**CSS**

CSS significa em inglês: Cascading Style Sheet, que traduzindo fica: Folha de Estilo em Cascata. É usado para alterar cores, fontes, tamanho, espaçamentos, etc. Substitui o marcador <font> da seguinte forma:

|  |
| --- |
| <style type="text/css">  .style1{  font-family:verdana;  font-size:15px;  font-weight:bold;  color:#FF0000;  }  </style> |

**Noções básicas**

Existem três métodos de utilizar o CSS para alterar o código HTML: o primeiro é com o marcador <style type="text/css"> dentro das delimitações do marcador <head>, chamado de *Método Incorporado*; outro é usando o atributo “style” dentro do marcador, como, por exemplo, em uma DIV, chamado de *Método InLine*. O código deve ficar assim:

|  |
| --- |
| <DIV style=“font-family:verdana;font-size:15px;font-weight:bold;  color:#FF0000;”>  </DIV> |

O último método, chamado de *Método Externo*, é criando um arquivo com atributos, mas será necessário adicionar o marcador <link href="arquivo.css" rel="stylesheet" type="text/css" /> dentro das limitações do marcador <head>. Na primeira opção, o código fica dentro da página HTML, e na outra, fica em um arquivo separado. A vantagem de usar em um **arquivo.css** é que as alterações servem para todas as páginas que estiverem ligadas AA esse arquivo.

Os atributos do CSS ficam dentro de chaves {}. Podemos fazer quantos estilos quisermos. Criando um estilo de texto com fonte Verdana, tamanho 12 e cor Verde, o código ficará dessa forma:

|  |
| --- |
| .style1{  font-family: verdana; /\* nome da fonte \*/  font-size: 12px; /\* tamanho da letra \*/  color: #00CC33; /\* cor da letra \*/  } |

O nome do estilo (style1) inicia com um ponto pois estamos fazendo um estilo para *texto*, mas quando mudamos para *tabelas* (table) ou uma *DIV*, substituímos o ponto por um jogo da velha (#nomedatabela ou #nomedadiv). Todos os atributos são finalizados com ponto e vírgula (;) e a atribuição dos valores é realizada pelos dois pontos (:).

Se quisermos alterar a propriedade de uma tabela <table>, devemos atribuir um “name” ou um “id” para o marcador <table>, ficando <table id=”table1”> ou <table name=”table1”>. No CSS ficará #table1 {}, a menos que o nome ou id designado para a tabela seja diferente de table1. O mesmo acontece com os marcadores DIV, SPAN, etc.

Dentro das chaves, vão os atributos. Por exemplo, se eu quiser que a table1 fique com o fundo preto, altura de 200 pixels e largura de 500 pixels usando CSS, o código ficará assim:

|  |
| --- |
| #table1{  background-color:#000000; /\* cor de fundo \*/  width:500px; /\* largura \*/  height:200px; /\* altura \*/  } |

Embora não seja aconselhável, você pode usar a palavra “importante” para alterar a precedência de um estilo. No exemplo abaixo, o estilo atribuído ao *H1* sobrepõe o estilo definido junto com o elemento graças ao uso da palavra *important*:

h1{

color:red !important;

font-weight:bold;

font-family:sans-serif !important;

}

**Criando Style Sheets para cada tipo de Mídia**

CSS permite também estilos diferentes para cada tipo de mídia utilizada para apresentação deste estilo. Isto dá um enorme poder para a CSS, permitindo a utilização de um documento por portadores de deficiência visual ou aditiva. Permite também a aplicação de estilos especificamente para a impressão de documentos, o que resolve uma grande deficiência dos navegadores. Tanto o IE quanto o Netscape possuem dificuldades na hora de paginar documentos para impressão. Observe a utilização nos exemplos abaixo:

|  |
| --- |
| @media print{  body{font-size:10pt;}  } |
| @media screen{  body{font-size:12pt;}  } |
| @media screen, print{  body{line-weight:1.2;}  } |

