing

Essa propriedade destina-se a definir a distância que separa as bordas das células de uma tabela.

<b>Valor inicial</b>	0
<b>Aplica-se</b>	a elementos table e inline-table
<b>Herdada</b>	sim
<b>Referência para porcentagem</b>	–
<b>Mídia</b>	visual

## Valores

Os valores possíveis para border-spacing são:

<length> <length>? | inherit

## box-shadow

Essa propriedade define os efeitos de sombreamento para um elemento.

Valor inicial	none
Aplica-se	a todos os elementos
Herdada	não
Referência para porcentagem	—
Mídia	visual
Criada na versão	CSS3

## Valores

Os valores possíveis para box-shadow são:

none | <shadow> [ , <shadow> ]\*

<shadow> = inset? && [ <length>{2,4} && <color>? ]

## box-sizing

Essa propriedade define como serão computadas as dimensões do box.

Valor inicial	content-box
Aplica-se	a todos os elementos que aceitam width e height
Herdada	não
Referência para porcentagem	—
Mídia	visual
Criada na versão	CSS3

## Valores

Os valores possíveis para box-sizing são:

content-box | border-box | inherit

## top

Essa propriedade destina-se a definir a distância que separa a borda superior de um bloco de conteúdo da linha de margem superior de um box (bloco de conteúdo) nele posicionado de forma absoluta. Para boxes posicionados de forma absoluta, a distância é medida em relação à borda superior do próprio box, e não do seu bloco de conteúdo.

<b>Valor inicial</b>	auto
<b>Aplica-se</b>	a elementos posicionados com position com um valor que não static
<b>Herdada</b>	não
<b>Referência para porcentagem</b>	altura do bloco de conteúdo
<b>Mídia</b>	visual
<b>Criada na versão</b>	CSS2

## Valores

Os valores possíveis para top são:

<length> | <percentage> | auto | inherit

## right

Essa propriedade destina-se a definir a distância que separa a borda direita de um bloco de conteúdo da linha de margem direita de um box (bloco de conteúdo) nele posicionado de forma absoluta. Para boxes posicionados de forma absoluta, a distância é medida em relação à borda direita do próprio box, e não do seu bloco de conteúdo.

<b>Valor inicial</b>	auto
<b>Aplica-se</b>	a elementos posicionados com position com valor que não static
<b>Herdada</b>	não

<b>Referência para porcentagem</b>	largura do bloco de conteúdo
<b>Mídia</b>	visual
<b>Criada na versão</b>	CSS2

## Valores

Os valores possíveis para right são:

<length> | <percentage> | auto | inherit

## bottom

Essa propriedade destina-se a definir a distância que separa a borda inferior de um bloco de conteúdo da linha de margem inferior de um box (bloco de conteúdo) nele posicionado de forma absoluta. Para boxes posicionados de forma absoluta, a distância é medida em relação à borda inferior do próprio box, e não do seu bloco de conteúdo.

<b>Valor inicial</b>	auto
<b>Aplica-se</b>	a elementos posicionados com position com um valor que não static
<b>Herdada</b>	não
<b>Referência para porcentagem</b>	altura do bloco de conteúdo
<b>Mídia</b>	visual
<b>Criada na versão</b>	CSS2

## Valores

Os valores possíveis para bottom são:

<length> | <percentage> | auto | inherit

## left

Essa propriedade destina-se a definir a distância que separa a borda esquerda de um bloco de conteúdo da linha de margem esquerda de um box (bloco de conteúdo) nele posicionado de forma absoluta. Para boxes posicionados de forma absoluta, a distância é



medida em relação à borda esquerda do próprio box, e não do seu bloco de conteúdo.

<b>Valor inicial</b>	auto
<b>Aplica-se</b>	a elementos posicionados com position com um valor que não static
<b>Herdada</b>	não
<b>Referência para porcentagem</b>	largura do bloco de conteúdo
<b>Mídia</b>	visual
<b>Criada na versão</b>	CSS2

### **Valores**

Os valores possíveis para left são:

<length> | <percentage> | auto | inherit

### **caption-side**

Essa propriedade destina-se a posicionar o bloco de conteúdo da legenda descritiva de uma tabela.

<b>Valor inicial</b>	top
<b>Aplica-se</b>	a elementos “table-caption”
<b>Herdada</b>	sim
<b>Referência para porcentagem</b>	–
<b>Mídia</b>	visual
<b>Criada na versão</b>	CSS2

### **Valores**

Os valores possíveis para caption-side são:

top | bottom | inherit

### **clear**

Essa propriedade destina-se a definir qual dos lados de um elemento não será adjacente a um bloco anterior que tenha sido flutuado (com uso da propriedade float).

<b>Valor inicial</b>	none
----------------------	------

<b>Aplica-se</b>	a elementos nível de bloco
<b>Herdada</b>	não
<b>Referência para porcentagem</b>	–
<b>Mídia</b>	visual
<b>Criada na versão</b>	CSS1

## Valores

Os valores possíveis para clear são:

none | left | right | both | inherit

## clip

Essa propriedade aplica-se somente a elementos posicionados de forma absoluta e destina-se a definir uma área de recorte do elemento.

<b>Valor inicial</b>	auto
<b>Aplica-se</b>	a elementos posicionados absolutamente
<b>Herdada</b>	não
<b>Referência para porcentagem</b>	–
<b>Mídia</b>	visual
<b>Criada na versão</b>	CSS2

## Valores

Os valores possíveis para clip são:

<shape> | auto | inherit

O valor possível para <shape> é:

<shape> = rect(<top>, <right>, <bottom>, <left>)

## color

Essa propriedade destina-se a definir uma cor para o conteúdo do elemento.

<b>Valor inicial</b>	depende do agente de usuário
<b>Aplica-se</b>	a todos os elementos
<b>Herdada</b>	sim

<b>Referência para porcentagem</b>	–
<b>Mídia</b>	visual
<b>Criada na versão</b>	CSS1

## Valores

Os valores possíveis para color são:

<keyword> | <numeric-values>

## column-width

Essa propriedade destina-se a definir a largura de colunas e de elementos multicolunares.

<b>Valor inicial</b>	auto
<b>Aplica-se</b>	a elementos nível de bloco (exceto tabelas), células de tabelas e elementos inline-block
<b>Herdada</b>	não
<b>Referência para porcentagem</b>	–
<b>Mídia</b>	visual
<b>Criada na versão</b>	CSS3

## Valores

Os valores possíveis para column-width são:

<length> | auto

## column-count

Essa propriedade destina-se a definir o número de colunas de um elemento multicolunar.

<b>Valor inicial</b>	auto
<b>Aplica-se</b>	a elementos nível de bloco (exceto tabelas), células de tabelas e elementos inline-block
<b>Herdada</b>	não
<b>Referência para porcentagem</b>	–
<b>Mídia</b>	visual

<b>Criada na versão</b>	CSS3
-------------------------	------

## Valores

Os valores possíveis para column-width são:

<integer> | auto

## columns

Essa propriedade é a forma abreviada de declarar as propriedades column-width e column-count.

<b>Valor inicial</b>	ver propriedades individuais
<b>Aplica-se</b>	a elementos nível de bloco (exceto tabelas), células de tabelas e elementos inline-block
<b>Herdada</b>	não
<b>Referência para porcentagem</b>	—
<b>Mídia</b>	visual
<b>Criada na versão</b>	CSS3

## Valores

Os valores possíveis para column-width são:

<column-width> || <column-count>

## column-gap

Essa propriedade destina-se a definir o espaçamento entre colunas.

<b>Valor inicial</b>	normal
<b>Aplica-se</b>	a elementos multicolunares
<b>Herdada</b>	não
<b>Referência para porcentagem</b>	—
<b>Mídia</b>	visual
<b>Criada na versão</b>	CSS3

## Valores

Os valores possíveis para column-gap são:

<lenght> | normal

## column-rule-width

Essa propriedade destina-se a definir a espessura de uma linha posicionada entre colunas.

<b>Valor inicial</b>	medium
<b>Aplica-se</b>	a elementos multicolunares
<b>Herdada</b>	não
<b>Referência para porcentagem</b>	–
<b>Mídia</b>	visual
<b>Criada na versão</b>	CSS3

### Valores

Os valores possíveis para column-rule-width são:

<border-width>{1,4} = <lenght> | thin | medium | thick

## column-rule-style

Essa propriedade destina-se a definir o estilo de uma linha posicionada entre colunas.

