

# FIAP

## GRADUAÇÃO

45697056



TDS

# FRONT-END DESIGN ENGINEERING

Prof. Alexandre Carlos [profalexandre.jesus@fiap.com.br](mailto:profalexandre.jesus@fiap.com.br)

Prof. Luís Carlos [lsilva@fiap.com.br](mailto:lsilva@fiap.com.br)



45697056



# INTRODUÇÃO CSS3

## *CASCADING STYLE SHEETS*





# Introdução CSS3

45697056



É uma linguagem de estilos utilizada para definir a apresentação de documentos escritos em HTML.





# Introdução CSS3

45697056



Linguagem criada pelo W3C para definir estilos (cores, tipologia, posicionamento, etc.) em páginas web. Possibilita que determinadas propriedades sejam aplicadas ao mesmo tempo a todos elementos de uma página ou site que estejam marcados com uma tag específica. Facilitam a formatação e manutenção de páginas web.

```

<div class="container">
  <div class="row">
    <div class="col-md-6 col-lg-8"> <!-- BEGIN NAVIGATION
      <nav id="nav" role="navigation">
        <ul>
          <li><a href="index.html">Home</a></li>
          <li><a href="home-events.html">Home Events</a></li>
          <li><a href="multi-col-menu.html">Multiple Column Menu</a></li>
          <li class="has-children"> <a href="#" class="current">
            <ul>
              <li><a href="tall-button-header.html">Tall But
              <li><a href="image-logo.html">Image Logo</a></li>
              <li class="active"><a href="tall-logo.html">Ta
            </ul>
          </li>
          <li class="has-children"> <a href="#">Carousels</a>
            <ul>
              <li><a href="variable-width-slider.html">Variab
              <li><a href="variable-width-slider.html">Testimoni
    </div>
  </div>
</div>

```



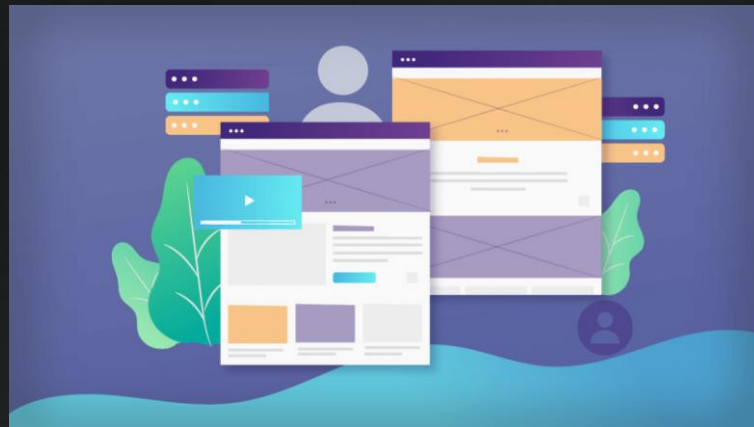


# Introdução CSS3

45697056



Criado no final de 1996, é um padrão definido pelo W3C e um dos principais conceitos relacionados aos Web Standards. A linguagem HTML estrutura um documento através de blocos de informação. A CSS controla a aparência e o layout do documento. É uma solução eficiente para administração de sites com alto volume de páginas.





# Introdução CSS3 - Declaração

45697056



IL

## CSS Inline

As regras de formatação são inseridas diretamente na tag e se aplicam apenas ao elemento marcado.

Não use !!!

IN

## CSS Interno

As regras de formatação são inseridas na seção <head> da página e se aplicam ao documento inteiro.

Use com moderação !!!



EX

## CSS Externo

As regras de formatação são inseridas em um documento separado do HTML e podem ser usadas por várias páginas.

Use à vontade !!!



# Introdução CSS3 - Declaração

45697056



## PRECEDÊNCIA

Pode-se usar um dos três, ou até mesmo os três, tipos de declaração CSS no mesmo documento, mas para isso temos de tomar cuidado com a precedência. Por isso, guarde a seguinte regra:

Inline > Interno > Externo







# Introdução CSS3 - Declaração

45697056



## DECLARAÇÃO INLINE

Com esse tipo de declaração você mistura a formatação de um elemento com o seu respectivo conteúdo. Caso você queira que a mesma regra seja usada por outro elemento, terá de refazer a mesma declaração no elemento desejado. É terrível dar manutenção nesse tipo de código. **PÉSSIMA PRÁTICA**

```
<p style="background-color:  #900;">Texto do Parágrafo</p>
```






# Introdução CSS3 - Declaração

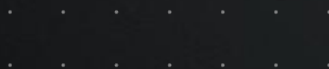
45697056



## DECLARAÇÃO INTERNA

Declaração feita na seção `<head>` da página. Todos os elementos da página que forem declarados na regra serão formatados. A regra é aplicada apenas naquela página que contém a respectiva formatação.

```
<STYLE>
  p{
    background-color:  #900;
  }
</STYLE>
```





# Introdução CSS3 - Declaração

45697056



## DECLARAÇÃO EXTERNA

A linha acima deve ser inserida na seção `<head>` de suas páginas, sua função é indicar onde está o arquivo que contém as regras CSS que devem ser usadas na estilização dos elementos da página. Uma mesma regra CSS dentro de um arquivo externo, pode formatar simultaneamente quantas páginas o desenvolvedor quiser.

