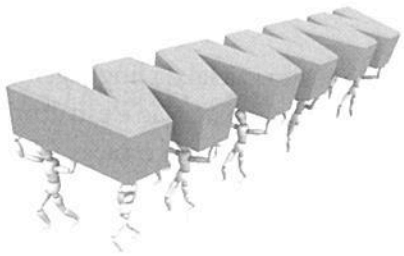

CAPÍTULO 3

Folhas de estilos em cascata

Neste capítulo teremos uma introdução às CSS. Veremos sua definição e um breve histórico de seu aparecimento. Em seguida, passaremos ao estudo da sintaxe, dissecando a regra CSS. Veremos a conceituação de unidades de medida CSS e os métodos para vincular folhas de estilos a documentos HTML. Ao final do capítulo o leitor terá uma visão geral da evolução das CSS, desde a sua criação, reconhecerá a terminologia dos fragmentos que compõem uma folha de estilo, terá uma noção sólida das unidades de medida CSS e saberá como linkar folhas de estilo a documentos (X)HTML.

3.1 CSS – Folhas de estilo em cascata

3.1.1 Definição

CSS é a abreviação para os termos em inglês Cascading Style Sheet, traduzido para o português como folhas de estilo em cascata. Neste livro adotaremos CSS como abreviação e folhas 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."

3.1.2 Finalidade

As CSS têm por finalidade devolver à (X)HTML o propósito inicial da linguagem. A HTML foi criada para ser uma linguagem exclusivamente de marcação e estruturação de conteúdos. Isto 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 função da HTML. Cabem às CSS todas as funções de apresentação de um documento, e esta é sua finalidade maior. Daí a já consagrada frase que resume a dobradinha CSS + (X)HTML:

"(X)HTML para estruturar e CSS para apresentar".

O termo agente de usuário é empregado para se fazer referência a qualquer dispositivo capaz de interpretar um documento escrito em linguagem de marcação. O exemplo mais comum e conhecido de agente de usuário é o navegador. Leitores de tela, robôs de indexação e busca, dispositivos móveis como PDAs e telefones são também alguns exemplos de agentes de usuário.

3.1.3 Histórico

Tim Berners-Lee, ao desenvolver o navegador Nexus que serviu para implementar seus inventos, conforme vimos no capítulo anterior, escreveu também, ainda que de forma bastante limitada, algumas funcionalidades intrínsecas que controlavam a apresentação dos documentos. Navegadores que se seguiram, nos anos de 1992 e 1993, também vinham com funcionalidades de estilização-padrão, tais como no modelo desenvolvido por Tim. No navegador Mosaic, lançado em 1993 e que popularizou a web, também foram previstas funcionalidades mínimas para aplicar estilos. Na verdade, apenas o controle de algumas fontes e de cores era possível. As funcionalidades intrínsecas aqui referidas são aquelas que hoje em dia conhecemos como folhas de estilo-padrão do navegador. Um conjunto de regras de estilo forma uma folha de estilo que o navegador aplica aos documentos por padrão nos casos em que o autor do documento ou o usuário não tenha definido qualquer regra de estilo. Adiante veremos detalhes da folha de estilo.

Em setembro de 1994 surgiu a primeira proposta para implementação das CSS. Até então, o próprio Tim considerava que a estilização era uma questão a ser resolvida pelo navegador, razão pela qual não se preocupou em publicar a sintaxe usada para criar a folha de estilo-padrão do seu navegador.

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

3.1.4 Sintaxe CSS

3.1.4.1 Regra CSS

Regra CSS é a unidade básica de uma folha de estilo. Entenda-se como unidade básica a menor porção de código capaz de produzir um 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 Figura 3.1 mostra a sintaxe para escrever uma regra CSS.

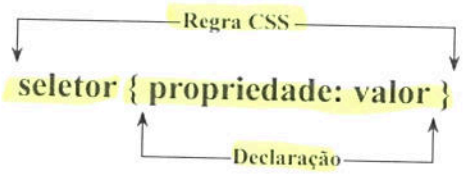

Figura 3.1 – Sintaxe da regra CSS.

