



TECNOLOGIA E INOVAÇÃO
EM PROL DA INDÚSTRIA



PROAJ – Desenvolvedor Web
com JAVA

Desenvolvimento de Sistemas

II – 180h

Prof^a: Francisleide Almeida

#What?

CSS em português foi traduzido para folha de estilo em cascata.

Definição mais precisa e simples para folha estílo encontra-se no site do W3C:

“ “ Folha de estílo em cascata é um mecanismo simples para adicionar estilos (p.ex., fontes, cores, espaçamentos) aos documentos Web. ”



FSdA1

Francisleide Silva de Almeida; 10/05/2017

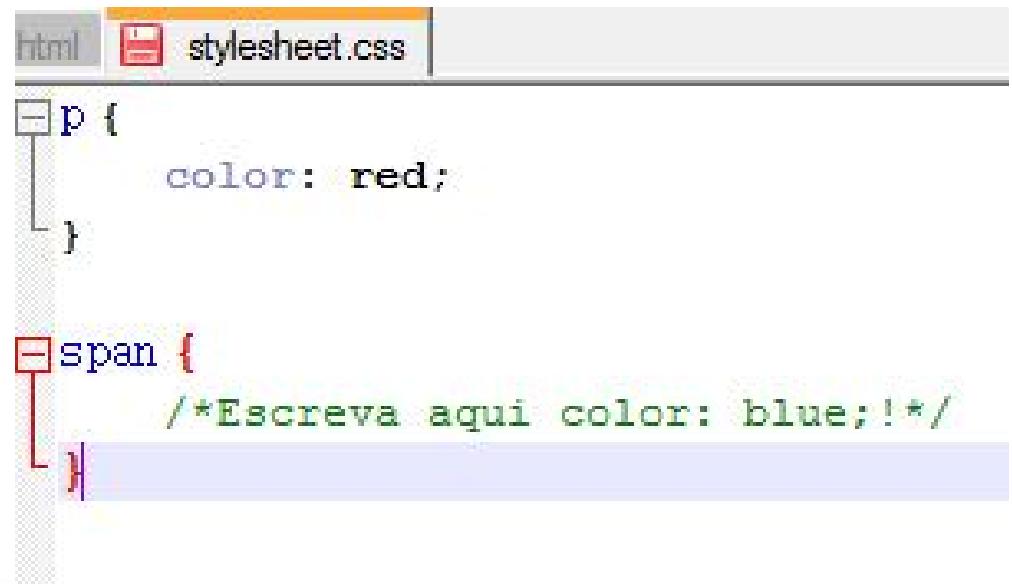
#NaPratica

Um arquivo “.css” contém todas as informações de estilo das CSS: onde elementos HTML devem estar, qual a cor que deveria ser, o quanto grande eles deveriam ser, e muito mais.



#PrimeiroPasso

```
<!DOCTYPE html>
<html>
    <head>
        <link type="text/css" rel="stylesheet" href="stylesheet.css"/>
        <title>Fancy Fonts</title>
    </head>
    <body>
        <p>
            Eu sou um parágrafo escrito em fonte vermelha,
            mas uma das minhas palavras é <span> azul</span>!
        </p>
    </body>
</html>
```



The code editor shows two tabs: 'html' and 'stylesheet.css'. The 'html' tab contains the provided HTML code. The 'stylesheet.css' tab contains the following CSS:

```
p {
    color: red;
}

span {
    /*Escreva aqui color: blue; !*/
}
```

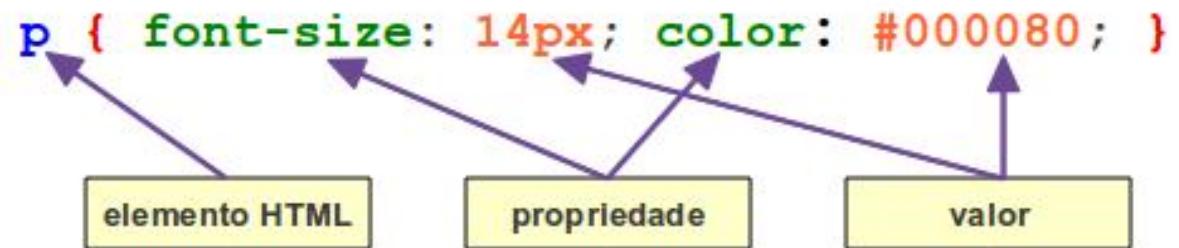
#Definições

- **Seletor:** é o alvo da regra CSS.
- **Declaração:** determinar os parâmetros de estilização. Compreende a propriedade e o valor.
- **Propriedade:** característica do seletor a ser estilizada.
- **Valor:** quantificação ou a qualificação da propriedade.

Regra CSS

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

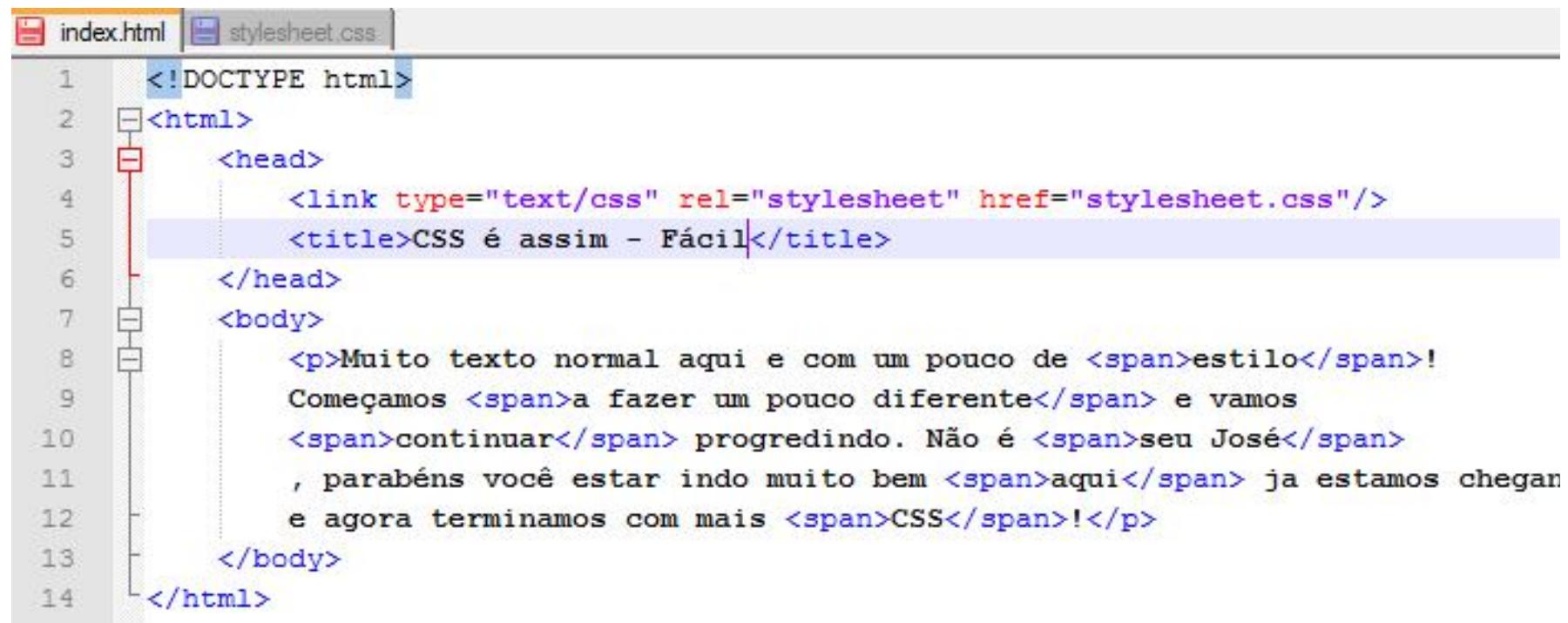
Declaração



#Separação do HTML e CSS

- Razões para separar HTML e CSS:
 1. Você pode aplicar a mesma formatação a vários elementos HTML, sem reescrever o código (por exemplo, style = "color: red" :)
 2. Você pode aplicar a aparência e formatação semelhante a várias páginas HTML a partir de um único arquivo CSS
 3. Melhora a manutenção do código