A tabela abaixo descreve cada um dos estilos possíveis com a CSS:

|  |  |
| --- | --- |
| **Tipo** | **Descrição** |
| Screen | Telas de computadores |
| Print | Impressoras |
| Projection | Projeções |
| Braile | Dispositivos Braile |
| Speech | Sintetizadores de voz |
| All | Todos os dispositivos |

**Edição de links**

Os links , por padrão, já vem na cor azul e sublinhado. Para mudar o estilo dos links, primeiro, deve-se criar um link, como <a href="#">Link</a>, por exemplo. Então, cria-se a conexão com o CSS utilizando qualquer um dos procedimentos previamente citados e inserimos e seguinte código:

|  |
| --- |
| <style type="text/css">  <!--  a:link {  text-decoration:none; /\* tira a decoração do texto \*/  color:#000000; /\* cor da letra \*/  }  -->  </style> |

Podemos, também, fazer com que as características mudem ao passar o cursor do mouse por cima do link pelo “hover”, tendo em vista que esta técnica se aplica a qualquer outro texto (ou imagem). No código a:hover {}, o “a”refere-se à marcação <a> e pode ser substituído por qualquer outra; dentro das chaves vão os parâmetros que vão se aplicar caso o cursor do mouse passe por cima da marcação <a>, no caso, o link. Continuando no assunto de links, temos:

*a:link -* Para editar os links de primeira impressão;

*a:visited -* Para editar apenas os links já visitados;

*a:hover -* Para editar os links quando o cursor do mouse estiver em cima deles;

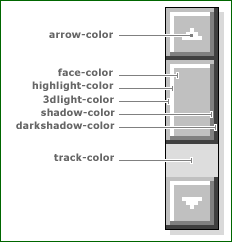
*a:active -* Para editar os links quando ele for ativado (clicado).

\*Essas chaves devem ser usadas na sequência que foi colocada, e nunca fora de ordem (*link*, *visited*, *hover* e *active*).

**Alterar a barra de rolagem (válido somente para Internet Explorer 7 e abaixo)**

Para alterar a barra de rolagem é necessário colocar o marcador <style> dentro do marcador <head> (ou quaisquer das opções dispostas) e alterar o marcador <body> da seguinte forma:

|  |
| --- |
| <style type="text/css">  body{  scrollbar-face-color:“Cor”;  scrollbar-shadow-color:“Cor”;  scrollbar-highlight-color:“Cor”;  scrollbar-3dlight-color:”Cor”;  scrollbar-darkshadow-color:“Cor”;  scrollbar-track-color:”Cor”;  scrollbar-arrow-color:”Cor”;  }  </style> |



**Agrupando seletores juntos em uma regra simples**

Caso seja necessária a definição de um mesmo estilo para vários elementos, esta pode ser feita listando os estilos individualmente. Entretanto, existe uma maneira mais eficiente de fazer a mesma ação. Os elementos (ou seletores) podem ser agrupados de modo a definir uma regra única para todo o grupo. No exemplo abaixo, os seletores são agrupados e uma única regra é atribuída:

|  |
| --- |
| p, ul, li{font-size:12pt;} |

**Definindo relacionamentos Pai-Filho nas regras**

Usando CSS, é possível especificar quando um estilo é aplicado à um elemento. Por exemplo, você pode querer definir dois estilos para o elemento <li>: um que se aplica quando ele for filho do elemento <ul> e outro quando ele for filho do elemento <ol>. Para tanto, você utiliza seletores contextuais.

|  |
| --- |
| ol li{list-style-type:decimal;} |
| ul li{list-style-type:square;} |

Note que a lista de seletores não é separada por vírgula. Separando cada seletor por vírgula, os estilos atribuídos serão aplicados àquele grupo de elementos.

**Entendendo Herança**

Os elementos herdam certas propriedades de seus pais. Por exemplo, todos os elementos dentro do elemento <body> (<p> e <ul>) herdam certas propriedades deste. Assim como o elemento <li> herda propriedades do elemento <ul> no qual ele está contido. Veja o exemplo:

|  |
| --- |
| <style type=”text/css”> p{color:blue;} </style>  <body>  <p>Olá. Este é um parágrafo do texto. <em>Este texto recebeu ênfase</em>  </p>  </body> |

A Style Sheet para este exemplo muda a cor do elemento <p> para azul. Não há definição para o elemento <em>. Devido ao fato do elemento <em> estar dentro do elemento <p>, <em> herda a propriedade color do elemento <p>.

