

INSTITUTO MÉDIO POLITÉCNICO PRIVADO

= MUZUZA =



TÉCNICA DE LINGUAGEM DE  
PROGRAMAÇÃO  
MODULO II  
CSS

## Introdução às CSS

CSS, sigla em inglês para **Cascading Style Sheet** que em português foi traduzido para **Folha de Estilo em Cascata**.

CSS criou o conceito que preconiza o uso dos elementos HTML exclusivamente para marcar e estruturar o conteúdo do documento. Nenhum elemento ou atributo HTML deve ser usado para alterar a apresentação, ou seja, para estilizar o conteúdo.

A tarefa de estilização fica a cargo das regras CSS que são gravadas em um arquivo independente do arquivo HTML no qual são declaradas propriedades e valores de estilização para os elementos da HTML.

As regras de estilos poderão ser escritas em um arquivo externo com extensão **.css** ou incorporadas ao documento HTML com uso do elemento .

### O efeito cascata

Que estilo será aplicado, quando há conflito de estilos especificados (por exemplo: uma regra de estilo determina que os parágrafos serão na cor preta e outra que serão na cor azul) para um mesmo elemento HTML?

Aqui entra o **efeito cascata**, que nada mais é, do que o estabelecimento de uma *prioridade* para aplicação da regra de estilo a um elemento.

Para determinar a prioridade são considerados diversos fatores, entre eles: 1º o tipo de folha de estilo; 2º o local físico da folha de estilo no seu todo; 3º o local físico da regra de estilo na folha de estilo e 4º especificidade da regra de estilo.

A prioridade para o efeito cascata em ordem crescente é a seguinte:

1. folha de estilo padrão do navegador do usuário;
2. folha de estilo do usuário;
3. folha de estilo do desenvolvedor;

### A regra CSS e sua sintaxe

#### Introdução

Uma regra CSS segue uma sintaxe própria que define como será aplicado estilo a um ou mais elementos da marcação HTML de uma página. Um conjunto de regras CSS formam uma folha de estilos. Uma regra CSS, na sua forma mais elementar, compõe-se de três

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

partes: um seletor, uma propriedade e um valor e tem a sintaxe conforme mostrado a seguir:

O significado de cada uma das partes da regra CSS é conforme descrito a seguir:

1. **Seletor:** é um elemento da marcação HTML identificado pelo seu nome (por exemplo: `<p>`, `<h1>`, `<form>`), pelo nome de uma classe aplicada ao elemento da marcação HTML (por exemplo: `.topo`, `.principal`, `.menu`), pelo nome de um identificador ID aplicado ao elemento da marcação HTML (por exemplo: `#tudo`, `#auxiliar`, `#rodape`) ou por qualquer outro identificador CSS de elementos ou trechos da marcação HTML, genericamente denominados de seletores CSS.
2. **Propriedade:** é a propriedade do elemento HTML ao qual será aplicada a estilização definida no valor (por exemplo: `tamanho da fonte`, `cor do texto`, `altura do elemento`).
3. **Valor:** é a característica específica a ser assumida pela propriedade (por exemplo: `letra tipo arial`, `cor azul`, `fundo verde`, `altura igual a 300px`).
4. Ao conjunto **propriedade: valor** denominamos declaração CSS.

**Na sintaxe de uma regra CSS, escreve-se o seletor e a seguir a propriedade e o valor separados por dois pontos e colocados entre chaves { }.**

É possível escrever mais de uma declaração CSS em uma regra CSS com o objetivo de estilizar-se várias propriedades de um mesmo seletor. Neste caso deve-se usar ponto-e-vírgula para separar as declaração CSS existente da regra CSS. O ponto-e-vírgula é facultativo quando a regra CSS for constituída de uma só declaração CSS e também facultativo após a última declaração CSS quando houver mais de uma declaração CSS.

**É recomendável que se use sempre, o ponto-e-vírgula após cada declaração CSS,** pois no caso de futuramente ter que se acrescentar mais declarações na regra CSS não se corra o risco de esquecer eventual ponto-e-vírgula inicialmente omitido.