#Separação do HTML e CSS



The screenshot shows a code editor with two tabs: "index.html" and "stylesheet.css".

index.html:

```
1 <!DOCTYPE html>
2 <html>
3   <head>
4     <link type="text/css" rel="stylesheet" href="stylesheet.css"/>
5     <title>CSS é assim - Fácil</title>
6   </head>
7   <body>
8     <p>Muito texto normal aqui e com um pouco de <span>estilo</span>!
9       Começamos <span>a fazer um pouco diferente</span> e vamos
10      <span>continuar</span> progredindo. Não é <span>seu José</span>
11      , parabéns você estar indo muito bem <span>aqui</span> ja estamos chegando
12      e agora terminamos com mais <span>CSS</span>!</p>
13   </body>
14 </html>
```

stylesheet.css:

```
1 span {
2   font-family: cursive;
3 }
```



The screenshot shows a code editor with two tabs: "index.html" and "stylesheet.css".

index.html:

```
1 <!DOCTYPE html>
2 <html>
3   <head>
4     <link type="text/css" rel="stylesheet" href="stylesheet.css"/>
5     <title>CSS é assim - Fácil</title>
6   </head>
7   <body>
8     <p>Muito texto normal aqui e com um pouco de <span>estilo</span>!
9       Começamos <span>a fazer um pouco diferente</span> e vamos
10      <span>continuar</span> progredindo. Não é <span>seu José</span>
11      , parabéns você estar indo muito bem <span>aqui</span> ja estamos chegando
12      e agora terminamos com mais <span>CSS</span>!</p>
13   </body>
14 </html>
```

stylesheet.css:

```
1 span {
2   font-family: cursive;
3 }
```

#Separação do HTML e CSS

1. Estilos inline:

```
1 <p style="width: 200px; color: white; background: red; font-size:1.8em;">
2   <!-- Parágrafo com aplicação de estilos inline -->
3 </p>
```

2. Estilos incorporados:

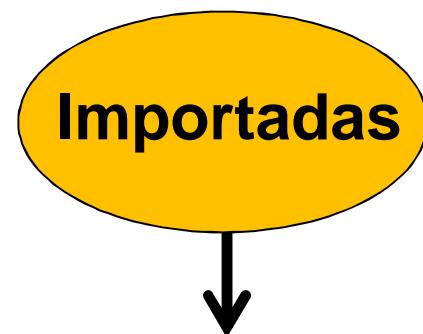
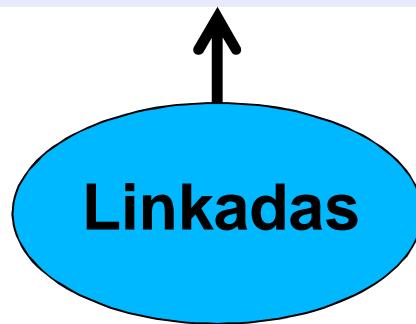
```
<head>
  <style type="text/css" media="all">
    body {
      margin: 0;
      padding: 0;
      font-size: 80%;
      color: black;
      background: white;
    }
  </style>
</head>
```

3. Estilos externos:

1. Linkadas
2. Importadas

#Separação do HTML e CSS

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



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

#NaPrática

Crie um arquivo html e um arquivo css. Crie a estrutura do html conforme abaixo. Escreva o estilo ao lado na CSS e siga os seguintes passos:

1. Faça todos os títulos h3 vermelho.
2. Defina todos os parágrafos do font-family: Courier.
3. O segundo parágrafo contém texto entre . Defina a cor de fundo do que de 'amarelo'.

```
p {  
    font-family: Arial;  
    color: blue;  
    font-size: 24px;  
}
```

```
<body>  
    <div>  
        <h3>What's CSS for?</h3>  
        <p>CSS is for styling HTML pages!</p>  
        <h3>Why use it?</h3>  
        <p>It makes webpages look <span>really rad</span>.</p>  
        <h3>What do I think of it?</h3>  
        <p>It's awesome!</p>  
    </div>  
</body>
```

#NaPrática

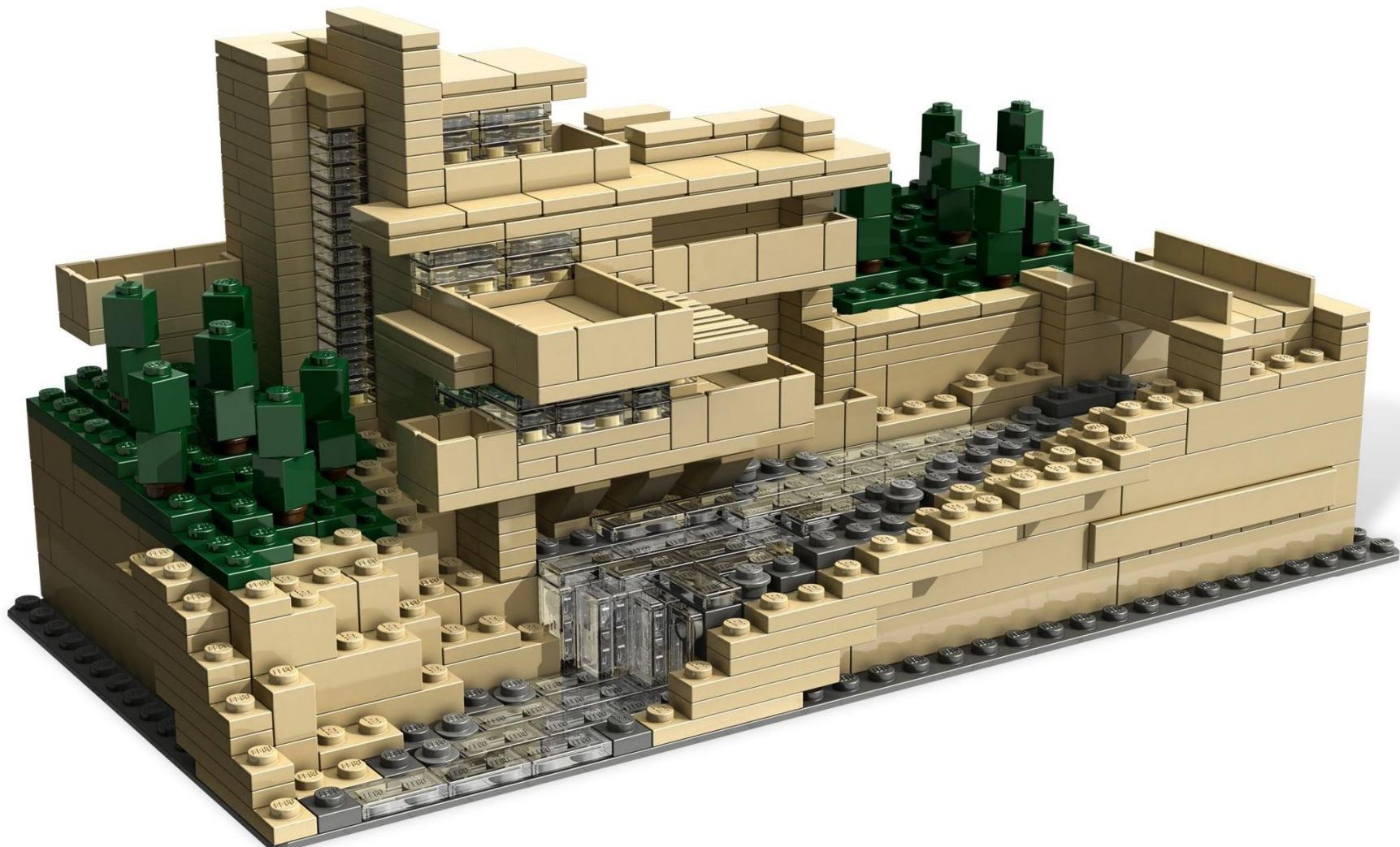
Crie um arquivo html e um arquivo css. Crie a estrutura básica do html insira uma div defina cor de fundo vermelho, largura e altura 100px
→

```
div{  
    background-color: #cc0000;  
    height:100px;  
    width: 100px;  
}
```

