Tutorial CSS

Desenvolvido por: Prof. Tiago Lopes Telecken
Instituto Federal do Rio Grande do Sul – Campus Rio Grande

1. Introdução

CSS é a abreviação de Cascading Style Sheet (em português: folhas de estilo em cascata). O CSS adiciona novos recursos gráficos e de design nas páginas Web, trata-se de um grande salto de qualidade e de novas possibilidades de se apresentar páginas Web.

Além de aumentar os recursos gráficos das páginas HTML, o CSS também possibilita uma organização do conteúdo Web mais avançada. Com o CSS é possível separar o conteúdo da especificação gráfica de uma página Web. Neste modelo o conteúdo de uma página web fica em uma página HTML. Já a aparência é especificada em uma página CSS. Tal separação traz inúmeras vantagens das quais destacam-se:

As páginas HTML ficam bem mais simples pois não precisam conter informações sobre a aparência. Esta característica facilita a tarefa de quem tem que construir e dar manutenção em páginas HTML. Além disto também é facilitada a tarefa de adicionar e dar manutenção em scripts no cliente (como códigos Javascript) e no servidor (como códigos PHP) pois estes scripts manipulam e são integrados as páginas HTML.

Uma mesma página HTML pode ter várias aparências, basta trocar o arquivo CSS que define a sua aparência. Pode existir uma página web padrão, uma para o natal, uma para o carnaval, uma especial para rodar em celulares, etc. Pode-se trocar de uma aparência para outra apenas alterando-se uma única linha na página HTML.

A aparência de um site inteiro pode estar definida em um arquivo CSS. Com isso pode-se alterar a aparência de centenas de páginas HTML alterando-se poucas linhas em um único arquivo CSS

A estrutura que permite a separação da aparência e do conteúdo é mostrada na figura 1:

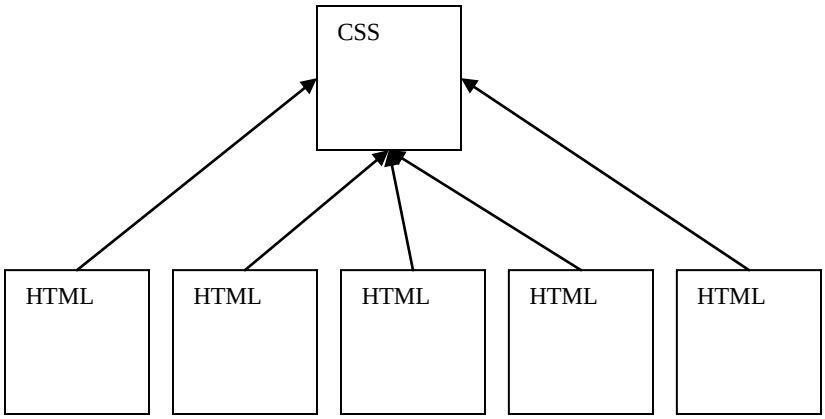

Figura 1 – Estrutura HTML + CSS

Na estrutura mostrada na figura 1, as páginas HTML contém somente o conteúdo do site (a página HTML deve conter textos, tabelas e listas sem especificações sobre suas cores, tipo de texto, tamanho de texto, cor de fundo, cor das bordas, etc). Adicionalmente as páginas HTML devem especificar a página CSS que definirá a sua aparência. A página CSS é que definirá cores, tipo de texto, tamanho de texto, cor de fundo, cor das bordas, etc.

Neste exemplo todas páginas HTML apontam para a mesma página CSS, logo se a página CSS define que os títulos das páginas (entre as tags <h1>) serão vermelhos, todos títulos de todas as páginas HTML serão vermelhos. Para trocar a cor de todos os títulos de todas as páginas basta alterar uma única linha no arquivo CSS.

2. Primeira página CSS

Exercício 1

Para construir sua primeira página HTML com a aparência definida numa página CSS recomendamos que os alunos leiam e executem todos os passos definidos no tutorial abaixo:

http://www.maujor.com/w3ctuto/firstcss.html

Este tutorial foi feito por Bert Bos e traduzido para o português por Maurício Samy Silva. O tutorial apresenta noções básicas sobre como trabalhar com arquivos CSS e como liga-los a páginas HTML. Após a leitura e execução deste tutorial deve-se ler os próximos capítulos do tutorial para fixar e expandir os conhecimentos sobre o CSS.

3. Estrutura básica dos comandos CSS

Os comandos CSS tem a seguinte estrutura básica.

Seletor {**propriedade**:**valor**}

Onde: **seletor** é uma tag HTML, **propriedade** é uma propriedade da tag e **valor** é o valor atribuido a esta propriedade. Abaixo um exemplo.