Esse é o padrão que deve ser usado.

```
<link rel="stylesheet" href="css/style.css">
```





# Introdução CSS3 - Declaração

45697056



## EFEITO CASCATA

O efeito cascata define qual regra será aplicada quando há mais de um estilo especificado para o mesmo elemento HTML. Neste caso ele seguirá o padrão abaixo:

- 1 - Estilo padrão do navegador.
- 2 - Folha de Estilo Externa (referenciada e/ou importada).
- 3 - Folha de Estilo Interna (definida na área de cabeçalho do documento).
- 4 - Folha de Estilo Inline (dentro de um elemento HTML).

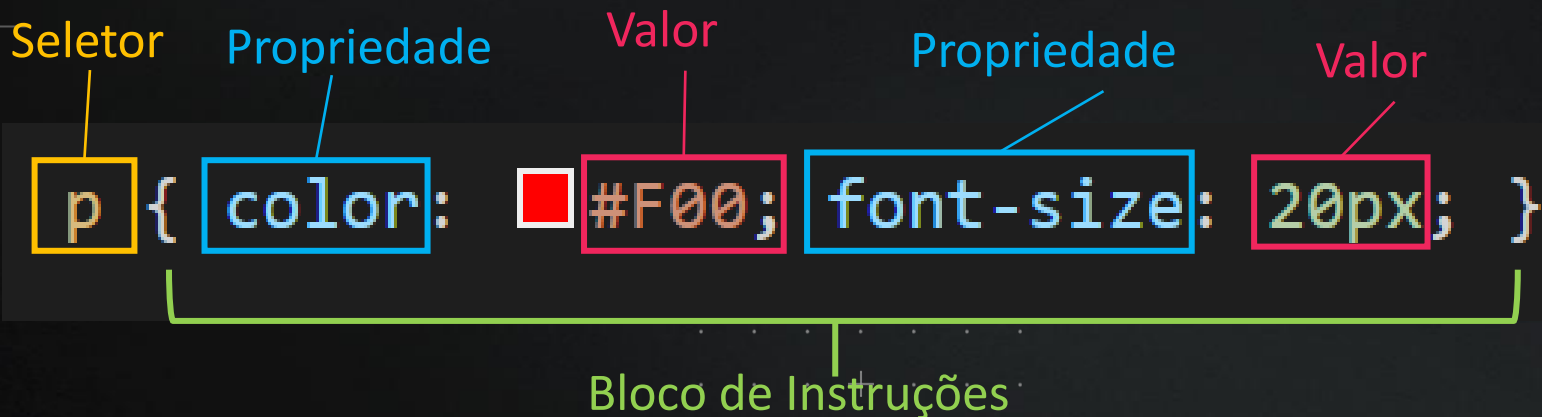


# Introdução CSS3 – Regras CSS

45697056

■ ■ ■

Uma regra CSS é uma declaração com sintaxe própria que indica como será o estilo de um ou mais elementos HTML. Um conjunto de regras CSS formam uma Folha de Estilos. É composta por: um seletor, uma propriedade e um valor.





# Introdução CSS3 - Seletores

45697056



Existem vários tipos de seletores CSS, os padrões são os seletores para tags, ids e classes.

**Seletor para TAG:** O seletor de tag é o mais simples de todos, basta informar o nome da tag, neste caso, todas as tags <p> existentes no documento receberão a formatação indicada.

Seletor

```
p { color: #F00; font-size: 20px; }
```



# Introdução CSS3 - Seletores

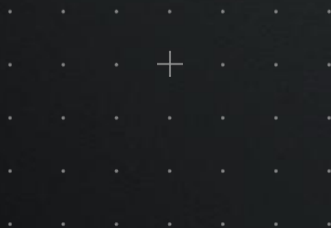
45697056



**Seletor para ID:** O seletor para id é mais específico do que o de tags, formata somente a tag que possui o id especificado, para indicar que o seletor será para um id basta anteceder o nome do id com o caractere hash “#”.

Seletor

```
#titulo { color: #333; font-family: cursive; }
```





# Introdução CSS3 - Seletores

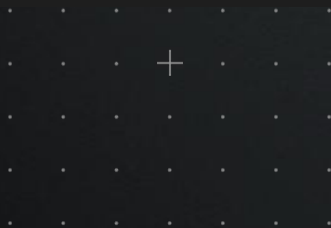
45697056



**Seletor para CLASS:** Este seletor formata todos os elementos que usam a classe indicada, desta maneira podemos ter várias tags <p> por exemplo, somente receberão a formatação aquelas que tiverem o atributo class com o nome da classe configurada, para identificar este seletor, basta usar um ponto “.” antes do nome da classe.