#UnidadesdeMedida

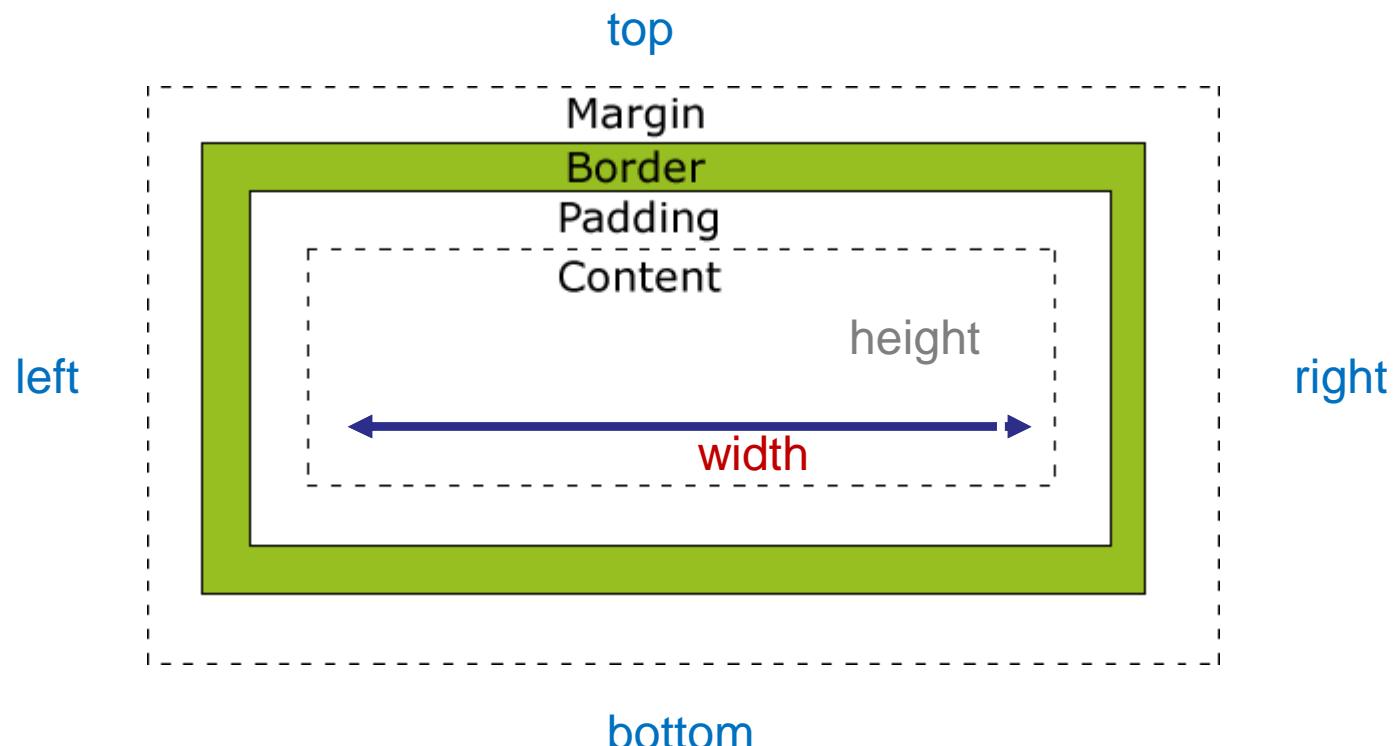
Unidade	Relativa
em	à font-size do elemento (ou do elemento pai)
px	a resolução do dispositivo em renderização
ex	ao valor x-height da fonte do elemento
%	a um outro valor anteriormente declarado

#BoxModel



#Box Model

- As CSS para montar o layout de um documento, consideram e tratam todos os elementos HTML como se fossem caixas (Box) a serem exibidas em uma mídia visual (a tela do monitor ou uma folha de papel impressa por exemplo).



```
08. <title>Box Model</title>
09. </head>
10. <body>
11.
12. <h1>Folhas de estilo em cascata</h1>
13.
14. <h3>A importância do Box Model</h3>
15.
16. <p>
17. Para projetar um layout CSS, seja para uma página
18. web ou um site inteiro é fundamental que o
19. desenvolvedor entenda com detalhes como as folhas
20. de estilo em cascata tratam os elementos HTML< para fi
21. de apresentação em uma mídia do usuário
22. por exemplo: a tela de um monitor).
23. </p>
24.
25. <p>
26. Todo elemento HTML é uma "caixa" retangular a ser
27. apresentada na tela com as estilizações determinadas
28. pelas regras CSS. As caixas são empilhadas uma após a
29. outra e constituídas de margens, bordas, espaçamentos
30. e o conteúdo propriamente dito.
31. </p>
32.
33. </body>
34. </html>
```

Folhas de estilo em cascata

A importância do Box Model

Para projetar um layout CSS, seja para uma página web ou um site inteiro é fundamental que o desenvolvedor entenda com detalhes como as folhas de estilo em cascata tratam os elementos HTML para fins de apresentação em uma mídia do usuário (por exemplo: a tela de um monitor).

Todo elemento HTML é uma "caixa" retangular a ser apresentada na tela com as estilizações determinadas pelas regras CSS. As caixas são empilhadas uma após a outra e constituídas de margens, bordas, espaçamentos e o conteúdo propriamente dito.

E ai, cadê os Boxes?

#Box Model

```

01. h1 {
02.   background-color: #cc9;
03.   border: 10px solid #f00;
04.   padding: 5px;
05. }
06.
07. h3 {
08.   background-color: #fc9;
09.   border: 20px solid #039;
10.   padding: 15px;
11. }
12.
13. p.um {
14.   background-color: #ff9;
15.   border: 5px solid #f0f;
16.   padding: 5px;
17.   text-align: justify;
18. }
19.
20. p.dois {
21.   background-color: #cff;
22.   border: 2px solid #039;
23.   padding: 10px;
24.   text-align: justify;
25. }
26.
27. Nota: As classes .um e .dois foram aplicadas
28. nos primeiro e segundo parágrafos

```

Folhas de estilo em cascata



Para projetar um layout CSS, seja para uma página web ou um site inteiro é fundamental que o desenvolvedor entenda com detalhes como as folhas de estilo em cascata tratam os elementos HTML para fins de apresentação em uma mídia do usuário (por exemplo: a tela de um monitor).

Todo elemento HTML é uma "caixa" retangular a ser apresentada na tela com as estilizações determinadas pelas regras CSS. As caixas são empilhadas uma após a outra e constituídas de margens, bordas, espaçamentos e o conteúdo propriamente dito.

Agora sim! Vejo as boxes.

#Box Model

Propriedade **display**

Bloco: faz com que o elemento HTML seja renderizado como bloco, tal como os parágrafos e os cabeçalhos o são. Um bloco contém um espaço em branco tanto em cima como embaixo e não permite outros elementos HTML ao lado, exceto quando tiver sido declarado ao contrário (por exemplo, declarar a propriedade `float` para o elemento próximo ao bloco)..