Definição dos componentes de uma regra CSS:

- **Seletor** – é o alvo da regra CSS. Genericamente, é a tag do elemento da marcação ou uma entidade capaz de definir com precisão em qual lugar da marcação será aplicada a regra CSS.
- **Declaração** – determina os parâmetros de estilização. Compreende a propriedade e o valor.
  - **Propriedade** – define qual será a característica do elemento, alvo do seletor, a ser estilizada.
  - **Valor** – é a quantificação ou a qualificação da propriedade.

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

Uma regra CSS pode conter mais de uma declaração de estilo. Isto significa que, para estilizar mais de uma propriedade de um mesmo seletor, você pode escrever todas as declarações de estilo em uma mesma regra. Por exemplo: queremos que os parágrafos do documento sejam na cor preta, em fundo amarelo, textos em itálico e alinhados à direita. A regra CSS para estilizar o seletor parágrafo é mostrada a seguir:

```
p {
   color: #000;
   background-color: #ff0;
   font-style: italic;
   text-align: right;
   }
```

Ao escrever uma regra CSS contendo mais de uma declaração, conforme mostrado anteriormente, deve-se usar ponto-e-vírgula para separá-las. O ponto-e-vírgula é facultativo no caso de propriedade única ou após a declaração da última propriedade no caso de mais de uma. Nada impede que você escreva a regra mostrada anteriormente da seguinte forma:

```
p {color:#000;background-color:#ff0;font-style:italic;text-align:right}
```

Observe que agora escrevemos de forma bem mais compacta, eliminando os espaços entre propriedade e valor e também não usamos ponto-e-vírgula após a última declaração. A grande vantagem desta forma de escrever a regra é a economia de espaço e tempo de digitação. A desvantagem é que a leitura e a visualização das propriedades tornaram-se mais difíceis. A legibilidade foi seriamente prejudicada. Isto pode trazer sérias complicações para a manutenção e a correções de bugs, sobretudo em folhas de estilos mais extensas. Usar uma ou outra forma fica a critério do desenvolvedor.

A experiência demonstra que é de boa prática adotar, para a escrita de regras CSS, a forma estendida, isto é, cada declaração em uma linha. É também de boa prática colocar o ponto-e-vírgula após a última declaração como forma de prevenir erro causado pelo esquecimento da colocação da pontuação no caso de acréscimo de mais declarações na regra.

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 a seguir:

```
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 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;  }
```

Eliminar o espaço entre o seletor e o sinal de abrir chaves pode causar confusão em alguns navegadores e deve ser evitado.

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. Isto torna o código consistente e facilita a manutenção. Com as folhas de estilo aplica-se esta prática, e você pode escolher qualquer 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. A forma estendida de declarar as propriedades em linhas distintas é escrita conforme mostrado a seguir, sendo a endentação a critério de desenvolvedor.

```
h1 {
   border: 1px solid red;
   }
```

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 no código de estilos, conforme mostrado nos exemplos a seguir:

Comentário em uma linha:

```
/* Este é um comentário em uma linha */
```

Bloco de comentário:

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

Você deve começar um comentário com o sinal /* e terminar com o sinal */.

A um conjunto de regras CSS denominamos *folha de estilo*. O conjunto das regras pode ser escrito no próprio documento onde as regras serão aplicadas ou em um arquivo externo gravado com a extensão-padrão `.css`. Por exemplo: `minha-folha.css`.

3.1.4.2 Agrupamento de seletores

Para aplicar uma regra CSS comum a vários seletores, você pode agrupá-los. Para escrever um agrupamento de seletores, separe-os com uma vírgula. No exemplo a seguir definimos que a cor dos cabeçalhos de níveis 1 e 2 e dos parágrafos será vermelha (`red`).

```
h1 {color: red;}
h2 {color: red;}
p {color: red;}
```

Agrupando os seletores, escrevemos as três regras em uma como a seguir:

```
h1, h2, p {color: red;}
```

3.1.4.3 Seletor classe

Nos exemplos apresentados anteriormente usamos as próprias tags da marcação (X) HTML como seletor. Seletores CSS não estão restritos às tags, e podem ser diversas entidades da marcação ou combinações delas. Seletores podem ser, por exemplo, atributos do (X)HTML ou seus valores, ou, ainda, combinações de tags entre si, de tags com atributos, e assim por diante, como veremos no Capítulo 5.