Observe os exemplos a seguir que esclarecem a sintaxe da regra CSS.

```
p { font-size: 12px; /* ponto-e-vírgula é recomendável */ }
```

```
body { color: #000; background-color: #fff; font-weight: bold; /* ponto-e-vírgula  
é recomendável */ }
```

No exemplo a seguir, o **seletor** é o "documento todo" (elemento *body*), a **propriedade** é a cor do fundo do documento e o **valor** é a cor branca.

```
body {background-color: #fff; }
```

Se o valor for uma palavra composta, deverá estar entre aspas duplas " ", ou simples ' ':

```
h3 { font-family: "Comic Sans MS"; } ou  
h3 { font-family: 'Comic Sans MS'; }
```

Para aumentar a **legibilidade** das folhas de estilo, é de boa prática usar linhas distintas para escrever cada uma das declarações — propriedade e seu valor —, bem como separar o valor da propriedade com uso de um espaço em branco, como mostrado a seguir.

```
p {text-align: right; color: #f00;}
```

Isto não é obrigatório. A regra CSS a seguir tem o mesmo efeito da regra anterior e ambas as sintaxes estão corretas.

```
p { text-align:right;color:#f00; }
```

## Agrupamento de Seletores

Agrupando seletores pode-se fazer com que uma regra CSS seja aplicada a mais de um seletor. Para agrupar seletores separe-os com uma vírgula. No exemplo abaixo agrupamos todos os elementos cabeçalho de níveis 1 e 2 os parágrafos e o elemento com a classe *.box*. A cor do texto de todos eles será verde.

```
h1, h2, p, .box {color: green;}
```

## Seletor classe

Mas você não está restrito somente aos elementos HTML (tags) para aplicar regras de estilo.

Você pode "inventar" um nome e defini-lo como valor a ser atribuído ao atributo *class* (classe) do elemento HTML. O nome "inventado" será o seletor para aplicar declarações CSS. E o mais interessante das classes é que elas podem ser aplicadas a **qualquer elemento** HTML. E mais ainda, você pode aplicar estilos diferentes para o mesmo tipo de elemento HTML, usando classes diferentes para cada um deles.

A sintaxe para o seletor classe é mostrada a seguir. Elemento HTML mais um nome qualquer que você "inventa" precedido de . (ponto):

```
elemento.nomedaclass { propriedade: valor; }
```

Uma nota importante a HTML5 acabou com a restrição dos caracteres e na sintaxe dela você pode usar qualquer caractere exceto espaço em branco, contudo a conselho a evitar, e usar somente os caracteres.

**Por exemplo:** suponha que você precisa de dois estilos para parágrafos em seu documento: um parágrafo com letras na cor preta e um parágrafo com letras na cor azul. Crie duas classes conforme mostrado a seguir.

```
1-exemplo: p.cor-um { color: #000; }
```

```
2-exemplo: p.cor-dois { color: #0ff; }
```

No seu documento HTML as classes seriam aplicadas conforme mostrado a seguir:

HTML

```
<p class="cor-um"> este parágrafo será na cor preta.</p>
```

```
<p class="cor-dois">este parágrafo será na cor azul.</p>
```

Ao nomear uma classe você talvez queira que ela seja aplicável a qualquer elemento HTML. Neste caso basta que se omita o nome do elemento antes da classe. Por exemplo: a regra CSS a seguir pode ser aplicada a qualquer elemento HTML ao qual você deseja atribuir cor azul:

```
.cor-tres { color: blue; }
```

No exemplo a seguir tanto o cabeçalho `<h2>` como o parágrafo `<p>` serão na cor azul:

HTML

```
<h2 class="cor-tres">
```

Este cabeçalho é na cor azul.

```
</h2>
```

```
<p class="cor-tres">
```

## Seletor ID

O seletor ID difere do seletor classe, por ser ÚNICO. Um seletor ID de determinado nome só pode ser definido a UM e somente UM elemento HTML dentro do documento.