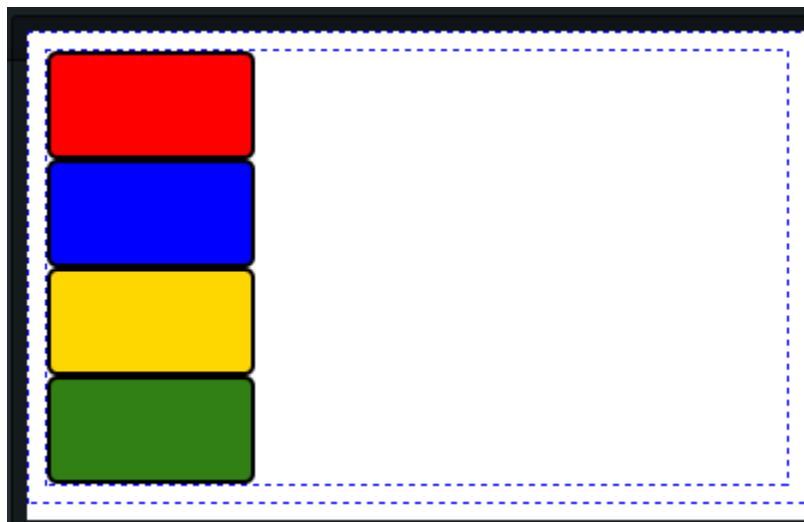
inline-block: bloco inline é colocado inline (ou seja, na mesma linha do conteúdo adjacente), mas comporta-se como se fosse um bloco.

Inline: faz com que o elemento HTML seja renderizado inline, dentro do bloco na mesma linha. Quando o elemento encontra-se entre dois blocos ele forma o chamado 'bloco anônimo' e é renderizado com a menor largura possível.

Nenhum(none): Isso faz com que o elemento e seu conteúdo desaparecer da página inteiramente!

#Display [block]

```
<body>
    <div id="one"></div>
    <div id="two"></div>
    <div id="three"></div>
    <div id="four"></div>
</body>
```



```
1  * {
2      border: 1px dashed blue;
3  }
4
5  div {
6      height: 50px;
7      width: 100px;
8      border: 2px solid black;
9      border-radius: 5px;
10     display: block;
11  }
12
13
14  #one {
15      background-color: #FF0000;
16  }
17
18  #two {
19      background-color: #0000FF;
20  }
21
22  #three {
23      background-color: #FFD700;
24  }
25
26  #four {
27      background-color: #308014;
28  }
```

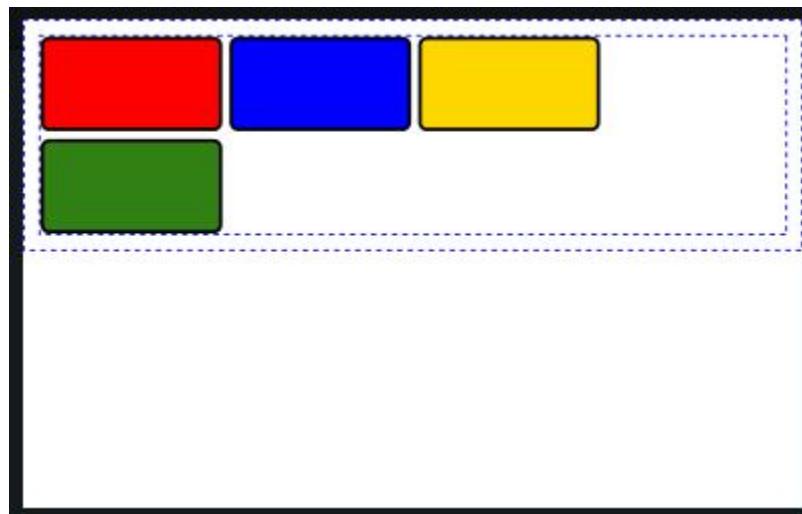
#Border

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

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

Display [inline-block]

```
<body>
    <div id="one"></div>
    <div id="two"></div>
    <div id="three"></div>
    <div id="four"></div>
</body>
```



```
* {
    border: 1px dashed blue;
}

div {
    height: 50px;
    width: 100px;
    border: 2px solid black;
    border-radius: 5px;
    display: inline-block;
}

#one {
    background-color: #FF0000;
}

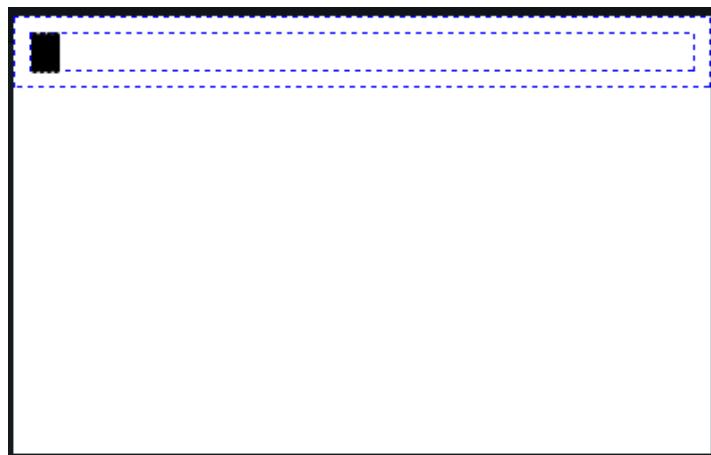
#two {
    background-color: #0000FF;
}

#three {
    background-color: #FFD700;
}

#four {
    background-color: #308014;
}
```

Display [inline]

```
<body>
    <div id="one"></div>
    <div id="two"></div>
    <div id="three"></div>
    <div id="four"></div>
</body>
```



Diferente do inline-block o inline mantêm um ao lado do outro, porem eles não mantêm suas dimensões.

```
* {
    border: 1px dashed blue;
}

div {
    height: 50px;
    width: 100px;
    border: 2px solid black;
    border-radius: 5px;
    display: inline;
}

#one {
    background-color: #FF0000;
}

#two {
    background-color: #0000FF;
}

#three {
    background-color: #ffd700;
}

#four {
    background-color: #308014;
}
```

#Border

```
<body>
    <div></div>
</body>
```



```
index.html stylesheet.css
1 * {
2     border: 1px dashed black;
3 }
4
5 div {
6     height: 50px;
7     width: 100px;
8     border: 4px solid #FF0000;
9     border-radius: 5px;
10    background-color: #308014;
11    margin: 20px 50px 10px 5px;
12 }
```

Display [none]

```
<body>
  <div id="one"></div>
  <div id="two"></div>
  <div id="three"></div>
  <div id="four"></div>
</body>
```



```
* {
  border: 1px dashed blue;
}

div {
  height: 50px;
  width: 100px;
  border: 2px solid black;
  border-radius: 5px;
  display: none;
}

#one {
  background-color: #FF0000;
}

#two {
  background-color: #0000FF;
}

#three {
  background-color: #FFD700;
}

#four {
  background-color: #308014;
}
```

#Box Model

- Existe um grande problema no box model que é a propriedade *width*. Quando você define a largura de um elemento, este elemento pode parecer maior do que o definido: as bordas - **border** - e o preenchimento - **padding** - vão esticar o elemento além da largura definida.
- Logo, criaram uma propriedade do CSS chamada **box-sizing**. Quando você define **box-sizing: border-box;** em um elemento, os valores do **padding** e do **border** não são adicionados em sua largura.