Um seletor amplamente empregado nas CSS é o seletor classe, que casa com os elementos (X)HTML da marcação aos quais tiver sido atribuída a classe.

Usaremos a expressão casar um seletor com o mesmo sentido que ela tem na sintaxe de expressões regulares. Dizer que o seletor S casa com o elemento E significa que as regras de estilo para o seletor S são aplicadas no elemento E.

A grafia para o atributo (X)HTML classe é: `class`. Aplicar uma classe significa associar um nome a um elemento. Uma classe pode ser aplicada à maioria dos elementos (X)HTML, enquanto ao mesmo elemento (X)HTML podem ser atribuídas várias classes. Observe, no exemplo a seguir, a sintaxe de aplicação de classe em elementos (X)HTML:

```
<h4 class="diferente">Cabeçalho da classe diferente</h4>
```

A sintaxe para seletores de classe consiste na combinação do sinal de ponto (.) imediatamente seguido do nome da classe. O uso do nome do elemento para completar a grafia do seletor é facultativo, e a conveniência de seu emprego depende da

especificidade do seletor, assunto que veremos adiante. Nos exemplo a seguir, as duas sintaxes estão corretas:

```
.diferente {
   color: black;
   }

h4.diferente {
   color: black;
   }
```

Você já deve ter concluído que o uso do seletor classe possibilita aplicar estilos diferentes para o mesmo tipo de elemento. Basta que se atribuam classes diferentes para um mesmo elemento, conforme mostrado a seguir:

- **HTML**

```
<p>Texto do parágrafo na cor azul</p>
<p class="cor-um">Texto do parágrafo na cor preta</p>
<p class="cor-dois">Texto do parágrafo na cor vermelha</p>
```

- **CSS**

```
p {
   color: blue;
   }
p.cor-um {
   color: black;
   }
p.cor-dois {
   color: red;
   }
```

O elemento parágrafo será na cor preta (`black`), se pertencer à classe `cor-um`, na cor vermelha (`red`), se pertencer à classe `cor-dois`, e na cor azul (`blue`), se não pertencer a nenhuma das duas classes.

Ao escolher um nome para a classe, evite usar números e caracteres especiais. Dê preferência a nomes de classe formados por letras minúsculas. Embora não seja proibido, há restrições quanto ao uso de números e caracteres.

É permitido declarar mais de um nome para a classe do mesmo elemento (X) HTML, e, neste caso, os nomes devem ser separados por um espaço:

```
<p class ="cor-um destaque">
   Texto do parágrafo pertencente às classes cor-um e destaque.
</p>
```

Regras de estilo definidas tanto para a classe `cor-um` quanto para a classe `destaque` serão aplicadas no parágrafo mostrado na marcação anterior.

Observe os códigos a seguir e descubra o que há de errado.

- **HTML**

```
<div class="esquerda">Coluna de navegação à esquerda</div>
<p class="vermelha">Texto na cor vermelha</p>
```

- **CSS**

```
div.esquerda {
    float:left;
    }
p.vermelha {
    color:red;
    }
```

Suponha que no futuro você conclua que o seu site está obsoleto e decida modernizar o design e o layout. Para o novo design, você concluiu que a cor vermelha para parágrafos agora deva ser azul e que a coluna de navegação do site, que se encontra à esquerda, deva ser colocada à direita. Imagine como ficaria a folha de estilos, após introduzidas as modificações nas regras CSS para contemplar o novo design, e descobrirá facilmente o que há de errado com os códigos. A conclusão:

Ao escolher nomes de classe, procure aqueles que se relacionem à estrutura do documento e evite escolher nomes que lembrem a apresentação.

Nomes como `esquerdo`, `direito`, `amarelo`, `grande`, `inclinado`, `forte` etc. são uma má escolha. Nomes como `menu`, `principal`, `auxiliar`, `destaque`, `cor-um`, `tamanho-dois` etc. são uma boa escolha.

Os seguintes elementos (X)HTML não admitem o atributo classe: `base`, `basefont`, `head`, `html`, `meta`, `param`, `script`, `style`, `title`. Ver o Apêndice A para uma lista completa dos elementos (X)HTML.

