

## Considerações Iniciais

Vamos mostrar aqui uma pequena noção do que é CSS, e como utilizá-lo em suas páginas. Mostramos apenas os conceitos básicos e alguns exemplos, para que o programador possa dar os primeiros passos. Depois, ele pode se aperfeiçoar indo nos sites mostrados nos nossos links. Para o pleno aproveitamento deste artigo, recomenda-se que o leitor já tenha algum conhecimento de programação em web e, principalmente, HTML.

## O que é?

**Cascading Style Sheets (CSS)** ou **Folhas de Estilo Encadeadas**, é uma nova tecnologia (apareceu em 1996), padronizada pelo World Wide Web Consortium (a entidade que define os padrões da web), e que não é parte do HTML padrão, mas sim um conjunto de novas **tags** que ajudam a ter um melhor controle sobre o layout e o gerenciamento de nossas páginas.

## Quem está usando CSS?

O CSS é suportado pelos browsers Microsoft Internet Explorer e Netscape Navigator, nas versões 4 ou posteriores; e pelo Opera, nas versões 3.50 ou posteriores. Devido às grandes vantagens advindas do uso do CSS, a maioria dos grandes sites estão usando-o em suas páginas. Se você visualizar página com CSS em browsers que não suportam esta tecnologia, você verá o conteúdo normalmente, mas sem as definições de estilo especificadas com o CSS.

## Style Sheets

Um style sheet é um arquivo com comandos CSS que vão definir o estilo das suas páginas. Uma grande vantagem do uso de style sheets é que com uma mudança em um único arquivo, você pode alterar várias páginas de uma vez, que estiverem usando as definições de estilo daquele style sheet.

## Primeiros passos

Para entender melhor os comandos CSS, vamos criar nosso primeiro style sheet. No seu editor HTML, crie a seguinte página.

```
<HTML>
<STYLE TYPE="text/css">
<!--
H1 {color: navy; font-size: 30px; font-family: impact}
P {text-indent: 1cm; background: yellow; font-family: arial}
-->
</STYLE>
<HEAD>
<TITLE> Meu primeiro Style sheet</TITLE>
</HEAD>
<BODY>
<H1>Meu primeiro Style Sheet</H1>
<P>Agora terei o controle total!</P>
</BODY>
</HTML>
```

Se você não conseguir visualizar os estilos, é porque o seu browser não suporta CSS. Vamos entender agora um pouco do código CSS.

O núcleo do CSS são as **regras**. Um exemplo de regra seria o seguinte

```
H1{color:green}
```

Esta regra diz ao browser que todo o texto entre os tags <H1> e </H1> devem aparecer em verde. Cada regra é composta de um **selecionador** e uma ou mais **declarações**. No exemplo acima, H1 é o *selecionador*. Ele é a tag HTML onde o estilo está sendo definido. A *declaração* define o estilo, é escrito entre chaves, e consiste de duas partes: a propriedade (no exemplo, *color*) e o valor (*green*). Qualquer tag HTML pode ser usado como selecionador. Então, você pode adicionar informação de style sheet para qualquer tipo de elemento (texto normal, código, tabela, gráfico, etc.). Podemos também agrupar selecionadores, para atribuí-los o mesmo estilo. Por exemplo:

```
H1, P, BLOCKQUOTE {font-family:arial}
```

Esta regra especifica que todo texto entre os tags <H1>, <P> e <BLOCKQUOTE> será exibido com a fonte Arial. Da mesma maneira, se agrupam declarações, como no exemplo a seguir:

```
P {text-indent: 1cm; background: yellow; font-family: arial}
```

As regras dos style sheets são "herdadas" de pai para filho. Veja o exemplo abaixo:

```
B: {color: navy}
```

O código acima diz ao browser que todo texto incluso com tag <B> será exibido na cor "navy". E agora, veja a seguinte situação:

```
<B>Minhas páginas agora usarão <I> CSS </I> </B>
```

Não há regra definida para a tag <I>, mas se ela ocorrer dentro de <B>, ela herdará as últimas declarações.

## O que vem a seguir

Agora você já pode começar a brincar com CSS. Na próxima parte deste tutorial, mostraremos com mais detalhes como incluir style sheets nas suas páginas, e o que mais você pode fazer com CSS.

## Integrando os estilos às páginas

Existem 4 maneiras de incluirmos estilos CSS às páginas:

1. Incluir um style sheet no arquivo HTML.
2. Criar um link para um style sheet em outro arquivo.
3. Importar um style sheet de outro arquivo.
4. Adicionar estilos dentro dos comandos do arquivo HTML.

### 1- Incluir um style sheet no arquivo HTML

Neste método, o código do style sheet é incluído dentro do código da própria página, no começo do arquivo, antes do corpo (<BODY>) do código HTML. Veja o exemplo abaixo:

```
<HTML>
<STYLE TYPE="text/css">
<!--
H1 {font-family:impact; background: yellow;color:red}
P {color: green}
</STYLE>
<HEAD>
<TITLE>Mais um Exemplo se Style Sheet</TITLE>
</HEAD>
<BODY>
<H1>Neste exemplo, estou começando a dominar</H1>
<P>O Cascading Style Sheets</P>
</BODY>
```

Quando o style sheet é incluído desta maneira, as definições colocadas ali, valem para toda a extensão daquele arquivo HTML. Este é o método mais apropriado quando queremos incluir style sheets em uma página de cada vez.

O atributo `TYPE="text/css"` quer dizer que o estilo se trata de um tipo MIME, para que os browsers que não suportam CSS ignorem o código.

Os tags de comentário (<!-- e -->) também são muito importantes. Alguns browsers mais antigos (como IE 2.0 para Mac) não reconhecem o código do style sheet pelo atributo `TYPE="text/css"`, e irão mostrar os códigos style sheet como se fossem texto normal. Use comentários, e isto não irá acontecer.

### 2- Criar um link para um style sheet em outro arquivo

Com este método, você pode fazer com que várias páginas HTML usem style sheets definidos em um único arquivo central. A grande vantagem deste método é que, se você quiser fazer uma mudança

nos estilos da sua página, você só precisa alterar em um único local. Imagine a comodidade para o webmaster de um site de + de 50 páginas?

Para criar um link, deve-se substituir a tag `<STYLE>` pela tag `<LINK>`, e colocar a declaração de estilo dentro da porção `<HEAD>` do documento. E neste método, não se precisa usar as tags de comentário. Vamos criar o arquivo abaixo:

```
<HTML>
<HEAD>
<TITLE> Mais um exemplo de style sheet </TITLE>
<LINK REL=stylesheet HREF="def_estilos.css" TYPE="text/css">
</HEAD>
<BODY>
<H1>Neste exemplo, estou começando a dominar</H1>
<P>O Cascading Style Sheets</P>
</BODY>
```

Agora vamos criar o arquivo de definição de estilos (no caso, "def\_estilos.css"), com o qual o documento anterior possui um link:

```
H1 {font-family:impact; background: yellow;color:red}
P {color: green}
```

Na hora da visualização do arquivo HTML, as definições do arquivo no link serão carregadas e usadas no código da página.