#Margens

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

As propriedades para margens são as listadas abaixo:

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

Valores válidos para a propriedade margin

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

#Margens [auto]

```
<body>
  <div></div>
</body>
```



```
index.html stylesheet.css
1  * {
2    border: 1px dashed black;
3  }
4
5  div {
6    height: 50px;
7    width: 100px;
8    border: 2px solid black;
9    border-radius: 5px;
10   background-color: #308014;
11   margin: auto;
12 }
```

Definir a largura (*width*) de um elemento de bloco previne que ele se estique para as pontas do elemento onde está contido tanto pra esquerda quanto pra a direita. Logo, você pode modificar as margens à esquerda e à direita em automático - *auto* - para centralizar o elemento horizontalmente aonde está contido

#Border

Valores validos para a propriedade borda

- **color:**

1. código hexadecimal: #FFFFFF
2. código rgb: rgb(255,235,0)
3. nome da cor: red, blue, green...etc

- **style:**

1. none: nenhuma borda
2. hidden: equivalente a none
3. dotted: borda pontilhada
4. dashed: borda tracejada
5. solid: borda contínua
6. double: borda dupla
7. groove: borda entalhada
8. ridge: borda em ressalto
9. inset: borda em baixo relevo
10. outset: borda em alto relevo

- **width:**

1. thin: borda fina
2. medium: borda média
3. thick: borda grossa
4. length: uma medida reconhecida pelas CSS (px, pt, em, cm, ...)

#Padding

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

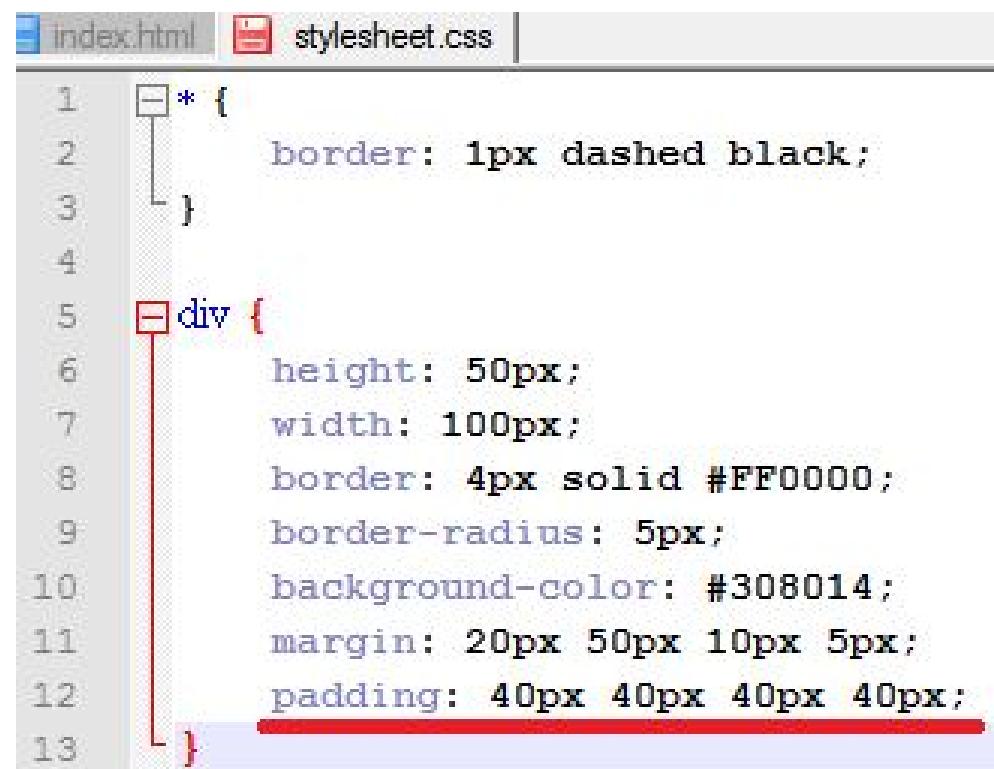
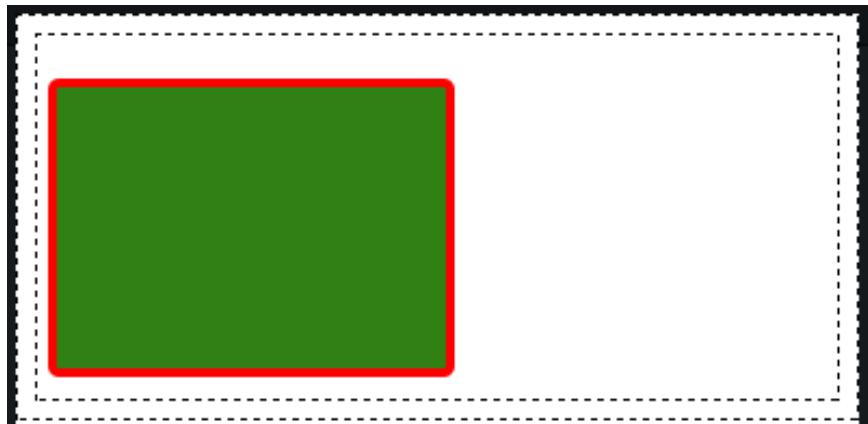
- padding-top.....define a espaçamento superior;
- padding-right.....define a espaçamento direita;
- padding-bottom.....define a espaçamento inferior;
- padding-left.....define a espaçamento esquerda;
- padding.....**maneira abreviada para todas os espaçamentos**

Valores validos:

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

#Padding

```
<body>
    <div></div>
</body>
```



The image shows a code editor with two tabs: "index.html" and "stylesheet.css".

index.html:

```
<body>
    <div></div>
</body>
```

stylesheet.css:

```
1  * {
2      border: 1px dashed black;
3  }
4
5  div {
6      height: 50px;
7      width: 100px;
8      border: 4px solid #FF0000;
9      border-radius: 5px;
10     background-color: #308014;
11     margin: 20px 50px 10px 5px;
12     padding: 40px 40px 40px 40px;
13 }
```

A red vertical line highlights the "padding: 40px 40px 40px 40px;" declaration in the CSS file, specifically the bottom-right value.

#Na Prática

Crie um <div> para você usar na guia HTML. Na guia CSS:

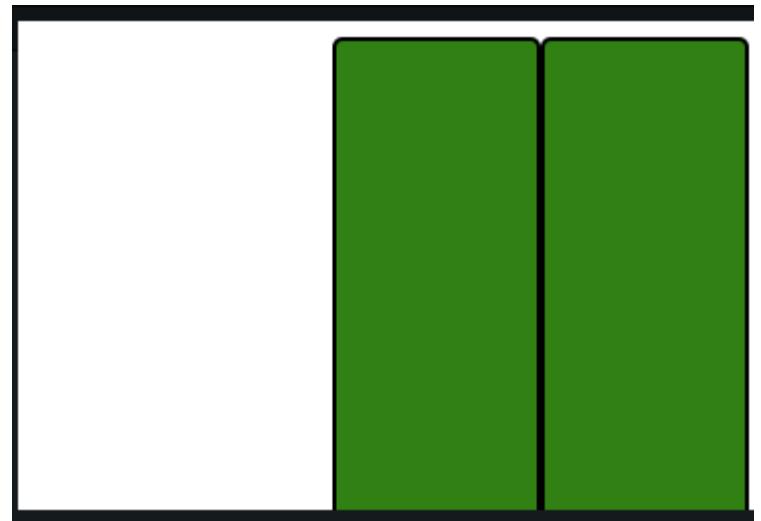
1. Dê a essa div uma borda preta de 1px solid.
2. Dê-lhe uma cor de fundo # CC0000.
3. Defina sua margem superior para 10px, a sua margem direita para 5px, a sua margem inferior para 5px, e sua margem esquerda para 50px.
4. Defina o seu espaçamento top para 0px, seu espaçamento direito de 30px, o seu espaçamento de fundo para 0px, e seu espaçamento à esquerda de 10px

#Float

› float serve para posicionar o elemento dentro do layout

```
<body>
    <div></div>
    <div></div>
</body>
```

```
1  div {
2      height: 300px;
3      width: 100px;
4      border: 2px solid black;
5      border-radius: 5px;
6      background-color: #308014;
7      float: right;
8
9  }
```