Seletor

```
.esporte { color: ■ #fff; text-align: center; }
```







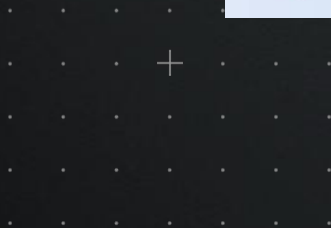
# Introdução CSS3 - Cores

45697056

■ ■ ■

Temos várias formas de especificar cores no CSS, as principais são:

- ✓ Cores Hexadecimais;
- ✓ Cores RGB;
- ✓ Nomes predefinidos entre navegadores.





# Introdução CSS3 - Cores

45697056



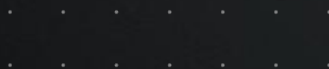
## Cores hexadecimais

Uma cor hexadecimal é especificada com: #RRGGBB, onde os inteiros hexadecimais RR (Vermelho), GG (verde) e BB (azul) especificam os componentes da cor. Todos os valores devem estar entre 00 e FF.

Por exemplo, o valor # 0000ff é renderizado como azul, porque o componente azul é definido como seu valor mais alto (ff) e os outros são definidos Como (00).

RED GREEN BLUE

```
.texto { color:  #0000ff; }
```





# Introdução CSS3 - Cores

45697056



## Coors RGB

Um valor de cor RGB E Especificado com uma função `rgb()`, que possui a seguinte sintaxe:

```
rgb(red, green, blue)
```

Cada parametro (Vermelho, Verde e Azul) irá definir uma intensidade da cor, pode ser um número inteiro Entre 0 e 255 ou um valor percentual (de 0% a 100%).

RED GREEN BLUE

```
.texto { color:  rgb(65,130,90); }
```





# Introdução CSS3 - Cores

45697056  
■ ■ ■

## Cores c/ Alfa

Os valores com RGB e Hexadecimal podem receber um quarto valor que define a opacidade da cor, chamamos ela de Alfa. Podemos definir a opacidade assim:

### RGB:

`rgba(red, green, blue, alfa)` <= O número alfa é um número entre 0.0 (totalmente transparente) e 1.0 (totalmente opaco).

### HEXADECIMAL:

`# red greenblue alfa` <= Já no hexadecimal a opacidade é configurada pelo mesmo range de número que as demais cores.



# Introdução CSS3 - Cores

45697056



## Nomes de Cores

Todos os navegadores modernos aceitam pouco mais de 140 nomes de cores :

MidnightBlue	#00008B	(0,0,128)
Navy	#00008B	(0,0,133)
DarkBlue	#00008B	(0,0,205)
MediumBlue	#0000CD	(0,0,255)
Blue	#0000FF	(0,0,255)
CornflowerBlue	#6495ED	(100,149,237)
RoyalBlue	#4169E1	(65,105,225)
DodgerBlue	#1E90FF	(30,144,255)
DeepSkyBlue	#00BFFF	(0,191,255)
LightSkyBlue	#87CEFA	(135,206,250)
SkyBlue	#87CEEB	(135,206,235)
LightBlue	#ADD8E6	(173,216,230)
SteelBlue	#4682B4	(70,130,180)
LightSteelBlue	#B0C4DE	(176,196,222)
SlateGray	#708090	(112,128,144)
LightSlateGray	#778899	(119,136,153)

Tons de Ciano		
Aqua / Cyan	#00FFFF	(0,255,255)
	#00CED1	(0,206,209)

MediumVioletRed	#C71585	(199,21,133)
DeepPink	#FF1493	(255,20,147)
HotPink	#FF69B4	(255,105,180)
PaleVioletRed	#DB7093	(219,112,147)
LightPink	#FFB6C1	(255,182,193)
Pink	#FFC0CB	(255,192,203)
LightCoral	#F08080	(240,128,128)
IndianRed	#CD5C5C	(205,92,92)
Crimson	#DC143C	(220,20,60)

Tons de Vermelho		
Maroon	#800000	(128,0,0)
DarkRed	#8B0000	(139,0,0)
FireBrick	#B22222	(178,34,34)
Brown	#A52A2A	(165,42,42)
Salmon	#FA8072	(250,128,114)
DarkSalmon	#E9967A	(233,150,122)
LightSalmon	#FFA07A	(255,160,122)
		(255,127,80)

[https://www.w3schools.com/colors/colors\\_names.asp](https://www.w3schools.com/colors/colors_names.asp)

45697056



# CSS - Preenchimentos de Fundo





# Preenchimento de Fundo

45697056



A propriedade `background` permite preencher o fundo da caixa dos elementos de diversas formas.