**Trabalhando com Classe em Style Sheets**

Uma classe define a variação de um estilo o qual é referenciado através de uma ocorrência específica de um elemento utilizando CLASS . Por exemplo, você pode definir três variações do estilo H1 e então usa cada variação em um contexto apropriado. Uma classe é definida normalmente, como é definido um estilo. Apenas é adicionado o nome da classe no final do elemento, separado por ponto, assim como no exemplo:

|  |
| --- |
| h1.blue{color:blue;}  h1.red{color:red;}  h1.black{color:black;} |

Quando o elemento <h1> é adicionado ao documento, o atributo CLASS é utilizado para atribuir qual estilo será utilizado.

|  |
| --- |
| <h1 class=”red”>Red Heading</h1> |

Uma classe não precisa estar ligada à um elemento. A classe podem ser declaradas sem nenhum elemento. Assim, qualquer elemento que utilize aquela classe utilizará aquele estilo:

|  |
| --- |
| .red {color:red;}  <h1 class=”red”>Red Heading</h1>  <p class=”red”>Red Paragraph</p> |

**Definindo um estilo para um elemento específico**

Como já discutido anteriormente, você pode declarar estilos junto com o elemento. Entretanto, para facilitar a leitura do documento, você pode atribuir um ID (nome) aos elementos e então associar estilos à este ID. Para declarar o estilo para um ID é usado o símbolo #. Somente aos atributos com aquele ID serão aplicados os estilos.

|  |
| --- |
| #test{color: red;}  <p id=test>This is a test paragraph</p> |

**Usando pseudo-classes e pseudo elementos**

Em CSS, um estilo normalmente é associado à um elemento com base na posição que este elemento ocupa dentro da estrutura hierárquica do documento. Isto é suficiente para uma grande variedade de estilos, mas não atende a alguns efeitos comuns. O conceito de pseudo-classes e de pseudo-elementos veio atender esta área, permitindo que informações exteriores possam influenciar o processo de formatação do documento. Pseudo-classes e pseudo-elementos são usados para especificar sub-partes de elementos, enquanto pseudo-classes permitem às folhas de estilo diferenciar entre tipos diferentes de elementos. Abaixo, vemos um exemplo de utilização:

|  |  |  |
| --- | --- | --- |
| **Seletor** | **Exemplos** | **Descrição dos exemplos** |
| :link | a:link | Seleciona os links não-visitados |
| :visited | a:visited | Seleciona os links visitados |
| :active | a:active | Seleciona os links ativos |
| :hover | a:hover | Seleciona os links em que o mouse estiver sobre |
| :focus | input:focus | Seleciona o elemento que tem o foco |
| :first-letter | p:first-letter | Seleciona a primeira letra de cada elemento <p> |
| :first-line | p:first-line | Seleciona a primeira linha de cada elemento <p> |
| :first-child | p:first-child | Seleciona os primeiros filhos de <p> |
| :before | p:before | Insere um conteúdo antes de cada elemento <p> |
| :after | p:after | Insere um conteúdo depois de cada elemento <p> |
| :lang(language) | p:lang(it) | Seleciona todo elemento <p> que começa com “it” |

Todos os elementos ‘a’ com um atributo ‘href’ serão classificados em um, e apenas um, destes grupos (ou seja, as definições das posições locais dos links não serão afetadas). Os navegadores podem decidir mover um elemento de um gruo à outro (de ‘ visitado’ para ‘não visitado’, por exemp0lo, após um certo tempo). Um ‘link ativo’ é aquele que está sendo selecionado no momento pelo leitor (está sendo ‘clicado’).

**Entendendo posicionamento de elementos**

No mundo das publicações, camadas são blocos retangulares e texto e imagens que podem ser posicionados em qualquer lugar da página. Estas camadas podem ser ocultas e sobrepostas. O CSS permite a construção de blocos de texto que se sobrepõe, assim como ocultar ou exibir determinados blocos.