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

Neste exemplo o comando determina que todo texto que estiver dentro da tag H1 deve ser pintado de azul.

4. Ligar arquivos HTML a comandos CSS

Existem 3 maneiras de se ligar um arquivo HTML á comandos CSS.

A primeira e mais recomendada forma é colocar os comandos CSS em um arquivo externo. Isto é feito através da tag <link> que por sua vez deve estar dentro da tag <head>. O valor do atributo href é o caminho (a url) para o arquivo css que conterá os comandos que serão aplicados na página. Os demais atributos podem ser sempre os mesmos. Mais de um arquivo HTML pode apontar para um mesmo arquivo CSS formando assim a estrutura mostrada na figura 1. Abaixo um exemplo de uma página HTML que aponta para o arquivo "estilo.css" que está localizado no mesmo diretório.

```
<HTML>
  <HEAD>
    <TITLE>title</TITLE>
    <LINK REL=STYLESHEET TYPE="text/css"
      HREF="estilo.css">
  </HEAD>
  <BODY>
    Ola Mundo!
  </BODY>
</HTML>
```

A segunda é colocar os comandos CSS na própria página HTML. Isto é feito através da tag <style> que deve ser colocada dentro da tag <head> do documento HTML. Entre <style> e </style> podem ser colocados comandos CSS que serão aplicados sobre a atual página HTML. Abaixo um exemplo:

```
<html>
<head>
  <title>Página CSS</title>
  <style type="text/css">
  body {
    color: red;
    background-color: yellow }
  </style>
</head>
<body> Ola Mundo! </body>
</html>
```

A terceira é colocar os comandos CSS no atributo "style" das tags HTML. Os comandos só serão aplicados na tag em que estiverem inseridos. Nos comandos basta informar o valor e a propriedade pois o seletor é a própria tag onde o comando está inserido. Abaixo um exemplo.

```
<P style="color: blue">
```

É importante notar que ao utilizar as duas últimas formas são perdidas grande parte das vantagens do CSS já que não é possível organizar uma estrutura como a da figura 1.

5. Variações e características dos comandos CSS

5.1 Adicionar comentários

Os comentários em códigos CSS são semelhantes aos comentários do Java. Ou seja usa-se "//" para colocar um comentário em 1 linha. Já para comentar várias linhas pode-se usar "/* *\". A seguir exemplos de comentários.

```
/* comentário de
   várias linhas  *\
H1 {color:blue}    // comentário sobre este comando
```

5.2 Agrupamentos

Pode-se especificar vários seletores em um só comando (deve-se separar os seletores por ","").

```
H1, H2, H3 {color:red}
```

Neste exemplo o comando determina que todo texto que estiver dentro das tags H1, H2 e H3 deve ser pintado de vermelho.

Pode-se especificar vários pares "propriedade:valor" em um só comando (deve-se separar os pares por ";").

```
H1 { Color:blue;  font-size: 12pt;  font-family: helvetica; }
```

Neste exemplo o comando determina que todo texto que estiver dentro da tag H1 deve ser pintado de vermelho, ter fonte do tamanho 12pt e ser do tipo helvetica.

5.3 Seletor contextual

Os seletores contextuais devem estar separados por um espaço. Estes seletores determinam que somente as tags do segundo seletor que estão dentro do primeiro seletor devem ser alteradas. Pode-se usar mais seletores como por exemplo 4 tags que devem estar uma dentro da outra.

```
P B {color:blue}
```

O seletor do exemplo acima é um seletor contextual. Ele determina que somente as tags <B> que estão dentro de tags <P> devem ser pintadas de azul.

5.4 Seletor classe

Quase todas tags HTML possuem o atributo class, qualquer nome pode ser colocado no valor do class. Tal nome vai definir a classe do correspondente elemento HTML. Com este atributo os elementos HTML podem ser classificados em classes, e estas classes podem ter sua aparência definida em um comando CSS. Veja o exemplo a seguir:

```
<HTML>
 <HEAD>
  <TITLE>Title</TITLE>
  <STYLE TYPE="text/css">
    H1.alerta { color: red }
  </STYLE>
 </HEAD>
 <BODY>
  <H1 CLASS="alerta">Atenção redobrada aos seguintes itens</H1>
  <H1> Aviso normal </H1>
```

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

No exemplo acima foi definido que o primeiro <H1> é da classe alerta (uma classe inventada pelo programador, pode-se inventar qualquer nome para classes). Na tag <style> foi colocado um comando CSS, tal comando especifica que o conteúdo das tags H1 da classe "alerta" devem ser pintados de vermelho. No exemplo dado só o primeiro H1 seria pintado de vermelho.

```
.alerta { color: red }
```

O comando acima especifica que qualquer tag HTML da classe alerta deverá ser pintada de vermelho. Logo na ausência de um identificador de tag a regra vale para qualquer tag que seja da referida classe.