**background-color:** É a propriedade que define a cor de fundo de um elemento.

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

**background-image:** É a propriedade que define uma imagem como fundo.

```
body{background-image: url('img/JavaScript-logo.png');}
```







## Preenchimento de Fundo

45697056

A propriedade **background-repeat** controla a repetição de uma imagem de fundo. Podemos configurar esta propriedade de 4 formas:

**background-repeat : repeat-x**: imagem repete na horizontal

**background-repeat : repeat-y** : imagem repete na vertical

**background-repeat : repeat**: imagem repete em ambas as posições

**background-repeat : no-repeat**: a imagem não se repete

```
body{background-image: url('img/JavaScript-logo.png');  
      background-repeat:no-repeat;}
```



# Preenchimento de Fundo

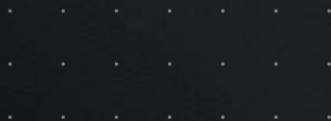
45697056



**background-attachment : fixed:** A imagem de fundo permanece na mesma posição da página.

**background-attachment : scroll:** A imagem de fundo move-se com o conteúdo quando o usuário rola a página.

```
body{background-image: url('img/JavaScript-logo.png');  
      background-repeat:no-repeat;  
      background-attachment: fixed;}
```





## Preenchimento de Fundo

45697056

Quando uma imagem não é repetida a propriedade background-position especifica a posição da imagem na janela. Podemos usar os valores:

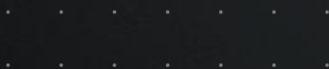
**Top, Bottom e Center:** para alinhamento vertical;

**Right, Left e Center:** para alinhamento horizontal;

```
body{background-image: url('img/JavaScript-logo.png');  
      background-repeat:no-repeat;  
      background-position: center center;}
```

Também podemos usar a **percentagem** para definir estes valores.

```
body{background-image: url('img/JavaScript-logo.png');  
      background-repeat:no-repeat;  
      background-position: 50% 50%;}
```







## Fontes – font-family

45697056

■ ■ ■

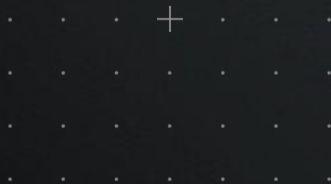
A propriedade font-Family permite especificar a fonte que deve ser usada para qualquer texto dentro do(s) elemento(s) ao qual a regra CSS se aplica.

Você pode especificar uma lista de fontes separadas por vírgulas de modo que, se o usuário não tiver a primeira fonte instalada, o navegador pode tentar usar a próxima da lista.

propriedade

valores

```
h1{font-family: arial,calibri;}
```





# Fontes – font-family

45697056

■ ■ ■

É uma boa prática terminar a lista com o tipo da fonte principal, pois se nenhuma delas for encontrada, o navegador busca uma fonte do mesmo tipo.

Temos 3 principais tipos de fonte:

**SERIF:** ela têm detalhes adicionais nas extremidades

**SANS-SERIF:** ela têm as extremidades retas, sem detalhes

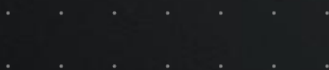
**MONOSPACE:** ela têm todas as letras com o mesmo espaçamento

propriedade

fonte

tipo da fonte

```
h1{font-family: arial, sans-serif;}
```





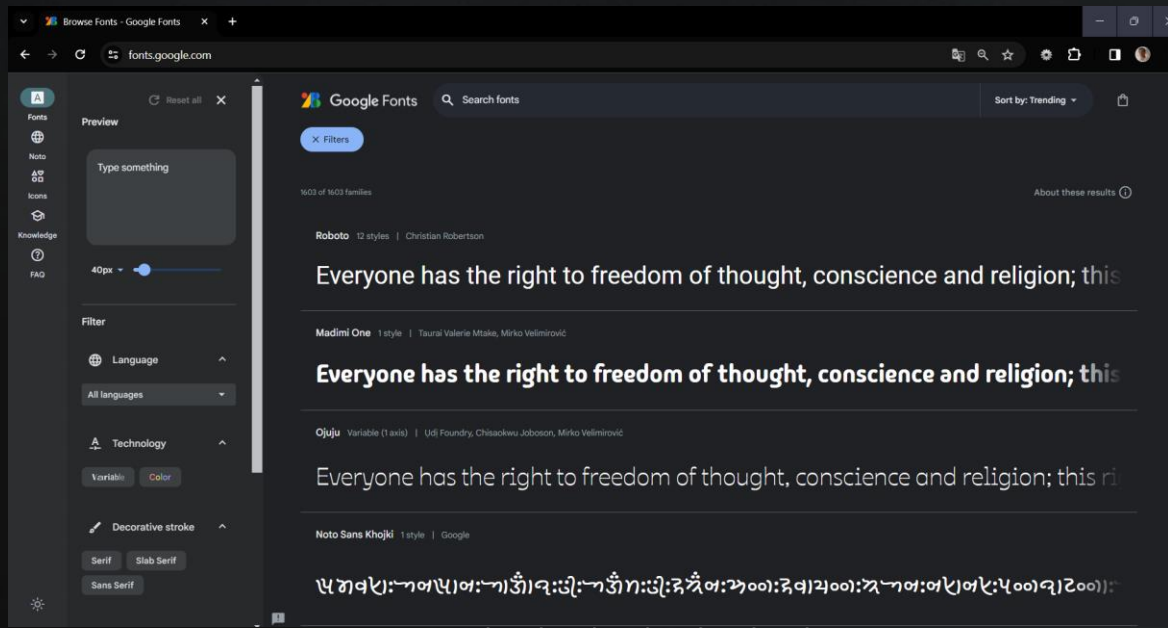
# Fontes – font-family

45697056



## Fonts-google

Podemos também utilizar uma fonte personalizada em nossas páginas sem ter que baixar e copiar em nosso projeto. Para isso podemos, por exemplo, utilizar as fontes que o google disponibiliza gratuitamente. Acesse <https://fonts.google.com/>.