<b>Valor inicial</b>	none
<b>Aplica-se</b>	a elementos multicolunares
<b>Herdada</b>	não
<b>Referência para porcentagem</b>	–
<b>Mídia</b>	visual
<b>Criada na versão</b>	CSS3

### Valores

Os valores possíveis para column-rule-style são:

<border-style>{1,4} = none | hidden | dotted | dashed | solid | double | groove | ridge | inset | outset

## column-rule-color

Essa propriedade destina-se a definir a cor de uma linha posicionada entre colunas.

<b>Valor inicial</b>	depende do agente de usuário

<b>Aplica-se</b>	a elementos multicolunares
<b>Herdada</b>	não
<b>Referência para porcentagem</b>	–
<b>Mídia</b>	visual
<b>Criada na versão</b>	CSS3

## Valores

Os valores possíveis para column-rule-color são:

<border-style>{1,4} = none | hidden | dotted | dashed | solid | double | groove | ridge | inset | outset

## column-rule

Essa propriedade é a forma abreviada de declarar as propriedades column-rule-width, column-rule-style, column-rule-color.

<b>Valor inicial</b>	ver propriedades individuais
<b>Aplica-se</b>	a elementos multicolunares
<b>Herdada</b>	não
<b>Referência para porcentagem</b>	–
<b>Mídia</b>	visual
<b>Criada na versão</b>	CSS3

## Valores

Os valores possíveis para column-rule são:

<column-rule-width> || <column-rule-style> || [ <column-rule-color> | transparent ]

## column-span

Essa propriedade destina-se a definir por quantas colunas um elemento se estende.

<b>Valor inicial</b>	none
<b>Aplica-se</b>	a elementos nível de bloco, exceto floats e elementos posicionados de forma absoluta
<b>Herdada</b>	não
<b>Referência para</b>	–

<b>porcentagem</b>	
<b>Mídia</b>	visual
<b>Criada na versão</b>	CSS3

## Valores

Os valores possíveis para column-span são:

none | all

## content

Essa propriedade destina-se a definir o conteúdo a ser renderizado dentro de um elemento ou pseudoelemento.

<b>Valor inicial</b>	normal
<b>Aplica-se</b>	a todos os elementos, ::before, ::after, ::alternate, ::marker, ::line-marker, área de margens, área de @footnotes
<b>Herdada</b>	não
<b>Referência para porcentagem</b>	–
<b>Mídia</b>	visual
<b>Criada na versão</b>	CSS2

## Valores

Os valores possíveis para content são:

[<uri> '']\* [normal | none | inhibit | <content-list>]

<content-list> = [ pending(<identifier>) | <string> | contents | footnote | endnote | section-note | list-item | <counters> | <strings> | open-quote | close-quote | no-open-quote | no-close-quote | <glyph> | <uri> | <date> | document-url | <target> ]+

## cursor

Essa propriedade destina-se a definir o aspecto do cursor do mouse.

<b>Valor inicial</b>	auto
<b>Aplica-se</b>	a todos os elementos

<b>Herdada</b>	sim
<b>Referência para porcentagem</b>	–
<b>Mídia</b>	visual
<b>Criada na versão</b>	CSS2

## Valores

Os valores possíveis para cursor são:

[[<uri> [<x> <y>]?,]\* [auto | default | none | context-menu | help | pointer | progress | wait | cell | crosshair | text | vertical-text | alias | copy | move | no-drop | not-allowed | e-resize | n-resize | ne-resize | nw-resize | s-resize | se-resize | sw-resize | w-resize | ew-resize | ns-resize | nesw-resize | nwse-resize | col-resize | row-resize | all-scroll ] ] | inherit

## direction

Essa propriedade destina-se a definir a direção da escrita.

<b>Valor inicial</b>	ltr
<b>Aplica-se</b>	a todos os elementos
<b>Herdada</b>	sim
<b>Referência para porcentagem</b>	–
<b>Mídia</b>	visual
<b>Criada na versão</b>	CSS2

## Valores

Os valores possíveis para direction são:

ltr | rtl | inherit

## display

Essa propriedade destina-se a definir o tipo de box gerado para um elemento.

<b>Valor inicial</b>	inline
<b>Aplica-se</b>	a todos os elementos
<b>Herdada</b>	não
<b>Referência para porcentagem</b>	–



<b>Mídia</b>	visual
<b>Criada na versão</b>	CSS2

## Valores

Os valores possíveis para display são:

inline | block | inline-block | list-item | run-in | compact | table | inline-table | table-row-group | table-header-group | table-footer-group | table-row | table-column-group | table-column | table-cell | table-caption | ruby | ruby-base | ruby-text | ruby-base-group | ruby-text-group | <template> | none

O valor <template> destina-se a definir valores, de acordo com o módulo template layout. Ver capítulo 16.

## empty-cells

Essa propriedade destina-se a definir o modo de renderização das bordas e do fundo de células vazias de tabela.

<b>Valor inicial</b>	show
<b>Aplica-se</b>	a células de tabelas
<b>Herdada</b>	sim
<b>Referência para porcentagem</b>	—
<b>Mídia</b>	visual
<b>Criada na versão</b>	CSS2

## Valores

Os valores possíveis para empty-cells são:

show | hide | inherit

## float

Essa propriedade destina-se a definir a posição de flutuação de um bloco.

<b>Valor inicial</b>	none
<b>Aplica-se</b>	a todos os elementos, exceto blocos posicionados de forma absoluta
<b>Herdada</b>	não

<b>Referência para porcentagem</b>	–
<b>Mídia</b>	visual
<b>Criada na versão</b>	CSS1

## Valores

Os valores possíveis para float são:

left | right | none | <page-floats>

O valor <page-float> destina-se a definir valores de acordo com as regras de geração de conteúdos para Paged Media.

## font-style

Essa propriedade destina-se a definir se a fonte será em itálico ou oblíqua.

<b>Valor inicial</b>	normal
<b>Aplica-se</b>	a todos os elementos
<b>Herdada</b>	sim
<b>Referência para porcentagem</b>	–
<b>Mídia</b>	visual
<b>Criada na versão</b>	CSS1

## Valores

Os valores possíveis para font-style são:

normal | italic | oblique | inherit

## font-variant

Essa propriedade destina-se a definir uma fonte renderizada como se estivesse em caixa-alta, mas com tamanho reduzido e diferentes proporções em relação àquelas em caixa-alta.

<b>Valor inicial</b>	normal
<b>Aplica-se</b>	a todos os elementos
<b>Herdada</b>	sim
<b>Referência para porcentagem</b>	–

<b>Mídia</b>	visual
<b>Criada na versão</b>	CSS1

## Valores

Os valores possíveis para font-variant nas CSS2.1 são:

normal | small-caps | inherit

As CSS3 criaram novos valores para essa propriedade. Tratamos desses valores no capítulo 8.

## font-weight

Essa propriedade destina-se a definir o peso das fontes, ou seja, a intensidade do negrito ou a espessura dos traços da fonte.

<b>Valor inicial</b>	normal
<b>Aplica-se</b>	a todos os elementos
<b>Herdada</b>	sim
<b>Referência para porcentagem</b>	–
<b>Mídia</b>	visual
<b>Criada na versão</b>	CSS1

## Valores

Os valores possíveis para font-weight são:

normal | bold | bolder | lighter | 100 | 200 | 300 | 400 | 500 | 600 | 700 | 800 | 900

## font-size

Essa propriedade destina-se a definir o tamanho da fonte.

<b>Valor inicial</b>	medium
<b>Aplica-se</b>	a todos os elementos
<b>Herdada</b>	sim
<b>Referência para porcentagem</b>	–
<b>Mídia</b>	visual
<b>Criada na versão</b>	CSS1

## Valores

Os valores possíveis para font-size são:

<absolute-size> | <relative-size> | <length> | <percentage> | inherit

## line-height

Essa propriedade destina-se a definir a distância entre linhas.