5.5 Seletor Id

Quase todas tags HTML possuem o atributo Id, qualquer nome pode ser colocado no valor do Id. Tal nome vai identificar o correspondente elemento HTML. Ele funciona de maneira similar ao class, logo pode-se definir a aparência de um elemento identificado.

A diferença para o atributo class é que enquanto vários elementos no documento HTML podem ter a mesma classe, só um elemento no documento inteiro pode ter o mesmo ID. O ID é portanto um identificador único de um elemento num documento HTML (erroneamente alguns browsers permitem a existência de 2 elementos com um mesmo id em um mesmo documento).

```
<HTML>
 <HEAD>
  <TITLE>Title</TITLE>
  <STYLE TYPE="text/css">
    #el123 { color: red }
  </STYLE>
 </HEAD>
 <BODY>
  <H1 ID=”el123”>Título</H1>
 </BODY>
</HTML>
```

No exemplo acima foi definido que o primeiro <H1> é identificado pelo nome "el123". Na tag <style> foi colocado um comando CSS, tal comando especifica que o conteúdo do elemento identificado por "el123" deve ser pintado de vermelho.

5.6 Pseudo elementos

Pseudo elementos são seletores especiais que ao invés de apontar para uma tag HTML, apontam para uma tag (ou parte dela) em uma determinada situação. Vejamos os seguintes exemplos:

```
A {color:blue}
```

O código acima é um seletor normal que aponta para a tag <A> (a tag dos links). O comando vai pintar os links de azul.

Agora observe os códigos abaixo.

```
A:visited {color:blue}
A:link {color:red}
```

Estes seletores apontam para pseudo elementos. O primeiro comando vai pintar de azul os links que já foram visitados (é o conteúdo da tag <A> em uma determinada situação), já o segundo vai pintar de vermelho os links que ainda não foram visitados.

Vejamos outros exemplos.

```
P:first-letter {font-size: 200%}
DIV:first-line{color:blue}
```

Os commandos acima mostram mais dois pseudo elementos. O primeiro determina que a primeira letra do texto que está dentro da tag <P> deve ter o tamanho alterado (200%, ou seja será o dobro do tamanho das demais letras). Portanto o seletor aponta para o conteúdo de uma tag (<P>) em uma determinada situação (só a primeira letra será alterada pelo CSS). Este pseudo elemento (first-letter) pode ser aplicado a várias outras tags como, por exemplo: DIV, H1,H2,B,etc.

O segundo comando tem um seletor similar ao first-line, só que este é aplicado a toda primeira linha. O comando do exemplo irá pintar de azul a primeira linha dos textos que estiverem dentro da tag <DIV>.

Existem vários outros pseudo elementos e novos são criados de tempos em tempos. Para uma lista mais completa de pseudo elementos acoselha-se a leitura dos livros e links apontados na bibliografia deste manual.

5.7 Herança

Um commando css é aplicado na tag informada pelo selector e em todos os seus filhos (em todas as tags que estão dentro da tag informada pelo seletor). Este mecanismo chama-se herança.

```
BODY {color: red;}
```

Neste exemplo o comando determina que todo texto que estiver dentro da tag <body> e dentro dos elementos que estão dentro desta tag devem ser pintados de vermelho.

5.8 Sobreposição de commandos CSS

Com o mecanismo da herança muitos já devem ter notado que pode haver uma sobre posição de comandos HTML sobre uma mesma tag. Por exemplo, veja os comandos a seguir:

```
BODY {color: red;}
H1 {color:blue;}
```

O commando sobre o <body> vai ser aplicado sobre a tag body e todas as tags que estão dentro do body, ou seja quase toda página HTML. Já o comando sobre <H1> é aplicado dentro das tags H1. Portanto, dentro das tags H1 deveriam ser aplicadas as regras definidas nos dois comandos acima (já que os H1 estão dentro do Body). Porém só um comando pode ser aplicado (ou pinta-se de azul ou pinta-se de vermelho). Esta situação caracteriza uma sobreposição.

No caso de sobreposição de comandos os navegadores agem da seguinte maneira:

Se os comandos afetarem propriedades diferentes o navegador aplica os dois comandos. Se os comandos afetarem a mesma propriedade (como no caso do exemplo acima onde os dois comandos afetam a propriedade cor)o navegador aplica o comando do seletor mais **específico**. No exemplo anterior o navegador vai pintar o que esta dentro do H1 de azul pois o seletor H1 é mais específico que o seletor body(que vale para toda a página).