# Fontes – font-family

45697056

■ ■ ■

Fonts-google

Devemos aplicar em nosso arquivo CSS da seguinte forma:

Referência da fonte

```
@import url('https://fonts.googleapis.com/css?family=Lobster');
```

```
h1{font-family: 'Lobster', cursive;}
```

aplicação

OBS. Sem a conexão com a internet as fontes não irão funcionar.





# Fontes – font-size

45697056



A propriedade `font-size` permite especificar o tamanho da fonte. Podemos utilizar algumas unidades de medida para isso:

**PIXELS:** referente aos pontos da tela, ela permite uma precisão maior para controlar o tamanho das fontes. Ex: **80px**.

**PERCENTAGENS:** Ele altera o tamanho da fonte usando uma percentagem relativa ao seu tamanho padrão. O padrão de textos é 16px. Ex: **75%** (sobre 16 seria 12px)

**EMs:** Um em é equivalente ao tamanho do elemento pai. O padrão das páginas é 16px. Ex: **1.5em**.

```
p{font-size: 24px;}
```



## Fontes – font-weight

45697056

A propriedade font-weight deixa o texto em negrito. Temos duas variáveis desta propriedade:

**Normal:** Isso faz com que o texto apareça em um peso normal.

```
p{font-weight: normal;}
```

**Bold:** Isso faz com que o texto apareça em negrito.

```
p{font-weight: bold;}
```





## Fontes – font-style

45697056

Para criar um texto em itálico usamos a propriedade font-style. Temos 3 valores possíveis para esta propriedade:

**Normal:** Isso faz com que o texto apareça em um estilo normal.

```
p{font-style: normal;}
```

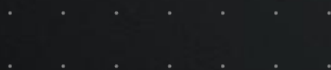
**Italic:** Isso faz com que o texto apareça em itálico.

```
p{font-style: italic;}
```

**Oblique:** Isso faz com que o texto apareça oblíquo.

```
p{font-style: oblique;}
```

**OBS.** Se o navegador não localizar a opção itálica da fonte o oblique faz com que ele incline a fonte original.





# Fontes – font-variant

45697056

■ ■ ■

É a propriedade que define a variação em tamanho da fonte:

**Normal:** não possui alteração no estilo da fonte. (default)

```
p{font-variant: normal;}
```

**Small-caps:** define um “caps” um pouco abaixo da caixa alta.

```
p{font-variant: small-caps;}
```







## Textos – text-align

45697056

■ ■ ■

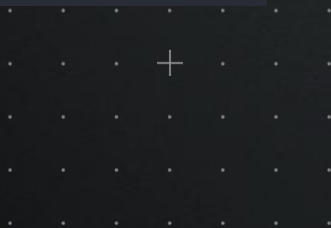
A propriedade text-align permite controlar o alinhamento do texto. A propriedade pode receber um dos 4 valores:

**left:** Isso indica que o texto irá ser alinhado a esquerda.

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

**right:** Isso indica que o texto será alinhado a direita.

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





# Textos – text-align

45697056



**center:** Isso faz com que o texto se alinhe ao centro.

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

**justify:** Isso faz com que o texto se alinhe dos dois dados.

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





## Textos – text-transform

45697056

O text-transform é usado para alterar o uso de maiúsculas e minúsculas no texto. Temo os seguintes valores:

**Uppercase:** Isso faz com que o texto apareça em maiúsculo.

```
p{text-transform: uppercase;}
```

**Lowercase:** Isso faz com que o texto apareça em minúsculo.

```
p{text-transform: lowercase;}
```

**Capitalize:** Isso faz com que a primeira letra de cada palavra do texto apareça em maiúsculo.

```
p{text-transform: capitalize;}
```





# Textos – text-decoration

45697056



O text-decoration permite aplicar 5 valores diferentes:

**none:** Isso remove qualquer decoração já aplicada no texto.

```
a{text-decoration: none;}
```

**underline:** Adiciona uma linha abaixo do texto.

```
p{text-decoration: underline;}
```

**overline:** Adiciona uma linha acima do texto.

```
p{text-decoration: overline;}
```



# Textos – text-decoration

45697056



**Line-through:** Adiciona uma linha no meio do texto.

```
p{text-decoration: line-through;}
```

**Underline overline:** Cria uma linha acima e outra abaixo do texto.

```
p{text-decoration: underline overline;}
```



# Textos – text-shadow

45697056  
■ ■ ■

A propriedade text-shadow cria uma sombra nas letras do texto .

