

# Introdução ao HTML5 e CSS3 – parte 2

Profª Mª Denilce Veloso  
[denilce.veloso@fatec.sp.gov.br](mailto:denilce.veloso@fatec.sp.gov.br)  
[denilce@gmail.com](mailto:denilce@gmail.com)

# CSS – O que é?

CSS é usado para definir o design e layout da página. Um mesmo arquivo HTML pode ser "formatado" de maneiras diferentes utilizando arquivos de CSS diferentes (**extensão.css**). Todas as diferenças de cores, fontes e formas de como os elementos são colocados nas páginas são o resultado do CSS.

Composto de regras de estilo que o browser interpreta e aplica no elemento correspondente no documento.



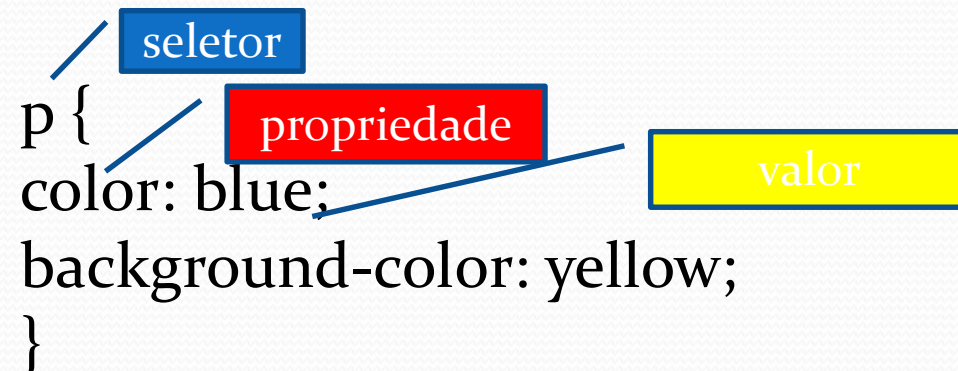
# CSS – O que é?

O CSS formata a informação entregue pelo HTML. Essa informação pode ser qualquer coisa: imagem, texto, vídeo, áudio ou qualquer outro elemento criado. Então, se CSS formata a informação, na maioria das vezes é visual, mas não necessariamente. No CSS Aural, pode se manipular o áudio entregue ao visitante pelo sistema de leitura de tela e aí controlar volume, profundidade, tipo da voz ou em qual das caixas de som a voz sairá.

# CSS – Sintaxe

A sintaxe CSS é constituída de três partes: um seletor (selector), um propriedade (property) e um valor (value): selector {property: value}

O seletor é normalmente o elemento/tag HTML que se deseja definir, a propriedade é o atributo que se deseja mudar, e cada propriedade pode ter um valor. A propriedade e o valor são separados por dois pontos e circundadas por chaves: Exemplo:



The diagram illustrates the CSS syntax with three labels: 'seletor' (selector) in a blue box pointing to 'p', 'propriedade' (property) in a red box pointing to 'color', and 'valor' (value) in a yellow box pointing to 'blue'. The CSS code is shown as follows:

```
p {  
  color: blue;  
  background-color: yellow;  
}
```



# CSS – estilos.css

Considere o seguinte arquivo CSS:

```
p {  
    background-color: yellow;  
    color: blue;  
}
```



Em relação às regras de folhas de estilo do CSS, o trecho acima corresponde aos seguintes números de ocorrências de: seletor, propriedade e valor:

- a) 2, 1, 3.
- b) 2, 2, 2.
- c) 2, 1, 1.
- d) 1, 2, 2.

# CSS – estilos.css

Além da melhor organização do projeto, a folha de estilo externa traz ainda as vantagens de manter o HTML mais limpo e do reaproveitamento de uma mesma folha de estilos para diversos documentos.

A indicação de uso de uma folha de estilos externa deve ser feita dentro da tag <head> do documento HTML. (forma mais utilizada)

```
<link rel="stylesheet" href="estilos.css">
```

E dentro do arquivo estilos.css coloca-se apenas o conteúdo do CSS:

Exemplo:

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



# Referenciar arquivos CSS e JavaScript no HTML5

<!-- O que antes era assim para carregar css e javascript ... -->

```
<link rel="stylesheet" href="estilo.css" type="text/css" />  
<script src="exemplo.js" type="text/javascript"></script>
```

<!-- ...pode ser escrito assim -->

```
<link rel="stylesheet" href="estilo.css">  
<script src="exemplo.js"></script>
```

# CSS – outras formas de utilizar

- **Com atributo style no próprio HTML**

`<p style="color: blue; background-color: yellow;">` O conteúdo desta tag será exibido em azul com fundo amarelo no navegador! `</p>`

- **Declarando suas propriedades dentro de uma tag `<style>` dentro da tag HEAD**

```
<!DOCTYPE html>
```

```
<html>
```

```
<head>
```

```
  <style>
```

```
    p {  
      background-color: yellow;  
      color: blue;  
    }
```

```
  </style>
```

```
</head>
```

```
<body>
```

```
<p>    O conteúdo desta tag será exibido em azul com fundo amarelo!    </p>
```

```
<p> <strong>Também em negrito</strong> será exibido em azul com fundo amarelo!
```

```
</p>
```

```
</body>
```

```
</html>
```



# CSS – outras formas de utilizar

Qual é o local mais indicado para referenciar um arquivo CSS externo em uma página HTML?

- a) antes de <html>
- b) depois de <head>
- c) no meio de doctype
- d) dentro de <body>
- e) dentro de <head>
- f) o correto é fazer tudo inline



# CSS – Comentários

Assim como no HTML, também é possível colocar comentários em um código CSS.

Na sintaxe de um comentário CS, o texto deve ser colocado entre os asteriscos.

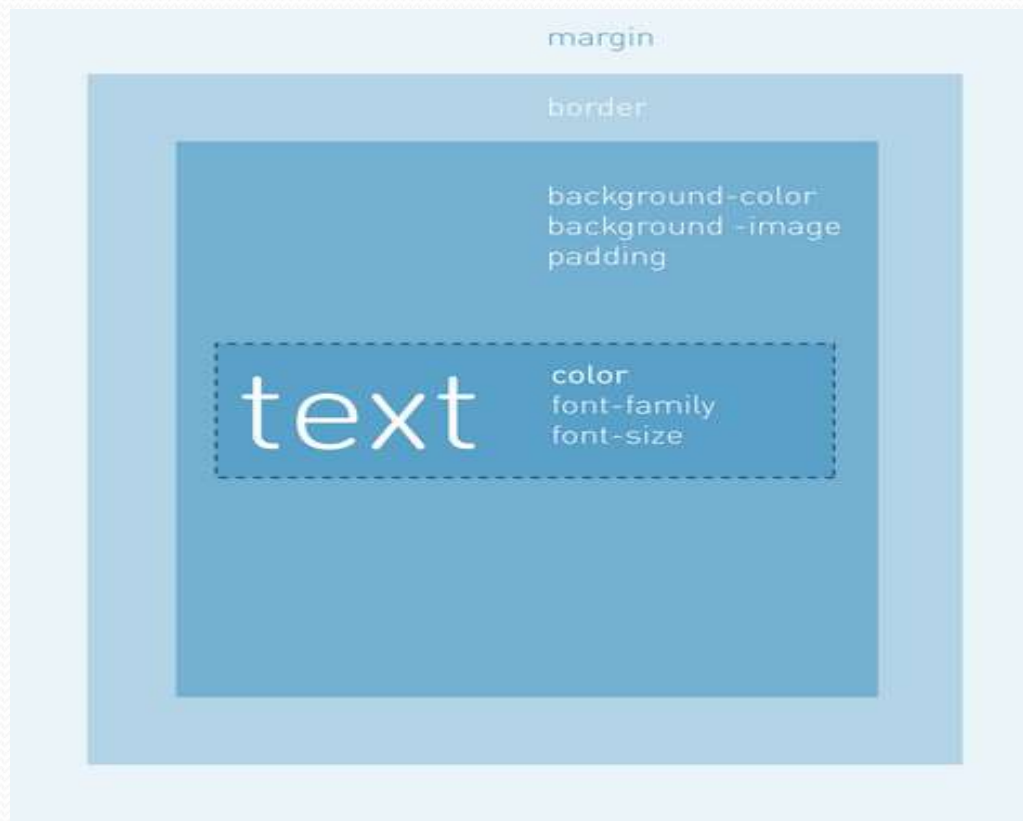
Exemplo:

```
/* Seu comentário aqui! */
```



# CSS – Propriedades

O CSS usa regras para definir o design de um elemento HTML. A figura mostra um elemento HTML marcado com as propriedades CSS que controlam diferentes aspectos de sua aparência. Quando for denominado um elemento HTML, é bom imaginar uma caixa em torno dele e aplicar essas propriedades para denominá-lo .



