

:: Introdução as CSS ::

O nascimento do HTML

A linguagem de marcação HTML (Hyper Text Markup Language) foi desenvolvida e aperfeiçoada até tornar-se tal como a conhecemos nos dias atuais a partir de uma "invenção" devida a um pesquisador físico, para trafégo de seus textos e informações de natureza científica.

Assim, o embrião do HTML surgiu para servir a uma comunidade bastante restrita, a comunidade de cientistas. Com a introdução gradativa de novas tags, atributos e aplicações específicas, o HTML tornou-se padrão mundial de apresentação de conteúdo na Web.

O HTML atual

E, já com várias inovações o HTML era usado para construção de páginas Web, que no início limitavam-se a exibir informações contidas nos documentos.

A evolução vinha atropelando tudo com uma avalanche de novos aplicativos, facilidades, softwares, hardware etc. E o HTML não passou ao largo, pelo contrário, a simples linguagem de marcação destinada a apresentar conteúdos carecia de uma maior flexibilidade no sentido de manipular visualmente os conteúdos.

Novas tags e atributos foram inventados, tais como a tag `font` e o atributo `color` que permitiam alterar a aparência de textos. Assim nasceu a **estilização** dos conteúdos.

E a evolução trazendo novas descobertas, corre célere neste dinamismo alucinante que estamos testemunhando até os dias de hoje. Novas tags e novos atributos de estilo foram introduzidos no HTML. Com isso, a velha linguagem de marcação passou a exercer uma dupla função. A função de estruturar o conteúdo através da marcação e a função de apresentá-lo ou seja de dar a aparência final.

Os problemas criados

Mas, esta dupla função do HTML, se por um lado resolveu uma necessidade dos designers e projetistas por outro acabou trazendo sérios problemas aos projetos criados. Os documentos Web publicados na Internet, cada vez mais sofisticados e extensos, estavam fugindo do controle de seus criadores.

Para ilustrar suponha o seguinte exemplo:

Seu melhor cliente telefona às 17:00h da tarde de uma sexta-feira (sempre ligam nesta hora para solicitar alguma coisa não é mesmo?) e diz o seguinte;

teremos uma reunião aqui na empresa, na segunda-feira às 0800h com um potencial comprador e é nossa intenção fazer uma apresentação dos nossos produtos através do

site que você criou e mantém. Seguindo uma sugestão do nosso departamento de marketing precisamos mudar a cor de todos os títulos no site de verde para vermelho, pois que esta é a cor principal da marca do nosso comprador e com isso pretendemos fixar uma cumplicidade subliminar. Isto é bem simples de fazer, não é? Afinal é só mudar a cor! Dá para você 'botar no ar' até às 19:30h ? Quero dar uma olhadinha antes de encerrar o expediente. OK?

Claro que você concorda e responde que vai providenciar rapidinho, afinal *é só para mudar a cor*. Mas, são 180 páginas no site! E os títulos são tags de cabeçalho deste tipo:

```
<h1><font color="#00FF00">Título</font></h1>
```

Supondo uma média de 3 títulos por página, você tem um total de 540 tags `font` para editar e mudar o atributo `color`. E se o seu cliente tivesse pedido para mudar a cor dos textos, e do fundo? Bem, este exemplo simples dá uma dimensão de um dos problemas criados com a mistura de marcação com apresentação - estilização!

A solução proposta

Cada vez mais ficava evidente que esta mistura que maravilhou os projetistas Web no início, tornara-se uma grande dor de cabeça. E é claro, a solução passava por dissociar linguagem de marcação da estilização.

Desta necessidade, eu diria mesmo uma imposição, nasceu as CSS, sigla em inglês para *Cascading Style Sheet* que em português foi traduzido para Folha de Estilo em Cascata.

A introdução deste conceito preconiza o uso dos elementos (tags) HTML, exclusivamente para marcar e estruturar o conteúdo do documento. Nenhum elemento HTML será usado para alterar a apresentação, ou seja estilizar o conteúdo.

A tarefa de estilização ficará a cargo das CSS que nada mais é do que um arquivo independente do arquivo HTML no qual são declaradas propriedades e valores de estilização para os elementos do HTML.

Estas declarações de estilo, quer sejam estruturadas em um arquivo externo com extensão `.css` quer sejam declaradas de outros modos (importadas, lincadas, incorporadas ou inline), contém todas as regras de estilo para os elementos do documento HTML.

Voltando àquela situação criada no item anterior, agora você mudaria a cor de TODOS os cabeçalhos `h1` em TODO o site em CINCO SEGUNDOS. Às 19:20h você retorna a ligação do cliente e pede para a secretária avisá-lo de que "já está no ar", sem maiores traumas, correrias e estresses. Ah e mais, mesmo que o site tivesse 1.800 páginas e não as 180 da situação criada, você gastaria os mesmos cinco segundos.

As restrições

A idéia, a filosofia mesmo, de projeto Web aponta para uso amplo das CSS, ainda não explorada em toda sua potencialidade por razões de incompatibilidades de certas

propriedades CSS com navegadores mais antigos e com as interpretações diferentes das CSS por parte das aplicações de usuários criadas por fabricantes distintos.

Contudo, há uma tendência - e torcemos para que se concretize rapidamente - de que as novas tecnologias voltadas para o desenvolvimento, não só das variadas aplicações de usuário como também de softwares e hardwares, atendam e se enquadrem dentro das recomendações e especificações dos órgãos normatizadores, notadamente as standards do W3C.

Quando o projeto Web em todas as suas incontáveis variantes, seguir a normatização e padronização recomendada pelo W3C, teremos uma Web muito mais fácil, dinâmica e agradável.

O efeito cascata

Que estilo será aplicado, quando há conflito de estilos especificados (por exemplo: uma regra de estilo determina que os parágrafos serão na cor preta e outra que serão na cor azul) para um mesmo elemento HTML?

Aqui entra o **efeito cascata**, que nada mais é, do que o estabelecimento de uma *prioridade* para aplicação da regra de estilo ao elemento.

Para determinar a prioridade são considerados diversos fatores, entre eles, o tipo de folha de estilo, o local físico da folha de estilo no seu todo, o local físico da regra de estilo na folha de estilo e a [especificidade](#) da regra de estilo.

A prioridade para o efeito cascata:

1. folha de estilo padrão do navegador do usuário;
2. folha de estilo do usuário;
3. folha de estilo do desenvolvedor;
 - o estilo inline (dentro de um elemento HTML);
 - o estilo incorporado (definido na seção `head` do documento);
 - o estilo externo (importado ou linkado).
4. declarações do desenvolvedor com `!important`;
5. declarações do usuário com `!important`;

Assim, uma declaração de estilo com `!important` definido pelo usuário prevalece sobre todas as demais, é a de mais alta prioridade. Entre as folhas de estilo definidas pelo desenvolvedor do site, os estilos inline (dentro de um elemento HTML) tem a prioridade a mais elevada, o que significa que prevalecerá sobre aquele definido na seção `head` e esta sobre o definido em uma folha de estilo externa e finalmente a última prioridade é para estilos padrão do navegador.

Agora você já sabe o porquê de "cascata" no nome Folha de estilo em cascata. Consulte os diversos tutoriais deste site para saber mais sobre o efeito cascata.

:: A regra CSS ::

A regra CSS e sua sintaxe

Uma regra CSS é uma declaração que segue uma sintaxe própria e que define como será aplicado estilo a um ou mais elementos HTML . Um conjunto de regras CSS formam uma Folha de Estilos. Uma regra CSS, na sua forma mais elementar, compõe-se de três partes: um seletor, uma propriedade e um valor e tem a sintaxe conforme mostrado abaixo:

```
seletor { propriedade: valor; }
```

Seletor: genericamente, é o elemento HTML identificado por sua tag, ou por uma classe, ou por uma ID, ou etc., e para o qual a regra será válida (por exemplo: <p>, <h1>, <form>, .minhaclasse, etc...);

Propriedade: é o atributo do elemento HTML ao qual será aplicada a regra (por exemplo: font, color, background, etc...).

Valor: é a característica específica a ser assumida pela propriedade (por exemplo: letra tipo arial, cor azul, fundo verde, etc...)

Na sintaxe de uma regra CSS, escreve-se o seletor e a seguir a propriedade e valor separados por dois pontos e entre chaves { }. Quando mais de uma propriedade for definida na regra, deve-se usar ponto-e-vírgula para separá-las. O ponto-e-vírgula é facultativo no caso de propriedade única e também após a declaração da última propriedade no caso de mais de uma.

No entanto é de boa técnica usar-se sempre o ponto-e-vírgula após cada regra para uma propriedade.

Ver os exemplos abaixo:

```
p {
font-size: 12px; /* ponto-e-vírgula é facultativo */
}

body {
color: #000000;
background: #FFFFFF;
font-weight: bold; /*ponto-e-vírgula é facultativo */
}
```

No exemplo abaixo, o **seletor** é o "documento todo" (body - a página web), a **propriedade** é o fundo do documento e o **valor** é a cor branca.

```
body {
background: #FFFFFF;
}
```

Se o valor for uma palavra composta, deverá estar entre aspas duplas " ", ou simples '':

```
h3 {
font-family: "Comic Sans MS"
}
```

Para **maior legibilidade** das folhas de estilo, é de boa prática usar linhas distintas para escrever cada uma das declarações — propriedade e seu valor —, como mostrado abaixo:

```
p {
text-align: right;
color: #FF0000;
}
```

Isto não é obrigatório! A regra abaixo tem o mesmo efeito da regra acima e ambas as sintaxes estão corretas:

```
p {text-align: right;color: #FF0000;}
```

NOTA: A razão do uso de ponto e vírgula na última declaração ou mesmo quando só há uma declaração é que durante a fase de projeto da Folha CSS quase sempre estaremos acrescentando novas declarações e a última declaração quase nunca é a última na fase de projeto. Assim, esta prática certamente nos poupará revisões por ter esquecido um ponto e vírgula!!!!

Agrupamento de Seletores

Uma regra CSS quando válida para vários seletores, estes podem ser agrupados. Separe cada seletor com uma vírgula. No exemplo abaixo agrupamos todos os elementos cabeçalho. A cor de todos os cabeçalhos será verde.

```
h1, h2, h3, h4, h5, h6 {
color: #00FF00;
}
```

O seletor classe

Mas você não está restrito somente aos elementos HTML (tags) para aplicar regras de estilo!

Você pode "inventar" um nome e com ele criar uma **classe** a qual definirá as regras CSS. E o mais interessante das classes, é que elas podem ser aplicadas a **qualquer elemento** HTML. E mais ainda, você pode aplicar estilos diferentes para o mesmo tipo de elemento do HTML, usando classes diferentes para cada um deles.

A sintaxe para o seletor classe é mostrada abaixo. Elemento HTML mais um nome qualquer que você "inventa" precedido de . (ponto):

```
elemento HTML.minhaclasse {
propriedade: valor;
}
```

Nota: Para o nome que você "inventa" evite usar números e caracteres especiais. Tanto quanto possível use só letras de a-z e de A-Z. Há restrições quanto ao uso de números e caracteres. Minha experiência e conselho: Use só letras!