**Horizontal**   **Vertical**   **Distorção**   **Cor**

```
p{text-shadow: 1px 1px 1px red;}
```





# Textos – Outras Configurações

45697056

■ ■ ■

**Line-height:** Regula o espaço entre linhas de um parágrafo.

```
p{line-height: 20px;}
```

**Letter-spacing:** Define o espaçamento entre as letras do texto.

```
p{letter-spacing: 8px;}
```

**Word-spacing:** Define o espaçamento entre as palavras do texto.

```
p{word-spacing: 8px;}
```

**Text-indent:** Recua a primeira linha de um objeto de texto.

```
p{text-indent: 20px;}
```



# Exercício

45697056



1. Crie uma página HTML chamada “exercicioCSS1.html”.
2. Na página coloque o título (h1) “Jornal da FIAP” e 3 Div’s com os títulos (h2) “Tecnologia, Esportes e Negócios”, um em cada Div. Cada uma delas deverá ter 3 parágrafos e uma cor de fundo.
3. Utilizando um bloco style interno, faça pelo menos 5 estilizações no título e parágrafos da área de Tecnologia.
4. Utilizando um folha de estilo externa, faça pelo menos 5 estilizações no título e parágrafos das áreas de Esporte e Negócios.



45697056



# CSS – Box Model



# Box Model

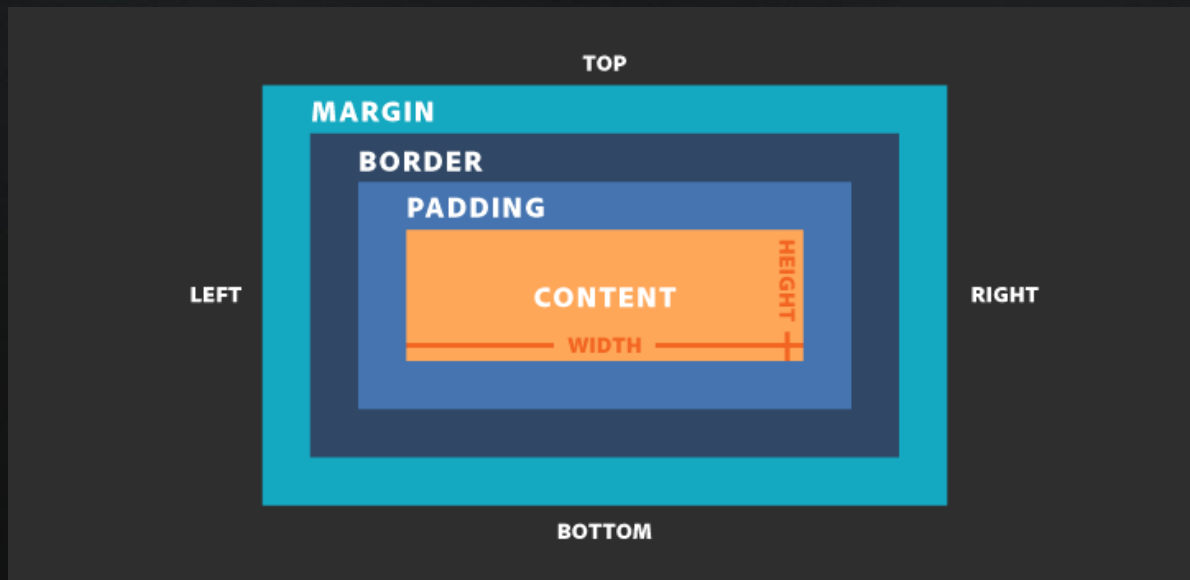
45697056



O Box Model é essencialmente uma caixa que envolve todos os elementos HTML.

É constituído por:

- Margin;
- Border;
- Padding;
- Content.





# Box Model - Dimensões

45697056

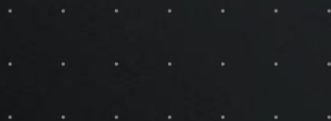


Por padrão, uma caixa recebe o tamanho exato para acomodar seu conteúdo. Podemos fixar estes valores utilizando as propriedades **width** e **height**. Para isso podemos fixar os valores em pixels ou percentagem.

**width:** Define a largura do elemento.

**height:** Define a altura do elemento.

```
div{width: 80%; height:150px;}
```







## Box Model – Limitando largura e altura

45697056

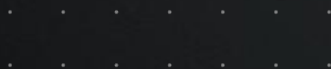


Alguns designs de página se expandem e se contraem para se adaptar ao tamanho das telas, com estas propriedades podemos limitar estas adaptações dando valores máximos e mínimos:

**Min-width e max-width:** Para definir mínimo e máximo para largura.

**Min-height e max-height:** Para definir mínimo e máximo para altura.

```
div{background-color: yellow;  
    max-height: 70px;}
```





# Box Model – Overflow

45697056



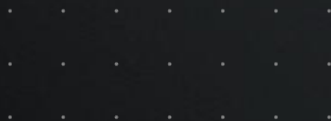
Podemos usar a propriedade `overflow` para tratar como o conteúdo das caixas devem se comportar quando ficam maiores que ela. Temos duas opções:

**`overflow: hidden`**: oculta qualquer conteúdo que não caiba na caixa.

**`overflow: scroll`**: adiciona uma barra de rolagem para rolar o conteúdo.

**`overflow: auto`**: se for necessário, adiciona uma barra de rolagem para rolar o conteúdo.

```
.content{overflow: auto; height: 200px}
```





## Box Model – Bordas Espessura

45697056

Cada caixa tem uma borda (mesmo que ela não estiver visível ou for especificada para ter 0px de largura). A borda contorna e determina a caixa. Para visualizarmos a borda precisamos definir as três propriedades principais: largura, estilo e cor.