Valor inicial	normal
Aplica-se	a todos os elementos
Herdada	sim
Referência para porcentagem	–
Mídia	visual
Criada na versão	CSS1

### Valores

Os valores possíveis para line-height são:

normal | <number> | <length> | <percentage> | none

## font-family

Essa propriedade destina-se a definir as famílias de fontes.

Valor inicial	depende do agente de usuário
Aplica-se	a todos os elementos
Herdada	sim
Referência para porcentagem	–
Mídia	visual
Criada na versão	CSS1

### Valores

Os valores possíveis para font-family são:

[ [ <family-name> | <generic-family> ] [, <family-name> | <generic-family>]\* ] | inherit

## font

Essa propriedade é a forma abreviada de declarar as propriedades para fontes.

--	--

<b>Valor inicial</b>	ver propriedades individuais
<b>Aplica-se</b>	a todos os elementos
<b>Herdada</b>	sim
<b>Referência para porcentagem</b>	–
<b>Mídia</b>	visual
<b>Criada na versão</b>	CSS1

## Valores

Os valores possíveis para font são:

[ [ <'font-style'> || <'font-variant-css21'> || <'font-weight'> ]? <'font-size'> [ / <'line-height'> ]? <'font-family'> ] | caption | icon | menu | message-box | small-caption | status-bar | inherit

Os valores de font-variant, quando declarados com uso da forma abreviada, devem ser os previstos nas especificações para as CSS2.1, ou seja, os valores criados pelas CSS3 não podem ser declarados. Para isso, use a declaração individual, e não a abreviada.

## font-size-adjust

Essa propriedade destina-se a preservar a legibilidade da fonte alternativa definida com uso da propriedade font-family. Vimos essa propriedade com detalhes no capítulo 8.

<b>Valor inicial</b>	none
<b>Aplica-se</b>	a todos os elementos
<b>Herdada</b>	sim
<b>Referência para porcentagem</b>	–
<b>Mídia</b>	visual
<b>Criada na versão</b>	CSS2/CSS3 (não existe na versão 2.1)

## Valores

Os valores possíveis para font-size-adjust são:

<number> | none | inherit

## font-stretch

Essa propriedade destina-se a definir o quanto uma fonte será comprimida ou expandida.

<b>Valor inicial</b>	normal
<b>Aplica-se</b>	a todos os elementos
<b>Herdada</b>	sim
<b>Referência para porcentagem</b>	–
<b>Mídia</b>	visual
<b>Criada na versão</b>	CSS3

### Valores

Os valores possíveis para font-stretch são:

normal | ultra-condensed | extra-condensed | condensed | semi-condensed |  
semi-expanded | expanded | extra-expanded | ultra-expanded | inherit

## height

Essa propriedade destina-se a definir a altura de uma área do box, de acordo com o Box Model.

<b>Valor inicial</b>	auto
<b>Aplica-se</b>	a todos os elementos, exceto elementos inline “non-replaced”, colunas e grupos de colunas e tabelas
<b>Herdada</b>	não
<b>Referência para porcentagem</b>	largura do bloco de conteúdo
<b>Mídia</b>	visual
<b>Criada na versão</b>	CSS3

Elementos inline “non-replaced” são os elementos inline que não têm dimensões intrínsecas. Ao contrário, elementos inline “replaced” são aqueles que têm dimensões intrínsecas, tais como os elementos img e elementos para marcar controles de formulários.

### Valores

Os valores possíveis para height são:

<length> | <percentage> | auto

## letter-spacing

Essa propriedade destina-se a definir as distâncias mínima, máxima e ótima entre os caracteres de uma fonte.

<b>Valor inicial</b>	normal
<b>Aplica-se</b>	a todos os elementos
<b>Herdada</b>	sim
<b>Referência para porcentagem</b>	distância padrão entre os caracteres
<b>Mídia</b>	visual
<b>Criada na versão</b>	CSS1

### Valores

Os valores possíveis para letter-spacing são:

<spacing-limit>{1,3}

<spacing-limit> = normal | <length> | <percentage>

## list-style-type

Essa propriedade destina-se a definir o estilo do marcador dos itens de uma lista.

<b>Valor inicial</b>	disc
<b>Aplica-se</b>	a todos os elementos com display: list-item
<b>Herdada</b>	sim
<b>Referência para porcentagem</b>	–
<b>Mídia</b>	visual
<b>Criada na versão</b>	CSS1

### Valores

Os valores possíveis para list-style-type são:

<string> | <counter-style> | inline | none

## list-style-position

Essa propriedade destina-se a definir uma imagem para servir como

marcador dos itens de uma lista.

<b>Valor inicial</b>	outside
<b>Aplica-se</b>	a todos os elementos com display: list-item
<b>Herdada</b>	sim
<b>Referência para porcentagem</b>	–
<b>Mídia</b>	visual
<b>Criada na versão</b>	CSS1

### **Valores**

Os valores possíveis para list-style-position são:

inside | hanging | outside

### **list-style-image**

Essa propriedade destina-se a definir uma imagem para servir como marcador dos itens de uma lista.

<b>Valor inicial</b>	none
<b>Aplica-se</b>	a todos os elementos com display: list-item
<b>Herdada</b>	sim
<b>Referência para porcentagem</b>	–
<b>Mídia</b>	visual
<b>Criada na versão</b>	CSS1

### **Valores**

Os valores possíveis para list-style-image são:

<image> | none

<image> = URL endereço de uma imagem

### **list-style**

Essa propriedade é a forma abreviada de declarar as propriedades list-style-type, list-style-position ou list-style-image para marcador dos itens de uma lista.

<b>Valor inicial</b>	ver propriedades individuais
<b>Aplica-se</b>	a todos os elementos com display: list-item



<b>Herdada</b>	sim
<b>Referência para porcentagem</b>	–
<b>Mídia</b>	visual
<b>Criada na versão</b>	CSS1

## Valores

Os valores possíveis para list-style são:

<list-style-type> || <list-style-position> || <list-style-image>

## margin-top

Essa propriedade destina-se a definir a dimensão da margem superior de um elemento.

<b>Valor inicial</b>	0
<b>Aplica-se</b>	a todos os boxes, exceto alguns boxes table-* e alguns blocos inline
<b>Herdada</b>	não
<b>Referência para porcentagem</b>	largura do bloco de conteúdo horizontal ou altura do bloco de conteúdo vertical
<b>Mídia</b>	visual
<b>Criada na versão</b>	CSS1

## Valores

Os valores possíveis para margin-top são:

<length> | <percentage> | auto

## margin-right

Essa propriedade destina-se a definir a dimensão da margem direita de um elemento.

<b>Valor inicial</b>	0
<b>Aplica-se</b>	a todos os boxes, exceto alguns boxes table-* e alguns blocos inline
<b>Herdada</b>	não
<b>Referência para porcentagem</b>	largura do bloco de conteúdo horizontal ou altura do bloco de conteúdo vertical
<b>Mídia</b>	visual

<b>Criada na versão</b>	CSS1
-------------------------	------

## Valores

Os valores possíveis para margin-right são:

<length> | <percentage> | auto

## margin-bottom

Essa propriedade destina-se a definir a dimensão da margem inferior de um elemento.

<b>Valor inicial</b>	0
<b>Aplica-se</b>	a todos os boxes, exceto alguns boxes table-* e alguns blocos inline
<b>Herdada</b>	não
<b>Referência para porcentagem</b>	largura do bloco de conteúdo horizontal ou altura do bloco de conteúdo vertical
<b>Mídia</b>	visual
<b>Criada na versão</b>	CSS1

## Valores

Os valores possíveis para margin-bottom são:

<length> | <percentage> | auto

## margin-left

Essa propriedade destina-se a definir a dimensão da margem esquerda de um elemento.

<b>Valor inicial</b>	0
<b>Aplica-se</b>	a todos os boxes, exceto alguns boxes table-* e alguns blocos inline
<b>Herdada</b>	não
<b>Referência para porcentagem</b>	largura do bloco de conteúdo horizontal ou altura do bloco de conteúdo vertical
<b>Mídia</b>	visual
<b>Criada na versão</b>	CSS1

## Valores

Os valores possíveis para margin-left são:

<length> | <percentage> | auto

## margin

Essa propriedade é a forma abreviada de definir as dimensões das quatro margens de um elemento.