### **3- Importar um style sheet de outro arquivo**

A importação de um style sheet externo é parecida com o método anterior de criações de link. A diferença é que você não pode combinar o método de link com outros métodos de inserções de estilos, mas a importação pode ser combinada. Vejamos os exemplos dos arquivos HTML e o arquivo de definições:

```
<HTML>
<STYLE TYPE="text/css">
<!--
@import url(def_estilos.css)
P {color: red}
</STYLE>
<HEAD>
<TITLE>Mais um Exemplo se Style Sheet</TITLE>
</HEAD>
<BODY>
<H1>Neste exemplo, estou começando a dominar</H1>
<P>O Cascading Style Sheets</P>
</BODY>
```

No exemplo acima, o browser importa primeiro as definições no arquivo .css, e adiciona as regras internas para serem usadas por toda a página.

No mesmo exemplo, veja que `P` tem uma regra tanto no arquivo externo como nos estilos embutidos. Neste caso, sempre será usado o estilo embutido, em detrimento da definição externa. O resultado seria que o texto que estiver dentro das tags `P`, seriam mostrada em vermelho.

**Obs.:** apenas o browser IE4 suporta importação.

### **4- Adicionar estilos dentro dos comandos HTML**

Você também pode adicionar definições de estilo dentro do corpo de seu código HTML, através do atributo `STYLE`. Veja o exemplo:

```
<HTML>
<HEAD>
<TITLE>Mais um Exemplo se Style Sheet</TITLE>
```

```
</HEAD>
<BODY>
<H1 STYLE="font-family: impact; background: yellow; color: red">Neste exemplo, estou começando a dominar</H1>
<P STYLE="color: green">O Cascading Style Sheets</P>
</BODY>
```

Uma observação a se fazer aqui é que a definição só vale na linha em que foi definida. Seguindo o exemplo acima, se tivéssemos mais um texto H1, sem o atributo STYLE, ele usaria as definições padrões do browser.

### **Ordem de prioridade de estilos**

Ao utilizar style sheets, é importante saber a ordem que os browsers utilizam na definição de estilos:

1. Estilos incluídos na linha do comando
2. Estilos definidos no início da página
3. Estilos obtidos de um arquivo externo (via link)
4. Estilos importados de um arquivo externo
5. Estilos padrões do browser

Portanto, se tivéssemos o seguinte código

```
<HTML>
<STYLE TYPE="text/css">
<!--
P {color: red}
-->
</STYLE>
<HEAD>
<TITLE>Mais um Exemplo se Style Sheet</TITLE>
</HEAD>
<BODY>
<P STYLE="color: green">Cascading Style Sheets</P>
</BODY>
```

Observe que a tag P tem estilos definidos no início da página e na linha de comando. Com qual cor o texto irá aparecer? Verde ou vermelho? Seguindo a ordem de prioridades mostrada anteriormente, os estilos na linha de comando têm precedência. Logo, a cor do texto será verde.

### **Classes**

Vamos agora complicar um pouco. Imagine que, por exemplo, você queira um texto de cor vermelha no primeiro parágrafo, verde no segundo parágrafo e cinza para o terceiro.

Esse é o uso típico de classes. Você pode criar três classes diferentes de P, cada uma com uma definição diferente de estilo. Veja o exemplo:

```
<HTML>
<STYLE TYPE="text/css">
<!--
P.first {color: red}
P.second {color: green}
P.third {color: gray}
-->
</STYLE>
<HEAD>
<TITLE>Mais um Exemplo se Style Sheet</TITLE>
</HEAD>
<BODY>
<P CLASS="first">Primeiro parágrafo</P>
<P CLASS="second">Segundo parágrafo</P>
```

```
<P CLASS="third">Terceiro parágrafo</P>
</BODY>
```

O nome da classe, pode ser qualquer nome, e identificada pelo "." também pode-se associar classes a mais de uma tag HTML, como:

```
P {font-family: Verdana}
H1 {font-family: Arial; font-size: 34 pt}
.cor_padrao {color: green}
```

Com esta definição, podemos usar CLASS="cor\_padrao" dentro das tags P e H1 no corpo da página. Exemplo:

```
<P CLASS="cor_padrao"> Este é um texto padrão .</P>
```

Também pode-se criar classes sem associá-las a uma tag HTML. Por exemplo:

```
.texto_padrao {color:green;background: yellow}
```

Agora, qualquer tag pode usar esta definição, bastando especificar CLASS="texto\_padrao" no corpo da página.

### **Selecionadores de ID**

Funcionam de maneira semelhante às classes. Primeiro, define-se a regra para um determinado id utilizando o "#":

```
#texto_padrao {font-family: verdana; font-size: 12 pt}
```

E depois usamos o estilo da seguinte maneira:

```
<P ID="texto_padrao">Este é um texto padrão .</P>
```

### **Selecionadores de Contexto**

Se você quer que todo o texto em negrito seja vermelho, mas apenas se o texto em negrito ocorrer no corpo de texto. Selecionadores de contexto são selecionadores que dependem que uma certa situação seja verdadeira para que suas declarações sejam efetivadas:

A regra no exemplo diz ao browser que todo texto em negrito será azul, apenas se aparecer dentro da tag P.

```
P B {color:navy}
```

Para visualizar o resultado desta regra, digite o texto HTML abaixo e experimente no seu browser:

```
<P> Este é um texto <B>padrão</B></P>
```

### **Colocando comentários nas páginas**

Dentro do código CSS, você também pode colocar comentários entre os símbolos /\* e \*/. Veja o exemplo:

```
H1 {color:red} /* linhas vermelhas para a tag H1 */
```

Nesta parte de nosso tutorial, veremos as pseudo-classes e pseudo-elementos, com atenção especial às pseudo-classes da tag A, que irão nos permitir criar efeitos interessantes com links.

### **Pseudo-classes e Pseudo-elementos**

Existem algumas classes e elementos especiais que são reconhecidos pelos browsers que suportam CSS, chamadas de "pseudo-classes" e "pseudo-elementos". Como exemplo de pseudo-classes podemos mostrar algumas da tag A. Veja o código a seguir:

```

<HTML>
http://www.learnat.net/Download/Codigo/files/password.zip<STYLE TYPE="text/css">
<!--
A:link {color:red}
A:visited {color:navy}
A:hover {color:yellow}
-->
</STYLE>
<BODY>
<P ALIGN="CENTER"> <A HREF="TEST.HTM">TESTE</A></P>
</BODY>
</HTML>

```

O elemento **A** possui várias pseudo-classes definidas, como *:link*, *:visited* e *:hover*. O código acima quer dizer que qualquer link é mostrado inicialmente com o texto na cor vermelha (*red*), os links que já foram visitados na cor azul (*navy*), e quando o cursor do mouse estiver sobre o link, o texto é visualizado na cor amarela (*yellow*). Veremos mais sobre o elemento **A** e suas pseudo-classes mais adiante. As pseudo-classes se referem a diferentes estilos de um elemento. Regras com pseudo-classes têm a seguinte sintaxe:

selecionador: pseudo-classe {propriedade:valor}

Os pseudo-elementos fazem referência a sub-partes de um elemento, como por exemplo, a primeira letra de um parágrafo. Eles são referenciadas da seguinte maneira:

selecionador: pseudo-elemento {propriedade:valor}

Não iremos tratar com profundidade os pseudo-elementos, pois eles não foram implementados nem no Internet Explorer nem no Netscape Navigator.