**Border-width:** Define a largura de uma borda. Podemos definir uma largura para todas as bordas do elemento ou definir a largura de cada borda.

```
p{border-width: 2px;}
```

Para definir cada lado usamos:

**border-top-width** (acima) | **border-right-width** (direita) | **border-bottom-width** (abaixo) | **border-left-width** (esquerda)

```
p{border-top-width: 2px; border-right-width: 1px;  
border-bottom-width: 3px; border-left-width: 4px;}
```

ou assim: 

```
p{border-width: 2px 1px 3px 4px;}
```



## Box Model – Bordas estilo

45697056



**Border-style:** Define o estilo de uma borda. Podemos definir um estilo para todas as bordas do elemento ou definir o estilo de cada borda.

```
p{border-style: solid;}
```

Para definir cada lado usamos:

**border-top-style** (acima) | **border-right-style** (direita) | **border-bottom-style** (abaixo) | **border-left-style** (esquerda)

```
p{border-top-style: solid; border-right-style: dotted;
border-bottom-style: dashed; border-left-style: double;}
```

ou assim:

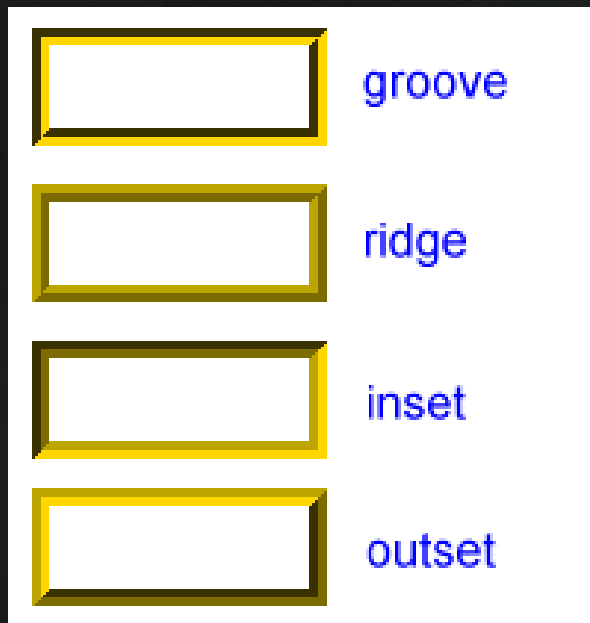
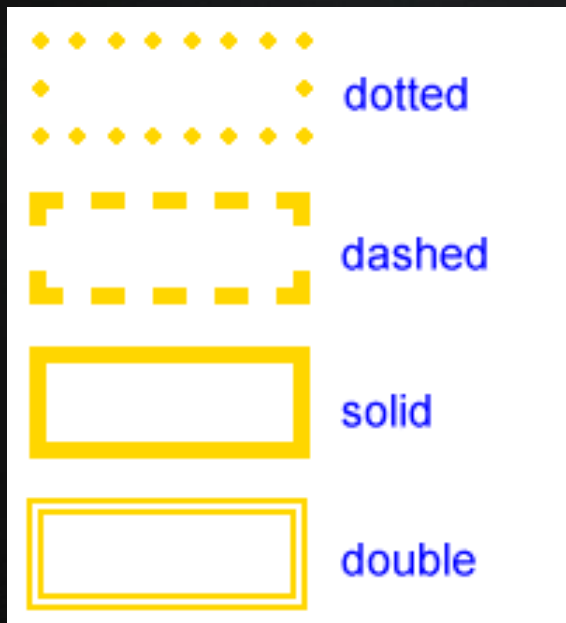
```
p{border-style: solid dotted dashed double;}
```

# Box Model – Bordas estilo

45697056



Podemos utilizar todos estes estilos de bordas em nossas aplicações:





## Box Model – Bordas cor

**Border-color:** Define a cor de uma borda. Podemos definir uma cor para todas as bordas do elemento ou definir a cor de cada borda.

```
p{border-color: black;}
```

Para definir cada lado usamos:

border-top-color (acima) | border-right-color (direita) | border-bottom-color (abaixo) |  
border-left-color (esquerda)

```
p{border-top-color: green; border-right: blue;  
border-bottom-color: yellow; border-left-color: red;}
```

ou assim: 

```
p{border-color: green blue yellow red;}
```

**OBS. É muito comum aplicarmos as três propriedades de uma só vez, da seguinte forma:**

```
p{border: 1px solid black;}
```



## Box Model – Preenchimento

45697056

■ ■ ■

A propriedade padding permite especificar quanto será o espaço terá entre o conteúdo e a borda do elemento.