Você pode "inventar" um nome e com ele criar um **ID** que será o seletor para definir a regra CSS. Um determinado **ID só pode ser aplicado a UM elemento HTML**.

A sintaxe para o seletor ID é mostrada a seguir. Um nome qualquer que você "inventa" precedido do sinal # ("cerquilha", "tralha" ou "jogo-da-velha" 😊):

```
CSS
#meuID {propriedade: valor;}
```

### Inserindo comentários nas CSS

Você pode inserir comentários nas CSS para explicar seu código, e principalmente ajudá-lo a lembrar de como você estruturou e qual a finalidade de partes importantes do código. Daqui há alguns meses a menos que você seja um privilegiado, terá esquecido a maior parte daquilo que você levou horas para "bolar". O comentário introduzido no código, será ignorado pelo navegador. Um comentário nas CSS começa com o sinal /\* e termina com \*/. Observe o exemplo a seguir:

### CSS

```
/* este é um comentário*/

p { font-size: 14px;

/* este é outro comentário*/

color: #000;

font-family: Arial, Serif;

}
```

### Vinculando folhas de estilo em documentos

#### Introdução

Folhas de estilo, segundo sua localização, podem ser classificadas em três tipos:

1. Externas;
2. Incorporadas;
3. Em escopo;
4. Inline.

Nesta seção estudaremos como vincular a um documento HTML cada um destes tipos de folhas de estilo.

## Folha de estilo externa

Uma folha de estilo é dita externa, quando as regras CSS são declaradas em um arquivo separado dos documentos HTML para os quais foram criadas. Um arquivo de folha de estilo deve ser gravado com a extensão **.css**

Uma folha de estilo externa é ideal para ser aplicada a várias páginas. Com uma folha de estilo externa, você pode mudar a aparência de um site inteiro mudando regras de estilos contidas em um arquivo apenas (o arquivo da folha de estilo).

O arquivo CSS da folha de estilo externa poderá ser lincado ou importado. A sintaxe geral para lincar uma folha de estilo denominada **main.css** é mostrada a seguir.

HTML

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

Conforme mostrado, usa-se o elemento *link* dentro da seção *head* do documento para importar uma folha de estilos em um documento HTML.

O navegador "lê" as regras de estilo do arquivo **main.css**, e formata o documento de acordo com elas.

A sintaxe geral para importar uma folha de estilo denominada **fontes.css** é mostrada a seguir.

CSS

```
/* essa folha de estilo demomina-se main.css */
```

```
@charset utf-8;
```

```
@import url("fontes.css");
```

```
body {
```

```
  margin: 0;
```

```
  background: #fff;
```

```
  color: #333;
```

```
}
```

```
/* mais regras CSS */
```

Conforme mostrado, usou-se a diretiva *@import* dentro da folha de estilos **main.css** para importar uma folha de estilos denominada **fontes.css**.

Ao usar essa diretiva em uma folha de estilo ela **obrigatoriamente** deverá ser a primeira declaração na folha logo a seguir da diretiva *@charset*. Caso contrário não haverá importação.

A seguir você precisa usar o elemento *link* dentro da seção *head* do documento para importar a folha de estilos main.css, conforme mostrado anteriormente.

Folhas de estilos externas podem ser escritas em qualquer editor de texto, devem ser gravadas com a extensão .css e a codificação de caracteres utf-8. O arquivo não deve conter nenhuma tag HTML.

### Notas complementares

A diretiva *@import* também pode ser usada para importação na seção *head* do documento HTML conforme mostrado a seguir.

### HTML

```
<head>
  <style rel="stylesheet" type="text/css">
    @import url("fontes.css");
  </style>
</head>
```

Contudo é aconselhável que esse tipo de importação seja evitado, pois o carregamento da folha de estilo pode trazer queda de performance. Então, prefira o uso do elemento

### Folha de estilo incorporada

Uma folha de estilo é dita incorporada ou interna, quando as regras CSS são declaradas na seção *head* do próprio documento HTML.