De maneira geral pode-se dizer que os seletores dos filhos são mais específicos que os seletores pais (onde pais são tags que contém outras tags que são as filhas). Além disso os seletores do tipo classe são mais específicos que os seletores normais. Já os seletores do tipo ID são mais específicos que os seletores do tipo classe.

Para concluir, uma observação: o algoritmo que escolhe o comando que será aplicado em uma tag é bem complexo e envolve todas as possibilidades de combinação de comandos. Entretanto, por em quanto, é mais produtivo aplicarmos as regras gerais de especificidades aqui apresentadas do que decorar um algoritmo mais complexo e que na grande maioria das vezes vai trazer o mesmo resultado.

5.9 Unidades de medida

Em diversos comandos CSS pode-se utilizar medidas para informar comprimentos, larguras, alturas, intensidades, localizações, etc. Em quase todos comandos pode-se utilizar dois tipos de medidas: as absolutas e as relativas.

5.9.1 Medidas absolutas

As medidas absolutas não variam em nehuma situação. Por exemplo o "centimetro" **cm:** ele é uma medida que pode ser usada para definir a expessura de uma borda. Se é dito que uma borda terá 1cm, ela terá 1cm em qualquer tela, em qualquer resolução, enfim em qualquer situação. Isto pode ser um problema pois se a sua página ser acessada de um celular e você preparou ela em cm ela pode até ficar com um bom design numa tela de computador mas provavelmente vai ficar muito ruim em um celular. Por isso cuidado ao definir medidas absolutas.

No CSS você pode usar as seguintes medidas (ao lado de cada medida é mostrado um exemplo de comando CSS que utiliza a referida medida para definir o tamanho de um letra):

In – polegada –p {font-size:23in}

Cm - centímetro –p{font-size:23cm}

Mm – milímetro –p{font-size:230mm}

Pt – ponto (1pt corresponde a 1/72 polegadas) – p{font-size:230pt}

Pc – pica (1pc corresponde a 12 pontos) –p{font-size:23pc}

5.9.2 Medidas Relativas

As medidas relativas variam seu tamanho conforme algumas situações ou referências (conforme a resolução da tela do usuário, o tamanho da janela utilizada, o tamanho da letra).

As unidades abaixo levam em conta o tamanho das letras

Em – 1em corresponde ao tamanho da fonte do elemento em questão – p {margin-left: 1em}. O exemplo dado define uma margem de 1em. Quanto maior for a fonte do texto que utiliza o em, maior será a margem.

Ex – 1ex corresponde a altura da letra xis minúscula - p {margin-left: 1ex}.

Já a medida pixel (px) depende da resolução do dispositivo onde esta sendo mostrada a página. Ela leva em conta se o usuário utiliza uma resolucao 800x600, 1200x800(típico de monitores), 200x200 (típico de celulares). É uma das medidas mais utilizada para páginas na web pois auxilia na adptaçào do layout a resolução do usuário.

Px - p{font-size:23px}

Por fim também pode-se usar a porcentagem **%.** Dependendo do contexto ela tem diferentes significados. As vezes é um valor em relacão ao valor original, abaixo o tamanho da letra corresponde a 200% do tamanho normal, ou corrente da letra. (Neste caso a letra terá o dobro do tamanho atual da fonte)

p{font-size: 200%}

Em outros casos a porcentagem é em relação ao tamanho da tela ou da caixa que o elemento pode ocupar. O exemplo abaixo informa que o paragráfo deve ocupar 50% da largura da tela ou da caixa onde o elemento está inserido.

p{width:50%}

6. Efeitos com textos e cores

Até agora foram vistas as características dos comandos CSS. A seguir serão mostrados uma série de comandos CSS que são aplicados para gerar efeitos em textos e cores.

Exercício 2

Coloque os códigos abaixo em páginas html e altere seus valores para ver como funcionam os comandos.

Colorindo textos:

```
<html>
<head>
<style type="text/css">
      body {color:red}
      h1 {color:green }
      p.destaque {color:blue}
</style>
</head>
<body>
      <h1>Título nível 1</h1>
      <p>Paragrafo comum, bem comum, bem comum, bem comum, bem comum mesmo.</p>
<p class="destaque">Paragrafo da class “destaque”</p>
</body>
</html>
```

Alinhando textos:

```
<html>
<head>
<style type="text/css">
h1 {text-align:center}
p.data {text-align:right}
p.comum {text-align:justify}
```

```
</style>
</head>

<body>
<h1>Titulo da pagina</h1>
<p class="data">Junho, 2009</p>
<p class="comum"> Paragrafo comum, bem comum, bem comum, bem comum, bem comum mesmo. Bla, bla bla bla. Paragrafo comum, comum, comum, bem comum, bem comum mesmo. Bla, bla bla bla. Paragrafo comum, bem comum, bem comum, bem comum, bem comum mesmo. Bla, bla bla bla. </p>
</body>
</html>
```