Por exemplo: suponha que você queira ter dois tipos de parágrafos em seu documento: um parágrafo com letras na cor preta e um parágrafo com letras na cor azul.

```
p.corum {
color:#000000;
}

p.cordois {
color:#0000FF;
}
```

No seu documento HTML as regras seriam aplicadas conforme abaixo:

```
<p class ="corum"> este parágrafo terá cor preta.</p>

<p class ="cordois">
este parágrafo terá cor azul.
</p>
```

Este item foi revisto em 22/06/04

Em CSS 1 não é válido atribuir mais de uma classe para um elemento HTML. O exemplo abaixo está errado:

```
<p class ="corum" class ="cordois">
Aqui há um erro.
</p>
```

Nota: CSS 2 mudou este conceito, permitindo declarar mais de uma classe, no entanto nenhum browser atual suporta mais de uma classe declarada.

Ao criar uma classe você talvez queira que ela seja aplicável a qualquer elemento HTML. Neste caso basta que se omita o nome do elemento antes da classe. Por exemplo: a regra CSS a seguir pode ser aplicada a qualquer elemento HTML ao qual você deseja atribuir cor azul:

```
.cortres {
    color: #0000FF;
}
```

No exemplo a seguir tanto o cabeçalho <h2> como o parágrafo <p> terão cor azul:

```
<h2 class="cortres">
Este cabeçalho é azul.
</h2>

<p class="cortres">
Este parágrafo é azul.
</p >
```

O seletor ID

O seletor ID difere do seletor de classe, por ser ÚNICO. Um seletor ID só pode ser aplicado a UM e somente UM elemento HTML dentro do documento.

Você pode "inventar" um nome e com ele criar uma **ID** a qual definirá as regras CSS. Uma **ID só pode ser aplicada a UM elemento HTML**.

A sintaxe para o seletor ID é mostrada abaixo. Um nome qualquer que você "inventa" precedido de # ("tralha", "jogo-da-velha" :-)):

```
#minhaID {  
  propriedade: valor;  
}
```

Nota: Para o nome que você "inventa" evite usar números e caracteres especiais. Tanto quanto possível use só letras de a-z e de A-Z. Há restrições quanto ao uso de números e caracteres. Minha experiência e conselho: Use só letras!

Inserindo comentários nas CSS

Você pode inserir comentários nas CSS para explicar seu código, e principalmente ajudá-lo a lembrar de como você estruturou e qual a finalidade de partes importantes do código. Daqui há alguns meses a menos que você seja um privilegiado, terá esquecido a maior parte daquilo que você levou horas para "bolar". O comentário introduzido no código, será ignorado pelo browser. Um comentário nas CSS começa com o "/*", e termina com "*/". Veja o exemplo abaixo:

```
/* este é um comentário*/  
p {  
  font-size: 14px;          /* este é outro comentário*/  
  color: #000000;  
  font-family: Arial, Serif;  
}
```

:: Vinculando folhas de estilo a documentos ::

Os três tipos de vinculação de folhas de estilo

As folhas de estilo podem ser vinculadas a um documento de três maneiras distintas:

1. Importadas ou lincadas;
2. Incorporadas;
3. Inline.

Folha de estilo externa

Uma folha de estilo é dita externa, quando as regras CSS estão declaradas em um documento a parte do documento HTML. A folha de estilo é um arquivo separado do arquivo html e que tem a extensão .css

Uma folha de estilo externa é ideal para ser aplicada a várias páginas. Com uma folha de estilo externa , você pode mudar a aparência de um site inteiro mudando um arquivo apenas (o arquivo da folha de estilo).

O arquivo css da folha de estilo externa deverá ser lincado ou importado ao documento HTML, dentro da tag <head> do documento. A sintaxe geral para lincar uma folha de estilo chamada estilo.css é mostrada abaixo.

```
<head>
.....
<link rel="stylesheet" type="text/css" href="estilo.css">
.....
</head>
```

A sintaxe geral para importar uma folha de estilo chamada estilo.css é mostrada abaixo:

```
<head>
.....
<style type="text/css">
@import url("estilo.css");
</style>
.....
</head>
```

O browser lerá as regras de estilo do arquivo estilo.css, e formatará o documento de acordo com elas.

Uma folha de estilo externa pode ser escrita em qualquer editor de texto. O arquivo não deve conter nenhuma tag HTML. As folhas de estilo devem ser "salvas" com uma extensão .css

Folha de estilo incorporada ou interna

Uma folha de estilo é dita incorporada ou interna, quando as regras CSS estão declaradas no próprio documento HTML.

Uma folha de estilo incorporada ou interna, é ideal para ser aplicada a uma única página. Com uma folha de estilo incorporada ou interna, você pode mudar a aparência de somente um documento, aquele onde a folha de estilo esta incorporada.

As regras de estilo, válidas para o documento, são declaradas na seção <head> do documento com a tag de estilo <style>, conforme sintaxe mostrada abaixo:

```
<head>
.....
<style type="text/css">
<!--
body {
background: #000000;
url("imagens/minhaimagem.gif");
}
h3 {
color: #FF0000;
}
p {
margin-left: 15px;
padding:1.5em;
}
-->
</style>
.....
</head>
```

O browser lerá agora as regras de estilo na própria página, e formatará o documento de acordo com elas.

Nota: Um browser ignora normalmente as tags desconhecidas. Isto significa que um browser velho que não suporte estilos, ignorará a tag <style>, mas o conteúdo da tag será mostrado na tela. É possível impedir que um browser velho mostre o conteúdo da tag, "escondendo-o" através do uso da marcação de comentário do HTML.

Observe a inclusão dos símbolos <!-- (abre comentário) --> (fecha comentário) no código acima.

Folha de estilo inline

Uma folha de estilo é dita inline, quando as regras CSS estão declaradas dentro da tag do elemento HTML.

Um estilo inline só se aplica a um elemento HTML. Ele perde muitas das vantagens de folhas de estilo pois mistura o conteúdo com a apresentação. Use este método excepcionalmente, como quando quiser aplicar um estilo a uma única ocorrência de um elemento.

A sintaxe para aplicar estilo inline é mostrada a seguir:

```
<p style="color:#000000; margin: 5px;">
Aqui um parágrafo de cor preta e com 5px nas 4 margens.
</p>
```

Folhas múltiplas de estilo

Se alguma propriedade for definida para um mesmo elemento em folhas de estilo diferentes, entrará em ação, o EFEITO CASCATA e prevalecerão os valores da folha de estilo mais específica.

Suponha, uma folha de estilo externa com as seguintes propriedades para o seletor h2:

```
h2 {
color: #FFCC00;
text-align: center;
font: italic 9pt Verdana, Sans-serif;
}
```

e, uma folha de estilo interna com as seguintes propriedades para o seletor h2:

```
h2 {
color: #FFCC00;
text-align: center;
font: italic 10pt Verdana, Sans-serif;
}
```

Se ambas as páginas estiverem vinculadas ao documento, como há um conflito no tamanho das letras para <h2>, prevalecerá a folha interna e a letra de <h2> terá o tamanho igual a 10 pt.

:: A propriedade font ::

As fontes nos elementos HTML

As propriedades para as fontes, definem as características (os valores na regra CSS) das letras que constituem os textos dentro dos elementos HTML.

As propriedades básicas para fontes e que abordaremos neste tutorial são as listadas abaixo:

- color:.....cor da fonte
- font-family:.....tipo de fonte
- font-size:.....tamanho de fonte
- font-style:.....estilo de fonte
- font-variant:.....fontes maiúsculas de menor altura
- font-weight:.....quanto mais escura a fonte é (negrito)
- font:.....maneira abreviada para todas as propriedades

Valores válidos para as propriedades da fonte

- **color:**
 1. código hexadecimal: #FFFFFF
 2. código rgb: rgb(255,235,0)
 3. nome da cor: red, blue, green...etc
- **font-family:**
 1. family-name: define-se pelo nome da fonte, p. ex:"verdana", "helvetica", "arial", etc.
 2. generic-family: define-se pelo nome genérico, p. ex:"serif", "sans-serif", "cursive", etc.
- **font-size:**
 1. xx-small
 2. x-small
 3. small
 4. medium
 5. large
 6. x-large
 7. xx-large
 8. smaller
 9. larger
 10. length: uma medida reconhecida pelas CSS (px, pt, em, cm, ...)
 11. %
- **font-style:**
 1. normal: fonte normal na vertical
 2. italic: fonte inclinada
 3. oblique:fonte oblíqua
- **font-variant:**
 1. normal: fonte normal
 2. small-caps: transforma em maiúsculas de menor altura

- **font-weight:**
 1. normal
 2. bold
 3. bolder
 4. lighter
 5. 100
 6. 200
 7. 300
 8. 400
 9. 500
 10. 600
 11. 700
 12. 800
 13. 900

Vamos a seguir analisar cada uma delas detalhadamente através de exemplos práticos.

Como estudar e entender os exemplos

Para cada propriedade apresento as regras CSS para um ou mais elementos HTML e definidas dentro de uma folha de estilos incorporada, bem como um trecho do documento HTML onde se aplicam as regras.

A seguir mostro o efeito que a regra produz. Observe a regra e o efeito e para melhor fixar seu aprendizado reproduza o código no seu editor, mude os valores e veja o resultado no browser. Esta é a melhor e mais rápida maneira de você aprender CSS. Bons estudos! E faça ótimo proveito dos tutoriais.

color ... A cor da fonte

```
<html>
<head>
<style type="text/css">
<!--
h1 {color: #FF0000;}
h2 {color: #00FF00;}
p {color: rgb(0,0,255);}
-->
</style>
</head>
<body>
<h1>Cabeçalho com letras vermelhas</h1>
<h2>Cabeçalho com letras verdes</h2>
<p>Parágrafo com letras azuis</p>
</body>
</html>
```

Este é o efeito da folha de estilo acima:

Cabeçalho com letras vermelhas

Cabeçalho com letras verdes

Parágrafo com letras azuis

font-family...O tipo de fonte

```
<html>
<head>
<style type="text/css">
<!--
h2 {font-family: arial, helvetica, serif;}
p {font-family: sans-serif;}
-->
</style>
</head>
<body>
<h2>Família por nome: arial, helvetica, serif;</h2>
<p>Família genérica: sans-serif;<p>
</body>
</html>
```

Este é o efeito da folha de estilo acima:

Família por nome: arial, helvetica, serif;

Família genérica: sans-serif;

Notas: A propriedade font-family é usada para definir uma lista de tipos de fontes.

O browser assume o primeiro nome que ele reconhece na lista.

Separar cada nome por uma vírgula e sempre prever um nome genérico.