Uma folha de estilo incorporada ou interna, é ideal para ser aplicada a uma única página. Com uma folha de estilo incorporada ou interna, você pode mudar a aparência de somente um documento, aquele onde a folha de estilo está incorporada.

As regras de estilo incorporadas e válidas para o documento, são declaradas na seção *head* do documento com a tag de estilo *<style>*, conforme sintaxe mostrada a seguir.



## HTML

```
<head>
<style rel="stylesheet" type="text/css">
  body { background: #000; url(imagens/minhaimagem.gif); }
  h3 { color: #f00; }
  p { margin-left: 15px; padding: 1.5em; }
</style>
</head>
```

### Folha de estilo em escopo

A HTML5 criou uma folha de estilo do tipo vinculada para ser usada dentro de um container da marcação HTML. As regras CSS declaradas nessa modalidade de folha de estilo se aplicam somente ao elementos da marcação que se encontram dentro do container, ou no escopo do container, daí seu nome.

A sintaxe para declarar uma folha de estilo deste tipo é mostrada a seguir.

## HTML

```
<!-- marcação HTML do documento -->
<div class="minha-classe">
  <style rel="stylesheet" type="text/css" scoped="scoped">
    /* regras de estilo */
  </style>
  <!-- marcação HTML dentro do div.minha-classe
  As regras CSS serão aplicadas somente aqui -->
</div>
<!-- mais marcação HTML do documento -->
```

### Folha de estilo inline

Uma folha de estilo é dita inline, quando as regras CSS são declaradas dentro da tag de abertura do elemento HTML com uso do atributo *style*.

Estilo inline só se aplica a um elemento HTML. Ele perde muitas das vantagens das folhas de estilo pois mistura a apresentação com a marcação. Use este método excepcionalmente, por exemplo: quando quiser aplicar um estilo a uma única ocorrência de um elemento.

A sintaxe para aplicar estilo inline é mostrada a seguir:

## HTML

```
<p style="color:#000; margin: 5px;">
  Aqui um parágrafo de cor preta e com 5px nas 4 margens.
</p>
```

## Folhas de estilo múltiplas

Se alguma propriedade for definida para um mesmo elemento em folhas de estilo diferentes, entrará em ação, o EFEITO CASCATA e prevalecerão os valores da folha de estilo mais específica.

Suponha, uma folha de estilo externa com as seguintes declarações para o seletor *h2*:

### CSS

```
h2 {  
  color: #fc0;  
  text-align: center;  
  font: italic 14px Verdana, Sans-serif;  
}
```

### CSS

```
h2 {  
  color: #fc0;  
  text-align: center;  
  font: italic 16px Verdana, Sans-serif;  
}
```

e, uma folha de estilo interna com as seguintes propriedades para o seletor *h2*:

Estando ambas as folhas vinculadas ao documento há um conflito no tamanho da fonte para o elemento *h2*. e Sendo a folha interna declarada depois da externa na seção *head* do documento, prevalecerá a folha interna e a fonte de *h2* terá o tamanho igual a 16px.

---

## Propriedades CSS para estilização de fontes

### Introdução

As propriedades para estilização de fontes, definem os diferentes aspectos de apresentação dos glifos (letras e caracteres) que compõem os conteúdos textuais dos diferentes elementos da marcação HTML.

As propriedades básicas para estilizar fontes que serão estudadas neste tutorial são as listadas a seguir:

**color:** cor da fonte;  
**font-family:** família (tipo) de fontes;  
**font-size:** tamanho da fonte;  
**font-style:** estilo da fonte;  
**font-variant:** fontes maiúsculas de menor altura;  
**font-weight:** peso da fonte;  
**font-stretch:** grau de expansão/contração dos glifos;  
**font:** maneira abreviada para declarar todas as propriedades anteriores (exceto cor).