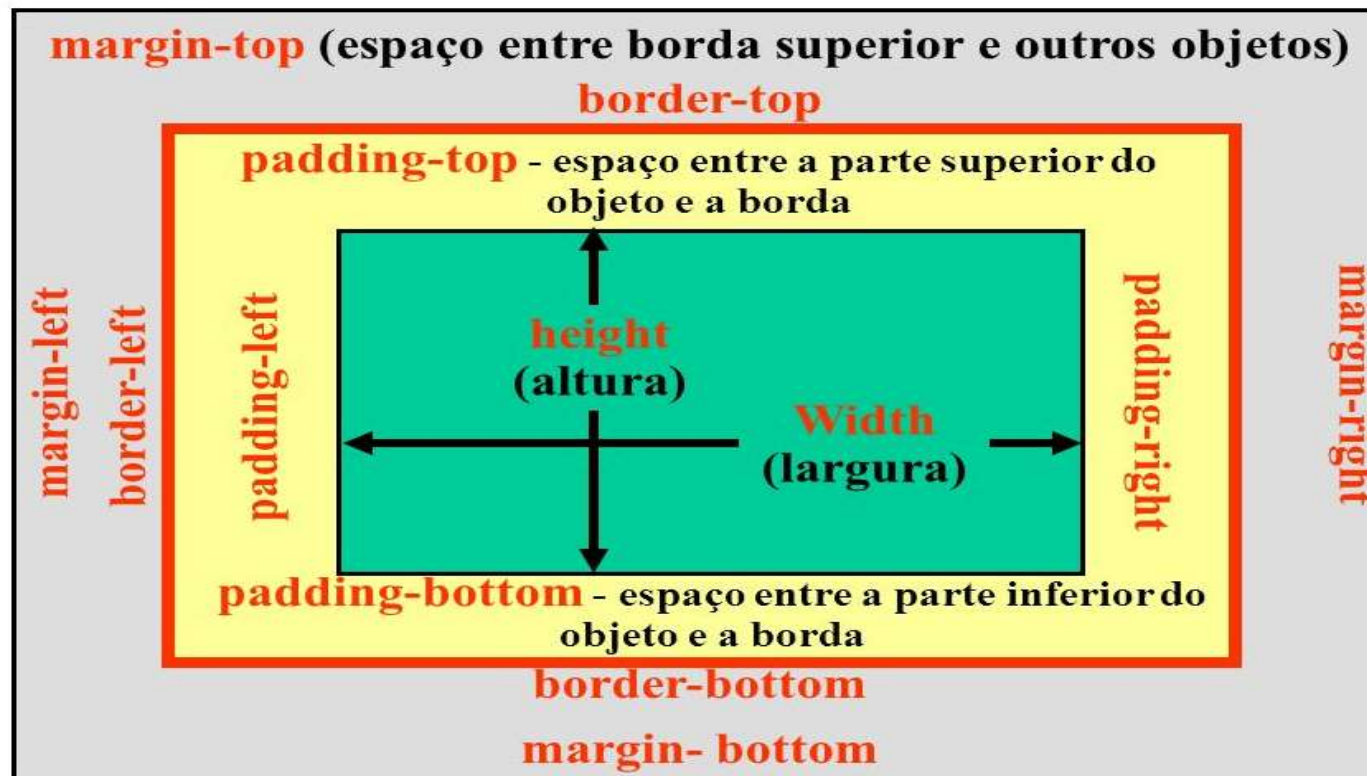
Exemplo:

```
p {  
    margin-top: 100px;  
    margin-bottom: 100px;  
    margin-right: 150px;  
    margin-left: 80px;  
}
```

```
p {  
    border: 5px solid red;  
}
```

# CSS – Propriedades

Observe cada caixa: MARGIN, BORDER, PADDING(espacamento interno)



Margens e Paddings Adaptado de : (Fonte:  
<http://slideplayer.com.br/slide/386386/>)



# CSS – Propriedade Margin

A margem é uma área transparente fora da borda de um elemento. São possíveis os 4 valores: **margin-top**, **margin-right**, **margin-bottom** e **margin-left**.

Se uma dessas propriedades (margin-right ou margin-left) estiver definido como automático, o navegador ajustar a margem. Se a margem à esquerda e à direita forem iguais o elemento será centrado .

```
p {  
margin: 0 auto; /* o top e bottom e automatica para left e right */  
}
```

```
h1 {  
margin : 25px 50px 75px 100px; /* top, right, bottom, left */  
}
```

```
h1 {  
margin: 30px; /* todas são 30 */  
border: 3px solid #ffffff;  
}
```

```
h1 {  
margin : 30px 35px /* 30 para top e bottom e 35 para left e right */  
}
```

# CSS – Exemplo 11UsandoMargens.html

```
<!DOCTYPE HTML>
<html>
<head>
<title>Margens</title>
<link rel="stylesheet" href="UsandoMargens.css">
</head>
<body>
```

**\*\*ATENÇÃO: ALTERAR VALORES DAS MARGENS NO CSS PARA TESTAR**

<p> Existem diferentes tipos de indentação e espaçamento, e um desses tipos são as margens.

Margens são espaçamentos que tem por função separar elementos com fins de organização e design. </p>

<p> As margens não tem cores, são sempre transparentes.</p>

<p>As margens, ou margin em inglês, são muito usadas em textos, imagens, para separar uma barra lateral do conteúdo, centralizar seu artigo, distanciar uma figura ou anúncio do texto etc.</p>

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



# CSS – Exemplo 11UsandoMargens.css

```
p {  
    margin-top: 200px;  
    margin-bottom: 100px;  
    margin-right: 150px;  
    margin-left: auto;  
    color: red;  
}
```

# CSS – Propriedade border

As propriedades do CSS para definirmos as bordas de um elemento nos apresentam uma série de opções. Podemos, para cada borda de um elemento, determinar sua cor, seu estilo de exibição e sua largura. Por exemplo:

```
body {  
border-width: 1px;  
border-style: solid;  
border-color: red;  
}
```

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

Para que o efeito da cor sobre a borda surta efeito, é necessário que a propriedade border-style tenha qualquer valor diferente do padrão none.



# CSS – exemplo 12UsandoBordas.html

```
<!DOCTYPE HTML>
<html>
<head>
<title>Bordas</title>
  <link rel="stylesheet" href="12UsandoBordas.css">
</head>
<body>
<h3>Borda média, contínua e azul</h3>
<p>Borda 6px, tracejada e vermelha</p>
  <br>
  <br>
  <h3 class='circulo'>CÍRCULO</h3>
  <br>
  <br>
  <h1 class="quadrado">QUADRADO</h1>
</body>
</html>
```

# CSS – exemplo 12 Usando Bordas.css

```
h3 {  
border-width: medium;  
border-style: solid;  
border-color: #00f; /* passe o mouse aqui em cima da cor o Brackets mostra a cor */  
}
```

```
p {  
border-width: 6px;  
border-style: dashed;  
border-color: #foo;  
}
```

/\* border-radius serve para criar aquelas bordas arredondadas ao redor de elementos da página\*/

/\* na verdade está se definindo a sequência

border-top-left-radius border-top-right-radius border-bottom-right-radius border-bottom-left-radius \*/

```
.circulo {  
border-radius: 50%; /* define o raio para os 4 lados */  
border: 5px solid #000;  
height: 150px;  
line-height: 100px; /* altura entre inicio e o texto */  
text-align: center;  
width: 150px;  
}
```

```
.quadrado {  
width: 300px;  
height: 100px;  
background-color: #660;  
border-radius: 10px 10px 10px 10px;  
/* todos os cantos como é um quadrado poderia colocar border-radius: 10px */}
```



## CSS – border



Como mostrar a seguinte borda corretamente:  
top = 10px, bottom = 5px, left=20px, right=1px.

- a) border-width: 5px 20px 10px 1px
- b) border-width: 10px 1px 5px 20px
- c) border-width: 10px 20px 5px 1px
- d) border-width: 10px 5px 20px 1px

# CSS – border - DESAFIO



A qual figura se refere o CSS abaixo?

```
.figura{  
  width: 0px;  
  height: 0px;  
  border-left: 125px solid transparent;  
  border-right: 125px solid transparent;  
  border-bottom: 125px solid black;  
}
```

- a) losango
- b) triângulo
- c) trapézio



# CSS – propriedade background-color

A propriedade background-color permite setar a cor de fundo de um elemento HTML. Por exemplo:

```
p {  
  background-color: #cccccc;  
}
```

# CSS – Propriedade background-image

A propriedade background-image permite indicar um arquivo de imagem para ser exibido ao fundo do elemento. Por exemplo:

```
h1 {  
background-image: url( 'https://goo.gl/ODpi3y');
```

ou

```
background-image: url("paper.gif");  
}
```

\*\*\* O navegador vai mostrar o arquivo do endereço, se não fosse url e somente o nome de um arquivo como nome1.jpg, ele deveria estar na mesma pasta do arquivo CSS.



# CSS – Propriedade Padding

Cria espaço entre o conteúdo e a margem de um elemento. Este espaço em branco é útil , a fim de melhorar a legibilidade e organização da página.

```
h1 {  
  padding: 23px;  
  border: 3px solid #ffffff;  
}
```

Podem ser passados, dois, três ou quatro valores (como a Margin). Se passados quatro valores, serão aplicados respectivamente a padding-top, padding- right, padding-bottom e padding-left.

```
p {  
  padding: 10px 20px 15px 5px;  
}
```

**\*\*** color do padding é afetado pelo color do background

# CSS – Propriedade font-family

As fontes podem ser alteradas com o uso da propriedade font-family. A propriedade font-family pode receber seu valor com ou sem aspas. Somente famílias de fontes genéricas como exemplos san-serif, serif e monospace deve ser escrito SEM aspas.

```
h1 {  
font-family: serif;  
}  
h2 {  
font-family: sans-serif;  
}  
p {  
font-family: "Arial";  
}
```

Nome família de fontes: serif, sans-serif,  
"cursive", "fantasy", monospace

Nome Fonte: "times", "courier", "arial",  
etc.