<b>Valor inicial</b>	ver propriedades individuais
<b>Aplica-se</b>	a todos os boxes, exceto alguns boxes table-* e alguns blocos inline
<b>Herdada</b>	não
<b>Referência para porcentagem</b>	largura do bloco de conteúdo horizontal ou altura do bloco de conteúdo vertical
<b>Mídia</b>	visual
<b>Criada na versão</b>	CSS1

## Valores

Os valores possíveis para margin são:

[<length> | <percentage> | auto]{1,4}

## max-width

Essa propriedade destina-se a definir a largura máxima de um box.

<b>Valor inicial</b>	none
<b>Aplica-se</b>	a todos os boxes, exceto elementos inline “non-replaced”, linhas e grupos de linhas de tabelas
<b>Herdada</b>	não
<b>Referência para porcentagem</b>	largura do bloco de conteúdo
<b>Mídia</b>	visual
<b>Criada na versão</b>	CSS2

## Valores

Os valores possíveis para max-width são:

<length> | <percentage> | none

## min-width

Essa propriedade destina-se a definir a largura mínima de um box.

<b>Valor inicial</b>	none
<b>Aplica-se</b>	a todos os boxes, exceto elementos inline “non-replaced”, linhas e grupos de linhas de tabelas
<b>Herdada</b>	não
<b>Referência para porcentagem</b>	largura do bloco de conteúdo
<b>Mídia</b>	visual
<b>Criada na versão</b>	CSS2

### Valores

Os valores possíveis para min-width são:

<length> | <percentage> | none

## max-height

Essa propriedade destina-se a definir a altura máxima de um box.

<b>Valor inicial</b>	none
<b>Aplica-se</b>	a todos os boxes, exceto elementos inline “non-replaced”, linhas e grupos de linhas de tabelas
<b>Herdada</b>	não
<b>Referência para porcentagem</b>	altura do bloco de conteúdo
<b>Mídia</b>	visual
<b>Criada na versão</b>	CSS2

### Valores

Os valores possíveis para max-height são:

<length> | <percentage> | none

## min-height

Essa propriedade destina-se a definir a altura mínima de um box.

<b>Valor inicial</b>	none
<b>Aplica-se</b>	a todos os boxes, exceto elementos inline “non-replaced”,

	linhas e grupos de linhas de tabelas
<b>Herdada</b>	não
<b>Referência para porcentagem</b>	altura do bloco de conteúdo
<b>Mídia</b>	visual
<b>Criada na versão</b>	CSS2

## Valores

Os valores possíveis para min-height são:

<length> | <percentage> | none

## opacity

Essa propriedade destina-se a definir o grau de opacidade de um elemento.

<b>Valor inicial</b>	1
<b>Aplica-se</b>	a todos os elementos
<b>Herdada</b>	não
<b>Referência para porcentagem</b>	–
<b>Mídia</b>	visual
<b>Criada na versão</b>	CSS3

## Valores

Os valores possíveis para opacity são:

<alphavalue> | inherit

<alphavalue> = varia de 0 (transparente) até 1 (opaco)

## outline-color

Essa propriedade destina-se a definir a cor de uma linha de contorno para dar destaque a um elemento.

<b>Valor inicial</b>	invert
<b>Aplica-se</b>	a todos os elementos
<b>Herdada</b>	não
<b>Referência para porcentagem</b>	–

<b>Mídia</b>	visual
<b>Criada na versão</b>	CSS2

## Valores

Os valores possíveis para outline-color são:

<color> | invert | inherit

A palavra-chave invert garante a inversão de cor em relação à cor de fundo, garantindo a visibilidade de outline, qualquer que seja a cor do fundo.

## outline-style

Essa propriedade destina-se a definir o estilo de uma linha de contorno para dar destaque a um elemento.

<b>Valor inicial</b>	none
<b>Aplica-se</b>	a todos os elementos
<b>Herdada</b>	não
<b>Referência para porcentagem</b>	–
<b>Mídia</b>	visual
<b>Criada na versão</b>	CSS2

## Valores

Os valores possíveis para outline-style são:

<border-style> | inherit

<border-style> = none | hidden | dotted | dashed | solid | double | groove | ridge | inset | outset

## outline-width

Essa propriedade destina-se a definir a espessura de uma linha de contorno para dar destaque a um elemento.

<b>Valor inicial</b>	medium
<b>Aplica-se</b>	a todos os elementos
<b>Herdada</b>	não
<b>Referência para porcentagem</b>	–

<b>Mídia</b>	visual
<b>Criada na versão</b>	CSS2

## Valores

Os valores possíveis para outline-width são:

<border-width> | inherit

<border-width> = <lenght> | thin | medium | thick

## outline

Essa propriedade é a forma abreviada de declarar as propriedades outline-color, outline-style, outline-width.

<b>Valor inicial</b>	ver propriedades individuais
<b>Aplica-se</b>	a todos os elementos
<b>Herdada</b>	não
<b>Referência para porcentagem</b>	–
<b>Mídia</b>	visual
<b>Criada na versão</b>	CSS2

## Valores

Os valores possíveis para outline são:

[<outline-color> || <outline-style> || <outline-width>] | inherit

## overflow-x

Essa propriedade destina-se a definir como será apresentado o conteúdo que ultrapassa a largura da área de conteúdo do elemento.

<b>Valor inicial</b>	visible
<b>Aplica-se</b>	a elementos nível de bloco e inline do tipo “non-replaced”
<b>Herdada</b>	não
<b>Referência para porcentagem</b>	–
<b>Mídia</b>	visual
<b>Criada na versão</b>	CSS3

## Valores

Os valores possíveis para overflow-x são:

visible | hidden | scroll | auto | no-display | no-content

## overflow-y

Essa propriedade destina-se a definir como será apresentado o conteúdo que ultrapassa a altura da área de conteúdo do elemento.

<b>Valor inicial</b>	visible
<b>Aplica-se</b>	a elementos nível de bloco e inline do tipo “non-replaced”
<b>Herdada</b>	não
<b>Referência para porcentagem</b>	—
<b>Mídia</b>	visual
<b>Criada na versão</b>	CSS3

## Valores

Os valores possíveis para overflow-y são:

visible | hidden | scroll | auto | no-display | no-content

## overflow

Essa propriedade é a forma abreviada de definir como será apresentado o conteúdo que ultrapassa a largura e a altura da área de conteúdo do elemento.

<b>Valor inicial</b>	visible
<b>Aplica-se</b>	a elementos nível de bloco e inline do tipo “non-replaced”
<b>Herdada</b>	não
<b>Referência para porcentagem</b>	—
<b>Mídia</b>	visual
<b>Criada na versão</b>	CSS2

## Valores



Os valores possíveis para overflow são:

[visible | hidden | scroll | auto | no-display | no-content ]{1,2}

## padding-top

Essa propriedade destina-se a definir a dimensão da área de enchimento superior de um elemento.

<b>Valor inicial</b>	0
<b>Aplica-se</b>	a todos os elementos
<b>Herdada</b>	não
<b>Referência para porcentagem</b>	largura do bloco de conteúdo horizontal ou altura do bloco de conteúdo vertical
<b>Mídia</b>	visual
<b>Criada na versão</b>	CSS1

### Valores

Os valores possíveis para padding-top são:

<length> | <percentage>

## padding-right

Essa propriedade destina-se a definir a dimensão da área de enchimento direita de um elemento.

<b>Valor inicial</b>	0
<b>Aplica-se</b>	a todos os elementos
<b>Herdada</b>	não
<b>Referência para porcentagem</b>	largura do bloco de conteúdo horizontal ou altura do bloco de conteúdo vertical
<b>Mídia</b>	visual
<b>Criada na versão</b>	CSS1

### Valores

Os valores possíveis para padding-right são:

<length> | <percentage>

## padding-bottom

Essa propriedade destina-se a definir a dimensão da área de enchimento inferior de um elemento.

<b>Valor inicial</b>	0
<b>Aplica-se</b>	a todos os elementos
<b>Herdada</b>	não
<b>Referência para porcentagem</b>	largura do bloco de conteúdo horizontal ou altura do bloco de conteúdo vertical
<b>Mídia</b>	visual
<b>Criada na versão</b>	CSS1

### **Valores**

Os valores possíveis para padding-bottom são:

<length> | <percentage>

### **padding-left**

Essa propriedade destina-se a definir a dimensão da área de enchimento esquerda de um elemento.