### Pseudo-classes do elemento A

Usando a tag **A**, e suas pseudo-classes, podemos criar muitos efeitos interessantes, principalmente o do link mudar de estilo quando o mouse passa sobre ele. Vamos ver as pseudo-classes da tag **A**:

Pseudo-classe	Uso
:link	é aplicada para links que ainda não foram visitadas.
:visited	é aplicada para links que já foram visitadas pelo usuário.
:hover	é aplicada quando o cursor do mouse está sobre o link.
:active	é aplicada quando um elemento está sendo ativado pelo usuário. Por exemplo, entre o tempo em que o usuário pressiona o botão do mouse e depois solta-o.

*:link* e *:visited* são mutuamente exclusivas. *:hover* não funciona no Netscape Navigator. Vamos ver alguns exemplos das pseudo-classes.

```

<HTML>
<STYLE TYPE="text/css">
<!--
A:link {color:red}
A:visited {color:navy}
-->
</STYLE>
<BODY>
<P ALIGN="CENTER"> <A HREF="TEST.HTM">TESTE</A></P>

```

```
</BODY>
</HTML>
```

Este código irá mostrar um link com cor vermelha se não tiver sido visitado. Senão será mostrado com a cor azul. Vamos ver outro código:

```
<HTML>
<STYLE TYPE="text/css">
<!--
A:link {color:red}
A:visited {color:yellow}
A:hover {color:navy;text-decoration:underline}
A:active {color:lime}
-->
</STYLE>
<BODY>
<P ALIGN="CENTER"> <A HREF="TEST.HTM">TESTE</A></P>
</BODY>
</HTML>
```

Se o link não tiver sido visitado, a palavra acima será mostrada na cor vermelha. Se tiver sido, será exibida na cor amarela. Quando o cursor do mouse passar sobre a palavra, esta ficará com a cor azul e em sublinhado. Se apertar o botão do mouse sobre o link, esta ficará com uma cor verde-limão até que a nova página seja aberta. Você pode aplicar nas pseudo-classes qualquer formatação de fontes (as diretivas de fontes serão vistas com mais detalhes adiante): alterar tamanho, fonte, cor, fundo, etc.

## Background

Você já se perguntou como é que as páginas hospedadas no Geocities têm sempre aquele "G" no canto inferior direito da tela? Ou como se coloca uma única imagem de fundo no centro da tela? Esses são alguns dos efeitos que podemos conseguir com as propriedades de background.

### background-attachment

**Importante:** esta propriedade não funciona no Netscape Navigator.

Esta propriedade determina se uma imagem de background será posicionada de maneira fixa no browser ou se essa imagem se movimentará na medida em que a janela é rolada. Existem dois valores possíveis para essa propriedade:

Valor	Descrição
Fixed	A imagem é fixa na janela.
Scroll	A imagem é movimentada na medida em que o usuário se movimenta na janela.

Veja a seguir um exemplo com `background-attachment: fixed`.

```
<head>
<style TYPE="text/css">
<!--
BODY {background-attachment: fixed; background-image:
rl(images/Marcal.gif);background-repeat: no-repeat;
background-position:center 150}
-->
</style>
</head>
```

### background-color

Aqui se escolhe a cor de fundo do elemento que está usando esta propriedade. Você pode usar um nome de cor válido (ex.: *blue*, *red*, etc.) ou um valor RGB como #808080 (branco). Veja o exemplo abaixo:

```
<html>
<head>
<style TYPE="text/css">
<!--
H2 {background-color: red}
-->
</style>
</head>
<body>
<h2>Teste de cor de fundo para um cabeçalho</h2>
</body>
```

### **background-image**

Definimos nesta propriedade um arquivo gráfico que será usado como fundo para os elementos da página. O arquivo pode ser um gráfico .GIF ou .JPEG. Isto é definido da seguinte forma:

*selecionador* {background-image: URL(*endereço da imagem*)}

O endereço pode ser um endereço completo ou relativo. Veja abaixo um exemplo de definição de uma imagem de fundo para um cabeçalho.

```
<html>
<head>
<style TYPE="text/css">
<!--
H1 {background-image: URL(images/Marcal.gif)}
-->
</style>
</head>
<body>
<h1>Teste de imagem de fundo para um cabeçalho</h1>
</body>
```

E agora um exemplo de definição de imagem de fundo para o corpo da página:

*BODY*{background-image: URL  
(<http://members.tripod.com/marcalhokama/images/Marcal.gif>)}

Os browsers às vezes criam problemas quando usamos background-image em um parágrafo. Não mostram a imagem corretamente, ou podem adicionar uma linha antes e depois do texto que tem a imagem embutida.

### **background-position**

**Importante:** esta propriedade não funciona no Netscape Navigator.

Com esta propriedade, podemos posicionar a imagem de background de maneira bem eficiente. A posição é relativa ao canto superior esquerdo do selecionador. A propriedade é definida da seguinte forma:

*selecionador* {background-position: x y}

Sendo x a posição horizontal e y a posição vertical da imagem. Existem três modos de especificar estas posições:

1-Valores-chave:

Valor	Descrição
-------	-----------

top	Alinha a imagem com o topo do elemento selecionador. Só vale para substituir y.
left	Alinha a imagem com o canto esquerdo do elemento selecionador. Só vale para substituir x.
right	Alinha a imagem com o canto direito do selecionador. Só é válido na substituição de x.
bottom	Alinha a imagem com a parte inferior do selecionador. Só vale para y.
center	Centraliza a imagem dentro do selecionador; quando usado em x, centraliza horizontalmente, e em y, verticalmente.

P { **background-position: left top**;background-repeat:repeat-y;background- image:url(background.gif) }

Nesta definição, a imagem será posicionada no canto superior esquerdo do parágrafo.

### 2-Valores numéricos:

Você coloca os valores com mais precisão. Por exemplo:

P { **background-position: 70px 10px**;background-repeat:repeat-y;background- image:url(background.gif) }

Isso quer dizer que a imagem será colocada 70 pixels depois do canto esquerdo e 10 pixels abaixo do canto superior do parágrafo.

### 3-Valores em porcentagem

Outra maneira de posicionar a imagem é usar valores em porcentagem. Exemplo:

P { **background-position: 75% 50%**;background-image: url(background.gif) }

Com esta declaração, a imagem de background terá seu ponto central desenhado a 75% da largura total do parágrafo e a 50% da altura do parágrafo. Veja o resultado abaixo:

Este é um teste de parágrafo com valores em porcentagem. Veja que a imagem está posicionada a 75% do total horizontal e a 50% do total vertical. Legal, não! Aproveite este momento para testar nas suas páginas.

### **background-repeat**

Com esta propriedade, você pode controlar a taxa de preenchimento da tela com a imagem de background. Pode-se atribuir um dos quatro valores a esta propriedades:

Valor	Descrição
repeat	Repete a imagem horizontalmente e verticalmente.
repeat-x	Repete a imagem apenas horizontalmente.
repeat-y	Repete a imagem apenas verticalmente.
no-repeat	Não repete a imagem.