É possível, declarar o nome de várias fontes. No ex. a seguir o navegador verificará se a fonte "Arial" está disponível e a utilizará para renderizar os textos de todos os elementos do documento que, por cascata, herdarão essa propriedade do elemento body. Caso a fonte "Arial" não esteja disponível, o navegador verificará a disponibilidade da próxima fonte declarada, no caso "Helvetica". Caso contrário, ele solicita qualquer fonte que pertença à família "sans-serif", declarada logo a seguir, e a utiliza para exibir o texto.

```
font-family: "Arial", "Helvetica", sans-serif;
```



# CSS – Exemplo 13 Usando Fontes.html

**\*\*ATENÇÃO: ALTERAR CSS PARA  
PRIORIZAR A FONTE TIMES NEW  
ROMAN PARA H<sub>1</sub>**

```
<HTML>
<HEAD>
  <meta charset="utf-8">
  <link rel="stylesheet" href="13UsandoFontes.css">
</HEAD>
<!-- teste novas fontes
-->

<BODY>
  <h1> Teste de Fontes </h1>

  <h2 class='icon-next'>Avançar</h2>

  <p class='icon-next'>Ir para a próxima página</p>

  <p class='icon-ok'>E-mail enviado com sucesso!</p>

  <p class='icon-invalid'>Atenção, preencha todos os campos do
  formulário
</p>

</BODY>
</HTML>
```

# CSS – Exemplo 13 Usando Fontes.css(1)

```
h1 {
```

```
    font-family: "Arial", "Helvetica", sans-serif;  
    font-size: 50px;
```

```
/* vai tentar encontrar as fontes nessa ordem */
```

```
@font-face {  
    /* possibilita utilizar fontes externas */  
    font-family: 'IconicFill'; /* fonte */  
    src: url('iconic_fill.ttf') format('ttf'); /* arquivo importado na pasta e formato */  
    font-weight: normal;  
    font-style: normal;  
}
```

Seletor é uma classe, inicia com ponto.  
after é um pseudo elemento

```
.icon-next::after {  
    content: '\2192'; /* é um símbolo, vai encontrar quando procurar sobre a fonte */  
    font-family: 'IconicFill';  
    margin-left: 10px;  
}
```

Outros exemplos  
fontes externas

[www.dafont.com](http://www.dafont.com)  
[www.fontsquirrel.com](http://www.fontsquirrel.com)  
[www.fonts2u.com](http://www.fonts2u.com)  
[www.1001fonts.com](http://www.1001fonts.com)  
[www.fontspace.com](http://www.fontspace.com)  
[www.actionfonts.com](http://www.actionfonts.com)



# CSS – Exemplo 13 Usando Fontes.css(2)

```
p.icon-next {  
  color: blue;  
}
```

```
.icon-ok {  
  color: #489D00;  
}  
.icon-ok::before {  
  content: "\2714";  
  font-family: 'IconicFill';  
  margin-right: 10px;  
}
```

```
.icon-invalid {  
  color: #990000;  
}
```

```
.icon-invalid::before {  
  content: "\2718";  
  font-family: 'IconicFill';  
  margin-right: 10px;  
}
```

/\* Enquanto o Iconic utiliza símbolos Unicode como caracteres para os seus ícones, outras fontes podem utilizar outros caracteres para isto. É necessário conferir a documentação - caso exista uma - a respeito de cada fonte antes de utilizá-la. \*/

# CSS – Fontes

Para pesquisar mais sobre fontes:

<https://www.w3.org/TR/css-fonts-3/#family-name-value>

<https://tableless.com.br/font-face-fonts-externas-na-web/>

<https://fonts.google.com/>

→ Melhores fontes para usar em 2021

<https://www.hostinger.com.br/tutoriais/melhores-fontes-html>



# CSS – Fontes

Para saber quais fontes tem na sua máquina:

<https://flippingtypical.com/>

# CSS – Propriedade Color

As cores podem ser alteradas com o uso da propriedade Color.

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

No exemplo o título h1 será alterado para vermelho.

Pode ser utilizada o nome da cor para mudar a cor do texto. Mas isso só funciona para 140 cores. Ao invés vez disso, pode ser utilizado valores RGB ou números hexadecimais. Eles podem representar milhões de cores. Valores RGB e números hexadecimais expressam cores como diferentes quantidades de vermelho, verde e azul. Valores RGB (*Red-Green-Blue*) variam de 0 a 255, sendo que 255 é o mais brilhante. Números hexadecimais variam de 00 a FF, sendo que ff é o mais brilhante. Por exemplo, a cor amarela é formada por 100% de vermelho, 100% de verde e 0% de azul, ou RGB(255,255,0).

*\*\* Veja tabela de cores em:*

<http://erikasarti.net/html/tabela-cores/>

<http://erikasarti.net/html/tabela-cores-seguras-web-safe/>

<http://html-color-codes.info/Codigos-de-Cores-HTML>



# CSS – Exemplo: 14UsandoCores.html(1)

```
<html>
<head>
    <link rel="stylesheet" href="14UsandoCores.css">
</head>
<body>
    <br><br>
```

**\*\*ATENÇÃO: ALTERAR CSS DA CLASSE AZUL PARA INICIAR COM ROXO E TERMINAR COM AMARELO.**

```
<figure>

<figcaption>
    San Francisco, Califórnia
<small>
    Por Salim Virji (http://www.flickr.com/photos/salim/402618628/)
</small>
</figcaption>
</figure>
```

# CSS – Exemplo: 14UsandoCores.html(2)

```
<ul>
<li>Um</li>
<li>Dois</li>
<li>Três</li>
<li>Quatro</li>
</ul>
<br> <br> <br> <br>
```

```
<div class='azul'>
Um gradiente azul clássico, utilizando 2 tons similares.
</div>
<br> <br>
```

```
<div class='cinza'>
Outro gradiente.
</div>
```

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



# CSS – Exemplo: 14UsandoCores.css (1)

```
figcaption {  
  bottom: 5px;  
  margin: 0 5px;  
  padding: 5px;  
  width: 260px;  
  background-color: rgba(0,0,0,0.5);  
  color: #FFF;  
}
```

Último atributo pode  
incluir a opacidade da cor,  
passe o mouse você vai  
vendo a cor

```
ul {  
  height: 35px;  
  padding: 0;  
  width: 210px;  
  background-color: #CCC; /*cinza */  
  color: white;  
  border: 1px solid rgba(0,0,0,0.3);  
}
```

# CSS – Exemplo: 14UsandoCores.css (2)

```
li {  
  float: left;  
  line-height: 35px;  
  font-size: 12px;  
  width: 50px;  
  text-align: center;  
  border-left: 1px solid rgba(255,255,255, 0.3);  
  border-right: 1px solid rgba(0,0,0, 0.3);  
}
```

```
li:first-child { /* para não colocar borda esq. no primeiro filho por causa outra  
borda */  
  border-left: none;  
}
```

```
li:last-child { /* para não colocar borda direita no último filho */  
  border-right: none;  
}
```



# CSS – Exemplo: 14UsandoCores.css (3)

```
.azul {  
font-weight: bold;  
height: 50px;  
line-height: 25px;  
margin-bottom: 10px;  
padding: 5px;  
text-align: center;  
width: 220px;
```

```
    background: -webkit-gradient(linear, left top, left bottom, from(#4377FA),  
to(#0537B7)); /*tipo gradiente, onde começa e termina, cor inicial, cor final #fff=branco  
#000=preto */  
}
```

```
.cinza{  
font-weight: bold;  
height: 50px;  
line-height: 25px;  
margin-bottom: 10px;  
padding: 5px;  
text-align: center;  
width: 220px;  
background: -webkit-gradient(linear, center top, center bottom, from(#fff), to(#000));  
}
```

Infelizmente ainda não se tem uma sintaxe universal e suporte total a alguns recursos mas existem algumas técnicas para contornar esse problema, usam-se os prefixos. No caso do Safari e Chrome usa-se -webkit. Dúvidas ver: [caniuse.com](http://caniuse.com)

# CSS – Exemplo: 14UsandoCores.css (4)

```
.verde {  
    width: 200px;  
    height: 200px;  
    background-color: #FFF;  
    background-image: -webkit-linear-gradient(green, red 20%);  
}
```

/\*O padrão é que o gradiente ocupe 100% do elemento, mas muitas vezes queremos fazer apenas um detalhe.

\*/



## CSS – Propriedade Color - RGBA e a diferença da propriedade OPACITY

Antes só era possível escrever cores sólidas, sem escolher a opacidade dessas cores. O CSS3 nos trouxe a possibilidade de modificar a opacidade dos elementos via propriedade opacity. Lembrando que quando for modificada a opacidade do elemento, tudo o que está contido nele também fica opaco e não apenas o background ou a cor dele. Veja o exemplo abaixo e compare as imagens

O RGBA funciona da mesma forma que o RGB, ou seja, definindo uma cor para a propriedade. No caso do RGBA, além dos 3 canais RGB (Red, Green e Blue) há um quarto canal, A (Alpha) que controla a opacidade da cor.