<b>Valor inicial</b>	0
<b>Aplica-se</b>	a todos os elementos
<b>Herdada</b>	não
<b>Referência para porcentagem</b>	largura do bloco de conteúdo horizontal ou altura do bloco de conteúdo vertical
<b>Mídia</b>	visual
<b>Criada na versão</b>	CSS1

### **Valores**

Os valores possíveis para padding-left são:

<length> | <percentage>

### **padding**

Essa propriedade é a forma abreviada de definir as dimensões dos quatro enchimentos de um elemento.

<b>Valor inicial</b>	ver propriedades individuais
<b>Aplica-se</b>	a todos os elementos

<b>Herdada</b>	não
<b>Referência para porcentagem</b>	largura do bloco de conteúdo horizontal ou altura do bloco de conteúdo vertical
<b>Mídia</b>	visual
<b>Criada na versão</b>	CSS1

## Valores

Os valores possíveis para padding são:

[<length> | <percentage>]{1,4}

## perspective() – função de transformação

Essa propriedade define uma matriz para projeção da perspectiva 3D.

<b>Valor inicial</b>	1
<b>Aplica-se</b>	a todos os elementos
<b>Herdada</b>	não
<b>Referência para porcentagem</b>	–
<b>Mídia</b>	visual
<b>Criada na versão</b>	CSS3

## Valores

O valor possível para perspective é:

<número>

## perspective-origin

Essa propriedade define o ponto de origem para a perspectiva dos efeitos de transformações 3D.

<b>Valor inicial</b>	50% 50%
<b>Aplica-se</b>	a elementos nível de bloco e inline
<b>Herdada</b>	não
<b>Referência para porcentagem</b>	dimensões do box
<b>Mídia</b>	visual
<b>Criada na versão</b>	CSS3

## Valores

Os valores possíveis para transform-origin são:

[ [ <percentage> | <length> | left | center | right ] [ <percentage> | <length> | top | center | bottom ]? ] | [ [ left | center | right ] || [ top | center | bottom ] ]

## position

Essa propriedade define como um elemento será posicionado.

Valor inicial	static
Aplica-se	a todos os elementos
Herdada	não
Referência para porcentagem	–
Mídia	visual
Criada na versão	CSS2

## Valores

Os valores possíveis para position são:

static | relative | absolute | fixed | inherit

## rotate(a) – função de transformação

Função para aplicar uma rotação 2D em torno de um ponto.

Valor inicial	0
Aplica-se	a todos os elementos
Herdada	não
Referência para porcentagem	–
Mídia	visual
Criada na versão	CSS3

## Valores

O valor possível para rotate é:

<medida de ângulo CSS>

## rotateX(x) – função de transformação

Função para aplicar rotação 3D em torno do eixo x.

--	--

<b>Valor inicial</b>	0
<b>Aplica-se</b>	a todos os elementos
<b>Herdada</b>	não
<b>Referência para porcentagem</b>	–
<b>Mídia</b>	visual
<b>Criada na versão</b>	CSS3

## Valores

O valor possível para rotateX é:

<medida de ângulo CSS>

## rotateY(a) – função de transformação

Função para aplicar rotação 3D em torno do eixo y.

<b>Valor inicial</b>	0
<b>Aplica-se</b>	a todos os elementos
<b>Herdada</b>	não
<b>Referência para porcentagem</b>	–
<b>Mídia</b>	visual
<b>Criada na versão</b>	CSS3

## Valores

O valor possível para rotateY é:

<medida de ângulo CSS>

## rotateZ(a) – função de transformação

Função para aplicar rotação 3D em torno do eixo z.

<b>Valor inicial</b>	0
<b>Aplica-se</b>	a todos os elementos
<b>Herdada</b>	não
<b>Referência para porcentagem</b>	–
<b>Mídia</b>	visual
<b>Criada na versão</b>	CSS3

## Valores

O valor possível para rotateZ é:

<medida de ângulo CSS>

## **scale(x, y) – função de transformação**

Função para aplicar escala 2D.

<b>Valor inicial</b>	1, 1
<b>Aplica-se</b>	a todos os elementos
<b>Herdada</b>	não
<b>Referência para porcentagem</b>	–
<b>Mídia</b>	visual
<b>Criada na versão</b>	CSS3

### **Valores**

Os valores possíveis para scale são:

<número> [, <número>]

## **scaleX(x) – função de transformação**

Função para aplicar escala na dimensão horizontal 2D.

<b>Valor inicial</b>	1, 1
<b>Aplica-se</b>	a todos os elementos
<b>Herdada</b>	não
<b>Referência para porcentagem</b>	–
<b>Mídia</b>	visual
<b>Criada na versão</b>	CSS3

### **Valores**

O valor possíveis para scaleX é:

<número>

## **scaleY(y) – função de transformação**

Função para aplicar escala na dimensão vertical 2D.

<b>Valor inicial</b>	1, 1
<b>Aplica-se</b>	a todos os elementos

<b>Herdada</b>	não
<b>Referência para porcentagem</b>	–
<b>Mídia</b>	visual
<b>Criada na versão</b>	CSS3

### **Valores**

O valor possível para scaleY é:

<número>

### **skew(a, b) – função de transformação**

Função para aplicar transformação ao longo dos eixos x e y.

<b>Valor inicial</b>	0
<b>Aplica-se</b>	a todos os elementos
<b>Herdada</b>	não
<b>Referência para porcentagem</b>	–
<b>Mídia</b>	visual
<b>Criada na versão</b>	CSS3

### **Valores**

Os valores possíveis para skew são:

<ângulo> [, <ângulo>]

### **skewX(a) – função de transformação**

Função para aplicar transformação ao longo do eixo x.

<b>Valor inicial</b>	0
<b>Aplica-se</b>	a todos os elementos
<b>Herdada</b>	não
<b>Referência para porcentagem</b>	–
<b>Mídia</b>	visual
<b>Criada na versão</b>	CSS3

### **Valores**

O valor possível para skewX é:

<ângulo>

## **skewY(b) – função de transformação**

Função para aplicar transformação ao longo do eixo y.

<b>Valor inicial</b>	0
<b>Aplica-se</b>	a todos os elementos
<b>Herdada</b>	não
<b>Referência para porcentagem</b>	–
<b>Mídia</b>	visual
<b>Criada na versão</b>	CSS3

### **Valores**

O valor possível para skewY é:

<ângulo>

## **table-layout**

Essa propriedade define como será o layout de células, linhas e colunas de tabelas.

<b>Valor inicial</b>	auto
<b>Aplica-se</b>	a elementos table e inline table
<b>Herdada</b>	não
<b>Referência para porcentagem</b>	–
<b>Mídia</b>	visual
<b>Criada na versão</b>	CSS2

### **Valores**

Os valores possíveis para table-layout são:

auto | fixed | inherit

## **text-align**

Essa propriedade define como será o alinhamento horizontal de conteúdos inline.

<b>Valor inicial</b>	start
<b>Aplica-se</b>	a blocos de conteúdos
<b>Herdada</b>	sim



<b>Referência para porcentagem</b>	–
<b>Mídia</b>	visual
<b>Criada na versão</b>	CSS1

## Valores

Os valores possíveis para text-align são:

[ start | <string> ]? [ start | end | left | right | center | justify | match-parent ]

## text-decoration

Essa propriedade define como será o elemento decorativo a ser anexado ao texto.

<b>Valor inicial</b>	none
<b>Aplica-se</b>	a todos os elementos
<b>Herdada</b>	não
<b>Referência para porcentagem</b>	–
<b>Mídia</b>	visual
<b>Criada na versão</b>	CSS1

## Valores

Os valores possíveis para text-decoration são:

none | [ underline || overline || line-through || blink ] | inherit

## text-indent

Essa propriedade define a endentação das linhas de um texto.

<b>Valor inicial</b>	0
<b>Aplica-se</b>	a blocos de conteúdos
<b>Herdada</b>	sim
<b>Referência para porcentagem</b>	largura do bloco de conteúdo
<b>Mídia</b>	visual
<b>Criada na versão</b>	CSS1

## Valores

Os valores possíveis para text-indent são:

[ <length> | <percentage> ] && [ hanging || each-line ]?