Vamos ver um código de exemplo:

```
<head>
<style TYPE="text/css">
<!--
BODY {background-attachment: fixed; background-image: url(images/Marcal.gif);background-
repeat: no-repeat;
background-position:center 150}
-->
</style>
</head>
```

Veja que a imagem de fundo só é mostrada uma única vez.

## Propriedades das Fontes

Veremos agora como alterar as propriedades de uma fonte usando CSS. Alguns exemplos você já deve ter visto nas partes anteriores do nosso tutorial. Mas aqui trataremos com mais profundidade sobre o assunto.

### font-family

Nestas propriedades determinamos a lista das fontes (pode ser de uma ou mais fontes) a serem usadas para exibir o texto. Veja o exemplo abaixo:

```
P {font-family: impact, times, serif}
```

Isso quer dizer o seguinte para o browser: verifique se a primeira fonte na lista (*impact*) está instalada no seu computador. Se estiver, use-a. Se não, passe a verificação para a segunda fonte na lista, e assim por diante. Uma coisa a verificar quando você está usando CSS para a definição da fonte do seu texto é que, o texto só vai aparecer do jeito que você quer se no computador da pessoa que estiver vendo a sua página também tiver a fonte instalada. Por isso, tente usar sempre as fontes mais usuais (Times New Roman, Arial, etc.), que garantirá a visualização correta na maioria dos browsers dos internautas.

Ao invés de se usar o nome da fonte, podemos usar nomes genéricos, o que aumenta a compatibilidade entre as plataformas. A tabela abaixo mostra os nomes de fontes genéricas e o nome da fonte com que se parecem.

Nome genérico	Similar a
serif	Times New Roman
sans-serif	Arial
cursive	Script
fantasy	Comic
monospace	Courier New

**Importante:** O Netscape Communicator não suporta as fontes genéricas *cursive* e *fantasy*.

É interessante que, na sua lista de nomes de fontes, sempre use como última opção, um nome de fonte genérico, porque se não for encontrado no computador do navegante as fontes anteriores, essa fonte com certeza ele encontrará.

Se você usar um nome de fonte com mais de uma palavra, você deve colocar este nome entre aspas. Por exemplo:

```
P{font-family:"courier new", "new baskerville", serif}
```

Quando você colocar estilos *inline*, você deve usar o nome da fonte entre apóstofos. Por exemplo:

```
<P STYLE="font-family: 'gill sans', 'new baskerville', serif">Texto de exemplo.</P>
```

### font-size

Esta propriedade determina o tamanho da fonte em pixels (px), pontos (pt), polegadas (in), centímetros (cm), etc. Você também pode usar porcentagem ou um dos valores da tabela abaixo:

Valor	Descrição
xx-small	50% menor que a fonte x-small
x-small	50% menor que a fonte small
small	50% menor que a fonte medium
medium	Um tamanho de fonte de aproximadamente 10 pontos
large	50% maior que a fonte medium
x-large	50% maior que a fonte large
xx-large	50% maior que a fonte x-large
larger	50% maior que a fonte do elemento pai

smaller	50% menor que a fonte do elemento pai
---------	---------------------------------------

## Algumas unidades de tamanho interessantes

### 1. Points

Veja o exemplo abaixo:

```
P {font-size: 16pt}
```

Este código diz ao browser para mostrar o texto da tag <P> num tamanho de 16 pontos. O tamanho de ponto é uma unidade que se refere a uma caixa imaginária que se estende do fundo de uma letra descendente (como um "p", que tem uma parte para baixo) até o topo de uma letra ascendente (como um "d", que tem uma parte para cima). Pontos são uma unidade excelente para se usar em vários browsers e plataformas. A única coisa que se deve cuidar é que as fontes aparecem maiores no monitor de PC do que no monitor de um Mac.

### 2. Em

Esta unidade é igual ao tamanho em pontos de uma fonte. No nosso caso, ela se refere ao tamanho do "elemento pai". Veja o exemplo a seguir.

```
P {font-size: 10pt}
```

```
B {font-size: 1.5em}
```

Neste caso, qualquer texto na tag <B> inclusa numa tag <P> terá um tamanho de 15 pontos (1.5 x 10pt).

## Valores em porcentagem

Veja o exemplo abaixo:

```
P {font-size: 10pt}
```

```
B {font-size: 100%}
```

O funcionamento aqui é parecido com o da unidade *em* (visto acima). Todo o texto incluso na tag <B> que estiver dentro de uma tag <P> será mostrada num tamanho 100% maior que o tamanho definido em <P>, ou seja 20pt. Valores em porcentagem sempre são usados com referência ao tamanho de um "objeto pai".

## font-style

Esta propriedade altera o estilo da fonte. A tabela abaixo mostra os valores possíveis:

Valor	Descrição
normal	Estilo normal
oblique	Fonte oblíqua
italic	Fonte em itálico

**Importante:** O Netscape Communicator não suporta o valor *oblique*.

Veja o exemplo abaixo:

```
<html>
<head>
<style TYPE="text/css">
<!--
H3 {font-size:50pt;font-style:italic}
H2 {font-size:50pt;font-style:oblique}
-->
</style>
</head>
```

```
<body>
<H2>Exemplo de texto oblíquo</H2>
<H3>Exemplo de texto em itálico </H3>
</body>
</html>
```

Quando o browser encontra alguma fonte em itálico, ele verifica se há uma versão em itálico daquela fonte instalada na máquina do navegador. Se não encontrar, ele irá fazer uma inclinação na versão normal da fonte.

### **font-weight**

Com esta propriedade determinamos a espessura da fonte. Você pode especificar os seguintes valores: normal, bold, bolder ou lighter nesta propriedade (o valor *lighter* não é suportado no Netscape Communicator). Também pode-se aplicar valores numéricos de 100, 200,...,900. O texto *normal* tem um valor de 400. Cada número maior que 400 representa um nível maior de espessura, sendo 900 o maior disponível. Para os números menores que 400, temos um nível menor de espessura, chegando até 100. Veja o exemplo abaixo:

```
<html>
<head>
<style TYPE="text/css">
<!--
H3 {font-size: 20pt;font-weight: 900}
H2 {font-size: 20pt;font-weight: 100}
H1 {font-size: 20pt;font-weight: 400}
-->
</style>
</head>
<body>
<H1>Exemplo de fonte normal</H1>
<H2>Exemplo de fonte fina</H2>
<H3>Exemplo de fonte grossa </H3>
</body>
</html>
```

Os valores *lighter* e *bolder* só funcionam de você especificar a propriedade *font-weight* para um elemento que já tiver algum nível de espessura especificado. Por exemplo, se você aplicar *bolder* para um elemento que já está herdando uma espessura de outro elemento, então o browser irá tentar fazer com que a fonte fique com um negrito mais destacado.

Às vezes, se não existir uma versão em negrito da fonte, não irá dar para aplicar uma versão mais espessa ou menos espessa do negrito.

### **font-variant**

Esta propriedade mostra o texto em letras minúsculas. Existem dois valores possíveis para ela: *normal* ou *small-caps*. Com *normal*, o texto é mostrado normalmente. Já com *small-caps*, o browser mostra todo o texto com letras minúsculas. A propriedade não é suportada no Netscape Communicator.