Exemplo:

```
p { background:rgba(255,255,0, 0.5);  
padding:10px;  
font:13px verdana; }
```

# CSS – Outros padrões de Cores: HEX e HSL

## Padrão de cor HSL

O padrão HSL é representado por Hue (tom), Saturation (saturação) e Lightness (luminosidade). O valor do tom é medido em ângulos expresso em graus. O valor de saturação é medido em porcentagem, sendo 0% um leve sombreado de saturação e 100% uma saturação total da cor. O valor de luminosidade também é expresso em porcentagem. O HSL foi projetado para ser legível por humanos e está ganhando popularidade, principalmente como uma alternativa RGB.

*Fonte: <https://tecnoblog.net/413962/o-que-sao-os-padroes-hex-rgb-e-hsl-de-cores/>*



# CSS – Outros padrões de Cores: HEX e HSL

## Padrão de cor HEX

O **Hexadecimal** é um padrão em que se misturam letras de **A** a **F** (**seis** letras) e números de **zero** a **nove**.

O padrão hexadecimal se divide em **três pares** de combinações, resultando em **seis** algarismos alfanuméricos.

Exemplo:

```
background: #000000;/*preto*/
```

```
background: #ffffff;/*branco*/
```

*Fonte: <https://tecnoblog.net/413962/o-que-sao-os-padroes-hex-rgb-e-hsl-de-cores/>*

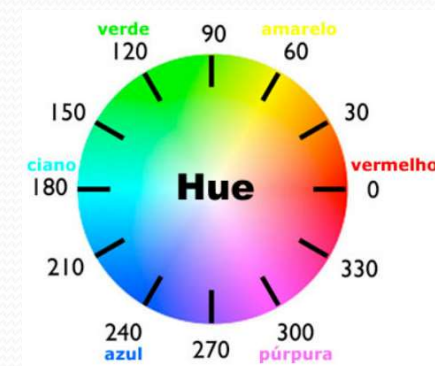
# CSS – Outros padrões de Cores: HEX e HSL/HSLA

## Padrão de cor HSL

O padrão HSL é representado por Hue (tom), Saturation (saturação) e Lightness (luminosidade). O valor do tom é medido em ângulos expresso em graus. O valor de saturação é medido em porcentagem, sendo 0% um leve sombreado de saturação e 100% uma saturação total da cor. O valor de luminosidade também é expresso em porcentagem. O HSL foi projetado para ser legível por humanos e está ganhando popularidade, principalmente como uma alternativa RGB.

Exemplo: `hsl(0, 100%, 50%)` → vermelho  
`hsl(240, 100%, 50%)` → azul

**\*\* HSL → permite configurar a opacidade no 4º canal**



Fonte: <https://tecnoblog.net/413962/o-que-sao-os-padroes-hex-rgb-e-hsl-de-cores/>  
[https://developer.mozilla.org/pt-BR/docs/Web/CSS/CSS\\_Colors/Color\\_picker\\_tool](https://developer.mozilla.org/pt-BR/docs/Web/CSS/CSS_Colors/Color_picker_tool)



# CSS – Outros padrões de Cores: HEX e HSL

## Padrão de cor HEX

O **código Hexadecimal** para **cores** consiste em seis letras ou números precedidos do “#” e seus números significam: Os dois primeiros elementos representam a intensidade de vermelho; O terceiro e quarto elementos representam a intensidade de verde; Os dois últimos a intensidade de azul.

*Fonte: <https://tecnoblog.net/413962/o-que-sao-os-padroes-hex-rgb-e-hsl-de-cores/>*

# CSS – Padrões de Cores: Qual usar

Escolha de um modelo de cores:

- Familiaridade com o formato;
- Mesmo formato que o resto da sua equipe de desenvolvimento usa para facilitar a manutenção;
- Mudar o tom de uma cor para mais escuro ou mais claro ou até mesmo aplicar uma opacidade é mais simples com hex e rgb;
- Há muita coisa definida com hex;
- O hsl foi projetado para ser legível por humanos;
- HSL também é utilizado em softwares do tipo Gimp, Photoshop, etc. (quem trabalha com design)

*Fonte: <https://tecnoblog.net/413962/o-que-sao-os-padroes-hex-rgb-e-hsl-de-cores/>*

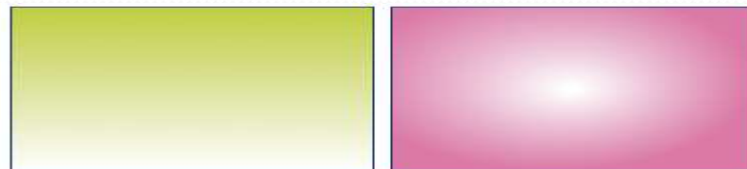


## CSS – Gradient

No contexto das CSS usa-se de forma genérica o termo gradiente para se referir a *gradiente de cores*(degradê).

Define-se gradiente CSS como sendo uma transição de cores em um determinado espaço. Com uso de declarações CSS é possível definir se a transição será linear ou radial, em que quantidade de espaço será feita, bem como se duas ou mais cores participam do gradiente.

Diz-se que um gradiente é linear quando a transição das cores se faz segundo um eixo (linha reta) que possui um *sentido* (por exemplo: horizontal) e uma *direção* (por exemplo: da esquerda para a direita). Radial irradia da origem (centro).



## CSS – Propriedade font-size (tamanho da fonte)

A propriedade font-size define o tamanho do texto de um elemento HTML. O tamanho do texto normalmente é utilizado em pixels, ems ou %.

```
h1 {  
  color: red;  
  font-family: 'Shift', sans-serif;  
  font-size: 48px;  
}
```



# CSS – Propriedade font-size (tamanho da fonte)

Ems "(EM)": uma unidade expansível que é usado em documento da Web. Um em é igual ao tamanho da fonte de corrente, por exemplo, se o tamanho da fonte do documento é de 12 pontos, 1em é igual a 12pt. Ems são escaláveis na natureza, assim 2em seria igual 24pt, 0,5em seria igual 6pt, etc. Ems estão se tornando cada vez mais popular nos documentos da web devido a escalabilidade.

Pixels (px): são unidades de tamanho fixo que são usados na tela do computador. Um pixel é igual a um ponto na tela do computador (a menor divisão da resolução do ecrã).

Porcentagem (%): é muito parecida com a unidade "em", salvo algumas diferenças fundamentais. Em primeiro lugar, o tamanho da fonte atual é igual a 100% (ou seja, 12pt = 100%). Se for 16 pixels = 100%, 10 pixels é igual a 62.5%. ( $16 \times 0.625 = 10$ ). Enquanto estiver usando a porcentagem, o texto permanece totalmente escalável para dispositivos móveis e para a acessibilidade.

	body { font-size: 100%; }	body { font-size: 120%; }
font-size: 1em	The quick brown fox	The quick brown
font-size: 12pt	The quick brown fox	The quick brown fox
font-size: 16px	The quick brown fox	The quick brown fox
font-size: 100%	The quick brown fox	The quick brown

# CSS – Propriedade text-align e spacing

## Alinhamento e espaçamento de texto

A propriedade que faz o alinhamento do texto é text-align.

```
p {  
text-align: right;  
}
```

No exemplo o parágrafo deve ter texto alinhado para a direita.

Center para o centro, justify para justificado e left para esquerda

Essa propriedade propaga-se em cascata.

É possível configurar também uma série de espaçamentos de texto com o CSS:

```
p { line-height: 3px; /* tamanho da altura de cada linha */  
letter-spacing: 3px; /* tamanho do espaço entre cada letra */  
word-spacing: 5px; /* tamanho do espaço entre cada palavra */  
text-indent: 30px; /* tamanho da margem da primeira linha do texto */ }
```



# CSS – Elementos Estruturais

O conjunto de tags do HTML é bem vasto mas é também limitado.

Quando não consegue achar a tag certa para aquele conteúdo pode ser utilizar alguns recursos, por exemplo as tags <div> e <span>. São tags sem nenhum significado especial mas que podem servir para agrupar um certo conteúdo, tanto um bloco da página quanto um pedaço de texto.

É possível estilizar esses divs e spans com CSS customizado. ***Por padrão, eles não têm estilo algum.***

# CSS – Elementos Estruturais

<DIV> – Vem de “divisão”, é possível criar um bloco, uma divisão, e dentro deste bloco ter uma imagem, links, textos e o que mais desejar. É possível aplicar o CSS nesse bloco, e tudo que estiver dentro dela vai receber aquelas regras de estilo.

As *divs* são IMPORTANTÍSSIMAS para estruturar, criar uma ordem lógica e organizar um site.

Por ex. cria-se uma div para o cabeçalho, outra para o conteúdo, outra div para os menus e uma última para o rodapé, então usa o CSS para estilizar cada uma dessas divs separadamente.