## text-shadow

Essa propriedade define os efeitos de sombreado para um texto.

Valor inicial	0
Aplica-se	a todos os elementos
Herdada	sim
Referência para porcentagem	–
Mídia	visual
Criada na versão	CSS3

### Valores

Os valores possíveis para text-shadow são:

none | [ <shadow>, ] \* <shadow>

<shadow> = inset? && [ <length>{2,4} && <color>? ]

## text-transform

Essa propriedade define os efeitos de capitalização para um texto.

Valor inicial	none
Aplica-se	a todos os elementos
Herdada	sim
Referência para porcentagem	–
Mídia	visual
Criada na versão	CSS1

### Valores

Os valores possíveis para text-transform são:

capitalize | uppercase | lowercase | none | inherit

## transform

Essa propriedade define efeitos de transformações 2D e 3D para um box.

Valor inicial	none

<b>Aplica-se</b>	a elementos nível de bloco e inline
<b>Herdada</b>	não
<b>Referência para porcentagem</b>	dimensões do box
<b>Mídia</b>	visual
<b>Criada na versão</b>	CSS3

## Valores

Os valores possíveis para transform são:

none | <função de transformação> [<função de transformação> ]\*

< função de transformação> = matrix() | matrix3d() | translate() | translate3d() | translateX() | translateY() | translateZ() | scale() | scale3d() | scaleX() | scaleY() | scaleZ() | rotate() | rotate3d() | rotateX() | rotateY() | rotateZ() | skew() | skewX() | skewY() | perspective

## transform-origin

Essa propriedade define o ponto de origem para aplicação de efeitos de transformações 2D e 3D.

<b>Valor inicial</b>	50% 50% 0
<b>Aplica-se</b>	a elementos nível de bloco e inline
<b>Herdada</b>	não
<b>Referência para porcentagem</b>	dimensões do box
<b>Mídia</b>	visual
<b>Criada na versão</b>	CSS3

## Valores

Os valores possíveis para transform-origin são:

[ [ [ <percentage> | <length> | left | center | right ] [ <percentage> | <length> | top | center | bottom ]? ] <length> ] | [ [ [ left | center | right ] || [ top | center | bottom ] ] <length>

## transform-style

Essa propriedade define o estilo das transformações 2D e 3D.

<b>Valor inicial</b>	flat

<b>Aplica-se</b>	a elementos nível de bloco e inline
<b>Herdada</b>	não
<b>Referência para porcentagem</b>	–
<b>Mídia</b>	visual
<b>Criada na versão</b>	CSS3

## Valores

Os valores possíveis para transform-style são:

flat | preserve 3d

## transition-property

Essa propriedade define a propriedade CSS à qual será aplicada uma transição.

<b>Valor inicial</b>	todas
<b>Aplica-se</b>	a todos os elementos e aos pseudoelementos :before e :after
<b>Herdada</b>	não
<b>Referência para porcentagem</b>	–
<b>Mídia</b>	visual
<b>Criada na versão</b>	CSS3

## Valores

Os valores possíveis para transition-property são:

none | all | [ <IDENT> ] [ ',' <IDENT> ]\*

<IDENT> = uma propriedade CSS

## transition-duration

Essa propriedade define o tempo de duração de uma transição.

<b>Valor inicial</b>	0
<b>Aplica-se</b>	a todos os elementos e aos pseudoelementos :before e :after
<b>Herdada</b>	não
<b>Referência para</b>	–

<b>porcentagem</b>	
<b>Mídia</b>	interativa
<b>Criada na versão</b>	CSS3

## Valores

O valor possível para transition-duration é:

<time> [, <time>]\*

<time> = número de segundos

## transition-timing-function

Essa propriedade define como serão calculados os valores intermediários de uma transição.

<b>Valor inicial</b>	ease
<b>Aplica-se</b>	a todos os elementos e aos pseudoelementos :before e :after
<b>Herdada</b>	não
<b>Referência para porcentagem</b>	—
<b>Mídia</b>	interativa
<b>Criada na versão</b>	CSS3

## Valores

Os valores possíveis para transition-timing-function são:

ease | linear | ease-in | ease-out | ease-in-out | cubic-bezier(<number>, <number>, <number>, <number>) [, ease | linear | ease-in | ease-out | ease-in-out | cubic-bezier(<number>, <number>, <number>, <number>)]\*

## transition-delay

Essa propriedade define o tempo de retardo para início de uma transição.

<b>Valor inicial</b>	0
<b>Aplica-se</b>	a todos os elementos e aos pseudoelementos :before e :after
<b>Herdada</b>	não

<b>Referência para porcentagem</b>	–
<b>Mídia</b>	interativa
<b>Criada na versão</b>	CSS3

### **Valores**

O valor possível para transition-delay é:

<time> [, <time>]\*

<time> = número de segundos

### **transition**

Essa propriedade é a forma abreviada de definição de uma transição.

<b>Valor inicial</b>	0
<b>Aplica-se</b>	a todos os elementos e aos pseudoelementos :before e :after
<b>Herdada</b>	não
<b>Referência para porcentagem</b>	–
<b>Mídia</b>	interativa
<b>Criada na versão</b>	CSS3

### **Valores**

Os valores possíveis para transition são:

[<'transition-property'> || <'transition-duration'> || <'transition-timing-function'> || <'transition-delay'> [, [<'transition-property'> || <'transition-duration'> || <'transition-timing-function'> || <'transition-delay'>]]\*

<time> = número de segundos

### **translate(x, y) – função de transformação**

Função para aplicar uma translação 2D.

<b>Valor inicial</b>	0, 0
<b>Aplica-se</b>	a todos os elementos
<b>Herdada</b>	não

<b>Referência para porcentagem</b>	–
<b>Mídia</b>	visual
<b>Criada na versão</b>	CSS3

### **Valores**

O valor possível para translate é:

<medida CSS> [, <medida CSS>]

### **translateX(x) – função de transformação**

Função para aplicar uma translação horizontal 2D.

<b>Valor inicial</b>	0, 0
<b>Aplica-se</b>	a todos os elementos
<b>Herdada</b>	não
<b>Referência para porcentagem</b>	–
<b>Mídia</b>	visual
<b>Criada na versão</b>	CSS3

### **Valores**

O valor possível para translateX é:

<medida CSS>

### **translateY(y) – função de transformação**

Função para aplicar uma translação vertical 2D.

<b>Valor inicial</b>	0, 0
<b>Aplica-se</b>	a todos os elementos
<b>Herdada</b>	não
<b>Referência para porcentagem</b>	–
<b>Mídia</b>	visual
<b>Criada na versão</b>	CSS3

### **Valores**

O valor possível para translateY é:

<medida CSS>

## vertical-align

Essa propriedade define o alinhamento vertical para elementos inline inseridos em blocos de conteúdos inline.

<b>Valor inicial</b>	não definido
<b>Aplica-se</b>	a elementos inline e table-cell
<b>Herdada</b>	não
<b>Referência para porcentagem</b>	line-height do elemento
<b>Mídia</b>	visual
<b>Criada na versão</b>	CSS1

### Valores

Os valores possíveis para vertical-align são:

auto | use-script | baseline | sub | super | top | text-top | central | middle | bottom | text-bottom | <percentage> | <length>

## white-space

Essa propriedade define o comportamento para espaços em branco nos textos.

<b>Valor inicial</b>	não definido
<b>Aplica-se</b>	a todos os elementos
<b>Herdada</b>	sim
<b>Referência para porcentagem</b>	–
<b>Mídia</b>	visual
<b>Criada na versão</b>	CSS1

### Valores

Os valores possíveis para vertical-align são:

normal | pre | nowrap | pre-wrap | pre-line

## width

Essa propriedade destina-se a definir a largura de uma área do box, de acordo com o Box Model.