#Float

```
<body>
    <div></div>
    <p>Confira! Eu sou um bloco de texto, mas o &lt;div&gt; e eu posso viver em harmonia. Barcos em um mar, o homem. Barcos em um mar.</p>
</body>
```

```
div {
    height: 300px;
    width: 60px;
    border: 2px solid black;
    border-radius: 5px;
    background-color: #308014;
    float: right;
}

p {
    font-family: Verdana, sans-serif;
    font-size: 20px;
    width: 280px;
    float:left;
}
```

Confira! Eu sou um bloco de texto, mas o <div> e eu posso viver em harmonia. Barcos em um mar, o homem. Barcos em um mar.

```
div {  
    border-radius: 5px;  
}  
  
#header {  
    height: 50px;  
    background-color: #F38630;  
    margin-bottom: 10px;  
}  
  
.left {  
    height: 300px;  
    width: 32%;  
    background-color: #A7DBD8;  
    float: left;  
    margin-bottom: 10px;  
}  
  
.center {  
    height: 300px;  
    width: 25%;  
    background-color: red;  
    margin-bottom: 10px;  
    margin-left: 10px;  
    float: right;  
}
```

```
.right {  
    height: 300px;  
    width: 40%;  
    background-color: #E0E4CC;  
    float: right;  
    margin-bottom: 10px;  
}  
  
#footer {  
    height: 50px;  
    background-color: #69D2E7;  
    clear: both;  
}
```

```
<body>  
    <div id="header"></div>  
    <div class="left"></div>  
    <div class="center"></div>  
    <div class="right"></div>  
    <div id="footer"></div>  
</body>
```





```
.right {  
    height: 300px;  
    width: 40%;  
    background-color: #E0E4CC;  
    float: right;  
    margin-bottom: 10px;  
}  
  
#footer {  
    height: 50px;  
    background-color: #69D2E7;  
    clear: both;  
}
```



Se você disser a ele para limpar: ambos, ele vai ficar fora do caminho dos elementos flutuantes à esquerda e à direita!

```
<!DOCTYPE html>
<html>
<head>
<meta name="description" content="Macete clearfix">
<meta charset="utf-8">
<title>Macete clearfix</title>
<style>
div{
    border:2px solid;
}
img{
    float:right;
}
.clearfix {
    overflow: auto;
}
</style>
</head>
<body>
| <div class="">
<p>
    Ops... esta imagem é maior do que o elemento que a contém, e é
    por isso que ela
    estoura os limites dele.!
</p>

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

Ops... esta imagem é maior do que o elemento que a contém, e é por isso que ela estoura os limites dele.!



Macete

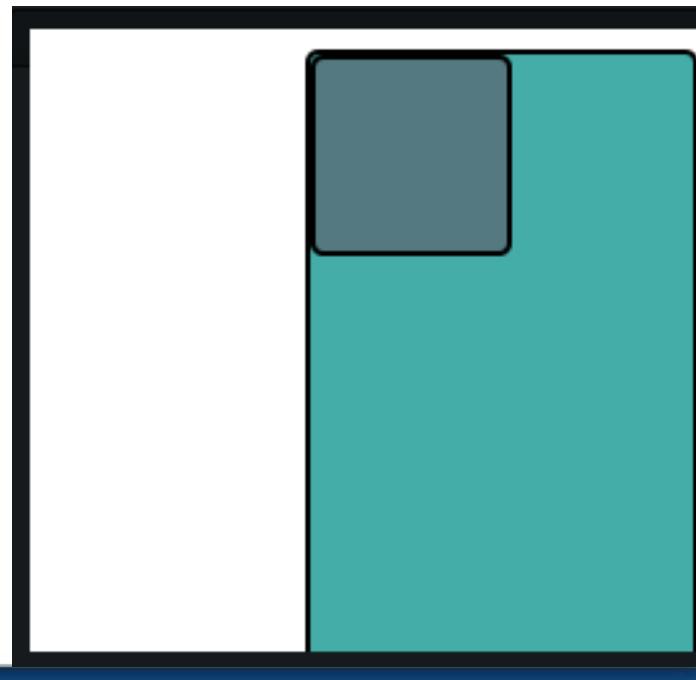


#Position

A propriedade CSS que possibilita posicionar um elemento qualquer é a propriedade **position**. Sua propriedade padrão é o **static**

```
div {  
    border-radius: 5px;  
    border: 2px solid black;  
}  
  
#inner {  
    height: 75px;  
    width: 75px;  
    background-color: #547980;  
}  
  
#outer {  
    height: 1500px;  
    width: 150px;  
    background-color: #45ADA8;  
    position: absolute;  
    margin-left: 100px;  
}
```

```
<body>  
    <div id="outer">  
        <div id="inner"></div>  
    </div>  
</body>
```

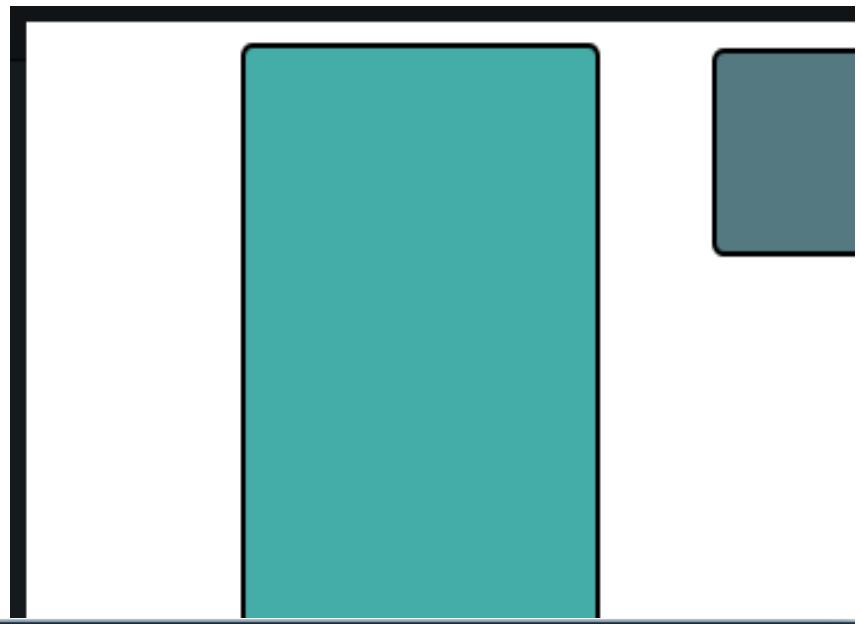


#Position [relative]

relative se comporta igualmente ao **static**, a menos que se adicione propriedades extras no estilo do elemento (top, right, bottom e left). O ponto zero será sempre o canto superior esquerdo.