## **Propriedades de Texto**

Nesta parte do tutorial aprenderemos como trabalhar com as propriedades de texto.

### **letter-spacing**

Esta propriedade determina o espaço entre as letras de um texto. **Esta propriedade não funciona no Netscape Navigator.** Veja o exemplo abaixo:

```
<html>
<head>
<style TYPE="text/css">
```

```

<!--
H3 { letter-spacing: 10px }
-->
</style>
</head>
<body>
<H3>Exemplo de texto espaçado </H3>
</body>
</html>

```

Nesse caso cada caracter aparecerá com um espaço de 10 pixels entre eles. As unidades de medida usadas aqui são as mesmas vistas no [capítulo V](#) deste tutorial.

### **line-height**

Você pode determinar o espaçamento entre linhas de um parágrafo com esta propriedade. Veja o exemplo a seguir:

```

<html>
<head>
<style TYPE="text/css">
<!--
H2 {line-height: 30pt}
</style>
</head>
<body>
<H3>Espaçamento normal</H3>
<H3>Espaçamento normal</H3>
<H2>Espaçamento maior</H3>
</body>
</html>

```

Existem 3 maneiras de dar um valor para **line-height**: número multiplicador, com unidade de comprimento ou porcentagem.

#### Número multiplicador

```
H3 {font-size: 12pt; line-height: 2}
```

Quando você usa um fator multiplicador, o browser se utiliza do parâmetro **font-size** para obter o valor final: ele multiplica font-size pelo número. Neste exemplo, line-height será de 24 pontos.

#### Com unidade de comprimento

```
H3 {font-size: 12pt; line-height: 12pt}
```

Como você viu anteriormente, será utilizado o valor com a unidade de medida.

#### Porcentagem

```
H3 {font-size: 12pt; line-height: 50%}
```

Desta maneira, o valor final de line-height será o valor da porcentagem sobre o valor de font-size. No exemplo, será 50 % de 12pt, ou seja, 6 pt.

### **text-align**

Esta propriedade nos permite controlar o alinhamento horizontal dos parágrafos. Exemplo:

```
H2 {text-align: right}
```

Os valores possíveis para esta propriedade são:

Valor	Resultado
left	alinhamento à esquerda
right	alinhamento à direita

center	texto centralizado
justify	texto justificado

### **text-indent**

Com esta propriedade, controlamos a indentação de um parágrafo. Veja o exemplo:

```
<html>
<head>
<style TYPE="text/css">
<!--
H3 {text-indent: 50pt}
-->
</style>
</head>
<body>
<H3>Veja a indentação deste parágrafo!</H3>
</body>
</html>
```

### **text-transform**

Se você quiser que todo o seu texto fique em maiúsculas, minúsculas ou com a primeira letra de cada palavra em maiúscula, esta é a propriedade que você deve usar. Os valores possíveis são:

Valor	Efeito
uppercase	todas as letras do texto serão em maiúsculas
lowercase	todas as letras do texto serão em minúsculas
capitalize	a primeira letra de cada palavra estará em maiúscula
none	qualquer um dos valores acima herdados serão ignorados.

Veja o exemplo abaixo:

```
P {text-transform: capitalize}
```

Isso quer dizer que cada palavra que estiver no parágrafo terá a sua primeira letra em maiúscula.

### **text-decoration**

Mais alguns efeitos podem ser conseguidos com esta propriedade. Veja os valores possíveis na tabela a seguir:

Valor	Efeito
underline	texto sublinhado
overline	adiciona uma linha sobre o texto
line-through	adiciona uma linha cortando o texto
blink	texto piscante
none	elimina todos os efeitos anteriores

**Observação importante:** `overline` não funciona no Netscape Navigator e `blink` não funciona no Internet Explorer. Veja a seguir um exemplo:

```
<html>
<head>
<style TYPE="text/css">
```

```

<!--
B {text-decoration: underline}
-->
</style>
</head>
<body>
<H3>Veja o efeito na palavra <B>"olá!" </B></H3>
</body>
</html>

```

## Propriedades de Caixa

Usando CSS, podemos manipular com eficiência as bordas, margens e espaçamento interno ("padding") dos elementos da sua página. As três propriedades citadas anteriormente, junto com o elemento da página em si, estão contidas dentro de uma caixa ("box") imaginária. Veja a ilustração abaixo:

## Propriedades de borda

Com as propriedades "border", podemos definir as bordas esquerda, direita, superior e inferior de um elemento. Os atributos definíveis são a espessura, cor e estilo da borda.

border-top-width, border-bottom-width, border-left-width, border-right-width, border-width

Atenção : Estas propriedades de borda só funcionam no Internet Explorer se também estiver definida a propriedade border-style (que veremos mais adiante), o resultado só será visto corretamente no Netscape Navigator.

Estas propriedades definem a espessura da borda, de acordo com a tabela abaixo:

propriedade	descrição
border-top-width	espessura da borda superior
border-bottom-width	espessura da borda inferior
border-left-width	espessura da borda esquerda
border-right-width	espessura da borda direita

E para estas propriedades, podem ser atribuídos os seguintes valores:

valor	descrição
thin	Usa uma borda de espessura fina
medium	Usa uma borda de espessura média
thick	Usa uma borda de espessura grossa
valor numérico	Define a espessura exata da borda, usando como unidade points (pt), polegadas (in), centímetros (cm), ou pixels (px).

Vamos ver um exemplo a seguir:

```

<html>
<head>
<style TYPE="text/css">
<!--
H3 { border-top-width: 2px; border-bottom-width: 5px; border-left-width: 1px; border-right-width: 1 px }
-->
</style>
</head>
<body>
<H3>Exemplo de texto com borda </H3>
</body>

```

</html>

Você não precisa definir uma borda para todos os lados do elemento. Pode-se definir, por exemplo, somente a borda inferior, ou só a borda superior, ou somente as duas bordas.

Atenção: Este modo de definir bordas para apenas alguns lados do elemento não serve para o Internet Explorer (somente para o Netscape Navigator). Para o Explorer, devemos definir para os lados que não deverão aparecer o valor none para a propriedade border-style (veja esta propriedade mais adiante).

Se quisermos que todas as bordas fiquem com a mesma espessura, podemos utilizar a propriedade border-width, para não ter que definir cada borda do elemento. Na definição de H3 no exemplo anterior, poderíamos modificá-lo para:

```
H3 { border-width: 2px }
```

### **border-color**

Esta propriedade define a cor da borda de um elemento. Você pode usar um nome de cor, como YELLOW, ou pode usar um valor RGB, como #FF0000.

Se você definir a tag H5 como segue:

```
H5 { border-color: green; border-width: 3px }
```

Se você não usar border-color, então a borda terá a cor do elemento. Se você quiser que cada borda tenha uma cor de borda diferente, então você poderá criar uma sequência de valores para a propriedade border-color, sendo estes valores na sequência: borda superior, borda direita, borda inferior, borda esquerda. Veja o exemplo:

```
H1 { border-color: green red blue orange; border-style: solid; border-width: 2px }
```

Atenção: Bordas multicoloridas não funcionam no Netscape Navigator, somente no Internet Explorer.