<b>Valor inicial</b>	auto



<b>Aplica-se</b>	a todos os elementos, exceto elementos inline “non-replaced”, colunas e grupos de colunas e tabelas
<b>Herdada</b>	largura do bloco de conteúdo
<b>Referência para porcentagem</b>	—
<b>Mídia</b>	visual
<b>Criada na versão</b>	CSS1

### **Valores**

Os valores possíveis para height são:

<length> | <percentage> | auto

### **word-spacing**

Essa propriedade destina-se a definir as distâncias mínima, máxima e ótima entre as palavras de um texto.

<b>Valor inicial</b>	normal
<b>Aplica-se</b>	a todos os elementos
<b>Herdada</b>	sim
<b>Referência para porcentagem</b>	distância padrão entre as palavras
<b>Mídia</b>	visual
<b>Criada na versão</b>	CSS1

### **Valores**

Os valores possíveis para letter-spacing são:

<spacing-limit>{1,3}

<spacing-limit> = normal | <length> | <percentage>

# APÊNDICE C

## Cores CSS

Neste apêndice, apresentamos uma tabela das cores CSS3 que podem ser definidas com uso de palavras-chave e seus respectivos códigos hexadecimal e RGB.

COR – Palavra-chave	HEX	RGB
aliceblue	#f0f8ff	240,248,255
antiquewhite	#faebd7	250,235,215
aqua	#00ffff	0,255,255
aquamarine	#7fffd4	127,255,212
azure	#f0ffff	240,255,255
beige	#f5f5dc	245,245,220
bisque	#ffe4c4	255,228,196
black	#000000	0,0,0
blanchedalmond	#ffebcd	255,235,205
blue	#0000ff	0,0,255
blueviolet	#8a2be2	138,43,226
brown	#a52a2a	165,42,42
burlywood	#deb887	222,184,135
cadetblue	#5f9ea0	95,158,160
chartreuse	#7fff00	127,255,0
chocolate	#d2691e	210,105,30
coral	#ff7f50	255,127,80
cornflowerblue	#6495ed	100,149,237
cornsilk	#fff8dc	255,248,220
crimson	#dc143c	220,20,60
cyan	#00ffff	0,255,255

<b>COR – Palavra-chave</b>	<b>HEX</b>	<b>RGB</b>
darkblue	#00008b	0,0,139
darkcyan	#008b8b	0,139,139
darkgoldenrod	#b8860b	184,134,11
darkgray	#a9a9a9	169,169,169
darkgreen	#006400	0,100,0
darkgrey	#a9a9a9	169,169,169
darkkhaki	#bdb76b	189,183,107
darkmagenta	#8b008b	139,0,139
darkolivegreen	#556b2f	85,107,47
darkorange	#ff8c00	255,140,0
darkorchid	#9932cc	153,50,204
darkred	#8b0000	139,0,0
darksalmon	#e9967a	233,150,122
darkseagreen	#8fbc8f	143,188,143
darkslateblue	#483d8b	72,61,139
darkslategray	#2f4f4f	47,79,79
darkslategrey	#2f4f4f	47,79,79
darkturquoise	#00ced1	0,206,209
darkviolet	#9400d3	148,0,211
deeppink	#ff1493	255,20,147
deepskyblue	#00bfff	0,191,255
dimgray	#696969	105,105,105
dimgrey	#696969	105,105,105
dodgerblue	#1e90ff	30,144,255
firebrick	#b22222	178,34,34
floralwhite	#fffaf0	255,250,240
forestgreen	#228b22	34,139,34
fuchsia	#ff00ff	255,0,255
gainsboro	#dcdcdc	220,220,220
ghostwhite	#f8f8ff	248,248,255
gold	#ffd700	255,215,0
goldenrod	#daa520	218,165,32

<b>COR – Palavra-chave</b>	<b>HEX</b>	<b>RGB</b>
gray	#808080	128,128,128
green	#008000	0,128,0
greenyellow	#adff2f	173,255,47
grey	#808080	128,128,128
honeydew	#f0fff0	240,255,240
hotpink	#ff69b4	255,105,180
indianred	#cd5c5c	205,92,92
indigo	#4b0082	75,0,130
ivory	#fffff0	255,255,240
khaki	#f0e68c	240,230,140
lavender	#e6e6fa	230,230,250
lavenderblush	#fff0f5	255,240,245
lawngreen	#7cfc00	124,252,0
lemonchiffon	#fffacd	255,250,205
lightblue	#add8e6	173,216,230
lightcoral	#f08080	240,128,128
lightcyan	#e0ffff	224,255,255
lightgoldenrodyellow	#fafad2	250,250,210
lightgray	#d3d3d3	211,211,211
lightgreen	#90ee90	144,238,144
lightgrey	#d3d3d3	211,211,211
lightpink	#ffb6c1	255,182,193
lightsalmon	#ffa07a	255,160,122
lightseagreen	#20b2aa	32,178,170
lightskyblue	#87cefa	135,206,250
lightslategray	#778899	119,136,153
lightslategrey	#778899	119,136,153
lightsteelblue	#b0c4de	176,196,222
lightyellow	#ffffe0	255,255,224
lime	#00ff00	0,255,0
limegreen	#32cd32	50,205,50
linen	#faf0e6	250,240,230

<b>COR – Palavra-chave</b>	<b>HEX</b>	<b>RGB</b>
magenta	#ff00ff	255,0,255
maroon	#800000	128,0,0
mediumaquamarine	#66cdaa	102,205,170
mediumblue	#0000cd	0,0,205
mediumorchid	#ba55d3	186,85,211
mediumpurple	#9370db	147,112,219
mediumseagreen	#3cb371	60,179,113
mediumslateblue	#7b68ee	123,104,238
mediumspringgreen	#00fa9a	0,250,154
mediumturquoise	#48d1cc	72,209,204
mediumvioletred	#c71585	199,21,133
midnightblue	#191970	25,25,112
mintcream	#f5fffa	245,255,250
mistyrose	#ffe4e1	255,228,225
moccasin	#ffe4b5	255,228,181
navajowhite	#ffdead	255,222,173
navy	#000080	0,0,128
oldlace	#fdf5e6	253,245,230
olive	#808000	128,128,0
olivedrab	#6b8e23	107,142,35
orange	#ffa500	255,165,0
orangered	#ff4500	255,69,0
orchid	#da70d6	218,112,214
palegoldenrod	#eee8aa	238,232,170
palegreen	#98fb98	152,251,152
paleturquoise	#afeeee	175,238,238
palevioletred	#db7093	219,112,147
papayawhip	#ffefd5	255,239,213
peachpuff	#ffdab9	255,218,185
peru	#cd853f	205,133,63
pink	#ffc0cb	255,192,203
plum	#dda0dd	221,160,221

<b>COR – Palavra-chave</b>	<b>HEX</b>	<b>RGB</b>
powderblue	#b0e0e6	176,224,230
purple	#800080	128,0,128
red	#ff0000	255,0,0
rosybrown	#bc8f8f	188,143,143
royalblue	#4169e1	65,105,225
saddlebrown	#8b4513	139,69,19
salmon	#fa8072	250,128,114
sandybrown	#f4a460	244,164,96
seagreen	#2e8b57	46,139,87
seashell	#fff5ee	255,245,238
sienna	#a0522d	160,82,45
silver	#c0c0c0	192,192,192
skyblue	#87ceeb	135,206,235
slateblue	#6a5acd	106,90,205
slategray	#708090	112,128,144
slategrey	#708090	112,128,144
snow	#ffaafa	255,250,250
springgreen	#00ff7f	0,255,127
steelblue	#4682b4	70,130,180
tan	#d2b48c	210,180,140
teal	#008080	0,128,128
thistle	#d8bfd8	216,191,216
tomato	#ff6347	255,99,71
turquoise	#40e0d0	64,224,208
violet	#ee82ee	238,130,238
wheat	#f5deb3	245,222,179
white	#ffffff	255,255,255
whitesmoke	#f5f5f5	245,245,245
yellow	#ffff00	255,255,0
yellowgreen	#9acd32	154,205,50

# Referências

GASSTON, Peter. *The Book of CSS3: A developer guide to the future of web design*. 1. ed. San Francisco: No Starch Press, 2011.

KEITH, Jeremy. *HTML5 for Web Designers*. New York: Jeffrey Zeldman, 2010.

LAWSON, Bruce; SHARP, Remy. *Introducing HTML5*. 2. ed. Berkeley: New Riders – Voices That Matter, 2011.