3.1.4.4 Seletor `id`

O seletor `id` é bem parecido com o seletor classe, e funciona de maneira idêntica para fins de estilização. As três diferenças fundamentais entre o atributo `id` e o atributo `class` são:

- Um atributo `id` e seu respectivo nome deve ser único no documento, isto é, aplica-se a um e somente um elemento (X)HTML dentro do documento.

- A natureza única do atributo `id` confere-lhe a característica de identificar exclusivamente um determinado elemento no documento, por isso é denominado genericamente *identificador*.

- A especificidade de um seletor `id` é maior do que a de um seletor classe. A especificidade de seletores será explicado em [C4 S4.1.3.1].

O identificador `id` pode ser aplicado à maioria dos elementos (X)HTML e, é bom repetir, cada identificador aplica-se uma só vez no mesmo documento. Observe, no exemplo a seguir, a sintaxe de aplicação do atributo `id` em elementos (X)HTML:

```
<div id="principal">Elemento DIV com identificador ID de nome principal</div>
```

A sintaxe CSS para seletores `id` consiste na combinação do sinal tralha (#) imediatamente seguido do nome do identificador. O uso do nome do elemento para completar a grafia do seletor é facultativo, e a conveniência de seu emprego depende da especificidade do seletor que está explicado em [C4 S4.1.3.1]. Nos exemplo a seguir, as duas sintaxes estão corretas:

```
#principal {
   color: black;
   }

div#diferente {
   color: red;
   }
```

3.2 Unidades CSS para medidas lineares

Vários aspectos da apresentação de um documento dependem da definição de uma medida de comprimento para serem aplicados. A espessura de uma borda ou a largura ou altura de um elemento são exemplos de medidas CSS. Medidas CSS lineares são expressas por um número real – inteiro ou fracionário – negativo ou positivo. Algumas propriedades admitem somente valores positivos, como veremos adiante. Uma medida CSS linear deve ser seguida por uma abreviação representando a unidade da medida declarada. São exemplos de unidade de medida CSS lineares o `px` (pixel) e o `mm` (milímetro).

Neste livro adotaremos a expressão simplificada medida CSS para designar medidas lineares de comprimento válidas na sintaxe para folhas de estilos.

As unidades de medida CSS podem ser *absolutas* ou *relativas*.

3.2.1 Unidades de medida absolutas

As unidades de medida absolutas são aquelas que não dependem de um valor de referência.

As cinco unidades de medida absolutas são:

Abreviatura	Unidade
`in`	Polegada.
`cm`	Centímetro.
`mm`	Milímetro.
`pt`	Ponto. Uma unidade de medida tipográfica. `1pt` é igual a 1/72inch.
`pc`	Pica. Outra unidade de medida tipográfica. `1pc` é igual a 12pt.

Tais unidades só devem ser usadas quando são conhecidas com detalhes as características físicas e as configurações da mídia para a qual o documento se destina. Um navegador, por exemplo, apresenta um documento de acordo com as configurações do monitor do usuário, entre elas o tamanho e a resolução. Obviamente, o desenvolvedor não tem como saber antecipadamente ou controlar aquelas configurações. Por esta razão, das unidades de medida absoluta apenas a unidade `pt` (ponto) é empregada com mais freqüência para a mídia `printer` (impressão), a qual é satisfatoriamente consistente com esta medida herdada da tipografia.

3.2.2 Unidades de medida relativas

As unidades de medida relativas são aquelas que tomam como base um valor de referência anteriormente definido.

As três unidades de medida relativas são:

Abreviatura	Unidade
`em`	`1em` é igual ao tamanho de fonte definido para o elemento em questão. Ver exemplos a seguir.
`ex`	`1ex` é igual a altura da letra xis minúscula (x) da fonte definida para o elemento em questão. Ver exemplos a seguir.
`px`	Pixel.

3.2.2.1 Unidade `em`

A unidade de medida `em` é calculada em relação a um tamanho de fonte predefinido no documento.

Considere os códigos a seguir:

- **HTML**

```
<h1>Cabeçalho nível 1</h1>
<h2>Cabeçalho nível 2</h2>
<h3>Cabeçalho nível 3</h3>
<h4>Cabeçalho nível 4</h4>
```

- **CSS**

```
h1 {font-size: 50px;}
h2 {font-size: 40px;}
h3 {font-size: 30px;}
h4 {font-size: 20px;}
h1, h2, h3, h4 {margin-left: 1em;}
```

Observe na Figura 3.2 a renderização dos códigos e a dependência da unidade `em` com o tamanho de fonte definido na propriedade `font-size`.