Se o nome da fonte for composto (mais de uma palavra, p. ex: Comic Sans MS), usar aspas duplas no nome. Se estiver definindo um atributo de "style" em HTML, onde as aspas duplas já estão presentes usar no nome de fonte composto, aspas simples.

font-size...O tamanho de fonte

```
<html>
<head>
<style type="text/css">
<!--
h1 {font-size: 14px;}
h2 {font-size: smaller;}
p {font-size: 100%;}
-->
</style>
</head>
<body>
<h1>Letras com tamanho: 14px</h1>
<h2>Letras com tamanho: smaller</h2>
<p>Letras com tamanho:100%</p>
```

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

Este é o efeito da folha de estilo acima:

Letras com tamanho: 14px

Letras com tamanho: smaller

Letras com tamanho:100%

font-style...O estilo de fonte

```
<html>
<head>
<style type="text/css">
<!--
h1 {font-style: italic;}
h2 {font-style: normal;}
p {font-style: oblique;}
-->
</style>
</head>
<body>
<h1>Este é o estilo italic</h1>
<h2>Este é o estilo normal</h2>
<p>Este é o estilo oblique</p>
</body>
</html>
```

Este é o efeito da folha de estilo acima:

Este é o estilo italic

Este é o estilo normal

Este é o estilo oblique

font-variant...fontes maiúsculas "menores"

```
<html>
<head>
<style type="text/css">
<!--
h3 {font-variant: normal;}
p{font-variant: small-caps;}
-->
</style>
</head>
<body>
<h3>Este cabeçalho com letras normais</h3>
<p>Este parágrafo com letras em "small-caps"</p>
</body>
</html>
```

Este é o efeito da folha de estilo acima:

Este cabeçalho com letras normais

ESTE PARÁGRAFO COM LETRAS EM "SMALL-CAPS"

font-weight...Peso das fontes - negrito (intensidade da cor)

```
<html>
<head>
<style type="text/css">
<!--
p {font-weight: bold;}
-->
</style>
</head>
<body>
<p>
Este é um parágrafo em negrito</p>
</body>
</html>
```

Este é o efeito da folha de estilo acima:

Este é um parágrafo em negrito

font...Todas as propriedades das fontes em uma declaração única

Esta é a maneira abreviada de você escrever uma regra para as propriedades das fontes.

A sintaxe geral é esta: font: [style] [variant] [weight] [**size**] [/line-height] [**family**] | caption | icon | menu | message-box | small-caption | status-bar | inherit

Você pode declarar todas ou algumas das propriedades.

Os valores `size` e `family` são obrigatórios. Os demais são facultativos (se você os omitir será adotado o valor default ou herdado do elemento pai).

Os valores `style`, `variant`, `weight` e `size`, podem ser declarados em qualquer ordem.

```
<html>
<head>
<style type="text/css">
<!--
p {
font: italic small-caps bold 14px
"Comic Sans MS", sans-serif;
```

```

}
-->
</style>
</head>
<body>
<p>Parágrafo em declaração única</p>
</body>
</html>

```

Este é o efeito da folha de estilo acima:

PARÁGRAFO EM DECLARAÇÃO ÚNICA

O valores `caption`, `icon`, `menu`, `message-box`, `small-caption` e `status-bar` referem-se às fontes usadas pelo sistema operacional do visitante do site e devem ser declarados **isolados** na propriedade `font`.

- `caption`.....fontes usadas em botões;
- `icon`.....fontes usadas em ícones;
- `menu`.....fontes usadas em menus;
- `message-box`...fontes usadas em caixas de mensagens;
- `small-caption`...fontes usadas em pequenos controles;
- `status-bar`.....fontes usadas na barra de status;

O valor `inherit` é usado para herdar a fonte usada pelo elemento pai e também deve ser declarados **isolados** na propriedade `font`.

```

<html>
<head>
<style type="text/css">
<!--
.p1 {
font: status-bar;
}
.p2 {
font: inherit;
.p3 {
font: small-caption ;
}
}
-->
</style>
</head>
<body>
<p class="p1">Parágrafo com fonte status-bar</p>
<p class="p2">Parágrafo com fonte inherit</p>
<p class="p3">Parágrafo com fonte small-caption</p>
</body>
</html>

```

Este é o efeito da folha de estilo acima:

Parágrafo com fonte status-bar

Parágrafo com fonte inherit

Parágrafo com fonte small-caption

Você poder fazer uso de um excelente editor para a propriedade font e descobrir mais efeitos para complementar este tutorial,

:: A propriedade text ::

Os textos nos elementos HTML

As propriedades para textos, definem as características (os valores na regra CSS) dos textos inseridos dentro dos elementos HTML.

As propriedades para textos são as listadas abaixo:

- color.....cor do texto;
- letter-spacing.....espaçamento entre letras;
- word-spacing.....espaçamento entre palavras;
- text-align.....alinhamento do texto;
- text-decoration.....decoração do texto;
- text-indent.....recuo do texto;
- text-transform.....forma das letras;
- direction.....direção do texto;
- white-space.....como o browser trata os espaços em branco;

Valores válidos para as propriedades do texto

- **color:**
 1. código hexadecimal: #FFFFFF
 2. código rgb: rgb(255,235,0)
 3. nome da cor: red, blue, green...etc
- **letter-spacing:**
 1. normal: é o espaçamento default
 2. length: uma medida reconhecida pelas CSS (px, pt, em, cm, ...) São válidos valores negativos
- **word-spacing:**
 1. normal: é o espaçamento default
 2. length: uma medida reconhecida pelas CSS (px, pt, em, cm, ...) São válidos valores negativos
- **text-align:**
 1. left: alinha o texto a esquerda
 2. right: alinha o texto a direita
 3. center: alinha o texto no centro
 4. justify: força o texto a ocupar toda a extensão da linha da esquerda a direita
- **text-decoration:**
 1. none: nenhuma decoração
 2. underline: coloca sublinhado no texto
 3. overline: coloca um sobrelinhado no texto
 4. line-through: coloca uma linha em cima do texto
 5. blink: faz o texto piscar
- **text-indent:**
 1. length: uma medida reconhecida pelas CSS (px, pt, em, cm, ...)
 2. % : porcentagem da largura do elemento pai

- **text-transform:**
 1. none: texto normal
 2. capitalize: todas as primeiras letras do texto em maiúsculas
 3. uppercase: todas as letras do texto em maiúsculas
 4. lowercase: todas as letras do texto em minúsculas
- **direction:**
 1. ltr: texto escrito da esquerda para a direita
 2. rtl: texto escrito da direita para a esquerda
- **white-space:**
 1. normal: os espaços em branco serão ignorados pelo browser
 2. pre: os espaços em branco serão preservados pelo browser
 3. nowrap: o texto será apresentado todo ele numa linha única na tela. Não há quebra de linha até ser encontrada uma tag

Vamos a seguir analisar cada uma delas detalhadamente através de exemplos práticos.

Como estudar e entender os exemplos

Para cada propriedade apresento as regras CSS para um ou mais elementos HTML e definidas dentro de uma folha de estilos incorporada, bem como um trecho do documento HTML onde se aplicam as regras.

A seguir mostro o efeito que a regra produz. Observe a regra e o efeito e para melhor fixar seu aprendizado reproduza o código no seu editor, mude os valores e veja o resultado no browser. Esta é a melhor e mais rápida maneira de você aprender CSS. Bons estudos! E faça ótimo proveito dos tutoriais.

color ... A cor do texto

```
<html>
<head>
<style type="text/css">
<!--
h1 {color: #FF0000;}
h2 {color: #00FF00;}
p {color: rgb(0,0,255);}
-->
</style>
</head>
<body>
<h1>Este cabeçalho é vermelho</h1>
<h2>Este cabeçalho é verde</h2>
<p>Este parágrafo é azul</p>
</body>
</html>
```

Este é o efeito da folha de estilo acima:

Este cabeçalho é vermelho

Este cabeçalho é verde

Este parágrafo é azul

letter-spacing...O espaço entre letras

```
<html>
<head>
<style type="text/css">
<!--
h2 {letter-spacing: 1.2em;}
p {letter-spacing: 0.4cm;}
-->
</style>
</head>
<body>
<h2> Este é o cabeçalho</h2>
<p> Este é o parágrafo</p>
</body>
</html>
```

Este é o efeito da folha de estilo acima:

E s t e é o c a b e ç a l h o

E s t e é o p a r a g r á f o

word-spacing...O espaço entre palavras

```
<html>
<head>
<style type="text/css">
<!--
h2 {word-spacing: 1.8em;}
p {word-spacing: 80px;}
-->
</style>
</head>
<body>
<h2> Este é o cabeçalho</h2>
<p> Este é o parágrafo</p>
</body>
</html>
```

Este é o efeito da folha de estilo acima:

Este é o cabeçalho

Este é o parágrafo

text-align...Alinhar o texto

```
<html>
<head>
```

```

<style type="text/css">
<!--
h1 {text-align: left;}
h2 {text-align: center;}
h3 {text-align: right;}
p {text-align: justify;}
-->
</style>
</head>
<body>
<h1>Este é o cabeçalho 1</h1>
<h2>Este é o cabeçalho 2</h2>
<h3>Este é o cabeçalho 3</h3>
<p>Este é o parágrafo cujo texto ...</p>
</body>
</html>

```

Este é o efeito da folha de estilo acima:

Este é o cabeçalho 1

Este é o cabeçalho 2

Este é o cabeçalho 3

Este é o parágrafo cujo texto foi alongado para mais de duas linhas para que você possa visualizar o efeito de `text-align: justify` que força o texto a estender-se desde a direita até a esquerda.

text-decoration...Decoração do texto

```

<html>
<head>
<style type="text/css">
<!--
h1 {text-decoration: underline;}
h2 {text-decoration: line-through;}
h3 {text-decoration: overline;}
a {text-decoration: none;}
-->
</style>
</head>
<body>
<h1>Texto com sublinhado</h1>
<h2>Texto com linha em cima</h2>
<h3>Texto com sobrelinhado</h3>
<p>
<a href="http://www.maujor.com">
Este é um link sem sublinhado</a>
</p>
</body>

```

```
</html>
```

Este é o efeito da folha de estilo acima:

Texto com sublinhado

~~Texto com linha em cima~~

Texto com sobrelinhado

[Este é um link sem sublinhado](#)

text-indent...Recuo do texto

```
<html>
<head>
<style type="text/css">
<!--
h3 {text-indent: 80px;}
p {text-indent: 3em;}
-->
</style>
</head>
<body>
<h3>Texto com recuo de 80 pixel</h3>
<p>Texto com recuo de 3.0em</p>
</body>
</html>
```

Este é o efeito da folha de estilo acima:

Texto com recuo de 80 pixels

Texto com recuo de 3.0em

text-transform...Forma das letras do texto

```
<html>
<head>
<style type="text/css">
<!--
h1 {text-transform: none;}
h2 {text-transform: capitalize;}
h3 {text-transform: uppercase;}
h4 {text-transform: lowercase;}
-->
</style>
</head>
<body>
<h1>Texto com letras como digitadas</h1>
<h2>Texto com primeira letra das palavras, maiúsculas</h2>
<h3>Texto com todas letras, maiúsculas</h3>
```

```
<h4>Texto com letras minúsculas</h4>  
</body>  
</html>
```

Este é o efeito da folha de estilo acima:

Texto com letras como digitadas

Texto com primeira letra das palavras, maiúsculas

TEXTO COM TODAS LETRAS, MAIÚSCULAS

Texto com letras minúsculas

:: A propriedade `margin` ::

As margens nos elementos HTML

A propriedade para margens, define um valor para espessura das margens dos elementos HTML.

As propriedades para margens são as listadas abaixo:

- `margin-top`.....define a margem superior;
- `margin-right`.....define a margem direita;
- `margin-bottom`.....define a margem inferior;
- `margin-left`.....define a margem esquerda;
- `margin`.....maneira abreviada para todas as margens

Valores válidos para a propriedade `margin`

1. `auto`: valor default da margem
2. `length`: uma medida reconhecida pelas CSS (`px`, `pt`, `em`, `cm`, ...)
3. `%`: porcentagem da largura do elemento pai

Vamos a seguir analisar cada uma delas detalhadamente através de exemplos práticos.

Como estudar e entender os exemplos

Para cada propriedade apresento as regras CSS válidas para um elemento HTML, e, definidas dentro de uma folha de estilos incorporada, bem como um trecho do documento HTML onde se aplicam as regras.