**Posicionando um elemento HTML**

O CSS fornece duas maneiras de posicionar elementos no navegador: RELATIVE e ABSOLUTE. No modo RELATIVE, o navegador posiciona o elemento em relação à sua posição normal, ou seja, onde o elemento deveria realmente aparecer. No modo ABSOLUTE, o navegador posiciona o elemento em relação ao elemento pai. Para posicionar o elemento, são utilizadas as propriedades TOP e LEFT:

|  |
| --- |
| #element{position:absolute;top:100;left:20;} |

No exemplo acima, o elemento é posicionado à 100 pixels da borda superior do elemento pai e à 20 pixels da borda esquerda do elemento pai. Quando o elemento a ser posicionado não possui pai, este é posicionado em relação às bordas do navegador.

A unidade de medida padrão para LEFT e TOP é pixels. Podem ser usados também, valores percentuais, que equivale ao percentual do elemento pai. O exemplo abaixo posiciona uma imagem à 40 pixels do topo e 100 pixels da borda esquerda do elemento <body>. Como no elemento <body> não existe posicionamento definido e ele não tem nenhum elemento pai, este assume a posição 0, 0.

|  |
| --- |
| <html>  <style type=”text/css”>  #example{  position: absolute; top:40; left:100;  }  </style>  <body>  <img id=”example” src=”init.gif”>  </body>  </html> |

**Mudando o tamanho de um elemento**

Assim como é possível controlar a posição de um elemento, é possível, também, controlar a dimensão do elemento Para isto, são usadas as propriedades WIDTH para largura e HEIGHT para altura. Assim como LEFT e TOP, a unidade de medida padrão é pixels, mas podem ser usados valores percentuais. Também pode ser usado o valor AUTO para que o navegador automaticamente determine o tamanho do elemento.

|  |
| --- |
| #element {position: aboslute; top:100; left:20; width:100; height:100} |

A propriedade WIDTH **não** é usada para definir exatamente a largura do elemento. Esta propriedade serve apenas para o caso em que haja algum elemento dentro do qual é definido pela propriedade WIDTH e este seja maior que o mesmo.

***Overflow***

Esta propriedade permite que seja ocultado o conteúdo de uma camada para mostrar unicamente o conteúdo que caiba, segundo suas dimensões. Para ter acesso ao conteúdo que não cabe na camada, pode se configurar overflow de modo que apareçam barras de rolagem.

Assim, passemos diretamente a ver quais são os atributos possíveis com o atributo overflow:

*visible:* Este valor indica que deve ser exibido todo o conteúdo da camada, ainda que não caiba no tamanho com a que a tenhamos configurado. No Internet Explorer ocorre que a camada cresce em tamanho o suficiente para que caiba todo o conteúdo que colocamos dentro. No Firefox, a camada tem o tamanho marcado, mas o conteúdo continua sendo visto, embora fora do espaço da camada, podendo sobrepor-se a um texto ou a uma imagem que haja embaixo. O conteúdo não se oculta em nenhum caso, ou seja, sempre estará visível.

*hidden:* Este valor faz com que os conteúdos que não caibam na camada pelo seu tamanho, ocultem-se. Por isso, a camada terá sempre o tamanho configurado, mas os conteúdos em, algumas ocasiões, poderão não ser vistos por completo.

*scroll:* Este valor indica que a camada deve ter o tamanho que foi configurado inicialmente e que, além disso, se devem mostrar barras de rolagem, para mover o conteúdo da camada dentro do espaço da mesma. As barras de rolagem sempre aparecem, solicitadas ou não.

*auto:* Com este valor também se respeitarão as dimensões atribuídas a uma caixa. O conteúdo será ocultado, mas aparecerão as barras de rolagem para movê-lo. No entanto, neste caso, as barras de rolagem poderão aparecer ou não, dependendo da necessidade ou não para exibir todo o conteúdo da camada.

Assim, o atributo overflow nos permitirá ter um maior controle sobre os espaços que destinamos a cada caixa de nosso design. É muito utilizado para mostrar textos longos, que se desejam integrar dentro de outro texto ou uma interface onde não temos espaço disponível para colocá-los ou não desejamos que cresçam mais da conta.