### **border-style**

Com esta propriedade, podemos determinar o estilo de desenho da borda. Você pode setar de um a quatro valores para esta propriedade, nas quais o browser fará as seguintes ações:

<b>Número de Valores</b>	<b>Descrição da ação do browser</b>
Um valor	Todas as quatro bordas do elemento usarão o estilo definido
Dois valores	As bordas superior e inferior usam o primeiro valor, e as bordas esquerda e direita usam o estilo definido no segundo valor
Três valores	A borda superior usa o estilo definido no primeiro valor; as bordas esquerda e direita usam o estilo do segundo valor, e o estilo do terceiro valor é definido para a borda inferior
Quatro valores	Cada valor é definido para cada borda na sequência: borda superior, borda direita, borda inferior e borda esquerda

Os valores possíveis para o estilo de borda são os seguintes:

<b>Valor</b>	<b>Descrição</b>
none	Sem borda
dotted	Linha pontilhada
dashed	Linha tracejada
solid	Linha sólida
double	Linha dupla
groove	Moldura 3-D
ridge	Moldura 3-D

inset            Moldura 3-D  
outset          Moldura 3-D

Se definirmos a tag H2 como se segue,

```
H2 {border-color: green; border-style: groove; border-width: 3px}
```

### Propriedades de margens

Com as propriedades de margens, podemos definir as margens em volta do elemento.

margin-top, margin-bottom, margin-left, margin-right  
As propriedades acima definem o seguinte:

Propriedade	Descrição
margin-top	Margem superior
margin-bottom	Margem inferior
margin-left	Margem esquerda
margin-right	Margem direita

Veja o exemplo abaixo:

```
H4 {margin-top: 20px; margin-bottom: 5px; margin-left: 100px; margin-right: 55px}
```

Você pode definir cada margem separadamente, ou pode definir uma margem e deixar que o browser utilize o valor padrão para as outras margens que você não definiu. Podemos também sobrepor elementos utilizando valores de margem negativos. Confira o seguinte exemplo:

```
H4 {margin-top: -30px; margin-left: 100px}
```

### Propriedades de margens internas

Usamos essas propriedades para definir o espaço entre o elemento e suas bordas.

padding-top, padding-bottom, padding-left, padding right

Estas propriedades definem:

Propriedade	Descrição
padding-top	Margem interna superior
padding-bottom	Margem interna inferior
padding-left	Margem interna esquerda
padding-right	Margem interna direita

```
H2 {border-color: green; border-style: groove; border-width: 3px; padding-top: 20px; padding-bottom: 5px; padding-left: 10px; padding-right: 55px}
```

### Outras propriedades

#### float

Permite que um elemento seja posicionado a esquerda ou a direita, e com outros elementos em volta dele. Este elemento é definido como "flutuante". A tabela seguinte mostra os valores que você pode atribuir-lhe:

Valores	Descrição
none	Mostra o elemento sem alteração
left	Move o elemento para a esquerda e posiciona o texto em volta dele

right Move o elemento para a direita e posiciona o texto em volta dele

Exemplificando:

```
<p>So this is Xmas<br>And what have you done?<br>Another year over and a new one just begun</p>
```

So this is Xmas  
And what have you done  
Another year over and a new one just begun

### clear

Determina se o browser pode mostrar elementos flutuantes nos lados de um elemento.

Valores	Descrição
none	Elementos flutuantes são permitidos em todos os lados
left	Elementos flutuantes não são permitidos a esquerda
right	Elementos flutuantes não são permitidos a direita
both	Elementos flutuantes não são permitidos em nenhum lado

Veja primeiramente que os dois parágrafos estão sem a propriedade clear definida.

```
<p>Parágrafo 1 - sem clear</p><p>Parágrafo 2 - sem clear</p>
```

Parágrafo 1 - sem clear  
Parágrafo 2 - sem clear

Agora veremos exemplo com o parágrafo 2 definido com a propriedade clear: left

```
<p>Parágrafo 1 - sem clear</p><p style="clear: left">Parágrafo 2 - com clear</p>
```

Parágrafo 1 - sem clear  
Parágrafo 2 - com clear

Observe que, no segundo exemplo, o browser não mostrou o parágrafo 2 ao lado da imagem por causa da propriedade clear, que com valor left, não permite que objetos flutuantes fiquem a esquerda do elemento.

### Posicionamento

Com certeza, esta é uma das características mais interessantes da especificação CSS, que permite ao programador um controle sobre os objetos da sua página que não seria possível usando apenas a linguagem HTML. Agora, podemos colocar nossos objetos exatamente onde quisermos.

O posicionamento em CSS funciona da seguinte maneira: quando o browser faz a "renderização" (ou seja, vai exibir um objeto na tela) de um objeto com posicionamento definido em CSS, ele é colocado dentro de um retângulo invisível, que chamaremos aqui de "caixa". Você então pode definir a distância exata desta caixa dos limites superior e inferior do seu browser ou definir a distância da caixa de outros elementos da página. Você pode ainda colocar os objetos em camadas, um acima do outro. Finalmente, podemos também tornar objetos visíveis ou invisíveis

**Importante:** o posicionamento em CSS é suportado somente pelos browsers Internet Explorer e Netscape Navigator nas versões 4 ou posterior.

## Propriedades do posicionamento

### position

Esta propriedade define o tipo de posicionamento do objeto, e pode ter os seguintes valores:

Valor	Descrição
Static	O elemento será posicionado normalmente dentro da página sem nenhuma característica especial aplicada a ele.
Absolute	O posicionamento é definido em relação às margens superior e esquerdo do browser
Relative	O posicionamento do objeto é definido em relação a outro objeto na página

Veremos exemplos destes três tipos de posicionamento a seguir.

### Posicionamento absoluto

Veja o exemplo a seguir:

```
<html>
<head>
<title>Teste de posicionamento absoluto</title>

<style>
<!--
H2 {position: absolute; left: 100px; top: 50px}
-->
</style>
</head>
<body>
<h2>Teste de posicionamento absoluto</h2>
</body>
</html>
```

A regra acima diz ao browser para posicionar todo o texto <H2> a exatamente 100 pixels de distância do canto esquerdo do browser e a 50 pixels do canto superior.

### left e top

Estas duas propriedades definem a distância do canto esquerdo (**left**) e do canto superior (**top**) que o seu objeto deve ficar em relação ao browser ou a outro objeto, dependendo do tipo de posicionamento adotado com a propriedade **position**. Para dedinição dos valores destas propriedades, você pode usar as unidades de medidas já conhecidas pelo CSS: pixels (px), points (pt), etc. Você também pode definir valores em porcentagem, que estará referenciando o tamanho do objeto de referência.