A seguir mostro o efeito que a regra produz. Observe a regra e o efeito e para melhor fixar seu aprendizado reproduza o código no seu editor, mude os valores e veja o resultado no browser. Bons estudos! E faça ótimo proveito do tutorial.

Nota: Coloquei um fundo cinza mais escuro nos exemplos para facilitar a visualização.

`margin-top`...a margem superior

```
<html>
<head>
<style type="text/css">
<!--
p {margin-top: 2cm;}
-->
</style>
</head>
<body>
<p>Uma margem superior de 2cm</p>
</body>
</html>
```

Este é o efeito da folha de estilo acima:

Uma margem superior de 2cm

margin-right...a margem direita

```
<html>
<head>
<style type="text/css">
<!--
p {margin-right: 300px;}
-->
</style>
</head>
<body>
<p>Uma margem direita de 300px nesta
frase mais longa dentro do parágrafo</p>
</body>
</html>
```

Este é o efeito da folha de estilo acima:

Uma margem direita de 300px nesta frase mais longa dentro do parágrafo

margin-bottom...a margem inferior

```
<html>
<head>
<style type="text/css">
<!--
p {margin-bottom: 2em;}
-->
</style>
</head>
<body>
<p>Uma margem inferior de 2.0em</p>
</body>
</html>
```

Este é o efeito da folha de estilo acima:

Uma margem inferior de 2.0em

margin-left...a margem esquerda

```
<html>
<head>
<style type="text/css">
<!--
p {margin-left: 10%;}
-->
</style>
```

```
</head>
<body>
<p>Uma margem esquerda de 10%</p>
</body>
</html>
```

Este é o efeito da folha de estilo acima:

Uma margem esquerda de 10%

margin...todas as quatro margens em uma declaração única

A propriedade da margin permitem que você controle o espaçamento em volta dos elementos HTML. São válidos **valores negativos** para margem, com o objetivo de sobrepor elementos.

Em declaração única a ordem das margens é: **superior, direita, inferior e esquerda**.

Há quatro modos de se declarar abreviadamente as margens:

1. margin: valor1.....as 4 margens terão valor1;
2. margin: valor1 valor2.....margem superior e inferior terão valor1 - margem direita e esquerda terão valor2
3. margin: valor1 valor2 valor3.....margem superior terá valor1 - margem direita e esquerda terão valor2 - margem inferior terá valor3
4. margin: valor1 valor2 valor3 valor4....margens superior, direita, inferior e esquerda nesta ordem.

```
<html>
<head>
<style type="text/css">
<!--
p {margin: 20px 40px 80px 5px;}
-->
</style>
</head>
<body>
<p>Uma margem superior de 20px, uma margem direita de 40px,
uma margem inferior de 80px e uma margem esquerda de 5px</p>
</body>
</html>
```

Este é o efeito da folha de estilo acima:

Uma margem superior de 20px, uma margem direita de 40px, uma margem inferior de 80px e uma margem esquerda de 5px

:: A propriedade border ::

As bordas nos elementos HTML

As propriedades para as bordas, definem as características (os valores na regra CSS) das quatro bordas de um elemento HTML.

As propriedades para as bordas são as listadas abaixo:

- border-width:.....espessura da borda
- border-style:.....estilo da borda
- border-color:.....cor da borda
- -----
- border-top-width:.....espessura da borda superior
- border-top-style:.....estilo da borda superior
- border-top-color:.....cor da borda superior
- -----
- border-right-width:.....espessura da borda direita
- border-right-style:.....estilo da borda direita
- border-right-color:.....cor da borda direita
- -----
- border-bottom-width:.....espessura da borda inferior
- border-bottom-style:.....estilo da borda inferior
- border-bottom-color:.....cor da borda inferior
- -----
- border-left-width:.....espessura da borda esquerda
- border-left-style:.....estilo da borda esquerda
- border-left-color:.....cor da borda esquerda
- -----
- border-top:...**maneira abreviada para todas as propriedades da borda superior**
- border-right:..**maneira abreviada para todas as propriedades da borda direita**
- border-bottom:..**maneira abreviada para todas as propriedades da borda inferior**
- border-left:..**maneira abreviada para todas as propriedades da borda esquerda**
- border:.....**maneira abreviada para todas as quatro bordas**

Valores válidos para as propriedades das bordas

- **color:**
 1. código hexadecimal: #FFFFFF
 2. código rgb: rgb(255,235,0)
 3. nome da cor: red, blue, green...etc
- **style:**
 1. none: nenhuma borda
 2. hidden: equivalente a none

3. dotted: borda pontilhada
 4. dashed: borda tracejada
 5. solid: borda contínua
 6. double: borda dupla
 7. groove: borda entalhada
 8. ridge: borda em relevo
 9. inset: borda em baixo relevo
 10. outset: borda em alto relevo
- **width:**
 1. thin: borda fina
 2. medium: borda média
 3. thick: borda grossa
 4. length: uma medida reconhecida pelas CSS (px, pt, em, cm, ...)

Vamos a seguir analisar algumas delas detalhadamente através de exemplos práticos.

Como estudar e entender os exemplos

Para cada propriedade apresento as regras CSS para um ou mais elementos HTML e definidas dentro de uma folha de estilos incorporada, bem como um trecho do documento HTML onde se aplicam as regras.

A seguir mostro o efeito que a regra produz. Observe a regra e o efeito e para melhor fixar seu aprendizado reproduza o código no seu editor, mude os valores e veja o resultado no browser. Bons estudos! E faça ótimo proveito dos tutoriais.

border-width, border-style e border-color

```
<html>
<head>
<style type="text/css">
<!--
h3 {
border-width: medium;
border-style: solid;
border-color: #0000FF;
}
p {
border-width: 6px;
border-style: dashed;
border-color: #FF0000;
}
-->
</style>
</head>
<body>
<h3>Borda média, contínua e azul</h3>
<p>Borda 6px, tracejada e vermelha</p>
</body>
</html>
```

Este é o efeito da folha de estilo acima:

Borda média, contínua e azul

Borda 6px, tracejada e vermelha

Nota: A propriedade `border-color` não é reconhecida pelo Internet Explorer se for usada isolada. Use a propriedade `border-style` para ser reconhecida pelo Internet Explorer.

Nota: A propriedade `border-color` não é reconhecida pelo Netscape. Use a propriedade `border` para ser reconhecida pelo Netscape.

border-style

Abaixo os estilos de bordas obtidos com a declaração `border-style: valor (dotted, dashed, etc..)`

Borda dotted

Borda dashed

Borda solid

Borda double

Borda groove

Borda ridge

Borda inset

Borda outset

border-width

Estude o código abaixo e tire suas conclusões de como obter outros efeitos com espessuras de bordas

```
<html>
<head>
<style type="text/css">
p {
border-style: solid;
border-bottom-width: 10px;
border-top-width: 0px;
```

```
border-right-width: 0px;
border-left-width: 0px;
}
</style>
</head>
<body>
<p>Borda com espessura inferior de 10px</p>
</body>
</html>
```

Este é o efeito da folha de estilo acima:

Borda com espessura inferior de 10px

Nota: A propriedade `border-bottom-width` não é reconhecida pelo Internet Explorer se usada isoladamente. Use `border-style` para ser reconhecida pelo Internet Explorer.

Definir a espessura das bordas superior, esquerda e direita

Proceda de modo semelhante ao mostrado acima.

border (declaração única)

Esta é a maneira abreviada de você escrever uma regra para as propriedades das bordas.

Você pode declarar todas as tres propriedadesdas bordas em uma regra única:

A sintaxe geral é esta: `border: size style color;` em qualquer ordem.

Nota: Aconselho a escolher, e adotar, sempre a mesma ordem.

```
<html>
<head>
<style type="text/css">
<!--
p    {
border: thick groove rgb(255,0,255)
    }
</style>
</head>
<body>
<p>Bordas em declaração única</p>
</body>
</html>
```

Este é o efeito da folha de estilo acima:

Bordas em declaração única

Propriedades CSS das bordas

As propriedades das bordas permitem que você controle o estilo a cor e a espessura das bordas de um elemento HTML.

As propriedades são muitas e como você viu, podem ser declaradas para cada uma das quatro bordas individualmente.

Neste tutorial abordei sumariamente algumas das propriedades, fornecendo as bases para seus estudos mais completos.

:: A propriedade padding ::

Os espaçamentos nos elementos HTML

A propriedade para espaçamentos (alguns traduzem como "enchimento"), define um valor para os espaçamentos entre o conteúdo e as bordas dos elementos HTML.

As propriedades para espaçamentos são as listadas abaixo:

- padding-top.....define a espaçamento superior;
- padding-right.....define a espaçamento direita;
- padding-bottom.....define a espaçamento inferior;
- padding-left.....define a espaçamento esquerda;
- padding.....**maneira abreviada para todas os espaçamentos**

Valores válidos para as propriedades de espaçamento

1. auto: valor default da margem
2. length: uma medida reconhecida pelas CSS (px, pt, em, cm, ...)
3. %: porcentagem da largura do elemento pai

Vamos a seguir analisar cada uma delas detalhadamente através de exemplos práticos.

Como estudar e entender os exemplos

Para cada propriedade apresento as regras CSS para um elemento HTML e definidas dentro de uma folha de estilos incorporada, bem como um trecho do documento HTML onde se aplicam as regras.

A seguir mostro o efeito que a regra produz. Observe a regra e o efeito e para melhor fixar seu aprendizado reproduza o código no seu editor, mude os valores e veja o resultado no browser. Bons estudos! E faça ótimo proveito dos tutorial.

Nota: Coloquei um fundo cinza mais escuro nos exemplos para facilitar a visualização.

padding-top...o espaçamento superior

```
<html>
<head>
<style type="text/css">
<!--
p {padding-top: 2cm;}
-->
</style>
</head>
<body>
<p>Um espaçamento superior de 2cm</p>
</body>
</html>
```

Este é o efeito da folha de estilo acima:

Um espaçamento superior de 2cm

padding-right...o espaçamento direito

```
<html>
<head>
<style type="text/css">
<!--
p {padding-right: 300px;}
-->
</style>
</head>
<body>
<p>Um espaçamento direito de 300px nesta
frase mais longa dentro do parágrafo</p>
</body>
</html>
```

Este é o efeito da folha de estilo acima:

Um espaçamento a direita de 300px nesta frase mais longa dentro do parágrafo

padding-bottom...o espaçamento inferior

```
<html>
<head>
<style type="text/css">
<!--
p {padding-bottom: 2em;}
-->
</style>
</head>
<body>
<p>Um espaçamento inferior de 2.0em</p>
</body>
</html>
```

Este é o efeito da folha de estilo acima:

Um espaçamento inferior de 2.0em

padding-left...o espaçamento esquerdo

```
<html>
<head>
<style type="text/css">
<!--
p {padding-left: 10%;}
-->
</style>
</head>
<body>
<p>Um espaçamento esquerdo de 10%</p>
```

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

Este é o efeito da folha de estilo acima:

Um espaçamento esquerdo de 10%

padding...todos os quatro espaçamentos em uma declaração única

A propriedade padding permite que você controle o espaçamento entre o conteúdo e as bordas dos elementos HTML.

Não são válidos valores negativos para espaçamento.

Em declaração única a ordem das espaçamentos é: **superior, direito, inferior e esquerdo**.