É possível especificar, também, em quais bordas desejamos que tenha a barra de rolagem através do overflow-x (para borda horizontal inferior) e overflow-y (para borda vertical à direita).

***Word-wrap***

No modelo de caixa dos navegadores, as palavras se vão colocando em linhas para ocupar toda a largura disponível na caixa. Pode surgir um problema quando temos uma palavra muito longa, que não cabe na largura disponível do container onde se colocou. Então o que ocorre é que a palavra aparece por fora da caixa, ou inclusive pode ocorrer que a caixa se deforme de tamanho, fazendo que o elemento container amplie a largura o suficiente para que fique a palavra. Em qualquer caso, o efeito resultante costuma ser desagradável, porque muitas vezes faz com que o layout fique mal construído.

Para evitar este efeito, existe o atributo word-wrap, que serve para especificar que as palavras que sejam longas demais devem se “cortar”, de maneira que caibam na largura disponível da caixa.

O atributo word-wrap tem dois possível valores: normal ou break-word.

*word-wrap: normal;*

Faz com que as palavras não se cortem, o que seria o comportamento normal que conhecíamos até agora, ocorrendo que as palavras longas possam desenquadrar nosso layout.

*word-wrap: break-word;*

Com este outro valor de word-wrap conseguimos que a palavra se “quebre” para caber na largura que havíamos definido.

**Sobrepondo elementos**

Utilizando as propriedades TOP e LEFT, é possível sobrepor elementos e a ordem de exibição é a mesma ordem em que eles estão no código, ou seja, o último bloco de código será o último bloco a ser montado na tela, sobrepondo os outros. Esta ordem de sobreposição pode ser alterada utilizando a propriedade Z-INDEX.

A propriedade Z-INDEX aceita valores inteiros positivos. Um elemento com a propriedade Z-INDEX igual à 10 sobrepõe um elemento com a propriedade Z-INDEX igual à 1.

O exemplo abaixo mostra 2 elementos usando a propriedade Z-INDEX para manipular a ordem de exibição:

|  |
| --- |
| <style type=”text/css”>  #ex1 {position: absolute; top:40; left:60;z-index:2;}  #ex2 {position: absolute; top:80;left:200;z-index:1;}  </style>  <body>  <div id=”ex1”>  <b>This is the first element. It’s in the middle.</b><br />  <b>This is the first element. It’s in the middle.</b><br /></div>  <div id=”ex2”>  <b>This is the second layer. It’s behind the first layer.</b><br />  <b>This is the second layer. It’s behind the first layer.</b><br /></div>  </body> |

**Aninhando posicionamento de elementos**

Nem sempre alguns elementos precisam estar contidos dentro de outros. Entretanto, esta prática pode ser interessante, visto que quando você posiciona o elemento pai, todos os elementos filho seguirão aquele posicionamento. Segue o estilo abaixo.

|  |
| --- |
| <style type=”text/css”>  #test1  {background-color: blue; position: absolute; top: 50; left: 60; width: 150}  #test2  {background-color: yellow; position: absolute; top: 70; left: 30; width: 150;}  </style> |

Veja o comportamento das páginas abaixo. A esquerda possui 2 elementos distintos. O posicionamento do ID test1 não influencia no ID test2. Já no outro exemplo, o posicionamento do ID test1 está diretamente ligado ao posicionamento do ID test2.

|  |  |
| --- | --- |
| <div id=“test1”>Título</div>  <div id=“test2”>  Aqui fica o texto da mensagem.  </div> | <div id=“test1”>Título<div id=“test2”>  Aqui fica o texto da mensagem.  </div>  </div> |

**Unidades para cores**

Uma cor é indicada ou por uma palavra-chave ou por um código RGB. As palavras-chave sugeridas para indicar cores são: ‘aqua’, ‘black’, ‘fuschia’, ‘gray’, ‘green’, ‘lime’, ‘maroon’, ‘navy’, ‘olive’, ‘purple’, ‘red’, ‘silver’, ‘teal’, ‘white’ e ‘ yellow’. Estas 16 cores foram tiradas da paleta de cores VGA do Windows, e seus valores RGB estão definidos nesta especificação.