## Valores válidos para as propriedades da fonte

- **color:**

- ✓ código hexadecimal: `#ffc6d9`
- ✓ código rgb: `rgb(255,235,0)`
- ✓ código rgba: `rgb(255,235,0, 0.7)`

- ✓ código hsl: `hsl(210,100%,40%)`
- ✓ código hsla: `hsla(155,80%,35%,0.4)`
- ✓ palavra-chave: *red, blue, green...etc*
- ✓ transparente: *transparent*

- **font-family:**

- ✓ nome da família de fonte: define-se pelo nome da fonte, exemplos: "verdana", "helvetica", "arial", etc.
- ✓ nome da família genérica: define-se pelo nome genérico da fonte, exemplos: "serif", "sans-serif", "cursive", etc.

- **font-size:**

- ✓ xx-small
- ✓ x-small
- ✓ small
- ✓ medium

- ✓ large
- ✓ x-large
- ✓ xx-large
- ✓ large

- ✓ x-large
- ✓ xx-large
- ✓ smaller
- ✓ larger
- ✓ length: medida CSS de comprimento exemplos: px, em, rem, % (porcentagem) ...

- **font-style:**

- ✓ normal: fonte normal (em pé)
- ✓ italic: fonte inclinada
- ✓ oblique: fonte oblíqua

- **font-variant:**

- ✓ normal: fonte normal
- ✓ small-caps: transforma em maiúsculas de menor altura

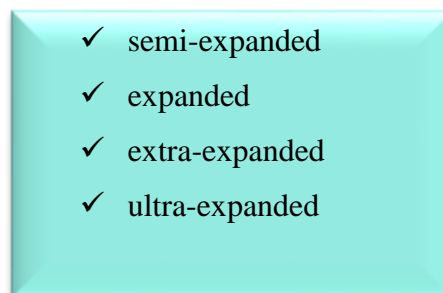
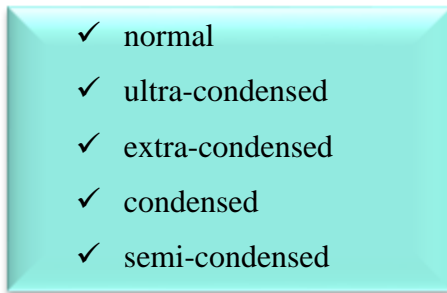
- **font-weight:** (ver explicação em [font-weight](#))

- ✓ Normal
- ✓ bold
- ✓ bolder
- ✓ lighter

- ✓ 100
- ✓ 200
- ✓ 300
- ✓ 400
- ✓ 500

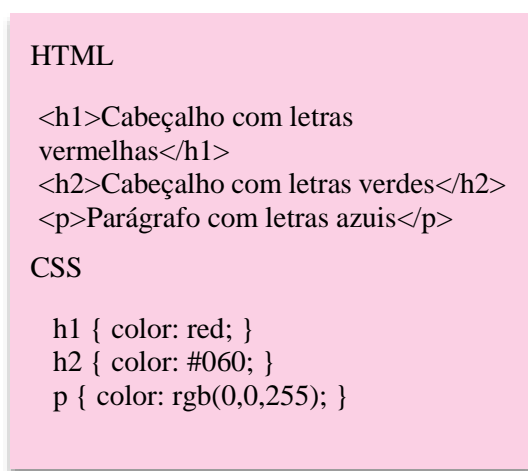
- ✓ 600
- ✓ 700
- ✓ 800
- ✓ 900

- **font-stretch:**

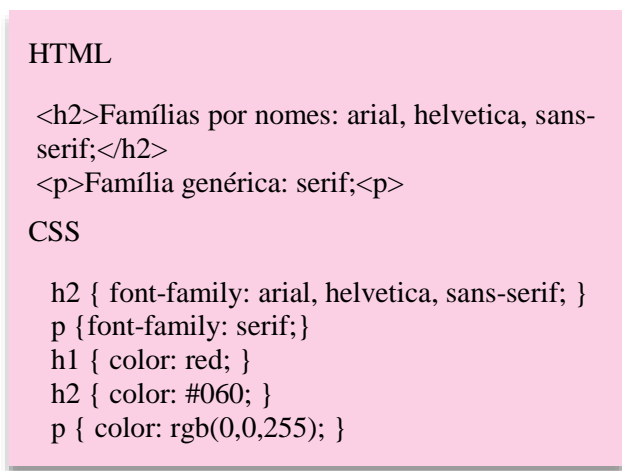


Vejamos a seguir alguns exemplos práticos de declarações para estilizar fontes.