### Posicionamento relativo

Veja o exemplo a seguir:

```
<html>
<head>
<title>Teste de posicionamento relativo</title>
<style>
<!--
#offset {position: relative; left: 100px; top: 50px}
-->
</style>
</head>
```

```

<body>
<p>Este texto está posicionado normalmente na página, enquanto que a linha de baixo
terá um distanciamento do final desta linha.</p>
<p id="offset">Este texto está usando o posicionamento relativo em relação a
linha de cima</p>
</body>
</html>

```

Enquanto que no posicionamento absoluto definimos o posicionamento de um objeto em relação às margens esquerda e superior do browser, no posicionamento relativo você define o posicionamento em relação ao objeto anterior no código HTML. Veja que no exemplo acima criamos uma regra chamada "offset" que define o posicionamento relativo. O primeiro parágrafo está posicionado normalmente, e o segundo está a 100 pixels à esquerda e 50 pixels abaixo das margens do parágrafo anterior.

Vamos ver outro exemplo de posicionamento relativo:

```

<html>
<head>
<title>Teste de posicionamento relativo</title>
<style>
<!--
I {position: relative; left: 10px; top: 50px}
-->
</style>
</head>
<body>
<H2>Este texto está posicionado normalmente na página, <I> e agora
este texto tem uma pequena distância do texto </I></H2>
</body>
</html>

```

A tag <I>, que estava com posicionamento relativo, usou como ponto inicial o final do texto anterior. Temos que tomar cuidado com isso, pois assim que você termina de aplicar o posicionamento relativo, o fluxo do texto volta ao normal, e podemos ter problemas com objetos sobrepostos, como mostra o exemplo abaixo:

```

<html>
<head>
<title>Teste de posicionamento relativo</title>
<style>
<!--
I {position: relative; left: 10px; top: 15px; color: navy}
-->
</style>
</head>
<body>
<H2>Este texto está posicionado normalmente na página, <I> e agora
este texto está usando o posicionamento relativo </I> e novamente o texto voltou
normal</H2>
</body>
</html>

```

## width e height

Como falamos anteriormente, no posicionamento em CSS os objetos estão contidos dentro de "caixas" invisíveis. Com as propriedades "width" e "height", podemos controlar, respectivamente, a largura e a altura desta "caixa" que contém o objeto. Veja o exemplo abaixo:

```

<html>
<head>
<title>Teste de posicionamento de caixa</title>
<style>
<!--
h2 {position: absolute; left: 100px; top: 150px; width: 200px; height: 200px; background-
color: navy;color:yellow}
-->
</style>
</head>
<body>
<H2>Veja como este texto está dentro de uma caixa azul com fontes amarelas</H2>
<P>E o resto do texto continua normal...</P>
</body>
</html>

```

**Importante:** a propriedade `width` trabalha apenas com elementos com posicionamento absoluto. O Netscape Navigator não suporta `height`.

Para os valores destas propriedades, pode-se usar uma das unidades de medida padrão do CSS, ou valores em porcentagem, que se referirão ao tamanho do objeto anterior. Nos browsers Internet Explorer versões 4 e 5, esta propriedade também trabalha com imagens, o que nos permitiria esticar ou comprimir um gráfico via CSS.

### DIV e SPAN

Estas tags na verdade, deveriam estar na primeira parte do nosso tutorial, mas como elas vão ser muito mais utilizadas aqui, decidimos deixá-las para esta parte. `<SPAN>` é usada para aplicar um estilo CSS para uma parte de uma linha de texto. Veja o exemplo abaixo:

```

<html>
<head>
<title>Teste do uso de SPAN</title>
<style>
<!--
#teste {position: relative; left: 10px; top: 50px}
-->
</style>
</head>
<body>
<H2>Este texto está posicionado normalmente na página, <I><SPAN id="teste"> e agora
este texto tem uma pequena distância do texto </SPAN></I></H2>
</body>
</html>

```

A vantagem do uso do `<SPAN>` neste exemplo, em relação a um exemplo parecido que vimos antes, é que não precisamos que todo o texto que contém tags `<I>` na página receba o estilo CSS. A tag `<DIV>` serve para aplicar um estilo CSS a um bloco de texto, mesmo que este bloco inclua outras tags HTML. Veja o exemplo a seguir:

```

<html>
<head>
<title>Teste do uso de DIV</title>
<style>
<!--
#teste {color: red}
-->
</style>
</head>
<body>

```

```

<DIV id=teste>
<H2>Linha 1 dentro da tag DIV</H2>
<P>Linha 2 dentro da tag DIV</P>
</DIV>
<P>Linha 3 fora da tag DIV</P>
</body>
</html>

```

Veja esta página [aqui](#). O que tiver dentro da tag <DIV> irá ter a cor vermelha.

### overflow

**Importante:** esta propriedade só funciona no Internet Explorer versões 4 ou superior. No caso do conteúdo do seu texto não caber no tamanho da "caixa" que você especificou podemos usar a propriedade [overflow](#), que pode ter os seguintes valores:

Valor	Descrição
visible	todo o conteúdo da caixa será mostrado, mesmo se "transbordar" os limites da caixa
hidden	o browser não irá mostrar o conteúdo que fica fora dos limites da caixa.
auto	o browser manterá o tamanho da caixa, e mostrará uma barra de rolagem para ver o resto do conteúdo, se necessário
scroll	o browser sempre mostrará uma barra de rolagem.

Veja o exemplo abaixo:

```

<html>
<head>
<title>Teste de posicionamento de caixa</title>
</head>
<body>
<DIV style="left: 100px; top: 150px; width: 200px; height: 100px; background-
color: navy; color: yellow; overflow: scroll">
<H2>Veja como este texto está dentro de uma caixa azul com fontes amarelas e se o texto
todo cabe dentro da caixa</H2>
</DIV>
</body>
</html>

```

### z-index

**Importante:** esta propriedade apresenta problemas no Internet Explorer, mas funciona bem no Netscape Communicator.

Com esta propriedade, já podemos posicionar nossos objetos em camadas de três dimensões! Quando temos objetos que se sobrepõem, podemos usar esta propriedade para definir qual irá ficar na frente. Vejamos alguns exemplos:

```

<html>
<head>
<title>Teste do z-index</title>
</head>
<body>
<H2 style="position: relative; left: 10px; top: 0px; z-index:5; color:red">Veja a linha
vermelha</H2>
<H1 style="position: relative; left: 30px; top: -50px; z-index:3; color:blue">Veja a linha
azul</H1>

```

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

Veja o resultado a seguir:

Veja a linha vermelha Veja a linha azul

Agora, se trocarmos o código para:

```
<H2 style="position: relative; left: 10px; top: 0px; z-index: 1; color: red">Veja a linha
vermelha</H2>
<H1 style="position: relative; left: 30px; top: -50px; z-index: 2; color: blue">Veja a linha
azul</H1>
```

Nos casos acima, quem tiver o maior valor de z-index, aparecerá no topo da tela. Os valores usados para esta propriedade devem ser inteiros. Pode ser usado tanto no posicionamento absoluto como no relativo.

### visibility

**Importante:** esta propriedade não funciona no Netscape Communicator.

Com esta propriedade, podemos tornar elementos da nossa página invisíveis. Os valores possíveis para esta propriedade são:

Valor	Descrição
visible	torna um elemento visível
hidden	torna um elemento invisível
inherit	herda a propriedade de visibilidade do objeto anterior

Quando um elemento está invisível, ele continua ocupando seu espaço na janela do browser, mas não podemos vê-lo. Esta propriedade é muito usada em DHTML. Vamos ver um exemplo:

```
<html>
<head>
<title>Teste da invisibilidade</title>
<STYLE type="text/css">
<!--
.hidden {position: relative; visibility: hidden}
-->
</STYLE>
</head>
<body>
<H2><SPAN class="hidden">Este texto está invisível na página, </SPAN> mas você vê que o
texto invisível ocupa espaço no browser</H2>
</body>
</html>
```

Veja este exemplo em funcionamento [aqui](#).