# CSS – Exemplo 15UsandoDivs.html (1)

```
<!DOCTYPE HTML>
<html>
<head>
<title>Testando Divs CSS</title>
<meta name="author" content=Denilce/>
<link href="15UsandoDivs.css" rel="stylesheet">
</head>
```

```
<body>
<div class="todo">
    <div class="topo">
        Aqui é o topo
    </div>
```

```
        <div class="meio">
            <div class="esquerda">
                <div class="menu">
                    <div class="itemMenu">Principal</div>
                    <div class="itemMenu">Empresa</div>
                </div>
            </div>
        </div>
```

**\*\*ATENÇÃO: ALTERAR NO CSS A COR DA CLASSE MEIO PARA FICAR IGUAL A DOS ITENS DO MIOLO. CRIAR MAIS UM ITEM DO MIOLO (PODE REDUZIR O TAMANHO DAS ANTERIORES SE DESEJAR)**

# CSS – Exemplo 15UsandoDivs.html (2)

```
<div class="miolo">  
    <div class="item_miolo">Seção 1 miolo</div>  
    <div class="item_miolo">Seção 2 miolo </div>  
</div>
```

```
<div class="direita">  
    <div class="item_direita">Seção 1 direita</div>  
    <div class="item_direita">Seção 2 direita</div>  
</div>
```

```
</div>
```

```
<div class="rodape">  
    <div class="rodape_direita">  
        Aqui está o rodapé  
    </div>
```

```
</div>
```

```
</div>
```

```
</body>
```

```
</html>
```



# CSS – Exemplo 15UsandoDivs.css (1)

```
body {  
    margin:0px;  
    background-color: #cccccc;  
    font-family:"Trebuchet MS", Tahoma, Arial, Verdana;  
    font-size:12px;  
    color:#000;  
}  
  
.todo {  
    width: 750px;  
    border: 1px solid black;  
    margin:0 auto; /* topo e bottom = 0 direita e esq. automatica */  
    background-color: greenyellow;  
}  
  
.topo {  
    width: 746px;  
    height: 50px;  
    border: 2px solid red;  
}
```

# CSS – Exemplo 15UsandoDivs.css (2)

```
.meio {  
    width: 750px;  
    min-height: 400px;  
    background-color: deeppink;  
}
```

```
.rodape {  
    width: 746px;  
    height: 75px;  
    border: 2px solid brown;  
    background-color: blue;  
}
```

```
.esquerda {  
    float: left;  
    width: 150px;  
    min-height: 400px;  
    border: 2px solid brown;  
}
```



# CSS – Exemplo 15 Usando Divs.css (3)

```
.menu {  
    width: 150px;  
    height: 400px;  
}  
  
.itemMenu {  
    width: 140px;  
    height: 21px;  
    background-color: lightyellow;  
    padding: 3px 0px 0px 10px;  
    border-bottom: 1px solid black;  
}  
  
.miolo {  
    float: left;  
    width: 438px;  
    min-height: 400px; /* altura minima, importante se tem imagem no mesmo  
div. usando float por ex.*/  
    border: 2px solid blue;  
}
```

# CSS – Exemplo 15UsandoDivs.css (4)

```
.item_miolo {  
    width: 428px;  
    height: 150px;  
    text-align: center;  
    background-color: #cccccc;  
    margin: 2px 5px 4px 5px;  
}
```

```
.direita {  
    float: right;  
    width: 150px;  
    min-height: 400px;  
    border: 2px solid yellow;  
}
```

```
.item_direita {  
    width: 140px;  
    height: 100px;  
    text-align: center;  
    background-color: #cccccc;  
    margin: 4px 5px 4px 5px;  
}
```



# CSS: Seletores de ID e filho

- É possível aplicar propriedades visuais a um elemento selecionado pelo valor de seu atributo id. Para isso, o seletor deve iniciar com o caractere "#" seguido do valor correspondente.

Ex.: `#cabecalho { color: white; text-align: center; }`

- O seletor acima fará com que o elemento do HTML que tem o atributo id com valor "cabecalho" tenha seu texto renderizado na cor branca e centralizado. Note que não há nenhuma indicação para qual tag a propriedade será aplicada. Pode ser tanto uma `<div>` quanto um `<p>`, até mesmo tags sem conteúdo como uma `<img>`, desde que essa tenha o atributo id com o valor "cabecalho".
- ***Como o atributo id deve ter valor único no documento, o seletor deve aplicar suas propriedades declaradas somente àquele único elemento e, por cascata, a todos os seus elementos filhos***

# CSS – Exemplo Seletor 16UsandoIDs.html

```
<html>
<head>
  <link rel="stylesheet" href="16UsandoIDs.css">
</head>
<body>
<div id="tudo">
  <div id="conteudo">
    <h1>Pagina Centrada com CSS sem scroll até 800x600</h1>
    <p> Não use nenhum elemento HTML figuras, tabelas, etc...) com
largura superior a 750px.</p>

    <p>Foi recomendada largura máxima de 750px para inserção de elementos
na página porque definimos 5px de padding esquerdo e direito. Então 760px -
5px - 5px = 750px
  </p>
  </div>
</div>
</body>
</html>
```



# CSS – Exemplo Seletor 16UsandoIDs.css

```
body {  
    margin:0;  
    padding:0;  
    background:#ccc;  
    text-align:center;  
}  
#tudo {  
    width: 760px;  
    margin:0 auto;  
    text-align:left;  
}  
#conteudo {  
    padding: 5px;  
    background-color: #eee;  
}
```

# CSS: Seletores Hierárquico

- Podemos ainda utilizar um seletor hierárquico que permite aplicar estilos aos elementos filhos de um elemento pai:

```
#rodape img {  
  margin-right: 35px;  
  vertical-align: middle;  
  width: 94px;  
}
```

No exemplo, o elemento pai rodape é selecionado pelo seu id. O estilo será aplicado apenas nos elementos img filhos do elemento com id=rodape.



# CSS: Seletores Encadeados

Exemplo 1:

```
div p strong a {  
  color: red;  
}
```

Este seletor formata o link (a), que está dentro de um Strong (negrito), que está dentro de p e que por sua vez está dentro de um div.

# CSS: Seletores Encadeados

## Exercício

```
.content .post-text p strong em {  
  color: red;  
}
```



O HTML deste seletor seria este:

```
<div class="content">  
  <div class="post-text">  
    <p>  
      Exemplo de <strong>seletores <em>encadeados</em></strong>.  
    </p>  
  </div>  
</div>
```

A quem o CSS irá afetar????



# CSS: Seletores Agrupados

Exemplo de seletor agrupado:

```
strong, em, span, .classea {  
  color: red;  
}
```

Elementos separados por vírgula herdam a mesma formatação.

## CSS – Tag Span

- Ao usar os seletores ID e CLASS em uma tag qualquer para estilizar, estará estilizando todo o conteúdo da tag, o que nem sempre é o que desejado.
- A tag **<span>**, permite que se use os seletores ID e CLASS em apenas um trecho da tag, ou seja, a tag *span* é uma tag *inline*, pois ela é usada dentro de outras tags, é uma espécie de elemento interno de outro elemento (de outra tag).



# CSS – Exemplo 15UsandoSpan.html

```
<html>
```

```
  <head>
```

```
    <title> Teste de Span </title>
```

```
    <link rel="stylesheet" href="15UsandoSpan.css">
```

```
  </head>
```

```
<body>
```

```
<p>Neste exemplo do <span class="principal">Curso de  
HTML</span>, vamos mostrar uma exemplo do <span  
class="destaque">SPAN</span> e sua importância.</p>
```

```
</body>
```

```
</html>
```

# CSS – Exemplo 15UsandoSpan.css

```
p{  
    font-size: 30px;  
    font-family: "Arial";  
}
```

```
.principal {  
    font-weight: bold;  
    color: red;  
    font-size: 25px;  
}
```

```
.destaque {  
    font-style: italic;  
    font-weight: bold;  
    font-size: 45px;  
    color: blue;  
}
```



## CSS – Quando usar ID ou CLASS

- Dependendo do caso tanto faz. Mas é bom lembrar que ids são mais fortes, devem ser únicos na página, sempre;
- Usar classes facilita reuso de código e flexibilidade;
- Além disso, um elemento pode ter mais de uma classe ao mesmo tempo, aplicando estilos de várias regras do CSS ao mesmo tempo.

## CSS – Quando usar ID ou CLASS



Como selecionar elementos dentro do arquivo CSS, em uma classe chamada "nova"?

- a) \*nova    b) nova    c) .nova    d) #nova



# CSS: Fluxo de Documento e Float

- Float: Esta propriedade permite que se tire um certo elemento do fluxo vertical do documento, o que faz com que o conteúdo abaixo dele flua ao seu redor.

Ex.

```
#familia{  
    float: right;  
    margin: 0 0 10px 10px;  
}
```

\*\* Uma imagem com id familia ficará a direita da página e o texto à esquerda.

# CSS – Exemplo 16UsandoFloat.html (1)

```
<!doctype html>
<html lang='pt-BR'>

<head>
  <meta charset="UTF-8">
  <title> Título da Página </title>
  <link rel="stylesheet" href="16UsandoFloat.css">
</head>
```

```
<body>
  <div class="container">
    <div class="texto1">
      <h1>CIDADES BRASILEIRAS</h1>
    <div />
```

```
    <div class="texto2">
      <h2>São Paulo é um município brasileiro, capital do estado de São Paulo e
principal centro financeiro, corporativo e mercantil da América do Sul. É a cidade mais
populosa do Brasil, do continente americano, e de todo o hemisfério sul.</h2>
      
    </div>
```

**\*\*ATENÇÃO: COLOCAR COMENTÁRIO NO CSS → CLASSE TEXTO2 IMG NA LINHA DO FLOAT, VER O QUE ACONTECE.**

**→ EM ALGUNS EDITORES COMENTÁRIO É CTRL+:**



## CSS – Exemplo 16 Usando Float.html (2)

```
<div class="texto3">  
  <h3>Locais para conhecer</h3>  
  
  <ul>  
    <li> MASP</li>  
    <li>Museu Ipiranga</li>  
    <li>Mercadão Municipal</li>  
    <li>25 de Março</li>  
  </ul>  
</div>  
</div>  
</body>  
</html>
```

# CSS – Exemplo 16UsandoFloat.css (1)

```
body {  
background: #FFF1D6 url(fundao.jpg);  
font-family: "Lucida Grande", "Lucida Sans Unicode", Verdana,  
sans-serif;  
line-height: 1.6;  
}
```

```
.container {  
margin: 0 auto;  
width: 960px; /* adequando tela 960px */  
}
```

```
.texto1 {  
background-color: #FFF;  
border: 1px solid #CCC;  
border-color: #CCC #999 #999 #CCC;  
margin-bottom: 20px;  
padding: 10px; /* espaçamento entre a borda e o título */  
}
```



## CSS – Exemplo 16UsandoFloat.css (2)

```
.texto2 {  
border-bottom: 1px dashed #7E9F19;  
border-color: blue;  
margin: 0;  
}
```

```
.texto2 img {  
border: 1px solid #7E9F19;  
float: left;  
margin: 10px 10px 0 0;  
padding: 2px;  
width: 300px;  
height: 300px;  
}
```

```
.texto3 {  
background-color: #FFFBE4;  
border: 1px solid #C9BC8F;  
margin: 10px 10px 0 0;  
padding: 10px;  
float: right;  
width: 260px;  
}
```

# CSS – Position

O Position não serve para diagramar a estrutura de layouts. Para isso, utiliza-se **propriedade float do css**. O Position vai servir para fazer coisas mais simples.