Há quatro modos de se declarar abreviadamente os espaçamentos:

1. padding: valor1.....os 4 espaçamentos terão valor1;
2. padding: valor1 valor2.....espaçamento superior e inferior terão valor1 - espaçamento direito e esquerdo terão valor2
3. padding: valor1 valor2 valor3.....espaçamento superior terá valor1 - espaçamento direito e esquerdo terão valor2 - espaçamento inferior terá valor3
4. padding: valor1 valor2 valor3 valor4....os espaçamentos superior, direito, inferior e esquerdo nesta ordem.

```
<html>
<head>
<style type="text/css">
<!--
p {padding: 20px 40px 80px 5px;}
-->
</style>
</head>
<body>
<p>Um espaçamento superior de 20px,
um espaçamento direito de 40px,
um espaçamento inferior de 80px
e um espaçamento esquerdo de 5px</p>
</body>
</html>
```

Este é o efeito da folha de estilo acima:

Um espaçamento superior de 20px, um espaçamento direito de 40px, um espaçamento inferior de 80px e um espaçamento esquerdo de 5px

:: A propriedade background ::

O fundo dos elementos HTML

A propriedade background define as características (os valores na regra CSS) do fundo dos elementos HTML.

As propriedades background são as listadas abaixo:

- background-color..... cor do fundo;
- background-image..... imagem de fundo;
- background-repeat..... maneira como a imagem de fundo é posicionada;
- background-attachment.....se a imagem de fundo "rola" ou não com a tela;
- background-position.....como e onde a imagem de fundo é posicionada;
- background.....**maneira abreviada para todas as propriedades;**

Valores válidos para as propriedades do fundo

- **background-color:**
 1. código hexadecimal: #FFFFFF
 2. código rgb: rgb(255,235,0)
 3. nome da cor: red, blue, green...etc
 4. transparente: transparent
- **background-image:**
 1. URL: url(caminho/imagem.gif)
- **background-repeat:**
 1. não repete: no-repeat
 2. repete vertical e horizontal: repeat
 3. repete vertical: repeat-y
 4. repete horizontal: repeat-x
- **background-attachment:**
 1. imagem fixa na tela: fixed
 2. imagem "rola" com a tela: scroll
- **background-position:**
 1. x-pos y-pos
 2. x-% y-%
 3. top left
 4. top center
 5. top right
 6. center left
 7. center center
 8. center right
 9. bottom left
 10. bottom center
 11. bottom right

Vamos a seguir analisar cada uma delas detalhadamente através de exemplos práticos.

Como estudar e entender os exemplos

Para cada propriedade apresento as regras CSS para um ou mais elementos HTML e definidas dentro de uma folha de estilos incorporada, bem como um trecho do documento HTML onde se aplicam as regras.

A seguir mostro o efeito que a regra produz. Observe a regra e o efeito e para melhor fixar seu aprendizado reproduza o código no seu editor, mude os valores e veja o resultado no browser. Esta é a melhor e mais rápida maneira de você aprender CSS. Bons estudos! E faça ótimo proveito dos tutoriais.

A cor do fundo

```
<html>
<head>
<style type="text/css">
<!--
body {background-color: #FFFFCC;} /*azul claro*/
h2 {background-color: #FF0000;} /* vermelho */
p {background-color: #00FF00;} /* verde */
-->
</style>
</head>
<body>
<h2>Estude CSS</h2>
<p>Com CSS você controla melhor seu layout</p>
</body>
</html>
```

Este é o efeito da folha de estilo acima:

Estude CSS

Com CSS você controla melhor seu layout

A imagem de fundo

```
<html>
<head>
<style type="text/css">
<!--
body { background-image: url("/images/css.gif");}
-->
</style>
</head>
<body>
</body>
</html>
```

Este é o efeito da folha de estilo acima:

Repetir verticalmente a imagem de fundo

```
<html>
<head>
<style type="text/css">
<!--
body {
background-image: url("/images/css.gif");
background-repeat: repeat-y;
}
-->
</style>
</head>
<body>
</body>
</html>
```

Este é o efeito da folha de estilo acima:

Repetir horizontalmente a imagem de fundo

```
<html>
<head>
style type="text/css">
<!--
body {
background-image: url("/images/css.gif");
background-repeat: repeat-x;
}
-->
</style
</head>
<body>
</body>
</html>
```

Este é o efeito da folha de estilo acima:

Posicionar uma imagem de fundo

```
<html>
<head>
<style type="text/css">
<!--
body {
background-image: url("/images/css.gif");
background-repeat: no-repeat;
background-position: 200px 70px;
}
-->
</style>
</head>
<body>
</body>
</html>
```

Este é o efeito da folha de estilo acima: a imagem esta posicionada a 200 pixel da margem esquerda e 70 pixel da margem superior

Ajustar uma imagem de fundo fixa, que não "rola" com a tela.

```
<html>
<head>
<style type="text/css">
<!--
body {
background-image: url("/images/css.gif");
background-repeat: no-repeat;
background-attachment: fixed;
}
-->
</style>
</head>
<body>
</body>
</html>
```

[Este é o efeito](#) da aplicação das regras CSS acima em uma página web.

Todas as propriedades do fundo em uma declaração única

Esta é a maneira abreviada de você escrever uma regra para as propriedades do fundo.

Você pode declarar todas ou algumas das propriedades estudadas em uma regra única:

A sintaxe geral é esta:

```
background: color image repeat attachment position;
em qualquer ordem, podendo ser omitido um mais valores.
```

Veja o exemplo abaixo:

```
<html>
<head>>
<style type="text/css">/>
<!--
body {
background: #00FF00 url("css.gif")
no-repeat fixed 200px 70px;
}
-->
</style>
</head>
```

:: A propriedade `list` ::

Mudando o estilo das listas HTML

A propriedade `list` define as características (valores) das listas HTML.

As propriedades `list` são as listadas abaixo:

- `list-style-image`..... imagem como marcador da lista;
- `list-style-position`.....onde o marcador da lista é posicionado;
- `list-style-type`.....tipo do marcador da lista;
- `list-style`.....**maneira abreviada para todas as propriedades;**

Valores válidos para as propriedades do lista

- `list-style-image`:
 1. `none`
 2. URL: `url(caminho/marcador.gif)`
- `list-style-position`:
 1. `outside`: marcador fora do alinhamento do texto
 2. `inside`: marcador alinhado com texto
- `list-style-type`:
 1. `none`: sem marcador
 2. `disc`: círculo (bolinha cheia)
 3. `circle`: circunferência (bolinha vazia)
 4. `square`: quadrado cheio
 5. `decimal`: números 1, 2, 3, 4, ...
 6. `decimal-leading-zero`
 7. `lower-roman`: romano minúsculo i, ii, iii, iv, ...
 8. `upper-roman`: romano maiúsculo I, II, III, IV, ...
 9. `lower-alpha`: letra minúscula a, b, c, d, ...
 10. `upper-alpha`: letra maiúscula A, B, C, D, ...
 11. `lower-greek`
 12. `lower-latin`
 13. `upper-latin`
 14. `hebrew`
 15. `armenian`
 16. `georgian`
 17. `CJK-ideographic`
 18. `hiragana`
 19. `katakana`
 20. `hiragana-iroha`
 21. `katakana-iroha`

Os tipos de 11 a 20 são de uso específico e sem suporte total pelos navegadores atuais e não serão tratados neste tutorial.

Vamos a seguir analisar cada uma delas detalhadamente através de exemplos práticos.

Como estudar e entender os exemplos

Para cada propriedade apresento as regras CSS para o elemento lista HTML e definidas dentro de uma folha de estilos incorporada, bem como um trecho do documento HTML onde se aplicam as regras.

A seguir mostro o efeito que a regra produz. Observe a regra e o efeito e para melhor fixar seu aprendizado reproduza o código no seu editor, mude os valores e veja o resultado no browser. Bons estudos! E faça ótimo proveito dos tutorial.

list-style-image...imagem para marcadores de lista

Este exemplo demonstra como definir uma imagem de marcador de listas

```
<html>
<head>
<style type="text/css">
<!--
ul
{
list-style-image: url("seta.gif");
}
-->
</style>
</head>
<body>
<ul>
<li>Item um</li>
<li>Item dois</li>
<li>Item tres</li>
</ul>
</body>
</html>
```

A folha de estilo acima resultará nesta lista:

- Item um
- Item dois
- Item tres

list-style-position...posição dos marcadores de lista

Este exemplo demonstra como posicionar os marcadores de listas

```
html>
<head>
<style type="text/css">
<!--
ul.inside
{
list-style-position: inside;
}
-->
```

```

ul.outside
{
list-style-position: outside;
}
-->
</style>
</head>
<body>
<ul class="inside">
<li>Este texto destina-se a demonstrar o
valor: "inside" dos marcadores de listas.</li>
<li>E aqui continuamos com mais texto para
fixar o valor:"inside" dosmarcadores de listas.</li>
</ul>
<ul class="outside">
<li>Este texto destina-se a demonstrar o
valor: "outside" dos marcadores de listas.</li>
<li>E aqui continuamos com mais texto para
fixar o valor:"outside" dos marcadores de listas.</li>
</ul>
</body>
</html>

```

A folha de estilo acima resultará nesta lista:

- Este texto destina-se a demonstrar o valor: "inside" dos marcadores
- E aqui continuamos com mais texto para fixar o valor:"inside" dos marcadores de listas.

- Este texto destina-se a demonstrar o valor: "outside" dos marcadores
- E aqui continuamos com mais texto para fixar o valor:"outside" dos marcadores de listas.

list-style-type...os tipos de marcadores de lista

Definir os marcadores de listas não ordenadas

Este exemplo demonstra como definir os marcadores de listas não ordenadas.

```

<html>
<head>
<style type="text/css">
<!--
ul.none {
list-style-type: none;
}
ul.disc {
list-style-type: disc;
}
ul.circle {
list-style-type: circle;
}
ul.square {
list-style-type: square;
}
-->

```

```
</style>
</head>
<body>
<ul class="none">
<li>Item um</li>
<li>Item dois</li>
<li>Item tres</li>
</ul>
<ul class="disc">
<li>Item um</li>
<li>Item dois</li>
<li>Item tres</li>
</ul>
<ul class="circle">
<li>Item um</li>
<li>Item dois</li>
<li>Item tres</li>
</ul>
<ul class="square">
<li>Item um</li>
<li>Item dois</li>
<li>Item tres</li>
</ul>
</body>
</html>
```

Este é o efeito da folha de estilo acima:

- Item um
 - Item dois
 - Item tres
-
- Item um
 - Item dois
 - Item tres
-
- Item um
 - Item dois
 - Item tres
-
- Item um
 - Item dois
 - Item tres

Definir os marcadores de listas ordenadas

Este exemplo demonstra como definir os marcadores de listas não ordenadas.

```
<html>
<head>
<style type="text/css">
<!--
ol.decimal
{
list-style-type: decimal;
}
```

```

ol.lroman
{
list-style-type: lower-roman;
}
ol.uroman
{
list-style-type: upper-roman;
}
ol.lalpha
{
list-style-type: lower-alpha;
}
ol.ualpha
{
list-style-type: upper-alpha;
}
-->
</style>
</head>
<body>
<ol class="decimal">
<li>Item um</li>
<li>Item dois</li>
<li>Item tres</li>
</ol>
<ol class="lroman">
<li>Item um</li>
<li>Item dois</li>
<li>Item tres</li>
</ol>
<ol class="uroman">
<li>Item um</li>
<li>Item dois</li>
<li>Item tres</li>
</ol>
<ol class="lalpha">
<li>Item um</li>
<li>Item dois</li>
<li>Item tres</li>
</ol>
<ol class="ualpha">
<li>Item um</li>
<li>Item dois</li>
<li>Item tres</li>
</ol>
</body>
</html>

```

Este é o efeito da folha de estilo acima:

1. Item um
2. Item dois
3. Item tres

1. Item um
2. Item dois
3. Item tres

1. Item um

2. Item dois
3. Item tres

1. Item um
2. Item dois
3. Item tres

1. Item um
2. Item dois
3. Item tres

list-style...duas propriedades das listas em uma declaração única

Esta é a maneira abreviada de você escrever uma regra para as propriedades das listas.