Decorando textos:

```
<html>
<head>
<style type="text/css">
h1 {text-decoration:overline}
h2 {text-decoration:line-through}
h3 {text-decoration:underline}
</style>
</head>
<body>
      <h1>Titulo 1</h1>
      <h2>Titulo 2</h2>
      <h3>Titulo 3</h3>
</body>
</html>
```

Espaços entre caracteres e entre linhas. Indentação.

```
<html>
<head>
<style type="text/css">
p.um { text-indent:70px}
p.dois {line-height: 400%;
        letter-spacing:6px }
p.tres {line-height: 30%;
      letter-spacing:-3px }
</style>
</head>
<body>
<p class=”um”>
```

```
Paragrafo comum, bem comum, bem comum, bem comum, bem comum mesmo. Bla, bla bla bla.
Paragrafo comum, comum, comum, bem comum, bem comum mesmo. Bla, bla bla bla.
Paragrafo comum, bem comum, bem comum, bem comum, bem comum mesmo. Bla, bla bla bla.
Paragrafo comum, bem comum, bem comum, bem comum, bem comum mesmo. Bla, bla bla bla.
Paragrafo comum, comum, comum, bem comum, bem comum mesmo. Bla, bla bla bla.
Paragrafo comum, bem comum, bem comum, bem comum, bem comum mesmo. Bla, bla bla bla.
</p>
<hr>
<p class="dois">
Paragrafo comum, bem comum, bem comum, bem comum, bem comum mesmo. Bla, bla bla bla.
Paragrafo comum, comum, comum, bem comum, bem comum mesmo. Bla, bla bla bla.
Paragrafo comum, bem comum, bem comum, bem comum, bem comum mesmo. Bla, bla bla bla.
Paragrafo comum, bem comum, bem comum, bem comum, bem comum mesmo. Bla, bla bla bla.
Paragrafo comum, comum, comum, bem comum, bem comum mesmo. Bla, bla bla bla.
Paragrafo comum, bem comum, bem comum, bem comum, bem comum mesmo. Bla, bla bla bla.
</p>
<hr>
<p class="tres">
Paragrafo comum, bem comum, bem comum, bem comum, bem comum mesmo. Bla, bla bla bla.
Paragrafo comum, comum, comum, bem comum, bem comum mesmo. Bla, bla bla bla.
Paragrafo comum, bem comum, bem comum, bem comum, bem comum mesmo. Bla, bla bla bla.
Paragrafo comum, bem comum, bem comum, bem comum, bem comum mesmo. Bla, bla bla bla.
Paragrafo comum, comum, comum, bem comum, bem comum mesmo. Bla, bla bla bla.
Paragrafo comum, bem comum, bem comum, bem comum, bem comum mesmo. Bla, bla bla bla.
</p>
</body>
</html>
```

Fontes:

```
<html>
<head>
<style type="text/css">
p.um{font-family:"Times New Roman"}
p.dois{font-family:Arial; font-size:300%}
</style>
</head>
<body>
<h1>CSS font-family</h1>
<p class="um">Parágrafo na fonte Times New Roman.</p>
<p class="dois">Parágrafo na fonte Arial e com o triplo do tamanho.</p>
</body>
</html>
```

Cor de fundo para a página e para os demais elementos.

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

```
body {background-color:gray;}
h1{background-color:yellow;}
p{background-color:green;}
div{background-color:red;}
</style>
</head>
<body>
<h1>Titulo com fundo alterado!O fundo da página é cinza mas o do título é amarelo</h1>
<div>
Elemento Div com o fundo alterado
<p>Paragrafo dentro do div.O paragrafo tem suaprópria cor de fundo</p>
ainda no elemento div.
</div>
</body>
</html>
```

Fundo da página com uma figura(substituir o nome figura.gif por uma figura que esteja no mesmo diretório da página html).

```
<html>
<head>
<style type="text/css">
body {background-image:url('figura.gif')}
</style>
</head>
<body>
<h1>Hello World!</h1>
</body>
</html>
```

7. Modelo de caixas do CSS (*"Box Model"*)

Para apresentar uma página HTML os Navegadores montam "caixas" ("*Box*") para cada elemento HTML. Depois as caixas são colocadas em seu devido lugar na página HTML. Quase todos elementos são caixas o HTML, BODY, P, DIV, B, I, H1, H2, TABLE, TR,TD, FORM, UL, LI, etc. Muitos deles podem ter dentro de seu conteúdo outras caixas. Assim pode-se imaginar que uma página HTML é montada como um conjunto de caixas que podem estar uma ao lado da outra, podem estar empilhadas e/ou contidas umas dentro das outras.