Mozilla Developer Network. Disponível em:  
<<https://developer.mozilla.org/en-US/>>.

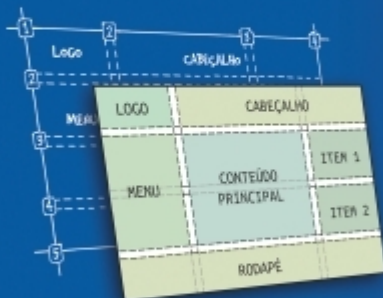
PILGRIM, Mark. *HTML5: Up and Running*. Sebastopol: O'Reilly, 2010.

W3C. Disponível em: <<http://www.w3.org/>>.

WHATWG. Disponível em: <<http://www.whatwg.org/>>.

CRIANDO LAYOUTS CSS PROFISSIONAIS

# CSS GRID LAYOUT



novatec

Mauricio Samy Silva  
[www.maujor.com](http://www.maujor.com)



# CSS Grid Layout

Silva, Maurício Samy

9788575228043

176 páginas

[Compre agora e leia](#)

A criação de Layout CSS sempre foi uma tarefa trabalhosa, mas agora os profissionais têm uma ferramenta poderosa ao seu alcance, o CSS Grid Layout, uma nova especificação do W3C, que veio para resolver praticamente todos os problemas de posicionamento na tela. Utilizando um novo método de layout CSS, bidimensional, com linhas e colunas, mais simples e muito mais fácil de usar, permite controlar o tamanho e a posição dos componentes da interface e seus conteúdos. Não será mais necessário inflar a marcação HTML com elementos desnecessários para a criação de layouts CSS. Explicações teóricas são abordadas em linguagem didática, dispensando, sempre que possível, o jargão técnico avançado, além de serem acompanhadas de exemplos práticos explicados passo a passo e complementados por arquivos HTML, disponíveis online para consulta e download. Neste livro, Maujor, o dinossauro das CSS, mostra de forma didática os recursos e as funcionalidades dessa nova ferramenta. Com auxílio de exercícios práticos, você aprenderá todas as potencialidades dessa nova tecnologia para criar layouts extremamente profissionais, que antes somente eram possíveis com a utilização de frameworks. Este livro

é indicado para desenvolvedores front-end, web designers e estudantes com alguma experiência em CSS.

[Compre agora e leia](#)

O'REILLY

# Python para Análise de Dados

TRATAMENTO DE DADOS COM  
PANDAS, NUMPY E IPYTHON



novatec

Wes McKinney

# Python para análise de dados

McKinney, Wes

9788575227510

616 páginas

[Compre agora e leia](#)

Obtenha instruções completas para manipular, processar, limpar e extrair informações de conjuntos de dados em Python. Atualizada para Python 3.6, este guia prático está repleto de casos de estudo práticos que mostram como resolver um amplo conjunto de problemas de análise de dados de forma eficiente. Você conhecerá as versões mais recentes do pandas, da NumPy, do IPython e do Jupyter no processo. Escrito por Wes McKinney, criador do projeto Python pandas, este livro contém uma introdução prática e moderna às ferramentas de ciência de dados em Python. É ideal para analistas, para quem Python é uma novidade, e para programadores Python iniciantes nas áreas de ciência de dados e processamento científico. Os arquivos de dados e os materiais relacionados ao livro estão disponíveis no GitHub.

- utilize o shell IPython e o Jupyter Notebook para processamentos exploratórios;
- conheça os recursos básicos e avançados da NumPy (Numerical Python);
- comece a trabalhar com ferramentas de análise de dados da biblioteca pandas;
- utilize ferramentas flexíveis para carregar, limpar, transformar, combinar e reformatar dados;
- crie visualizações informativas com a matplotlib;
- aplique o recurso

groupby do pandas para processar e sintetizar conjuntos de dados; • analise e manipule dados de séries temporais regulares e irregulares; • aprenda a resolver problemas de análise de dados do mundo real com exemplos completos e detalhados.

[Compre agora e leia](#)

O'REILLY®

# Padrões para Kubernetes

Elementos reutilizáveis no design de aplicações  
nativas de nuvem



novatec

Bilgin Ibryam  
Roland Huß

# Padrões para Kubernetes

Ibryam, Bilgin

9788575228159

272 páginas

[Compre agora e leia](#)

O modo como os desenvolvedores projetam, desenvolvem e executam software mudou significativamente com a evolução dos microsserviços e dos contêineres. Essas arquiteturas modernas oferecem novas primitivas distribuídas que exigem um conjunto diferente de práticas, distinto daquele com o qual muitos desenvolvedores, líderes técnicos e arquitetos estão acostumados. Este guia apresenta padrões comuns e reutilizáveis, além de princípios para o design e a implementação de aplicações nativas de nuvem no Kubernetes. Cada padrão inclui uma descrição do problema e uma solução específica no Kubernetes. Todos os padrões acompanham e são demonstrados por exemplos concretos de código. Este livro é ideal para desenvolvedores e arquitetos que já tenham familiaridade com os conceitos básicos do Kubernetes, e que queiram aprender a solucionar desafios comuns no ambiente nativo de nuvem, usando padrões de projeto de uso comprovado. Você conhecerá as seguintes classes de padrões:

- Padrões básicos, que incluem princípios e práticas essenciais para desenvolver aplicações nativas de nuvem com base em contêineres.
- Padrões comportamentais, que exploram conceitos mais

específicos para administrar contêineres e interações com a plataforma. • Padrões estruturais, que ajudam você a organizar contêineres em um Pod para tratar casos de uso específicos. • Padrões de configuração, que oferecem insights sobre como tratar as configurações das aplicações no Kubernetes. • Padrões avançados, que incluem assuntos mais complexos, como operadores e escalabilidade automática (autoscaling).

[Compre agora e leia](#)



# CANDLESTICK

Um método para ampliar lucros na Bolsa de Valores



novatec

Carlos Alberto Debastiani

# Candlestick

Debastiani, Carlos Alberto

9788575225943

200 páginas

[Compre agora e leia](#)

A análise dos gráficos de Candlestick é uma técnica amplamente utilizada pelos operadores de bolsas de valores no mundo inteiro. De origem japonesa, este refinado método avalia o comportamento do mercado, sendo muito eficaz na previsão de mudanças em tendências, o que permite desvendar fatores psicológicos por trás dos gráficos, incrementando a lucratividade dos investimentos.

Candlestick – Um método para ampliar lucros na Bolsa de Valores é uma obra bem estruturada e totalmente ilustrada. A preocupação do autor em utilizar uma linguagem clara e acessível a torna leve e de fácil assimilação, mesmo para leigos. Cada padrão de análise abordado possui um modelo com sua figura clássica, facilitando a identificação. Depois das características, das peculiaridades e dos fatores psicológicos do padrão, é apresentado o gráfico de um caso real aplicado a uma ação negociada na Bovespa. Este livro possui, ainda, um índice resumido dos padrões para pesquisa rápida na utilização cotidiana.

[Compre agora e leia](#)



AVALIANDO  
EMPRESAS

# INVESTINDO EM AÇÕES

A APLICAÇÃO PRÁTICA DA  
ANÁLISE FUNDAMENTALISTA NA  
AVALIAÇÃO DE EMPRESAS

novatec

CARLOS ALBERTO DEBASTIANI  
FELIPE AUGUSTO RUSSO

# Avaliando Empresas, Investindo em Ações

Debastiani, Carlos Alberto

9788575225974

224 páginas

[Compre agora e leia](#)

Avaliando Empresas, Investindo em Ações é um livro destinado a investidores que desejam conhecer, em detalhes, os métodos de análise que integram a linha de trabalho da escola fundamentalista, trazendo ao leitor, em linguagem clara e acessível, o conhecimento profundo dos elementos necessários a uma análise criteriosa da saúde financeira das empresas, envolvendo indicadores de balanço e de mercado, análise de liquidez e dos riscos pertinentes a fatores setoriais e conjunturas econômicas nacional e internacional. Por meio de exemplos práticos e ilustrações, os autores exercitam os conceitos teóricos abordados, desde os fundamentos básicos da economia até a formulação de estratégias para investimentos de longo prazo.

[Compre agora e leia](#)