Você pode declarar duas das propriedades estudadas em uma regra única:

A sintaxe geral é esta: `list-style: position; imagem` OU `list-style: position; type` podendo inverter a ordem.

Veja o exemplo abaixo:

```
<html>
<head>
<style type="text/css">
<!--
ul
{
list-style: inside url("seta.gif");
}
-->
</style>
</head>
<body>
<ul>
<li>Texto para demonstrar a propriedade
de declaração única para listas usando
CSS - Folhas de Estilo em Cascata;</li>
<li>Item dois;</li>
<li>Item tres.</li>
</ul>
</body>
</html>
```

A folha de estilo acima resultará nesta lista:

- Texto para demonstrar a propriedade de declaração única para listas usando CSS - Folhas de Estilo em Cascata;
- Item dois;
- Item tres.

:: Pseudo-elementos CSS ::

Sintaxe

São usados em CSS, para adicionar efeitos a um seletor, ou a parte de um seletor.

A sintaxe dos pseudo-elementos:

```
seletor:pseudo-elemento {propriedade: valor;}
```

As classes em CSS podem também ser usadas com pseudo-elementos.

Esta regra permite que você defina diferentes efeitos para pseudo-elementos localizados em diferentes lugares em uma mesma página.

```
seletor.classe:pseudo-elemento {propriedade: valor;}
```

O pseudo-elemento `first-letter`

O pseudo-elemento `first-letter` é usado para obter um efeito especial na **primeira letra** de um texto.

```
<html>
<head>
<style type="text/css">
p {
font-size: 12pt
}
p:first-letter {
font-size:300%;
}
</style>
</head>
<body>
<p>Este texto destina-se a demonstrar o
pseudo-elemento first-letter, bla...bla...bla...
bla... bla...bla...bla...bla...bla... bla...bla...
bla... bla...bla...bla...bla...bla... bla...bla...</p>
</body>
</html>
```

O código acima produzirá esse efeito

Este texto destina-se a demonstrar o pseudo-elemento `first-letter`, bla...bla...bla... bla... bla...bla...bla...bla...bla... bla...bla... bla... bla...bla...bla...bla...bla... bla...bla... bla... bla...bla...bla...bla...bla... bla...bla...

O pseudo-elemento `first-letter` somente pode ser usado com elementos de bloco.

Propriedades aplicáveis ao pseudo-elemento `first-letter`

- font - propriedades de letras
- color - propriedades de cores
- background - propriedades de fundo
- margin - propriedades de margens
- padding - propriedades de espaçamentos
- border - propriedades de bordas
- text-decoration
- vertical-align (somente para "float: none)
- text-transform
- line-height
- float
- clear

O pseudo-elemento `first-line`

O pseudo-elemento `first-line` é usado para obter um efeito especial na **primeira linha** de um texto.

```
<html>
<head>
<style type="text/css">
p {
font-size: 12pt
}
p:first-line {
color: #0000FF;
font-variant: small-caps;
}
</style>
</head>
<body>
<p>Um texto qualquer dentro
de um pseudo-elemento first-line,
para um efeito especial na primeira linha</p>
</body>
</html>
```

O código acima produzirá esse efeito

Um texto qualquer dentro de um pseudo-elemento `first-line`, para um efeito especial na primeira linha. Notar a mudança de cor e o tipo de letra `small-caps` na primeira linha.

No exemplo acima toda a primeira linha sofre o efeito da definição do pseudo-elemento. A "quebra" da linha depende do tamanho da janela do browser.

O pseudo-elemento `first-line` somente pode ser usado com elementos de bloco.

Propriedades aplicáveis ao pseudo-elemento `first-line`

- font - propriedades de letras
- color - propriedades de cores
- background - propriedades de fundo

- word-spacing - espaçamento entre palavras
- letter-spacing - espaçamento entre letras
- text-decoration
- vertical-align
- text-transform
- line-height
- clear

Pseudo-elementos em classes CSS

Pseudo-elementos podem ser combinados com classes CSS

```
<html>
<head>
<style type="text/css">
p.combinado:first-letter {
color: #FF0000;
font-size:xx-large;
}
</style>
</head>
<body>
<p class="combinado"> Uma frase com efeito
especial combinado </p>
</body>
</html>
```

O código acima produzirá esse efeito

Uma frase com efeito especial combinado

:: Controlando as entrelinhas e o espaçamento entre elementos HTML ::

As propriedades `line-height` e `margin`

A propriedade CSS de dimensionamento `line-height` permite controlar o espaçamento entre linhas e a propriedade CSS `margin` permite controlar o espaçamento entre elementos HTML.

Observe abaixo o código HTML para um texto composto de dois parágrafos:

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

<p>
1o. Parágrafo....Lorem ipsum dolor sit
amet, consectetur adipiscing elit.
Nulla pharetra egestas neque.
Duis dolor lacus, volutpat ac,
vestibulum nec, suscipit a, felis.
Aenean pharetra orci id elit.
Duis non dui. Suspendisse potenti.
Ut ac risus. Etiam dignissim.
Quisque nec felis.
</p>

<p>
2o.Parágrafo.....Sed blandit est non
ante. Ut imperdiet sagittis mi.
Sed gravida sodales nisl. Ut hendrerit
ipsum eu enim. Duis tempus consequat mauris.
In hac habitasse platea dictumst.
Vivamus lectus justo, commodo in, rutrum non,
eleifend eget, pede. Sed ac lacus. In tortor.
</p>

</body>
</html>
```

O código acima é renderizado pelo navegador conforme mostrado abaixo.

Notar a distância entre as linhas em cada parágrafo, ou seja as entrelinhas (não confunda com distância entre parágrafos):

1o. Parágrafo....Lorem ipsum dolor sit amet, consectetur adipiscing elit. Nulla pharetra egestas neque. Duis dolor lacus, volutpat ac, vestibulum nec, suscipit a, felis. Aenean pharetra orci id elit. Duis non dui. Suspendisse potenti. Ut ac risus. Etiam dignissim. Quisque nec felis.

2o.Parágrafo.....Sed blandit est non ante. Ut imperdiet sagittis mi. Sed gravida sodales nisl. Ut hendrerit ipsum eu enim. Duis tempus consequat mauris. In hac habitasse platea dictumst. Vivamus lectus justo, commodo in, rutrum non, eleifend eget, pede. Sed ac lacus. In tortor.

Alterando o espaçamento entre linhas

No código HTML mostrado acima vamos inserir uma regra CSS para `line-height` que é a propriedade CSS que controla as entrelinhas. Observe abaixo o mesmo código com a regra, definindo uma entrelinha igual a 200%.

Nota: A entrelinha default do browser é 100%.

Você pode usar qualquer medida de comprimento, válida em CSS (px, cm, em, %, in...) para o valor da propriedade `line-height`.

```
<html>
<head>
<style type="text/css">
<!--
p {
line-height:200%;
}
-->
</style>

</head>
<body>

<p>
1o. Parágrafo....Lorem ipsum dolor sit
amet, consectetur adipiscing elit.
Nulla pharetra egestas neque.
Duis dolor lacus, volutpat ac,
vestibulum nec, suscipit a, felis.
Aenean pharetra orci id elit.
Duis non dui. Suspendisse potenti.
Ut ac risus. Etiam dignissim.
Quisque nec felis.
</p>

<p>
2o.Parágrafo.....Sed blandit est non
ante. Ut imperdiet sagittis mi.
Sed gravida sodales nisl. Ut hendrerit
ipsum eu enim. Duis tempus consequat mauris.
In hac habitasse platea dictumst.
Vivamus lectus justo, commodo in, rutrum non,
eleifend eget, pede. Sed ac lacus. In tortor.
</p>

</body>
</html>
```

O código acima é renderizado pelo navegador conforme mostrado abaixo.

Notar que a entrelinha que era default 100%, agora está 200% ou seja dobrou:

Nota: Faça algumas experiências com o valor de `line-height`, usando inclusive valores abaixo de 100% e também outras medidas válidas (por exemplo: 12px, 2.3em, 3cm...etc...) e você vai constatar que tem o controle total das entrelinhas.

1o. Parágrafo....Lorem ipsum dolor sit amet, consectetur adipiscing elit. Nulla pharetra egestas neque. Duis dolor lacus, volutpat ac, vestibulum nec, suscipit a, felis. Aenean pharetra orci id elit. Duis non dui. Suspendisse potenti. Ut ac risus. Etiam dignissim. Quisque nec felis.

2o.Parágrafo.....Sed blandit est non ante. Ut imperdiet sagittis mi. Sed gravida sodales nisl. Ut hendrerit ipsum eu enim. Duis tempus consequat mauris. In hac habitasse platea dictumst. Vivamus lectus justo, commodo in, rutrum non, eleifend eget, pede. Sed ac lacus. In tortor.

E o espaçamento (a distância) entre os parágrafos?

Aqui também o controle é todo seu via CSS.

E quem dita as regras para este espaçamento é a propriedade `margin`.

Vamos acrescentar mais uma regra CSS no nosso código.

Se voce não lembra da propriedade `margin`, leia este [tutorial sobre margens](#)

```
<html>
<head>
<style type="text/css">
<!--
p {
line-height:200%;
margin: 40px 0 40px 0;
-->
</style>

</head>
<body>

<p>
1o. Parágrafo....Lorem ipsum dolor sit
amet, consectetur adipiscing elit.
Nulla pharetra egestas neque.
Duis dolor lacus, volutpat ac,
vestibulum nec, suscipit a, felis.
```

```
Aenean pharetra orci id elit.  
Duis non dui. Suspendisse potenti.  
Ut ac risus. Etiam dignissim.  
Quisque nec felis.  
</p>  
  
<p>  
2o.Parágrafo.....Sed blandit est non  
ante. Ut imperdiet sagittis mi.  
Sed gravida sodales nisl. Ut hendrerit  
ipsum eu enim. Duis tempus consequat mauris.  
In hac habitasse platea dictumst.  
Vivamus lectus justo, commodo in, rutrum non,  
eleifend eget, pede. Sed ac lacus. In tortor.  
</p>  
  