```
div {  
    border-radius: 5px;  
    border: 2px solid black;  
}  
  
#inner {  
    height: 75px;  
    width: 75px;  
    background-color: #547980;  
    position: relative;  
    margin-left: 200px;  
}  
  
#outer {  
    height: 1500px;  
    width: 150px;  
    background-color: #45ADA8;  
    position: absolute;  
    margin-left: 100px;  
}
```

```
<body>  
    <div id="outer">  
        <div id="inner"></div>  
    </div>  
</body>
```

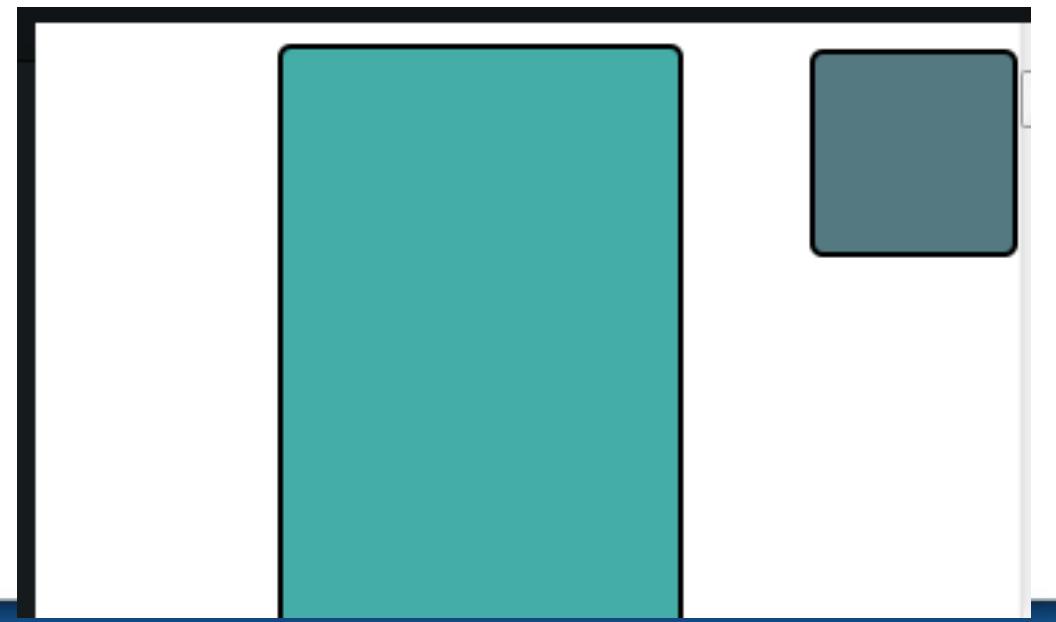


#Position [fixed]

- Um elemento fixo - **fixed** - é posicionado relativamente ao "viewport", isso significa que ele sempre ficará no mesmo lugar mesmo que haja rolagem na página. Assim como o **relative**, as propriedades **top**, **right**, **bottom** e **left** também são utilizadas.

```
div {  
    border-radius: 5px;  
    border: 2px solid black;  
}  
  
#inner {  
    height: 75px;  
    width: 75px;  
    background-color: #547980;  
    position: fixed;  
    margin-left: 200px;  
}  
  
#outer {  
    height: 1500px;  
    width: 150px;  
    background-color: #45ADA8;  
    position: absolute;  
    margin-left: 100px;  
}
```

```
<body>  
    <div id="outer">  
        <div id="inner"></div>  
    </div>  
</body>
```

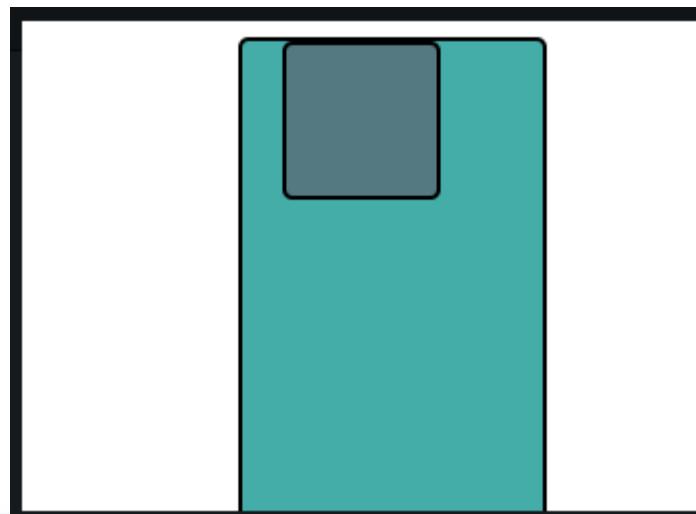


#Position [absolute]

- Este valor se comporta como o **fixed**, porém tendo como referência a posição do elemento relativo mais próximo de onde está contido. Se não possuir elementos ancestrais posicionados relativamente, ele utilizará o **body** como referência.

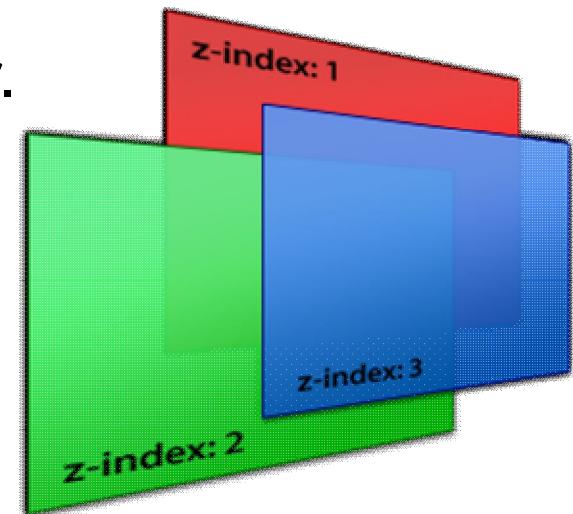
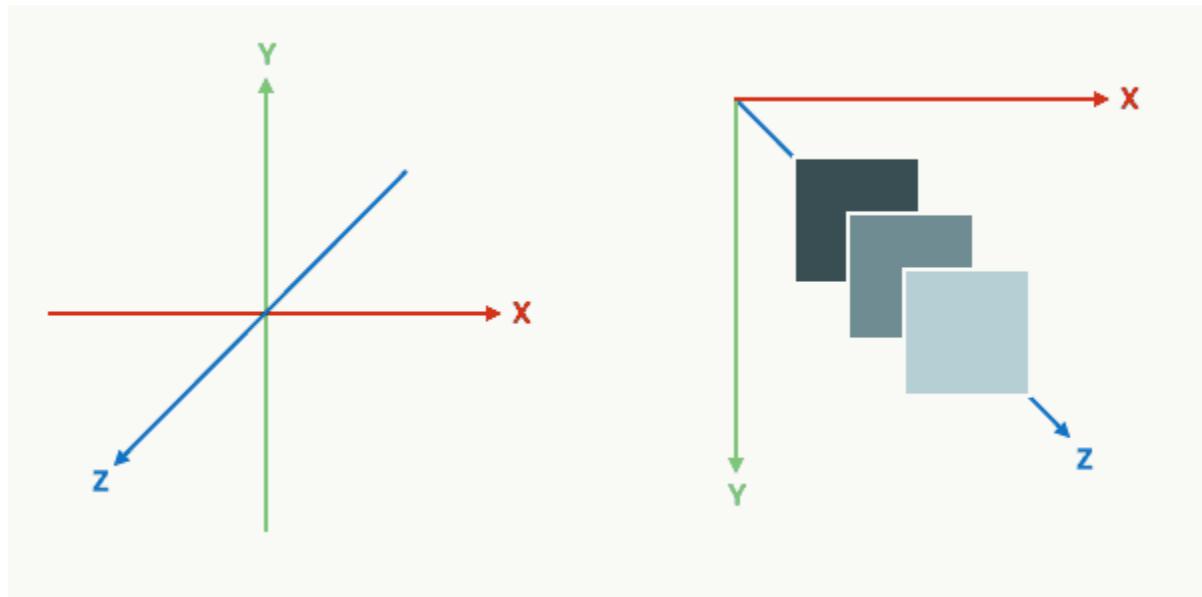
```
div {  
    border-radius: 5px;  
    border: 2px solid black;  
}  
  
#inner {  
    height: 75px;  
    width: 75px;  
    background-color: #547980;  
    position: absolute;  
    margin-left: 20px;  
}  
  
#outer {  
    height: 1500px;  
    width: 150px;  
    background-color: #45ADA8;  
    position: absolute;  
    margin-left: 100px;  
}
```

```
<body>  
    <div id="outer">  
        <div id="inner"></div>  
    </div>  
</body>
```



#Z-Index

A propriedade CSS z-index rege o empilhamento de boxes segundo o **eixo z**.
O eixo Z é perpendicular à tela do monitor.



A propriedade CSS **z-index** determina qual box fica à frente quando dois ou mais boxes ocupam a mesma posição ou se sobrepõe parcialmente.

#Z-Index