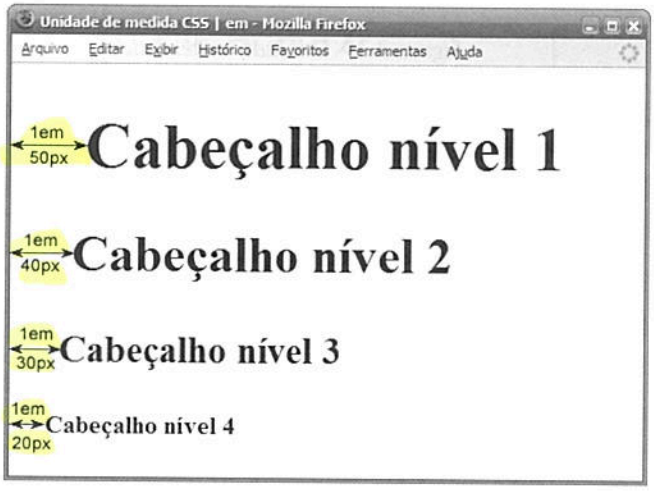

Figura 3.2 – Unidade `em` *no navegador.*

A última regra CSS definiu a margem esquerda para os quatro cabeçalhos igual a `1em`. Como a unidade `em` é relativa ao tamanho da fonte, para cada nível de cabeçalho a medida da margem esquerda é diferente, embora todas elas tenham sido definidas iguais a `1em`. Tivéssemos definido para o cabeçalho nível 4 `margin-left: 2em`, esta medida seria igual a `40px`, resultando em uma margem igual à do cabeçalho nível 2.

Não sendo definido um tamanho de fonte para um elemento, será tomado como base para a unidade `em` o tamanho de fonte do elemento definido para o ancestral mais próximo.

Considere os códigos a seguir:

- **HTML**

```
<p>Um parágrafo com uma <b>palavra em negrito</b> e mais texto</p>
```

- **CSS**

```
p {font-size: 20px;}
b {font-size: 0.8em;}
```

Observe na Figura 3.3 a renderização do HTML estilizado de acordo com as regras CSS.

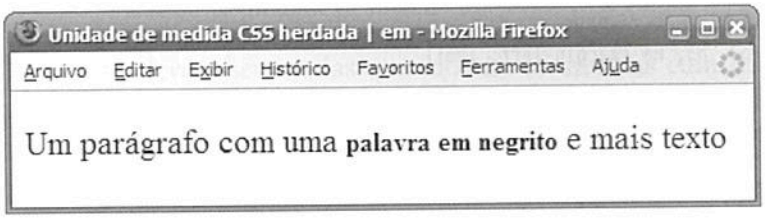

Figura 3.3 – Unidade em *herdada.*

O tamanho da fonte para o texto *palavra em negrito* é de 16px. O elemento parágrafo é pai do elemento negrito. Assim, o tamanho de fonte em unidade em para ele será tomado com base no tamanho de fonte do parágrafo, ou seja, 20px, resultando 0.8 x 20px = 16px.

Caso não tenha sido definido um tamanho de fonte para um elemento ancestral, este será tomado, por padrão, igual a 16px, resultando 1em = 16px. Uma conseqüência imediata disto é que, se definirmos um tamanho de fonte igual a 62,5% para o elemento-pai, teremos 1em = 10px, pois 62,5% de 16px é igual a 10px. Segue-se que 1.2em = 12px, 2em = 20px, e assim por diante, facilitando a definição de medidas de comprimento na unidade em.

3.2.2.2 *Unidade* ex

A unidade de medida ex é igual à altura da letra *xis* minúscula da fonte adotada. Ver a unidade de medida tipográfica x-height na Figura 6.1.