- **color**



- **font-family**



: A propriedade *font-family* é usada para definir uma lista de família de fontes.

O navegador renderiza o primeiro nome de fonte que ele reconhece na lista e ignora os demais **Notas**.

Ao declarar famílias de fontes separe cada nome por uma vírgula e sempre declare por último na lista o nome de fonte genérico daqueles que estão sendo declarados.

Se o nome da fonte for composto por mais de uma palavra, por exemplo: Comic Sans MS, deve-se usar aspas duplas ou simples na grafia do nome. Se for definido o atributo "style" na marcação HTML (para estilização inline - evite isso), onde as aspas duplas já estão presentes usar obrigatoriamente aspas simples na sintaxe para grafar o nome de fonte composto.

- *font-size*

#### HTML

```
<h1>Letras com tamanho: 14px</h1>
<h2>Letras com tamanho: smaller</h2>
<p>Letras com tamanho: 100%</p>
```

#### CSS

```
h1 { font-size: 16px; }
h2 { font-size: smaller; }
p { font-size: 100%; }
```

- *font-style*

#### HTML

```
<h1>Este é o estilo italic</h1>
<h2>Este é o estilo normal</h2>
<p>Este é o estilo oblique</p>
```

#### CSS

```
h1 { font-style: italic; }
h2 { font-style: normal; }
p { font-style: oblique; }
```

- *font-variant*

#### HTML

```
<p>Este cabeçalho com letras normais</p>
<p>Este parágrafo com letras em "small-caps"</p>
```

#### CSS

```
p1 { font-variant: normal; }
p2 { font-variant: small-caps; }
```

### *font-weight*

Esta propriedade CSS consagrou-se como sendo aquela destinada a obter o efeito conhecido como "negrito" cuja finalidade é realçar palavras de um texto, contudo, os efeitos que ela causa vão muito além do simples negrito.

A tradução de *font-weight* é: "peso da fonte" e estas palavras foram usadas para dar nome a uma propriedade CSS cujos valores definem o ... peso da fonte.

### *bolder e lighter*

O mapeamento dos valores *bold* e *lighter* é conforme mostrado na tabela a seguir:

#### HTML

```
<p>Este é um parágrafo com glifos de peso bold</p>
<p>Este é um parágrafo com glifos de peso 700</p>
<p>Este é um parágrafo com glifos de peso 400</p>
<p>Este é um parágrafo com glifos de peso 900</p>
```

```
p1 { font-weight: bold; }
p2 { font-weight: 700; }
p3 { font-weight: 400; }
p4 { font-weight: 900; }
```

## A propriedade CSS *margin*

### Introdução

As propriedades para as margens, definem a dimensão de cada uma das quatro margens de um elemento HTML e são as listadas a seguir:

**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 definir todas as 4 margens.

### Valores válidos para as propriedades da margem

- **auto:** margens laterais iguais (para centrar na horizontal elementos nível de bloco)
- **Comprimento:** um valor CSS para comprimento (**px, em, pt, etc**)
- **Porcentagem:** um valor expresso em percentagem;

É válido declarar **valores negativos** para margem, com o objetivo de sobrepor elementos.