A figura 2 mostra a estrutura destas caixas.

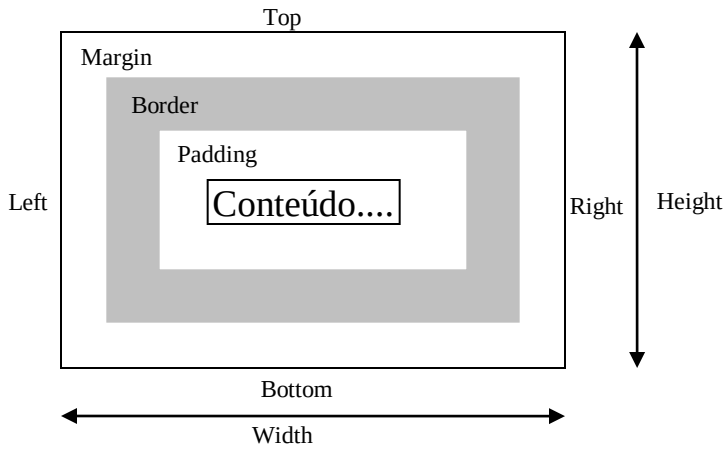

Figura 2 – Box Model

Cada caixa contém todos os elementos mostrados na figura 2. Todo elemento HTML tem um valor padrão para os elementos mostrados. Normalmente, no estado padrão estes elementos são finos e transparentes sendo praticamente imperceptíveis. Entretanto com o CSS é possível alterar o valor padrão dos elementos aumentando consideravelmente as possibilidades gráficas das páginas HTML. Antes de aprender a alterar os valores dos elementos das caixas, vamos entende-los. A seguir está a desscrição de cada ítem do modelo.

Conteúdo: é o conteúdo do elemento HTML. São os textos, as figuras e/ou outros elementos HTML(que por sua vez são outras caixas).

Padding: é o espaço entre a área do conteúdo e a borda do elemento.

Border: é a borda do elemento.

Margin: é o espaço entre a borda e a próxima caixa.

As caixas possuem 4 lados: topo, direita, baixo e esquerda (**top, right, bottom, left**). As prorpiedades do padding, border e margin podem ser diferentes em cada lado da caixa.

Por fim as caixas possuem uma largura (o **width**) e uma altura (o **height**).

Agora um exemplo de página que define a aparência das caixas(box) de vários elementos <DIV>.

```
<html>
<head>
    <title>Entendendo o box model</title>
<STYLE type="text/css">
body{
    font-family: Arial, Verdana, Tahoma, Sans-Serif;
    font-size: 12px;
}
#box1{
    background-color: #999999;
```

```
    padding: 20px;
}
#box2{
    background-color: #009999;
    padding: 10px;
    width: 300px;
    height: 50px;
}
#box3{
    padding: 10px;
    border: 2px solid #FF0000;
    margin: 20px;
    width: 150px;
}
#box4{
    background-color: #D8D8C5;
    padding: 5px 10px 15px 20px;
    height: 150px;
    width: 150px;
}
#box5{
    background-color: #BBF3A3;
    border: 2px dotted #008000;
    padding: 10px 5px 10px 5px;
    margin: 10px 0px 10px 0px;
    width: 300px;
    height: 100px;
}
</style>
</head>
<body>
<div          id="box1">Exemplo          de          box          model          1</div>
<div          id="box2">Exemplo          de          box          model          2</div>
<div          id="box3">Exemplo          de          box          model          3</div>
<div          id="box4">Exemplo          de          box          model          4</div>
<div id="box5">Exemplo de box model 5</div>
</body>
</html>
```

A seguir um outro exemplo onde a estrutura dos box de uma lista são alterados. A lista do código abaixo é uma box, o <UL>, com duas caixas dentro, as duas <LI>. As propriedades destas 3 box são alteradas pelo código CSS contido na página. Rode o exemplo abaixo e veja o resultado.

```
<HTML>
  <HEAD>
    <TITLE>Caixas</TITLE>
    <STYLE type="text/css">
      UL {
```

```
        background: yellow;
        margin: 12px 12px 12px 12px;
        padding: 3px 3px 3px 3px;
                                    /* sem bordas */
      }
      LI {
        color: white;               /* Cor do texto branca */
        background: blue;           /* o conteudo e o padding serão azuis */
        margin: 12px 12px 12px 12px;
        padding: 12px 0px 12px 12px;
        list-style: none
      }
      LI.comborda {
        border-style: dashed;
        border-width: medium;       /* características da borda */
        border-color: lime;
      }
    </STYLE>
  </HEAD>
  <BODY>
    <UL>
      <LI>Primeiro elemento da lista
      <LI class="comborda">Segundo elemento da lista.
    </UL>
  </BODY>
</HTML>
```

A seguir vamos explicar com mais detalhes os comandos que podemos usar para definir cada uma das características de uma box.