Considere os códigos a seguir:

- **HTML**

```
<p class="ari">Fonte Arial - x</p>
<p class="geo">Fonte Georgia - x</p>
<p class="cou">Fonte Courier - x</p>
<p class="imp">Font Impact - x</p>
<p class="tim">Fonte Times - x</p>
```

- CSS

```
p {font-size: 40px;}
p.ari {font-family: arial;}
p.geo {font-family: georgia;}
p.cou {font-family: courier;}
p.imp {font-family: impact;}
p.tim {font-family: times;}
```

Observe na Figura 3.4 que a altura da letra *x* depende do tipo de fonte definida na propriedade `font-family`, e não mais no tamanho de fonte como foi o caso para a unidade `em` que acabamos de estudar.

Na prática, a maioria dos navegadores não implementa corretamente a unidade ex devido às complicações para computar a altura de letras. Adotam uma simplificação que consiste em fazer `1ex` igual a `0.5em`. Não é comum usar esta unidade em desenvolvimento de sites.

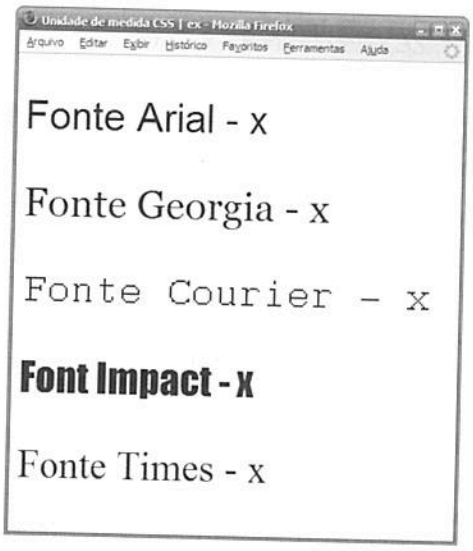

Figura 3.4 – Unidade ex.

3.2.2.4 Unidade px

A unidade de medida `px` é calculada em relação à resolução do dispositivo no qual o documento é apresentado, na maioria dos casos: a tela de um monitor. As definições para as unidades de medida descritas anteriormente são simples e de entendimento imediato como você acabou de constatar. Quando se trata de pixel, a teoria passa por considerações de óptica física, cálculos de densidade e uma série de outras implicações cujo detalhamento fogem ao escopo deste livro. Caso você

esteja interessado em aprofundar-se neste assunto, recomendo começar lendo o que prescrevem as Recomendações do W3C para as CSS2.1 em: http://www.w3.org/TR/CSS21/syndata.html#length-units.

O uso do pixel é útil na criação de layouts com dimensões fixas. É o caso clássico de sites projetados com uma largura de 760px ou algo próximo a tal valor para serem renderizados sem barra de rolagem horizontal em resoluções-padrão de 800 x 600px. Apresentam a desvantagem de ocupar pouca área útil do monitor em resoluções maiores, tais como, 1280 x 1024px. A definição de unidades em pixel proporciona ainda ao autor um controle rígido sobre o layout no qual a precisão absoluta é necessária. Outra desvantagem da adoção do pixel diz respeito à acessibilidade. Afinal, o uso da unidade de medida é uma decisão que depende de cada caso.

3.2.3 Porcentagens

Porcentagens são calculadas em relação a um valor preexistente, em geral uma unidade de medida.

O valor de referência para o cálculo da porcentagem depende da propriedade CSS. Em geral, o valor de referência é determinado pelo valor de uma propriedade do próprio elemento, ou o valor de uma propriedade do seu elemento ancestral ou ainda um valor dependente do contexto geral de formatação, como, por exemplo, a largura de um elemento de bloco.

A sintaxe para escrever porcentagem consiste de um valor de 0 a 100 seguido do sinal de porcentagem (%).

O exemplo a seguir ilustra a aplicação e o cálculo de porcentagem em folhas de estilo.

- **HTML**

```
<p>Neste parágrafo uma <span>palavra</span> maior</p>
```

- **CSS**

```
p {
   width: 50%;
   font-size: 30px;
   }

span {
   font-size: 200%;
   }
```

Observe na Figura 3.5 que o navegador colocou o parágrafo com uma largura total igual a 50% da janela do navegador, e a palavra dentro do parágrafo com fonte igual a 200% do tamanho de fonte do parágrafo.