```
div.um {  
    width: 200px;  
    height: 80px;  
    background: #cff;  
    z-index: 3;  
    opacity: 0.8; /* CSS3 */  
    /* regras proprietarias para opacidade - para fins de clareza do exemplo */  
    -moz-opacity: 0.8;  
    filter: alpha(opacity=80);  
}
```

```
div.dois {  
    width: 110px;  
    height: 120px;  
    left: 120px;  
    top: 20px;  
    background: #6f9;  
    z-index: 2;  
    opacity: 0.8; /* CSS3 */  
    /* regras proprietarias para opacidade - para fins de clareza do exemplo */  
    -moz-opacity: 0.8;  
    filter: alpha(opacity=80);  
}
```

```
div.tres {  
    width: 90px;  
    height: 60px;  
    left: 70px;  
    top: 50px;  
    background: #fcf;  
    z-index: 1;  
}
```

Todo o elemento posicionado com o valor *relative*, *absolute* ou *fixed* cria um contexto de posicionamento, estabelecendo uma origem para contagem das coordenadas da posição horizontal e vertical.

De modo similar ocorre com z.

Criando um padrão para demonstrar funcionando

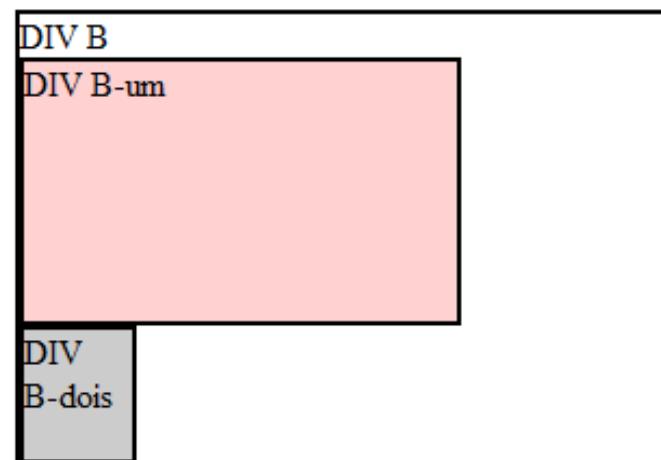
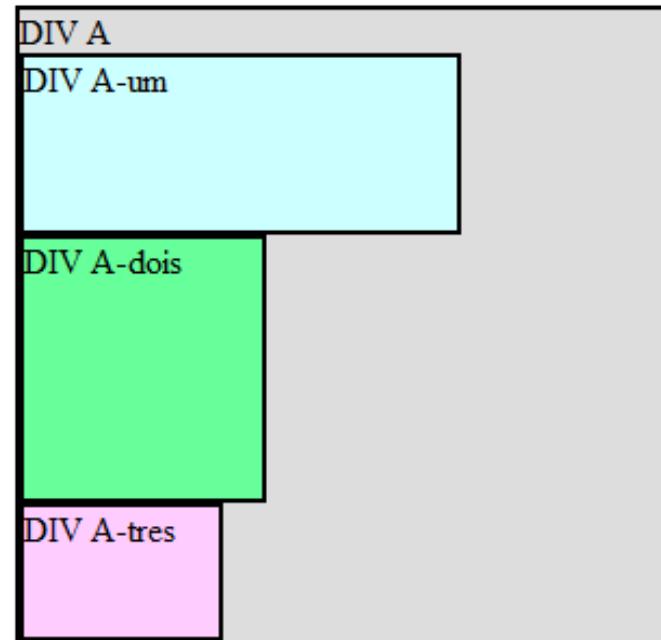
```
1   <!DOCTYPE html>
2   <html lang="pt-br">
3   <head>
4   <meta charset="utf-8" />
5   <head>
47  <body>
48  <div id="a">
49    DIV A
50    <div class="a-um">DIV A-um</div>
51    <div class="a-dois">DIV A-dois</div>
52    <div class="a-tres">DIV A-tres</div>
53  </div>
54
55  <div id="b">
56    DIV B
57    <div class="b-um">DIV B-um</div>
58    <div class="b-dois">DIV B-dois</div>
59  </div>
60 </body>
61 </html>
```

#Z-Index – Na Prática

Criando css para os componentes html criados

```
8  div {  
9      border: 2px solid #000;  
10     }  
11     div#a, div#b {  
12         width:300px;  
13         margin-bottom:20px;  
14     }  
15     div#a {  
16         background: #ddd;  
17     }  
  
35    div.b-um {  
36        width: 200px;  
37        height: 120px;  
38        background: #ffd1d1;  
39    }  
40    div.b-dois {  
41        width: 50px;  
42        height: 60px;  
43        background: #ccc;  
44    }  
  
18    div.a-um {  
19        width: 200px;  
20        height: 80px;  
21        background: #cff;  
22    }  
23    div.a-dois {  
24        top:70px;  
25        width: 110px;  
26        height: 120px;  
27        background: #6f9;  
28    }  
29    div.a-tres {  
30        top:90px;  
31        width: 90px;  
32        height: 60px;  
33        background: #fcf;  
34    }
```

O resultado foi o comportamento Esperado, com o padrão de empilhamento

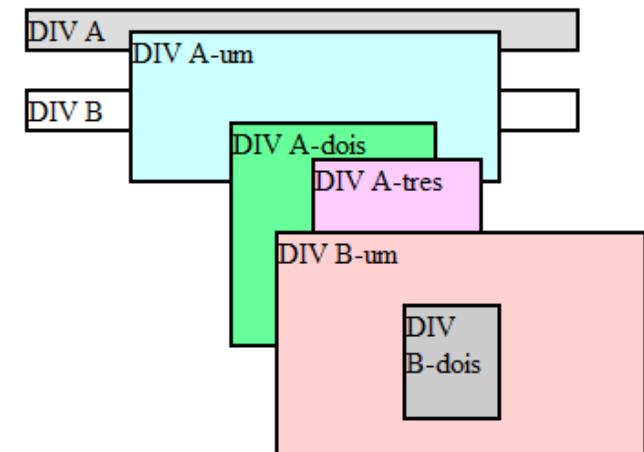


Inserindo posicionamento absolute para as divs

```
18  div.a-um {  
19      position: absolute;  
20      left: 65px;  
21      top: 20px;  
22      width: 200px;  
23      height: 80px;  
24      background: #cff;  
25  }  
26  div.a-dois {  
27      position: absolute;  
28      left: 120px;  
29      top: 70px;  
30      width: 110px;  
31      height: 120px;  
32      background: #6f9;  
33  }  
34  div.a-tres {  
35      position: absolute;  
36      left: 165px;  
37      top: 90px;  
38      width: 90px;  
39      height: 60px;  
40      background: #fcf;  
41  }  
42  div.b-um {  
43      position: absolute;  
44      left: 145px;  
45      top: 130px;  
46      width: 200px;  
47      height: 120px;  
48      background: #ffd1d1;  
49  }  
50  div.b-dois {  
51      position: absolute;  
52      left: 215px;  
53      top: 170px;  
54      width: 50px;  
55      height: 60px;  
56      background: #ccc;  
57  }
```

`div#a` e `div#b` encolheram até uma altura suficiente para conter apenas seus nomes; isso porque seus elementos-filhos foram posicionados de forma absoluta, liberando o espaço que ocupavam no fluxo do documento e esvaziando-o.

A ordem de empilhamento obedece a ordem em que os elementos-filhos posicionados aparecem na marcação; isso porque nenhum elemento filho tem elemento ancestral posicionado.



Em um main com padrão de 1000px de width, crie um layout como o desenho abaixo usando os princípios ensinados na sala

Topo do site

- Menu 01
- Menu 02
- Menu 03

Conteúdo1

Conteúdo2

Rodape