### clip

Com esta propriedade, podemos tornar que partes do elemento estarão visíveis e as que ficarão invisíveis. Veja um exemplo abaixo:

```
<html>
<head>
<title>Teste de clipping</title>
<STYLE type="text/css">
<!--
.plain {position: absolute; top: 200px; left: 200px; width: 150px; height: 150px; background-
color: yellow;}
.clip {position: absolute; top: 200px; left: 200px; width: 150px; height: 150px; color: yellow;
```

```

background-color: blue; clip: rect(25 px 125px 125px 25 px); }
-->
</STYLE>
</head>
<body>
<DIV class="plain"><P>Este texto é coberto pelo quadrado azul. Mas uma vez que o
quadrado está cortado, alguma parte do seu texto será mostrado</P></DIV>
<DIV class="clip"><P>Este texto é de cor amarela num quadrado azul, mas está sendo
cortado pelo clip</P></DIV>
</body>
</html>

```

Definimos duas caixas de mesmo tamanho nas classes "plain" e "clip". Só que em "clip", recortamos através da propriedade clip uma área retangular (através da expressão `rect`), que é a única suportada no momento. A sequência de dados na expressão `rect` é a seguinte: distância da parte superior do elemento; o canto direito está a a uma distância x da esquerda da caixa; o canto inferior está a uma distância x do topo da caixa; e o canto esquerdo está a uma distância x do canto esquerdo da caixa.

Nem tudo é perfeito. CSS era para ser uma especificação padrão (foi definido pela WWW Consortium - W3C), que traria grandes inovações no desenvolvimento de páginas web, mas é barrada pelo suporte dos browsers a esta especificação.

O primeiro browser a tentar suportar style sheets foi o IE 3.0, ainda sem a especificação oficial da W3C, e por isso ele suporta muitas propriedades do CSS, mas não todas.

As versões 4 do Netscape Navigator e Internet Explorer suportam style sheets, mas cada um implementou à sua maneira as propriedades CSS. Existem propriedades que um navegador suporta e outro não. E certas propriedades funcionam de maneira diferente nos dois browsers. Isto torna a tarefa de criar páginas com CSS que sejam multiplataformas e multibrowsers bem difícil. A única maneira é testar nos diversos ambientes para verificar se as definições de estilo funcionam corretamente.

Existe uma tabela no site *WebReview.com* (<http://webreview.com/wr/pub/guides/style/mastergrid.html>) que mostra as declarações CSS e a compatibilidade delas com os browsers. É importantíssimo para quem programa com CSS ter esta tabela à mão.

Quando você coloca um radio button numa tabela com cor de fundo, geralmente o efeito obtido no Netscape Navigator (não detectamos este problema no Internet Explorer 5) não é bom. Veja o código abaixo:

```

<table border="1" cellpadding="0" cellspacing="0">
<tr>
<td bgcolor="#0000FF"><font color="#FFFF00">Tabela de Teste - Veja os radio button
abaixo: </font>
<form method="POST" action="--WEBBOT-SELF--">
<p><input type="radio" value="V1" checked name="R1">
<font color="#FFFF00">Opção 1</font></p>
<p><input type="radio" name="R1" value="V2"> <font color="#FFFF00">Opção 2</font></p>
</form>
</td>
</tr>
</table>

```

Veja a cor branca de fundo no radio button. Qualquer webdesigner que se preze não pode deixar que isso aconteça. Recentemente, vi em ([http://www.netmechanic.com/news/vol3/html\\_no1.htm](http://www.netmechanic.com/news/vol3/html_no1.htm)) uma pequena adição de um estilo CSS ao código que resolve este problema. Veja a mudança:

```

<input type="radio" value="V1" checked name="R1" style="background:#0000FF;color:blue">

```

O valor `background` deve ser preenchido com o código da cor (saiba mais sobre cores em CSS [aqui](#)), que deve ser o mesmo da cor de fundo da tabela. O valor `color` não influi em nada aqui, mas deve ser colocado. Ponha apenas um valor válido.

Para atribuir uma determinada cor a alguma propriedade em CSS, usamos uma das seguintes maneiras de representá-la:

1- Usando um nome de cor conhecido. Veja o exemplo abaixo:

```
<html>
<head>
<style TYPE="text/css">
<!--
H2 {background-color: red}
-->
</style>
</head>
<body>
<h2>Teste de cor de fundo para um cabeçalho</h2>
</body>
```

Este código gera um cabeçalho com cor de fundo vermelho (red). A tabela abaixo mostra os nomes de cores válidos:

nome	cor
aqua	azul claro
black	preto
blue	azul
fuchsia	rosa
gray	cinza
green	verde
lime	verde-limão
maroon	marrom
navy	azul escuro
olive	marrom claro
purple	roxo
red	vermelho
silver	prata
teal	verde-água
white	branco
yellow	amarelo

2- Usando uma representação RGB - Aqui, cada cor é formada por 3 componentes: **vermelho** (red), **verde** (green) e **azul** (blue). A combinação das intensidades destes 3 componentes forma uma cor. A intensidade é representada por números entre 0 e 255. Por exemplo, a cor branca é formada pela combinação das intensidades máximas dos três componentes: 255 de vermelho, 255 de verde e 255 de azul. A cor verde é formada apenas pelo componente verde: 0 de vermelho, 255 de verde e 0 de azul. Se quiser um verde mais escuro, basta diminuir a intensidade do componente verde. Por exemplo, 0 de vermelho, 140 de verde e 0 de azul. Vamos fazer um teste? Use o código abaixo, alterando os valores dos componentes RGB (de 0 a 255) para verificar novas cores.

```
<html>
<head>
<style TYPE="text/css">
<!--
H2 {background-color: rgb(100,120,100)}
-->
</style>
</head>
<body>
```

```
<h2>Teste de cor de fundo para um cabeçalho</h2>  
</body>
```

### **Teste de cor de fundo para um cabeçalho**

Em CSS, podemos representar uma cor RGB das seguintes maneiras:

```
#<vermelho_hex><verde_hex><azul_hex>  
rgb(<vermelho>,<verde>,<azul>)  
rgb(<%vermelho>,<%verde>,<%azul>)
```

Na primeira representação, os valores para cada componente estão na forma de números hexadecimais de dois dígitos. Portanto, para representar a cor verde, que tem apenas o componente verde com 255 (que equivale a FF em hexadecimal), teríamos o seguinte: #00FF00.

Na segunda forma, que é a mais fácil, basta colocar os valores para cada componente, entre 0 e 255. Por exemplo, para a cor azul teríamos: rgb(0,0,255)

Na terceira forma, representamos as intensidades das cores em forma de porcentagem, entre 0% a 100%. Para a cor azul, teríamos: rgb(0%,0%,100%)