Para o elemento parágrafo, a referência foi a largura total da janela do navegador e para a palavra marcada com o elemento `span` contido no elemento `p`, a referência foi o tamanho da fonte do parágrafo. O fundo cinza no parágrafo foi destacado.

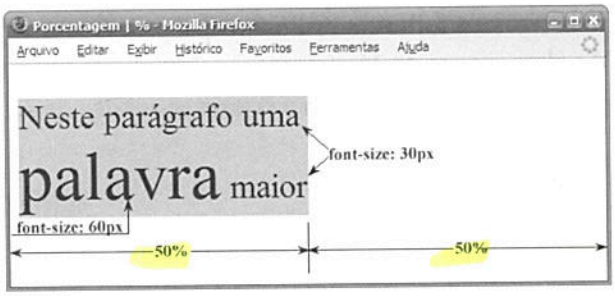

Figura 3.5 – Porcentagem CSS.

Ao escrever regras de estilo, quando uma medida qualquer tem valor 0 (zero), não há necessidade de se especificar a unidade da medida, seja ela qual for. Escrever, por exemplo, 0px, 0%, 0em etc., não tem propósito nenhum. Escreva simplesmente 0.

3.3 Vinculando folhas de estilo a documentos

Depois que você acabou de escrever sua folha de estilo para um documento (X)HTML, precisa informar ao documento onde ele deve buscar a sua folha de estilo. Ou seja, você precisa de um método capaz de vincular a folha de estilo ao documento no qual ela será aplicada.

As folhas de estilo podem ser escritas no próprio documento (X)HTML onde serão aplicadas ou serem arquivos externos independentes, gravados com a extensão `.css`, como, por exemplo, um arquivo chamado de `meuestilo.css`, e linkados ao documento.

3.3.1 Estilos inline

O método direto e simples de aplicar estilos a um elemento da marcação é com o emprego do atributo `style` do (X)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:red; background:#eee; font-size:1.8em;">
Parágrafo com aplicação de estilos inline.
</p>
```

Este 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.

3.3.2 Estilos incorporados

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

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

... outras regras...
</style>
</head>
...
```

A vantagem deste método sobre o anterior é que, agora, localizamos com mais facilidade a folha de estilo, mas tem a desvantagem de colocar a folha de estilos dentro do próprio documento. Não seria sensato vincular uma mesma folha de estilo a vários documentos empregando este 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 estilos em todos eles. Uma boa escolha para uso deste método seria para o caso de aplicação de estilos específicos a um documento do site.

O elemento `style` deve estar contido na seção `head` do documento. Notar o uso de atributos no elemento style. O atributo `type` informa qual o tipo de dado está sendo enviado e o atributo `media` informa para qual tipo de mídia deve ser aplicado aos estilos. Os valores do atributo `media` e a mídia a que se destina são listados a seguir:

Valor	Mídia
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.
all	Todos os tipos de mídia.

3.3.3 Estilos externos

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

3.3.3.1 Folhas de estilo linkadas

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 onde ele esta 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>
<body>
   ...
```

O atributo `href` aponta para o endereço onde se encontra o arquivo da folha de estilo.

3.3.3.2 Folhas de estilo importadas

Neste método, você vincula uma folha de estilo externa a um documento usando a diretiva `@import` dentro do elemento `style`. A sintaxe para este tipo de vinculação é mostrada a seguir.

```
...
<head>
   ...
   <style type="text/css">
      @import url("estilos.css") screen, projection;
   </style>
   ...
</head>
...
```

Alternativamente, você pode usar uma forma abreviada da sintaxe, como se mostra a seguir:

```
<style type="text/css">
   @import "estilos.css" screen, projection;
</style>
```

As duas sintaxes apresentadas (com ou sem a notação `url (...)` na diretiva) são equivalentes. O tipo de mídia é definido em uma lista separada por vírgula na própria diretiva. Na ausência do tipo de mídia, os estilos serão aplicados para todas as mídias, ou seja, o efeito é o mesmo que o de declarar mídia `all`. Declarando ou não a mídia, deve-se terminar a diretiva com um ponto-e-vírgula.

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 declaradas após a diretiva.

O método de vincular folhas de estilo externa 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 tamanho de fonte na folha de estilo imediatamente se reflete no site inteiro quer ele tenha 10 ou 10.000 páginas.