Borda

border-width: 5px; //define a espessura da bordas

border-color: blue; // define a cor da borda

border-style: solid; // define o estilo da borda. As alternativas são: solid, dotted, dashed, groove, double,

//ridge,inset, outset

Margin

Margin: 20px; //define a espessura da margin

Padding

Padding: 6px; //define a espessura do padding

Box

Ainda existem comandos que são aplicados a toda box.

Background: green // define que a cor de fundo da caixa será verde. O background defina a cor de fundo

// do conteudo e do padding da caixa pois ambos são um espaçamento.

Width: 200px; // define que a largura da box será de 200px.

Height: 100px; // define que a altura da box será de 100px.

Atalhos e variações

Os comandos acima podem ser aplicados em apenas um lado da caixa. Basta acrescentar aos comandos as palavras left, right, top ou bottom.

border-color-left: blue; // a borda esquerda será azul

border-color-top: red; // a borda superior será vermelha

margin-top: 20px; //a espessura da margin superior será de 20px

Pode-se definir propriedades dos 4 lados de uma caixa em um só comando. O comando abaixo define uma espessura diferente para cada borda. Será atribuido os tamanhos 5px 10px 20px 30px respectivamente para as bordas top, right, bottom, left. Nestes casos segue-se sempre a sequência top, right, bottom, left.

border:5px 10px 20px 30px; // espessura das bordas

padding: 4px 6px 8px 12px //espessura dos paddings

No caso da borda ainda é possivel definir diversas propriedades em um só comando.

border: 1px solid red // define que todas as bordas terão espessura 1, estilo solid e cor vermelha

border-right: 1px solid red // define que a borda direita terá espessura 1, estilo solid e cor vermelha

8. Posicionando caixas

Agora que já foi apresentado como definir as caixas do HTML, resta saber onde coloca-las na página. Falaremos aqui dos dois principais modos de definir as posições das caixas em uma página HTML.

8.1 Posicionamento Absoluto

O posicionamento absoluto precisa que do comando “position: absolute” combinado com pelo menos um dos seguintes comandos left, top. O comando position:absolute define que o posicionamento será absoluto, ou seja as distâncias informadas nos demais comandos serão em relação a coordenada 0,0 da página (que é o canto superior esquerdo da página).

O comando left irá definir a distância lateral entre o canto superior esquerdo da caixa e a coordenada 0,0 da página. De modo análogo o comando top informa a distância vertical.

A seguir um exemplo:

```
p.principal {  position: absolute;
   left:70px;
```

```
        top: 100px    }
```

O canto superior esquerdo da caixa do parágrafo da classe "principal" será colocado a uma distância horizontal de 70px e a uma distancia vertical de 100px do ponto 0,0; conforme mostra a figura 2.

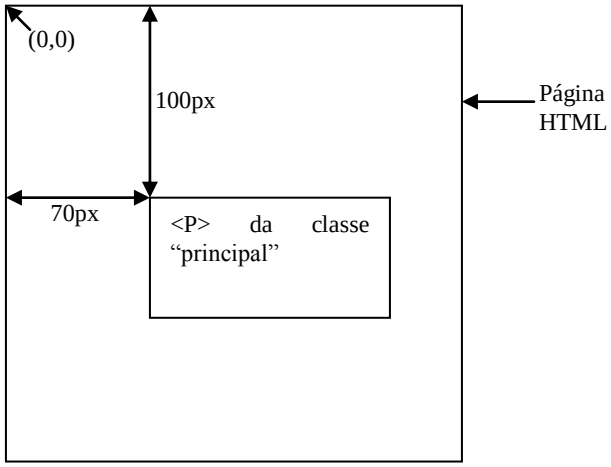

Figura 3 – Posicionamento Absoluto

Index-z

Quando o posicionamento absoluto é utilizado, pode acontecer de duas caixas ficarem sobrepostas (uma por cima da outra). Nesta situação só uma caixa vai aparecer completamente para o usuário, já as demais ficariam escondidas em baixo da caixa que está em primeiro plano. Para controlar qual caixa é mostrada em primeiro plano utiliza-se o comando index-z. O valor da propriedade index-z é um numero qualquer. As caixas que tiverem o maior valor do index-z aparecem em primeiro plano. Ou seja a caixa que tem o número z-index maior aparece na frente da caixa que tem o z-index menor. Abaixo um exemplo.

```
P.1 { position: absolute;
      left:70px;
      top: 100px
      z-index:2}
P.2 { position: absolute;
      left:80px;
      top: 110px
      z-index:1}
P.3 { position: absolute;
      left:90px;
      top: 120px
      z-index:3}
```