## COORDENADAS

Para posicionar seus elementos, é necessário inserir uma coordenada. Essas coordenadas são comandadas pelas propriedades: top, left, right ou bottom. Todos os valores de positions só trabalham com essas coordenadas. Obviamente, se definir um left para o seu elemento, não faz sentido definir um right. A mesma coisa para o bottom e o top. Em código ficaria assim:

```
div { position: absolute; top: 150px; left: 150px;}
```



## •POSITION FIXED

Irá fixar a posição do elemento na coordenada definida. A medida que a página é rolada, o elemento continua fixo na posição definida e o conteúdo da página rola normalmente. Geralmente é usado para fixar elementos como cabeçalhos ou sidebars.

## •POSITION RELATIVE

Todos os positions precisam de um ponto para iniciar o cálculo da coordenada para assim posicionar o elemento na tela. Ao contrário do que muitos acham, esse ponto não é o ponto central do elemento, o ponto base é o canto superior esquerdo do elemento. O position relative posiciona o elemento em relação a si mesmo. Ou seja, o ponto zero será o canto superior esquerdo, e ele começará a contar a partir dali.

## •POSITION ABSOLUTE

É um tanto diferente do Relative. Enquanto o elemento com Position Relative utiliza seu próprio canto para referenciar sua posição, o elemento com Position Absolute se utiliza do ponto superior esquerdo de outros elementos. Estes elementos são os “pais” dele do elemento com position absolute. Mais especificamente o pai.

## •POSITION STATIC

É o padrão para todos os elementos, e a sua posição não é afetada por nenhuma das propriedades de coordenadas, deixando que o navegador posicione o elemento no seu lugar de origem.

# CSS – Position Exemplo: 18UsandoPosition.html (1)

```
<!DOCTYPE html>
<html lang="pt-br">
<head>
  <meta charset="UTF-8">
  <title>Document</title>
  <link rel="stylesheet" href="18UsandoPosition.css">
</head>
<body>

<div class='boxabsolute'>
  Teste absolute
</div>

  <br>
  <br>
  <br>
  <br>
  <br>
  <br>

<div class='boxrelative'>
  Teste relative
</div>
  <br>
  <br>
  <br>
```



# CSS – Position Exemplo: 18UsandoPosition.html (2)

```
<br> <br> <br> <br> <br> <br>  
<div class='boxfixed'>  
Teste fixed  
</div>
```

```
<br><br><br>
```

```
<div class='boxstatic'>  
Teste Static  
</div>
```

```
<br><br><br>
```

```
<p> ROLE A TELA E VEJA O QUE ACONTECE </p>
```

```
<p> POSITION FIXED
```

Irá fixar a posição do elemento na coordenada definida. A medida que a página é rolada, o elemento continua fixo na posição que você definiu e o conteúdo da página rola normalmente. Geralmente é usado para fixar elementos como cabeçalhos ou sidebars. </p>

```
<p>POSITION RELATIVE
```

Todos os positions precisam de um ponto para iniciar o cálculo da coordenada para assim posicionar o elemento na tela. Ao contrário do que muitos acham, esse ponto não é o ponto central do elemento, o ponto base é o canto superior esquerdo do elemento. O position relative posiciona o elemento em relação a si mesmo. Ou seja, o ponto zero será o canto superior esquerdo, e ele começará a contar a partir dali.</p>

# CSS – Position Exemplo: 18UsandoPosition.html (3)

<p>POSITION ABSOLUTE

É um tanto diferente do Relative. Enquanto o elemento com Position Relative utiliza seu próprio canto para referenciar sua posição, o elemento com Position Absolute se utiliza do ponto superior esquerdo de outros elementos. Estes elementos são os parentes dele do elemento com position absolute. Mais especificamente o pai.

</p>

<p>POSITION ABSOLUTE

É um tanto diferente do Relative. Enquanto o elemento com Position Relative utiliza seu próprio canto para referenciar sua posição, o elemento com Position Absolute se utiliza do ponto superior esquerdo de outros elementos. Estes elementos são os parentes dele do elemento com position absolute. Mais especificamente o pai.

</p>

</body>

</html>



# CSS – Position Exemplo: 18UsandoPosition.css

```
.boxabsolute {  
background-color: green;  
height: 100px;  
width: 100px;  
position: absolute;  
top: 30px;  
left: 15px;  
}
```

```
.boxrelative {  
background-color: PapayaWhip;  
height: 200px;  
width: 200px;  
position: relative;  
top: 50px;  
left: 15px;  
}
```

```
.boxfixed {  
background-color: red;  
height: 50px;  
width: 50px;  
position: fixed;  
top: 30px;  
left: 15px;  
}
```

# Qual é mais indicado? Float ou Position?



Supondo que se deseja colocar cupom de desconto em cima de uma foto (sobreposição-> colocar por cima).

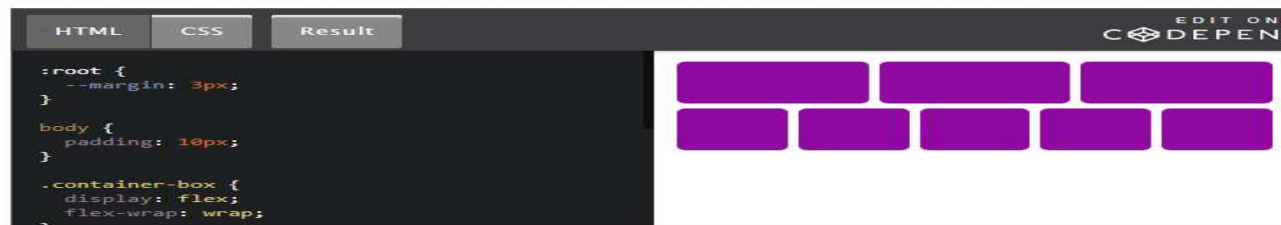
- a) Float
- b) Position



# FlexBox

Em FlexBox a ideia principal é dar ao container a capacidade de alterar a largura / altura e ordem de seus itens para um melhor preenchimento do espaço disponível e, principalmente, para acomodar todos os tipos de dispositivos de exibição e tamanhos de tela. Ele é responsivo.

*Flexbox* é para layouts unidimensionais – qualquer coisa que precisa ser disposta em uma linha reta.



*\*\* Ver mais em: <https://origamid.com/projetos/flexbox-guia-completo/>*

# FlexBox – FlexBox Exemplo: 16\_1UsandoFlexBox.html

```
<!DOCTYPE html>
<html lang="pt-br">

<head>
  <meta charset="UTF-8">
  <title>Testando FlexBox</title>
  <link rel="stylesheet" href="16_1UsandoFlexBox.css">
</head>

<body>
  <div class="container-box">
    <!-- primeiro propriedades da
    classe box e depois de a -->
    <div class="box a">teste1</div>
    <div class="box a">teste2</div>
    <div class="box a">teste3</div>
    <div class="box b">teste4</div>
    <div class="box b">teste5</div>
    <div class="box b">teste6</div>
    <div class="box b">teste7</div>
    <div class="box b">teste8</div>
  </div>
</body>
</html>
```



# FlexBox Exemplo: 16\_1UsandoFlexBox.css (1)

```
:root {  
  --margin: 10px;  
}
```

```
/* variavel fica  
no elemento raiz */
```

```
body {  
  padding: 10px;  
}
```

```
.container-box {  
  display: flex;  
  flex-wrap: wrap;  
  /* Define se os itens devem quebrar ou não a linha*/  
}
```

```
.a {  
  width: calc((100%/3) - (var(--margin)* 2));  
}
```

## FlexBox Exemplo: 16\_1UsandoFlexBox.css (2)

Para saber mais sobre FlexBox veja:

[https://www.alura.com.br/artigos/css-guia-do-flexbox?gclid=CjwKCAjw1JeJBhB9EiwAV612yxkQN-IwnxqowKkdU8WYyFzJ2HWOkajduRNSamnRk3lYHwGf4MdNwxoCg1kQAvD\\_BwE](https://www.alura.com.br/artigos/css-guia-do-flexbox?gclid=CjwKCAjw1JeJBhB9EiwAV612yxkQN-IwnxqowKkdU8WYyFzJ2HWOkajduRNSamnRk3lYHwGf4MdNwxoCg1kQAvD_BwE)



## CSS – FlexBox

O *Grid Layout* serve para controlar o dimensionamento e o alinhamento em duas dimensões.