Podemos definir um preenchimento para todos os lados do conteúdo ou definir o preenchimento de cada lado.

```
p{padding: 5px;}
```

Para definir cada lado usamos:

padding-top (acima) | padding-right (direita) | padding-bottom (abaixo) | padding-left (esquerda)

```
p{padding-top: 5px; padding-right: 4px;  
padding-bottom: 3px; padding-left: 2px;}
```

. . . + . . .

ou assim: 

```
p{padding: 5px 4px 3px 2px;}
```

. . . + . . .



## Box Model – Margem

45697056

A propriedade `margin` define o espaço entre o elemento e os demais elementos da página.

Podemos definir o espaço para todos os lados da caixa ou definir o espaço de cada lado.

```
p{margin: 5px;}
```

Para definir cada lado usamos:

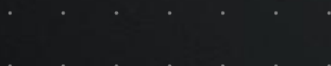
`margin-top` (acima) | `margin-right` (direita) | `margin-bottom` (abaixo) | `margin-left` (esquerda)

```
p{margin-top: 5px; margin-right: 4px;  
margin-bottom: 3px; margin-left: 2px;}
```



ou assim:

```
p{margin: 5px 4px 3px 2px;}
```







## Box Model – Margem

45697056

■ ■ ■

Para centralizarmos uma caixa podemos utilizar o valor **auto** para as margens direita e esquerda.

```
div{margin: 5px auto 3px auto;}
```

Podemos também resumir agrupando top com bottom e right com left, fazendo assim: \_\_\_\_\_

```
div{margin: 5px auto;}
```

**OBS.** Só podemos usar este tipo de resumo quanto quisermos que os valores top com bottom e right com left sejam iguais.





## Box Model – Sombras

45697056

■ ■ ■

A propriedade **box-shadow** permite adicionar uma sombra em torno da caixa.

**Vertical**

**Cor**

```
p{box-shadow: 2px 2px 10px green;}
```

**Horizontal** **Distorção**

Resultado:

Vamos fazer um teste.



## Box Model – Cantos Arredondados

45697056

■ ■ ■

A propriedade `border-radius` permite que arredondemos as bordas da caixa.

```
p{border: 1px solid black;  
border-radius: 2px;}
```

ou assim:

```
p{border: 1px solid black;  
border-radius: 2px 2px 10px 5px;}
```





# Box Model – Box-Sizing

45697056

■ ■ ■

A propriedade **box-sizing** permite incluir o preenchimento e a borda na largura e altura total de um elemento.

**border-content:** é o valor default, o preenchimento e a borda não fazem parte do tamanho total do elemento;

```
div{box-sizing: content-box;}
```

**Border-box:** o preenchimento e a borda fazem parte do tamanho total do elemento;

```
div{box-sizing: border-box;}
```





# Box Model – Posição dos Elementos

45697056  
■ ■ ■

A propriedade `position` define a posição do elemento em relação aos demais da página. Para configurar podemos usar as opções:

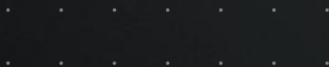
**position: static:** é o valor default do elemento;

**position: relative:** move um elemento em relação ao local onde ele estaria no fluxo normal;

**position: absolute:** remove o elemento do fluxo normal e ele não mais afeta o fluxo dos demais elementos . Precisa usar as propriedades `top`, `bottom`, `right` e `left` para configurar

**position: fixed:** marca o posicionamento em relação a janela. Precisa usar as propriedades `top`, `bottom`, `right` e `left` para configurar ;

```
.bloco{position:fixed; top: 20px; left: 30px;}
```





## Box Model – Sobrepondo Elementos

45697056

■ ■ ■

Ao usar o posicionamento relativo, fixo ou absoluto os elementos podem se sobrepor. Assim, com a propriedade **z-index**, podemos ordenar a sequência de sobreposição. Por exemplo, um elemento com z-index de 5 ficará sobre um outro com o z-index de 4.

```
.bloco{position:fixed;  
    top: 20px;  
    left: 30px;  
    z-index: 4;}
```





## Box Model – Elementos Flutuantes

45697056

■ ■ ■

A propriedade **float** permite selecionar um elemento do fluxo normal e posicioná-lo o máximo possível a direita ou a esquerda do elemento container. Ao usar a propriedade float você também deve usar a propriedade width, pois você pode ter um resultado inconsistente.

```
.bloco{float: left;  
        width: 50%;}
```

Podemos isolar elementos flutuantes utilizando a propriedade **clear**, ela determina que nenhum elemento deve flutuar do lado selecionado do elemento. Temos os valores: **left**, **right**, **both** e **none**.

```
.bloco2{background-color: blue;  
        width: 50%;  
        clear: both;}
```

# DUVIDAS







Copyright © 2015 - 2021 Prof. Luís Carlos S. Silva  
Prof. Alexandre Carlos de Jesus

Todos direitos reservados. Reprodução ou divulgação total ou parcial deste documento é expressamente proibido sem o consentimento formal, por escrito, do Professor (autor).