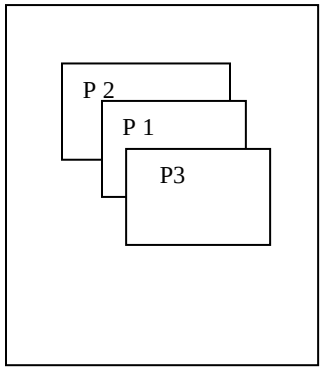

Figura 4- zindex

Neste exemplo, a caixa do p.3 será mostrada em primeiro plano, em baixo dela estará a caixa p.1 e em baixo de p.1 será colocada a p.2.

8.2 Flutuação

Normalmente os navegadores colocam as caixas no ponto mais alto que podem. Este tipo de colocação ou flutuação das caixas é mostrado na figura 3(a). Nela o navegador coloca o paragrafo 1 no canto superior esquerdo, o parágrafo 2 logo abaixo do paragrafo 1, o 3 logo abixo e assim por diante. Este é o comportamento padrão do navegador, não é preciso qye se faça nenhum comando para o navegador se comportar assim.

Entretanto é possível se alterar este comportamento. Ao colocar o comando css "float: left" a flutuação passa a ser para a esquerda, conforme mostra a figura 3(b). No nosso exemplo o navegador coloca o p1 no canto superior esquerdo, o p2 ao lado de p1, p3 ao lado de p2 (sempre coloca o mais a esquerda possível). Quando termina a largura da página e o navegador não pode colocar outra caixa ao lado de p3, ele coloca a caixa na linha abaixo (novamente a esquerda).

Já o comando css "float:right" especifica uma flutuação para a direita conforme mostra a figura 3(c). Neste caso p1 é colocado no canto superior direito. P2 é colocado ao lado de p1 (sempre coloca o mais a direita possível), p3 ao lado e p4 abaixo e a direita (pois não cabe na primeira linha).

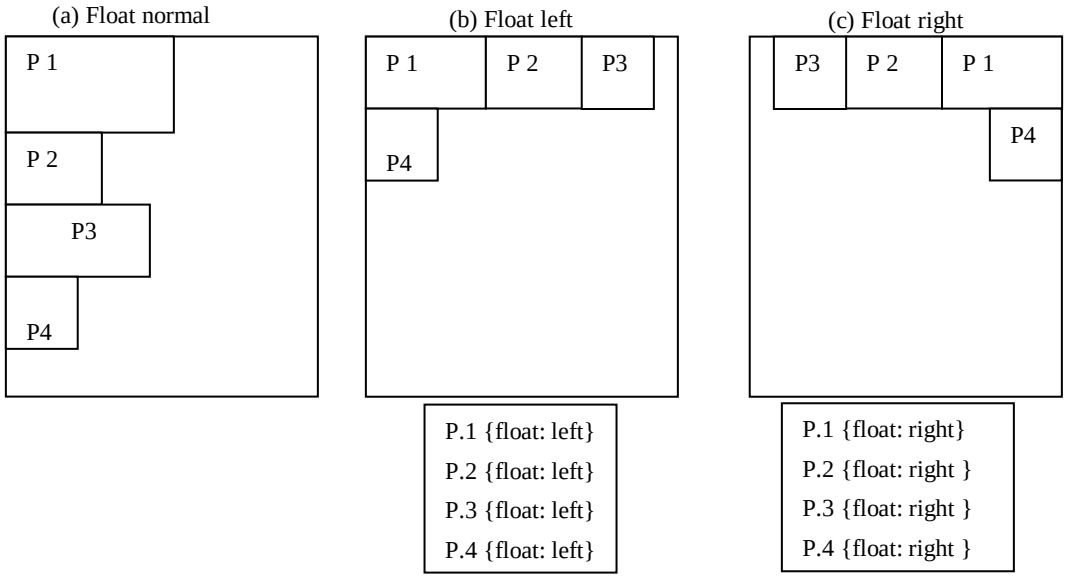

Figura 5 - Float

Exercício: Acessar o link para a W3Schools abaixo, olhar os códigos CSS, alterar seus valores e verificar o resultado das mudanças.

http://www.w3schools.com/css/css_examples.asp