Grid Lines - linhas que definem o grid, que podem ser distribuídas de forma horizontal ou vertical.

Grid Tracks - espaço horizontal ou vertical entre duas *Grid Lines*.

Grid Cell ou Grid Item - espaço entre quatro *Grid Lines*, sendo a menor unidade no grid (“célula”)

Grid Areas - espaço no *Grid* usado para exibir um ou mais *Grid Cells/ Items*.

Grid Containers - pai direto de todos os itens do *grid*, sendo o elemento que recebe a propriedade *display: grid*.

Grid items - itens que representam o conteúdo do grid; cada filho direto do *grid container* torna-se um *grid item*.

\*\* Ver: <https://imasters.com.br/desenvolvimento/adeus-flexbox-bem-vindo-css-grid-layout/?trace=1519021197&source=single>

## CSS – Grid Layout

O *Grid Layout* serve para controlar o dimensionamento e o alinhamento em duas dimensões.

Grid Lines - linhas que definem o grid, que podem ser distribuídas de forma horizontal ou vertical.

Grid Tracks - espaço horizontal ou vertical entre duas *Grid Lines*.

Grid Cell ou Grid Item - espaço entre quatro *Grid Lines*, sendo a menor unidade no grid (“célula”)

Grid Areas - espaço no *Grid* usado para exibir um ou mais *Grid Cells/ Items*.

Grid Containers - pai direto de todos os itens do *grid*, sendo o elemento que recebe a propriedade *display: grid*.

Grid items - itens que representam o conteúdo do grid; cada filho direto do *grid container* torna-se um *grid item*.

\*\* Ver: <https://imasters.com.br/desenvolvimento/adeus-flexbox-bem-vindo-css-grid-layout/?trace=1519021197&source=single>



## CSS – Grid Layout

Veja mais sobre Grid Layout em:

<https://css-tricks.com/snippets/css/complete-guide-grid/>

## CSS — Grid Layout – Exemplo:16\_2UsandoGridLayout.html

```
<!DOCTYPE html>
<html lang="pt-br">

<head>
  <meta charset="UTF-8">
  <title>Testando FlexBox</title>
  <link rel="stylesheet" href="16_2UsandoGridLayout.css">
</head>

<body>
  <div class="container-box">
    <div class="box a"></div>
    <div class="box b"></div>
    <div class="box c"></div>
    <div class="box d"></div>
    <div class="box e"></div>
    <div class="box f"></div>
  </div>
</body>
</html>
```



# CSS — Grid Layout – Exemplo:16\_2UsandoGridLayout.css (1)

```
body {  
  padding: 10px;  
}  
  
.container-box {  
  display: grid;  
  /* espaçamento entre as linhas e colunas */  
  grid-gap: 10px;  
  grid-template-columns: auto;  
  grid-template-rows: 50px 100px;  
  
  /* disposição na tela observe que a aparece 2x na primeira coluna e f aparece 2x na segunda */  
  grid-template-areas: 'a b c d' 'a e f f';  
}  
  
.box {  
  background-color: #FF8C00;  
  border-radius: 5px;  
}  
  
.a {  
  grid-area: a;  
}  
  
.b {  
  grid-area: b;  
}
```

## CSS — Grid Layout — Exemplo:16\_2UsandoGridLayout.css (2)

```
.c {  
  grid-area: c;  
}  
  
.d {  
  grid-area: d;  
}  
  
.e {  
  grid-area: e;  
}  
  
.f {  
  grid-area: f;  
}
```

**\*\* Ver também 17UsandoGrid.html**



## CSS – Block vs Inline

- Os elementos do HTML, quando renderizados no navegador, podem comportar-se basicamente de duas maneiras diferentes no que diz respeito à maneira como eles interferem no documento como um todo: em bloco (block) ou em linha (inline).
- Elementos em linha são aqueles que ocupam somente o espaço necessário para que seu próprio conteúdo seja exibido, permitindo que outros elementos em linha possam ser renderizados logo na sequência, exibindo diversos elementos nessa mesma linha. Em bloco ocupa o espaço inteiro.

*\*\* Cada elemento tem um valor padrão para o display dependendo de seu tipo. Ex. Div, p, h1, form, header, footer, section são block. Span é inline.*

## CSS – Block vs Inline

As tags de heading `<h1>` a `<h6>` são colocados em block ou inline?

E os parágrafos `<p>`?

E as divisões `<div>`?





## CSS – Block vs Inline Exemplo: 19UsandoDisplayBlockInline.html (1)

```
<html>
<head>
<link rel="stylesheet" href="19UsandoDisplayBlockInline.css">
</head>
<body>
  <FORM method="post" ACTION="">

  <label for='content'>Mensagem:</label>
  <textarea id='content'></textarea>
  <span class='hint'>
    Escreva a sua mensagem, em até 500 caracteres.
  </span>

  <br>
  <br>
```

## CSS — Block vs Inline Exemplo: 19UsandoDisplayBlockInline.html (2)

```
<p>
  <label for="email">EMAIL DA PESSOA:</label>
  <input type='email' id='email'>
  <span class='error'>Preencha o seu e-mail corretamente.</span>
</p>
```

```
<br><br><INPUT TYPE="submit" VALUE="Gravar">
<INPUT TYPE="reset" VALUE="Limpar">
</form>
```

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



## CSS — Block vs Inline Exemplo: 19UsandoDisplayBlockInline.css

```
.hint {  
display: none;  
} /* limpa o hint inicialmente */
```

```
input:focus + .hint, textarea:focus + .hint {  
display: inline-block; /* mostra do lado */  
} /* quando der o foco no input, mostra o hint e dá o foco no  
textarea */
```

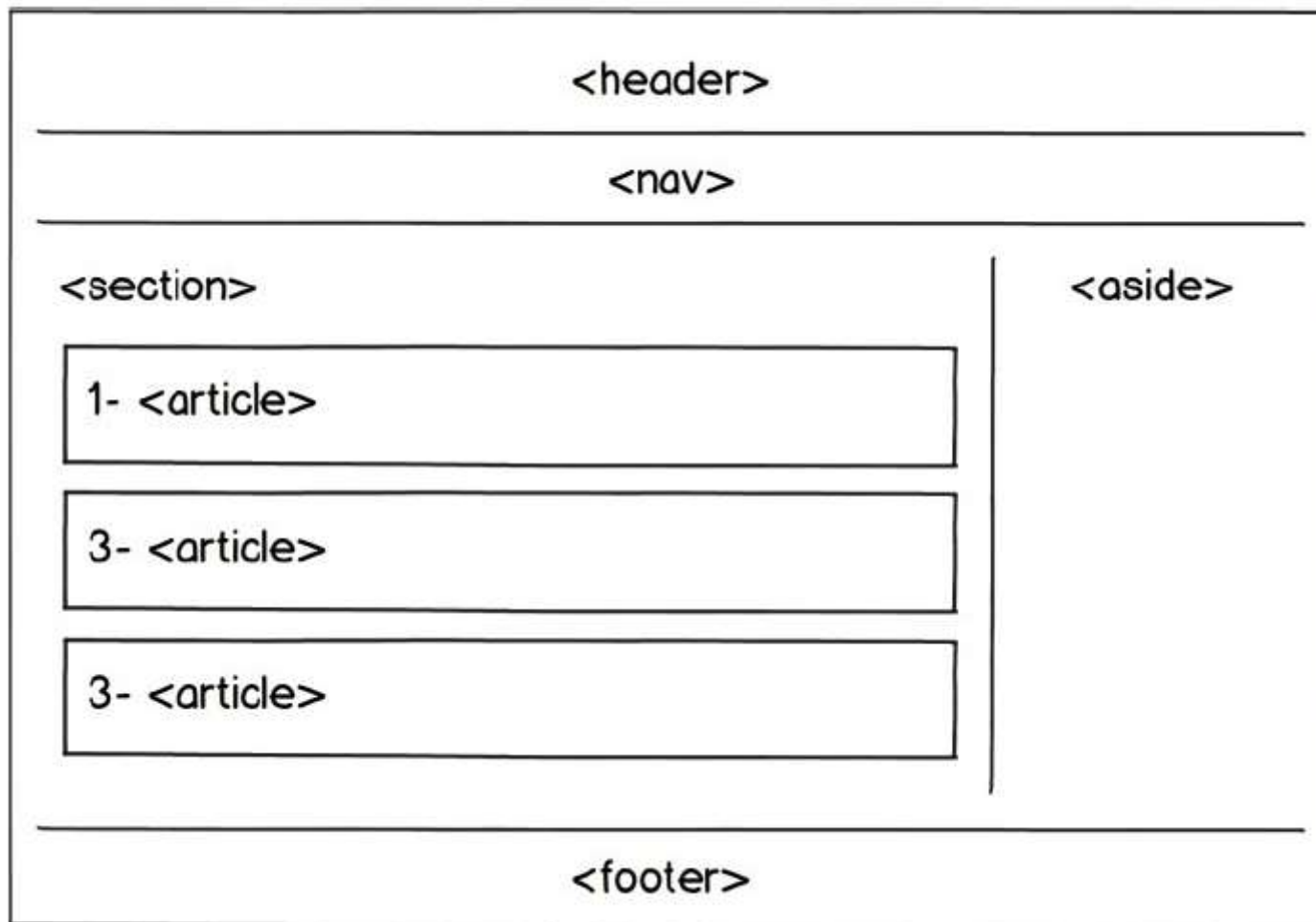
```
.error {  
color: red;  
display: block; /* mostra abaixo */  
font-size: 0.8em;  
font-style: italic;  
margin: 5px 0 0 90px;  
}
```

## CSS – HTML Semântico

- As tags <header>, <section> e <footer> - são tags do HTML5.
- Facilita a leitura do HTML.
- HTML semântico carrega significado independente da sua apresentação visual. Isso é importante quando o conteúdo será consumido por clientes não visuais. Neste caso, a estrutura semântica do HTML é essencial para ele entender as partes do conteúdo. Os robôs de busca como o Google também são leitores não visuais da sua página.