A codificação RGB é usada para a indicação de cores no formato numérico. Todas as atribuições a seguir indicam a mesma cor (vermelho):

EM{color: #F00} /\*#rgb\*/

EM{color: #FF0000} /\*#rrggbb\*/

EM{color: rgb(255,0,0)} /\*inteiro entre 0 e 255\*/

EM{color: rgb(100%,0%,0%)} /\*percentual entre 0.0% e 100.0%\*/

**Outros atributos**

**Retirar borda padrão de formulários dos navegadores**

Usando exemplo do Google Chrome, temos uma borda amarela padrão ao passar o mouse acima dos formulários. Em alguns formulários, é necessário deixar somente a borda designada e para retirar a borda padrão do navegador, basta utilizar o seguinte código:

|  |
| --- |
| input{  outline:none;} |

Caso queria aplicar somente à campos específicos, basta inserir após o input, colchetes com as especificações do campo.

|  |
| --- |
| input[type=text]{outline:none;} /\*Remove do tipo texto\*/  input[type=password]{outline:none;} /\*Remove do tipo password\*/ |

**Atributos de CSS**

|  |  |  |
| --- | --- | --- |
| **Atributo** | **Função** | **Valores** |
| background | Cor de fundo, imagem, transparência. | - |
| background-attachment | Rolagem do fundo/Marca d'água. | *Fixed* e *scroll*. |
| background-image | Imagem de fundo. | *URL*. |
| background-color | Cor de fundo ou transparência. | *Hexadecimal*, *rgb*, *nome da cor* e *transparent*. |
| background-position | Posicionamento da imagem de fundo. | *X-pox*, *y-pos*, *top*, *bottom*, *left*, *right* e *center* (devem ser usados 2 desses valores). |
| background-repeat | Configuração lado-a-lado da imagem de fundo. | *No-repeat*, *repeat*, *repeat-y* e *repeat-x*. |
| border | Largura, estilo e cor de todas as 4 bordas. | *Color (código hexadecimal, código rgb e nome da cor)*, *style (none, hidden, dotted, dashed solid, double groove, ridge, inset, outset),* e *width (thin, medium, thick e length (em pixels)).* |
| border-bottom | Largura, estilo e cor da borda inferior. | - |
| border-bottom-color | Cor da citada borda. | - |
| border-bottom-style | Estilo da citada borda. | - |
| border-bottom-width | Largura da citada borda. | - |
| border-color | Cor das 4 bordas. | - |
| border-left | Largura, estilo e cor da borda esquerda. | - |
| border-left-color | Cor da borda citada. | - |
| border-left-style | Estilo da borda citada. | - |
| border-left-width | Largura da borda citada. | - |
| border-right | Largura, estilo e cor da borda direita. | - |
| border-right-color | Cor da borda citada. | - |
| border-right-style | Estilo da borda citada. | - |
| border-right-width | Largura da borda citada. | - |
| border-style | Estilo de todas as 4 boras. | - |
| border-top | Largura, estilo e cor da borda superior. | - |
| border-top-color | Cor da borda citada. | - |
| border-top-style | Estilo da borda citada. | - |
| border-top-width | Largura da borda citada. | - |
| border-width | Largura de todas as 4 bordas. | - |
| box-shadow | Põe sombra nos elementos | *hor*, *ver*, *blur* e *cor* |
| clear | Elementos flutuantes à esquerda ou à direita de um elemento. | - |
| clip | Parte visível de um elemento. | - |
| color | Cor de primeiro plano. | *Cód. hexadecimal*, *cód. rgb* e *nome da cor.* |
| cursor | Tipo de ponteiro do mouse. | - |
| display | Se o elemento é exibido e o espaço é reservado para ele. | - |
| direction | Direcionamento da escrita do texto. | *Ltr (left to right)* e *rtl (right to left).* |
| filter | Tipo de filtro aplicado ao elemento. | - |
| first-letter | Deixa a primeira letra com o efeito especificado. | *Font*, *color*, *background*, *margin*, *padding*, *border*, *text-decoration*, *vertical-align (somente em ”float:none;”)* , *text-transform*, *line-height*, *float* e *clear*. |