</body>  
</html>
```

O código acima é renderizado pelo navegador conforme mostrado abaixo.

Notar que a entrelinha continua em 200% e agora o espaçamento entre parágrafos cresceu para 40 pixels, cumprindo a regra CSS, escrita.

1o. Parágrafo....Lorem ipsum dolor sit amet, consectetur adipiscing elit. Nulla pharetra egestas neque. Duis dolor lacus, volutpat ac, vestibulum nec, suscipit a, felis. Aenean pharetra orci id elit. Duis non dui. Suspendisse potenti. Ut ac risus. Etiam dignissim. Quisque nec felis.

2o.Parágrafo.....Sed blandit est non ante. Ut imperdiet sagittis mi. Sed gravida sodales nisl. Ut hendrerit ipsum eu enim. Duis tempus consequat mauris. In hac habitasse platea dictumst. Vivamus lectus justo, commodo in, rutrum non, eleifend eget, pede. Sed ac lacus. In tortor.

Você deve ter notado que o espaçamento do 1o. parágrafo para a borda superior do quadro amarelo e também a do 2o. parágrafo para a borda inferior do quadro amarelo, ambas AUMENTARAM.

Sim, este aumento no espaçamento cumpriu o prescrito na nova regra, ou seja: 40 pixel de margem superior e 40 pixel de margem inferior nos parágrafos.

Mas lembre-se o controle é SEU. Tem como evitar este espaçamento não previsto :-)
Veja o item 1-) abaixo.

Dicas adicionais

1-) Para evitar aquele espaçamento referido acima, crie e aplique uma classe no parágrafo superior com `margin-top: 0;` (ou `n pixels`) e outra classe ao parágrafo inferior com `margin-bottom: 0;` (ou `n pixels`);

Você pode também declarar: `margin: 0 0 40px 0;` e suprimir o espaçamento superior, ou ainda `margin: 40px 0 0 0;` e suprimir o espaçamento inferior. E, uma série de outras combinações que ficam a título de exercícios para você.

2-) Se você deseja aplicar regras CSS em alguns elementos do documento e não em todos (por exemplo: alguns parágrafos na página seguirão uma regra `line-height` outros não) crie classes e aplique aos elementos.

: As medidas CSS de comprimento ::

Introdução

Unidades de medida de comprimento CSS

As unidades de medida de comprimento CSS referem-se a medidas na horizontal ou na vertical (e em sentido mais amplo, em qualquer direção).

O formato para declarar o valor de uma unidade de medida CSS é um número com ou sem ponto decimal **imediatamente precedido** do sinal '+' (mais) ou do sinal '-' (menos), sendo o sinal '+' (mais) o valor "default" e **imediatamente seguido** por uma unidade identificadora (medida CSS válida - p.ex., px, em, deg, etc...). A unidade identificadora é opcional quando se declara um valor '0' (zero).

Algumas das propriedades CSS permitem que sejam declarados valores negativos para unidades de medida. A adoção de valores negativos podem complicar a formatação do elemento e devem ser usados com cautela. Se valores negativos não forem suportados pela aplicação de usuário, eles serão convertidos para o valor mais próximo suportado (e isso pode tornar-se desastroso para um layout).

Unidades de medida de comprimento CSS válidas

São dois os tipos de unidade de medida de comprimento CSS:

UNIDADE RELATIVA

- em
- ex
- px - pixel
- % - percentagem

as unidades relativas são referenciadas a outras unidades como veremos a seguir.

UNIDADE ABSOLUTA

- **pt - point** :1/72 in;
- **pc - pica** :12 points ou 1/6 in;
- **mm - milímetro** :1/10 cm;
- **cm - centímetro** :1/100 m;
- **in - polegada** :2,54 cm;

Unidade relativa- é aquela tomada em relação a uma outra medida. Folhas de Estilo em Cascata que usam unidades de comprimento relativas são mais apropriadas para ajustes de uso em diferentes tipos de mídia. (p. ex., de uma tela de monitor para uma impressora laser).

O valor é tomado em relação:

- **em**: ...ao tamanho da fonte ('font-size') herdada;
- **ex**: ...a altura da letra x (xis) da fonte herdada;
- **px**: ...ao dispositivo (midia) de exibição;
- **%**: ... a uma medida previamente definida.

Unidade absoluta - é aquela que não esta referenciada a qualquer outra unidade e nem é herdada. São unidades de medida de comprimento definidas nos sistemas de medidas pela física e em fim são os conhecidos "centímetros, polegadas etc..."). São indicadas para serem usadas quando as mídias de exibição são perfeitamente conhecidas.

Abaixo exemplos ilustrativos do uso destas medidas de comprimento CSS:

```
div { margin: 1.5em; }
h4 { margin: 2ex; }
p { font-size: 14px; }
.classe { padding: 90%; }
hr { width: 14pt; }
h1 { margin: 1pc; }
h2 { font-size: 4mm; }
p.classe { padding: 0.3cm; }
h5.classe { padding: 0.5in; }
```

Nota: Relembro que uma regra CSS tem a seguinte sintaxe

```
seletor {propriedade: valor;}
```

Entendendo as unidades de medida CSS

Vamos a seguir definir e analisar cada uma das unidades de medida CSS e apresentar exemplos práticos.

A unidade de medida - pixel

A unidade de medida de comprimento pixel é relativa a resolução do dispositivo de exibição (p.ex: a tela de um monitor).

Sem entrar em maiores considerações teóricas a mais simplista definição de pixel que encontrei é esta:

Pixel é o menor elemento em um dispositivo de exibição, ao qual é possível atribuir-se uma cor.

Considere um dispositivo de exibição construído com uma densidade de 90 dpi (dpi = dots per inch = pontos por polegada). Por definição, a **referência padrão para pixel** é igual a um ponto no citado dispositivo. Daí pode-se concluir que **1 pixel** naquele dispositivo de exibição é igual a $1/90$ inch = 0,28 mm.

Para uma densidade de 300 dpi 1 pixel é igual a $1/300$ inch = 0,085mm

Assim, pixel é uma medida relativa a resolução do dispositivo de exibição.

A unidade de medida - em

A unidade de medida de comprimento **em** referencia-se ao tamanho da fonte (letra) do seletor onde for declarada. Quando **em** for declarada para a propriedade `font-size` referencia-se ao tamanho da fonte (letra) do elemento pai. Quando **em** for declarada para o elemento raiz do documento (p. ex: `<html>` em documentos html) referencia-se ao valor inicial (default) do tamanho de fonte (letra).

Os exemplos abaixo esclarecem as definições:

```
h1 { line-height: 1.2em }
```

line-height de `<h1>` será 20% maior do que o tamanho das letras de `<h1>`

```
h1 { font-size: 1.2em }
```

font-size de `<h1>` será 20% maior do que o tamanho das letras herdado por `<h1>` p.ex.: se h1 estiver contido numa div com `font-size=10px` então font-size de h1 = 12px

A unidade de medida - ex

A unidade de medida de comprimento **ex** é igual a altura da letra **x**(xis) minúscula).

A unidade de medida - percentagem, %

Valores em percentagem são relativos a um outro valor anterior declarado. Este valor anterior há que estar bem definido e em geral esta definição está em uma determinada propriedade do mesmo elemento, na propriedade do elemento "pai" (por exemplo: uma medida CSS de comprimento) ou mesmo no contexto geral da formatação (por exemplo: a largura do bloco de conteúdo).

```
p { font-size: 10px }  
p { line-height: 120% } /*120% de 'font-size'=12px*/
```

:: Definindo cores em uma regra CSS ::

Objetivo

Detalhar as diferentes maneiras de se escrever a sintaxe para os valores das cores em uma regra CSS

Valores válidos para cores em CSS

Observe as regras de estilo a seguir:

- 1-) `div.um {background-color: #FF0000;}`
- 2-) `div.dois {background-color: #F00;}`
- 3-) `div.tres {background-color: rgb(255, 0, 0);}`
- 4-) `div.quatro {background-color: rgb(100%, 0%, 0%);}`
- 5-) `div.cinco {background-color: red;}`
- 6-) `div.seis {background-color: ThreeDShadow;}`

Como você já deve ter concluído apresentei 06 (seis) maneiras diferentes de definir uma cor de fundo para uma DIV .

E, se considerarmos que para as duas primeiras regras é válido usar letras minúsculas, existem 08 (oito) maneiras de se definir uma cor em uma regra CSS.

As maneiras mais usadas são as mostradas em 1 e em 2, ou seja, com uso do código hexadecimal de cores.

O efeito das regras no navegador

Observe agora no screenshot a seguir como estas seis DIV's serão renderizadas.



As cinco primeiras estão com a mesma cor de fundo, vermelha o que nos leva a concluir que as cinco primeiras regras mostradas são equivalentes, ou seja são cinco maneiras diferentes de definir um mesmo valor para uma cor.

`#FF0000 = #F00 = rgb(255,0,0) = rgb(100%,0%,0%) = red`

A sexta cor, `ThreeDShadow` depende do equipamento do usuário.

Vejamos cada uma delas detalhadamente.

Definir uma cor pelo seu código hexadecimal

Esta é a maneira mais conhecida de definir uma cor.

Convém ressaltar que em uma regra CSS é indiferente usar letras maiúsculas ou minúsculas na sintaxe hexadecimal de cores e também que é válido abreviar a notação para três dígitos.

Na notação abreviada cada um dos três dígitos é automaticamente dobrado conforme exemplos a seguir:

`#FFF = #FFFFFF`

`#CF9 = #CCFF99`

`#cde = #ccddee`

`#49c = #4499cc`

Não é do escopo deste tutorial detalhar o código hexadecimal, contudo ressalto que os dezesseis dígitos hexadecimais são: 0,1,2,3,4,5,6,7,8,9,A,B,C,D,E,F e somente eles são válidos para definir uma cor, podendo em geral ser usada qualquer combinação deles. Assim: #FFDDHH não define uma cor, pois H não é válido.

Existem várias ferramentas online para determinar o código hexadecimal de uma cor. Uma das que eu costumo usar e indico para vocês é esta:
<http://www.colorschemer.com/online.html>

Definir uma cor pelo seu código rgb

rgb é abreviatura para:
r = red (vermelha)
g = green (verde)
b = blue (azul)

Assim o código `rgb(xxx, yyy, zzz)` indica uma cor obtida com a mistura de uma quantidade xxx de vermelho com yyy de verde e com zzz de azul.

Duas são as maneiras de se definir a quantidade de cada uma das três cores:

Uma faixa de numeração de 0 (zero) até 255
Em percentagem de 0% até 100%

Não é válido usar em uma definição número e percentagem.

Exemplos:

definições válidas
`rgb(145, 230, 50)` - `rgb(20%, 0%, 70%)`

definição não válida
`rgb(255, 20%, 120)`

Não é do escopo deste tutorial detalhar as misturas de cor rgb.

No link indicado no item anterior é possível determinar também, o código rgb de uma cor

Definir cor por palavra-chave

Você pode definir uma cor usando o nome da cor. Os nomes de cor válidos são os listados nas recomendações CSS do W3C.

As Recomendações para CSS 2.1 listam as seguintes 17 cores:

aqua, black, blue, fuchsia, gray, green, lime, maroon, navy, olive, orange, purple, red, silver, teal, white, e yellow



Assim, as regras a seguir são válidas para definir cor

```
p {color: aqua;}
div {background-color: teal;}
```

Definir cor baseado no sistema operacional do usuário

As recomendações para CSS2.1 preconizam a definição da cor baseado nas cores adotadas pelo sistema operacional do usuário.

Este tipo de unidade de definição de cor denominado System Colors está em desuso e não deverá constar das futuras Recomendações CSS3.

Trata-se de uma lista de nomes de cores válidas à semelhança da listagem de cores por palavra-chave e que se refere a áreas do sistema operacional.

As cores previstas são:

ActiveBorder, ActiveCaption, AppWorkspace, Background, ButtonFace, ButtonHighlight, ButtonShadow, ButtonText, CaptionText, GrayText, Highlight, HighlightText, InactiveBorder, InactiveCaption, InactiveCaptionText, InfoBackground, InfoText, Menu, MenuText, Scrollbar, ThreeDDarkShadow, ThreeDFace, ThreeDHighlight, ThreeDLightShadow, ThreeDShadow, Window, WindowFrame, WindowText

Embora os valores CSS sejam "case insensitive" recomenda-se usar a grafia com letras maiúsculas e minúsculas ao se escrever o nome das cores de sistema por razões de legibilidade.

Exemplos:

```
p {color: ThreeDLightShadow;}
div {background: ButtonShadow;}
```

Veja [NESTE LINK](#) as cores definidas por regras de estilo para o seu sistema operacional.

:: Abreviando declarações e valores em regras CSS ::

Relembrando a sintaxe e a terminologia de uma regra CSS

É comum encontrar-se em muitos artigos sobre CSS escritos em blogs e sites, em textos de posts em fóruns, em listas de discussão e até mesmo em revistas e jornais, diferentes referências e denominações equivocadas para os componentes de uma regra CSS

Seletores são chamados de elementos ou de tags, *propriedades* são chamadas de seletores ou de atributos, *valores* são chamados de atributos ou de propriedades, *declarações* são chamadas de regras ou funções CSS e por aí vai em uma diversificada combinação dos termos acima citados em uma salada que acaba por confundir iniciantes e as vezes até mesmo outros já com alguma experiência com folhas de estilo em cascata.

Com a finalidade de facilitar o entendimento desta matéria e esclarecer a confusão que vem se formando em torno do assunto, vamos rever a sintaxe e a terminologia de uma regra CSS para que quando eu escrever seletor, declaração, propriedade e valor, não haja dúvidas sobre a porção da regra CSS a que estou me referindo e você não fique se perguntando onde estão os "atributos CSS, as tags CSS, e outros tantos termos equivocados".

Vejam os que diz as [Recomendações do W3C para Folhas de Estilo, nível 1](#) na seção intitulada [Conceitos Básicos](#)

“O projeto, ou desenho do layout, das folhas de estilos é fácil. É preciso apenas conhecer um pouco da linguagem HTML e possuir noções básicas dos termos usados em publicação eletrônica. Como exemplo, para ajustar a cor das letras de um elemento 'H1' para azul, basta fazer:

```
H1 {color: blue}
```

Este exemplo mostra o que é uma 'regra' simples em CSS. Uma regra é composta de duas partes principais: um selector ('H1') e uma declaração ('color: blue'). Por sua vez, esta declaração também possui duas partes: uma propriedade ('color') e seu valor ('blue'). Embora este exemplo especifique apenas uma das várias propriedades necessárias para montar um documento HTML, ela constitui por si só uma 'folha de estilo'. Quando for combinada com outras folhas de estilo ela determinará a apresentação final do documento (uma característica fundamental é que as folhas de estilo podem ser combinadas).

O seletor funciona como a ponte de ligação entre o documento HTML e a folha de estilo, e todos os elementos HTML podem funcionar como possíveis seletores. Os vários elementos HTML estão definidos na Recomendação HTML

etc.”

A Recomendação do W3C define claramente que uma regra CSS é composta de um seletor e uma declaração e que a declaração compreende uma propriedade e um valor.

Na regra CSS a seguir:

```
H1 {color: blue}
```

a terminologia correta é:

- H1 - seletor;
- {color: blue} - declaração;
- color - propriedade;
- blue - valor.

e a sintaxe correta é:

- Escrever o seletor e a seguir a declaração;
- A declaração deve estar entre { } (chaves);
- Na declaração, separar a propriedade e o valor por : (dois pontos);
- É permitido usar espaços em branco em qualquer quantidade entre cada um dos caracteres da regra;
- É permitido agrupar declarações em uma mesma regra e neste caso as declarações deverão ser separadas por ; (ponto-e-vírgula) podendo todas elas estar em uma mesma linha ou em linhas distintas. É facultativo o uso de ; (ponto-e-vírgula) após a última declaração na regra;
- É indiferente o uso de maiúsculas e minúsculas em uma regra CSS, contudo as *classes* e *ID's* devem seguir a mesma grafia constante da marcação.

Estes são os termos normatizados de uma regra CSS e os que usaremos. Portanto, não existe "atributo CSS" ou "tag CSS" ou "elemento CSS" ou "função CSS" ou tantos outros equivocadamente escritos.

Não é do escopo deste tutorial detalhar as boas práticas de escrita das regras em uma folha de estilos.

Sobre este assunto escrevi e recomendo a leitura do tutorial [Dicas básicas para projetar folhas de estilos](#).

Abreviando valores de cores hexadecimais

O formato hexadecimal é uma das opções sintáticas mais usadas para se escrever o [valor das cores em regras CSS](#). A regra a seguir define que os parágrafos serão na cor vermelha (#ff0000).

```
p {color: #ff0000;}
```

e que poderá ser abreviada para:

```
p {color: #f00;}
```

É válido abreviar cores hexadecimais para 3 dígitos. Valores escritos com 3 dígitos são interpretados como se cada um dos dígitos tivesse sido declarado duas vezes, isto é:

genericamente, `#abc` é equivalente `#aabbcc`

Exemplos:

```
#c30 = #cc3300  
#999 = #999999  
#ff0 = #ffff00  
#d61 = #dd6611
```

É fácil concluir que a abreviação de cores hexadecimais somente é possível para as cores constituídas por 3 pares de dígitos hexadecimais.

Valores para os quatro lados de um elemento nível de bloco

Um elemento nível de bloco ou uma 'caixa' admite estilização em seus quatro lados para algumas propriedades como `border` e `padding` entre outras.

Por exemplo: você pode definir um `padding` superior, um `padding` à direita, um `padding` inferior e um `padding` à esquerda para uma `div`.

A sequência em que você escreve os valores para estilizar os quatro lados de uma 'caixa' é rígida e fixa em uma regra CSS e obedece a seguinte ordem:

em cima , lado direito, embaixo, lado esquerdo

Faça uma analogia com o relógio para não esquecer a sequência.

12 horas (superior), 3 horas (direita), 6 horas (inferior), 9 horas (esquerda).

A regra `div {padding: 2px 3px 8px 7px;}` define para a `div`:

um `padding` inferior igual a 8px;

um `padding` superior igual a 2px;

um `padding` à esquerda igual a 7px;

um `padding` à direita igual a 3px.

Além da mostrada acima é válido abreviar declarações que envolvem os quatro lados de uma 'caixa' de outras 3 maneiras diferentes como mostradas a seguir:

1. `div {padding: 10px;} padding` de 10px para os 4 lados;
2. `div {padding: 6px 8px;} padding` de 6px para os lados superior e inferior e de 8px para os lados direito e esquerdo;
3. `div {padding: 2px 4px 9px;} padding` de 2px para o lado superior, de 4px para os lados direito e esquerdo e de 9px para o lado inferior.

Propriedades que admitem abreviação

Veremos ao longo deste tutorial, como abreviar as seguintes propriedades CSS:

1. [margin](#);
2. [padding](#);
3. [background](#);
4. [font](#);
5. [list](#);
6. outline;
7. [border](#).

Abreviando `margin`

As regras a seguir definem valores para as 4 margens para uma div:

```
div {  
margin-top:10px;  
margin-right:8px;  
margin-bottom:0;  
margin-left:5px;  
}
```

E pode ser abreviada para:

```
div {  
margin:10px 8px 0 5px;  
}
```

Abreviando `padding`

As regras a seguir definem valores para os 4 paddings de um parágrafo:

```
p {  
padding-bottom:6px;  
padding-top:12px;  
padding-left:1px;  
padding-right:2px;  
}
```

E pode ser abreviada para:

```
div {  
padding:12px 2px 6px 1px;  
}
```

Abreviando `background`

As regras a seguir definem valores para propriedades background de uma div:

```
div {  
background-color:#ffffcc;  
background-image:url(fundo.gif);  
background-repeat:no-repeat;  
background-attachment:fixed;  
background-position:20px 10px;  
}
```

E pode ser abreviada para:

```
div {
background:#ffc url(fundo.gif) no-repeat fixed 20px 10px;
}
```

Abreviando `font`

As regras a seguir definem valores para propriedades de font em um documento:

```
body {
font-style:italic;
font-variant:small-caps;
font-weight:bold;
font-size:11px;
line-height:15px;
font-family:Arial, Helvetica, Sans-serif;
}
```

E pode ser abreviada para:

```
body {
font:italic small-caps bold 11px/15px Arial, Helvetica, Sans-serif;
}
```

Nota: Para abreviar a propriedade `font` é obrigatório definir no mínimo os valores de tamanho e família da `font`. Os demais valores são facultativos. A ordem de declaração dos valores é importante e deve ser assim:

1. começar com `font-style`, `font-variant` e `font-weight` sedo que estes três valores são facultativos e podem ser escritos em qualquer ordem;
2. a seguir declarar obrigatoriamente `font-size` e opcionalmente `line-height` (`font-size/line-height`);
3. finalmente declarar obrigatoriamente `font-family`.

Abreviando `list`

As regras a seguir definem valores para propriedades de listas:

```
ul {
list-style-type:square;
list-style-position:inside;
list-style-image:url(image.gif);
}
```

E pode ser abreviada para:

```
ul {list-style:square inside url(image.gif);}
```

A propriedade: `list-style-type` pode ser abreviada para **`list-style`**.

Por exemplo: `list-style-type:none` pode ser abreviada para **`list-style:none`**;

Abreviando `outline`

A propriedade `outline` é pouco conhecida e empregada. Serve para colocar uma margem ao redor de um elemento, com a finalidade de destacá-lo no contexto. Difere da propriedade `border` por não interferir com as dimensões do box model, isto é, não ocupa espaço no box do elemento e em consequência não afeta o posicionamento do box e nem dos boxes adjacentes.

As regras a seguir definem a marcação de um 'destaque' em linha vermelha sólida de 1px ao redor do elemento `h2`:

```
h2 {  
  outline-color:#f00;  
  outline-style:solid;  
  outline-width:1px;  
}
```

E pode ser abreviada para:

```
h2 {  
  outline:#f00 solid 1px;  
}
```

Exemplo: Para este parágrafo eu defini um destaque (`outline`) de 5px em linha tracejada na cor azul que será visualizado nos Mozillas e no Ópera, mas não no Internet Explorer que não suporta `outline`.

Abreviando `border`

As regras a seguir definem valores para propriedades de borda:

```
div {  
  border-width:1px;  
  border-style:solid;  
  border-color:#f00;  
}
```

E pode ser abreviada para:

```
div {border:1px solid #f00;}
```

As regras a seguir definem valores para espessuras de borda:

```
p {  
  border-top-width:2px;  
  border-right-width:1px;  
  border-bottom-width:3px;  
  border-left-width:5px;  
}
```

E pode ser abreviada para:

```
p {border-width:2px 1px 3px 5px;}
```

As regras a seguir definem valores para cores de borda:

```
h1 {  
border-top-color:#f00;  
border-right-color:#ccc;  
border-bottom-color:#00f;  
border-left-color:#999;  
}
```

E pode ser abreviada para:

```
p {border-color:#f00 #ccc #00f #999;}
```

As regras a seguir definem valores para estilos de borda:

```
p {  
border-top-style:solid;  
border-right-style:ridge;  
border-bottom-style:double;  
border-left-style:dotted;  
}
```

E pode ser abreviada para:

```
p {border-style:solid ridge double dotted;}
```