# CSS – HTML Semântico



# CSS – HTML Semântico

```
<header> O meu blog sobre HTML5 </header>

<nav> Página principal | Sobre o autor | Contato </nav>

<article>
  <header> Utilizando as tags <header> e <article>... </header>
  Publicado a 3 dias atrás

  ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore
  et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi
  ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate velit
  esse cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non proident, sunt
  in culpa qui officia deserunt mollit anim id est laborum.Lorem ipsum dolor sit amet, consectetur
  adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad
  minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo
  consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat

</article>
```

Adaptado de (Mazza,2012)



# CSS – HTML Semântico

**section** - utilizado para representar uma seção genérica, geralmente com um cabeçalho próprio e o seu conteúdo;

- **nav** - representação de um bloco principal de links de navegação - nem todo grupo de links deve ser tratado como um nav;
- **aside** - a tag aside pode ser utilizada para representar uma seção de conteúdo secundário ou auxiliar a outro pedaço de maior importância. Por ex.: Citações, links de referência ou notas adicionais.
- **header** - referente ao cabeçalho de uma seção específica (ou da própria página) contendo títulos, introduções e outros elementos similares;
- **footer** - o rodapé referente a um bloco de conteúdo;
- **article** - normalmente identifica o conteúdo em si, como uma notícia de um portal, um post em blog ou um comentário em uma lista de comentários.

## CSS – HTML Semântico – Exemplos

ver [16\\_3UsandoHtmlSemantico.html](#)

[16\\_4UsandoHtmlSemanticoComFlexBox.html](#)

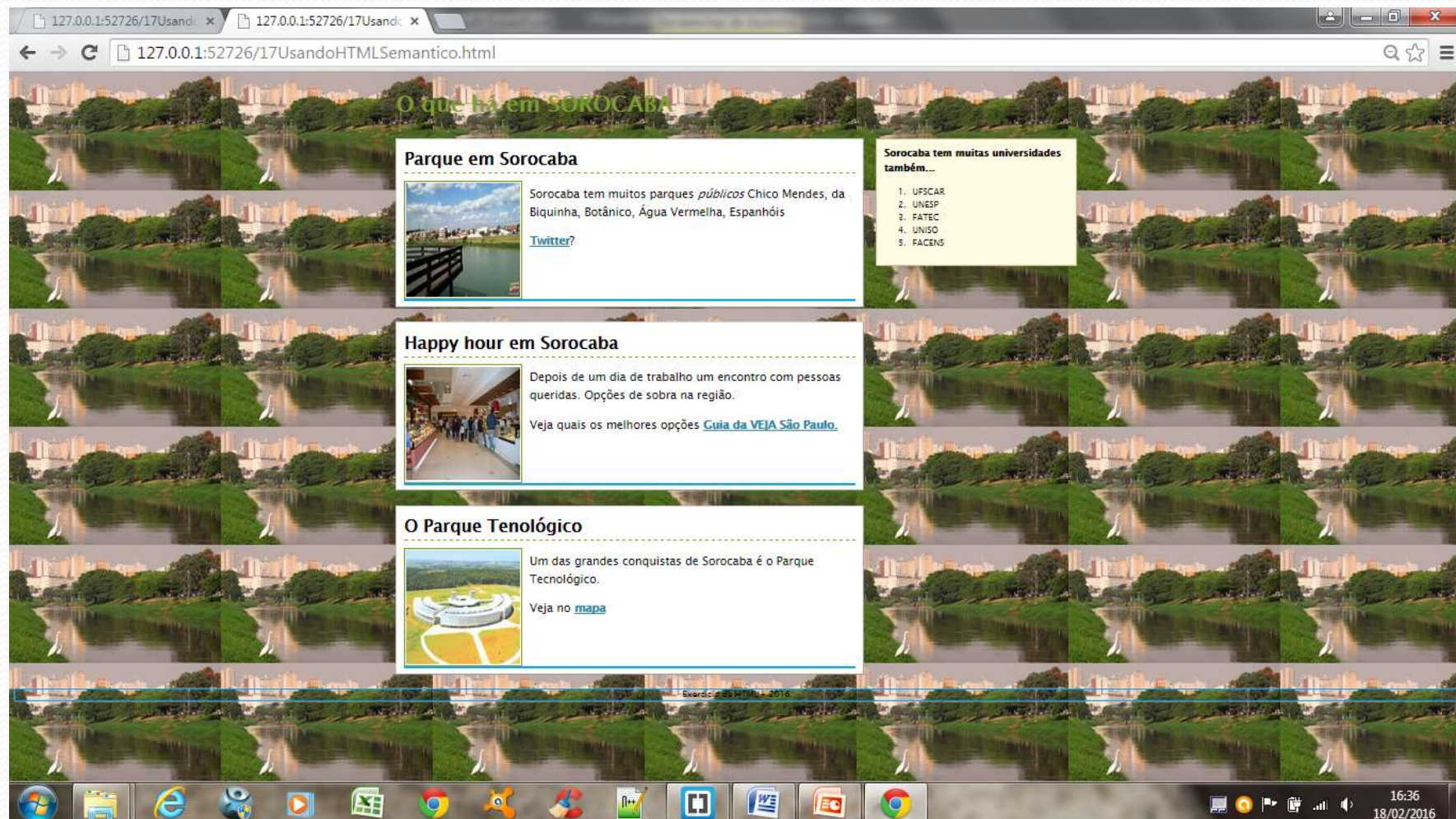
**\*\*** com o flexbox, ele fica flexível, responsivo.



# Atividade 4 - CSS – HTML Semântico

Criar uma página utilizando html semântico, disponibilizar no GitHub → seuusuario\PWEB\Atividade4

Exemplo:





# ATIVIDADE 5 – Site de Notícias

Criar um site de notícias utilizando flexbox, disponibilizar no GitHub

→ seuusuario\PWEB\Atividade5

→ Dá uma olhada nos sites:

<https://www.theguardian.com/international>

<https://www.terra.com.br/>



# Referências

ACESSIBILIDADELEGAL, Disponível em: <<http://www.acessibilidadelegal.com/23-webstandards.php>> Acesso em: Jan. 2015.

BOOTSTRAP. Disponível em: <http://www.tutorialwebdesign.com.br/o-que-e-bootstrap/> Acesso em: Fev.2016.

CAELUM. Apostilas Cursos Gratuitas Disponível em: <<https://www.caelum.com.br/apostilas/>> Acesso em: Jan. 2015.

CODEACADEMY. Cursos Gratuitos. Disponível em: <<https://www.codecademy.com/pt>> Acesso em: Jan. 2015.

CSS. Disponível em: <<https://css-tricks.com/>> Acesso em: Fev.2016.

CSS(2). CSS - CURSO W3C ESCRITÓRIO BRASIL Disponível em: <<http://www.w3c.br/pub/Cursos/CursoCSS3/css-web.pdf>> Acesso em: Ago.2016.

FONTSIZW. Disponível em: <<http://kyleschaeffer.com/development/css-font-size-em-vs-px-vs-pt-vs/>> Acesso em: Fev.2016

GRADIENT. Disponível em: <http://www.maujor.com/tutorial/css3-gradientes-lineares.php> Acesso em: Jun.2016

JOBSTRAIBIZER, Flávia. Criação de Sites com Css - Desenvolva páginas Web mais leves e dinâmicas em menos tempo. 1ª Ed. São Paulo: Digerati Books.,2009.

MARCOTTE, Ethan. Responsive Web Design. New York: A Book Apart, 2011.

MAZZA, Lucas. HTML5 e CSS3. Domine a Web do Futuro. Casa do Código, 2012.

# Referências

RESPONSIVO. Disponível em: <<http://wpmidia.com.br/desenvolvimento-web/design-responsivo-em-3-passos/comment-page-1/#comments>> Acesso em: Fev.2016

RESPONSIVO2. Disponível em: <<http://blog.popupdesign.com.br/design-responsivo-i-o-que-e-e-por-que-usar/>> Acesso em: Fev.2016

W3C. Site da W3C oficial. Disponível em: <<http://www.w3.org>> Acesso em: Mai.2015.

WILTON, Paul, McPeak, Jeremy. Beginning JavaScript. 3rd Edition. Indianapolis (EUA): Wiley Publishing, Inc., 2007.

WROBLEWSKI, Luke. *Mobile First*. New York: A Book Apart, 2011.

ZEMEL, Tércio. Web Design Responsivo: Páginas adaptáveis para todos os dispositivos. 1ª Ed. São Paulo: Casa do Código, 2012.