| first-line | Deixa a primeira linha com o efeito especificado. | *Font*, *color*, *background*, *word-*spacing, *letter-spacing*, *text-decoration*, *vertical-align*, *text-transform*, *line-height* e *clear*. |
| float | Se o elemento flutua. | - |
| font | Estilo, variante, peso, tamanho e altura do tipo de fonte. | - |
| @font-face | Incorporação da fonte ao arquivo HTML. | - |
| font-family | Tipo de fonte. | - |
| font-size | Tamanho da fonte. | - |
| font-style | Fonte itálico. | - |
| fonte-variant | Fonte bold. | - |
| font-weight | Peso da fonte de claro a negrito. | - |
| height | Altura exibida ao elemento. | - |
| @import | Folha de estilo a importar. | - |
| left | Posição do elemento em relação a margem esquerda da página. | - |
| letter-spacing | Distância entre as letras. | *Normal* e *length (em pixels).* |
| line-height | Distância entre linhas de base. | - |
| list-style | Tipo, imagem e posição do estilo da lista. | - |
| list-style-image | Marcador de item de lista. | *None* e *URL*. |
| list-style-position | Posição do marcador de item da lista. | *Outside* e *inside*. |
| list-style-type | Marcador de item de lista alternativo. | *None*, *disc*, *circle*, *square*, *decimal*, *decimal-leading-zero*, *lower-roman*, *upper-roman*, *lower-alpha*, *upper-alpha*, *lower-greek*, *lower-tain*, *upper-latin*, *hebrew*, *armenian*, *georgian*, *cjk-ideographic*, *hirigana*, *katakana*, *hiragana-iroha* e *katakana-iroha*. |
| margin | Tamanho de todas as 4 margens. | *Auto*, *length (em pixels)* e *%*. |
| margin-left | Tamanho da margem esquerda. | *-* |
| margin-right | Tamanho da margem direita. | *-* |
| margin-bottom | Tamanho da margem inferior. | *-* |
| margin-top | Tamanho da margem superior. | *-* |
| overflow | Exibição de imagens que são maiores do que suas molduras. | *Visible*, *hidden*, *scroll* e *auto.* |
| padding | Espaço em torno de um elemento em todos os lados. | *Auto*, *length (em pixels)* e *%*. |
| padding-bottom | Espaço a partir da margem inferior de um elemento. | *-* |
| padding-left | Espaço à esquerda do elemento. | *-* |
| padding-right | Espaço à direita do elemento. | *-* |
| padding-top | Espaço a partir da margem superior do elemento. | *-* |
| page-break-after | Inserir quebra de página depois de um elemento. | *-* |
| page-break-before | Inserir quebra de página antes de um elemento. | *-* |
| position | Como o elemento é posicionado na página. | *-* |
| text-align | Alinhamento do texto. | *Left*, *right*, *center*,e *justify*. |
| text-decoration | Sublinhado, sobrelinhado ou riscado. | *None*, *underline*, *overline*, *line-through* e *blink*. |
| text-indent | Recuo da primeira linha do parágrafo. | *Length (em pixels)* e *%*. |
| text-transform | Transformação para todas maiúsculas e minúsculas. | *None*, *capitalize*, *uppercase* e *lowercase*. |
| text-shadow | Põe sombra no texto. | *Top*, *left*, *down*, *right* e *color*. |
| top | Posição do elemento em relação à parte superior da página. | *-* |
| vertical-align | Alinhamento vertical do elemento. | *-* |
| visibility | Se elemento é visível ou invisível. | *-* |
| white-space | Regula leitura de espaços. | *Normal*, *pre* e *nowrap*. |
| width | Largura do elemento. |  |
| word-spacing | Espaçamento entre palavras. | *Normal* e *length (em pixels)*. |
| word-wrap | Quebra de linha de texto com tamanho em excesso. | *Normal* e *Break-word*. |
| z-index | Posição do elemento na pilha. | *-* |