#### *margin-top*

##### HTML

```
<p>
  Uma margem superior de 2cm
</p>
```

##### CSS

```
p { margin-top: 2cm; }
```

#### *margin-right*

##### HTML

```
<p>
  Uma margem direita igual a 80% nesta
  frase do parágrafo.
</p>
```

##### CSS

```
p { margin-right: 80%; }
```

#### *margin-bottom*

##### HTML

```
<p>
  Uma margem inferior de 30mm
</p>
```

##### CSS

```
.p { margin-bottom: 30mm; }
```

#### *margin-left*

##### HTML

```
<p>
  Uma margem esquerda de 5em
</p>
```

##### CSS

```
.p { margin-left: 5em; }
```

## margin

A propriedade da margin é a maneira abreviada de se declarar as 4 margens. A ordem de declaração das margens é: **superior, direita, inferior e esquerda**.

Há quatro maneiras de se declarar abreviadamente as margens:

- *margin: valor1*  
as 4 margens terão valor1;
- *margin: valor1 valor2;*  
margem superior e inferior terão valor1 - margem direita e esquerda terão valor2
- *margin: valor1 valor2 valor3;*  
margem superior terá valor1 - margem direita e esquerda terão valor2 - margem inferior terá valor3
- *margin: valor1 valor2 valor3 valor4;*  
margens superior, direita, inferior e esquerda nesta ordem.

### HTML

```
<p>  
  Uma margem superior de 20px, uma margem direita de 40px,  
  uma margem inferior de 80px e uma margem esquerda de 15px  
</p>
```

### CSS

```
.p { margin: 20px 40px 80px 15px; }
```

## Sobreposição de margens verticais

É preciso que o autor entenda o algoritmo de cálculo das CSS para determinar a margem vertical resultante entre dois boxes. A margem resultante é obtida por sobreposição também conhecida como "**collapsing margins**" (**sobreposição de margens**)

Nota: Margens horizontais nunca se sobrepõem.

Observe a marcação HTML e as regras CSS para dois boxes

### HTML

```
<div class="um">1  
</div>  
<div class="um">2  
</div>
```

### CSS

```
div.um {  
  width: 100%;  
  background: lightblue;  
  margin-bottom: 40px;  
}
```

```
div.dois {  
  width: 100%;  
  background: lightcoral;  
  margin-top: 50px;  
}
```

## A propriedade CSS *padding*

### Introdução

A propriedade *padding* se destina a criar um espaçamento interno em um box entre seus 4 lados e a área de conteúdo do box.

As propriedades para declarar *padding*, definem a dimensão de cada uma dos quatro espaçamentos entre a área das bordas e a área de conteúdo de um elemento HTML e são as listadas a seguir:

**Padding-top:** define o espaçamento superior;

**Padding-right:** define o espaçamento à direita;

**Padding-bottom:** define o espaçamento inferior;

**Padding-left:** define o espaçamento à esquerda;

**Padding:** maneira abreviada para

definir todos os 4 espaçamentos.

### Valores válidos para as propriedades *padding*

- **Comprimento:** um valor CSS para comprimento (px, em, pt, etc)
- **Porcentagem:** um valor expresso em percentagem;

Valores declarados com uso de percentagem são calculados, **para os quatro lados do box**, tomando-se a percentagem em relação à largura (width) do box.

Não é válido declarar **valores negativos** para *padding*.

### *Padding-top*

HTML

<p>

Um espaçamento superior de 2cm

</p>

CSS

p { padding-top: 2cm; }



### ***Padding-right***

#### HTML

<p>

Um espaçamento à direita igual a 80% neste parágrafo.

</p>

#### CSS

p { padding-right: 80%; }

A renderização do código acima é conforme mostrado a seguir:

Um espaçamento à direita igual a 80% neste parágrafo.

### ***Padding-bottom***

#### HTML

<p>

Um espaçamento inferior de 30mm

</p>

#### CSS

.p { padding-bottom: 30mm; }

A renderização do código acima é conforme mostrado a seguir:

### ***Padding-left***

<p>

Um espaçamento à esquerda de 5em

</p>

#### CSS

.p { padding-left: 5em; }

A renderização do código acima é conforme mostrado a seguir:

Um espaçamento à esquerda de 5em

### ***Padding***

A propriedade *padding* é a maneira abreviada de se declarar os 4 espaçamentos. A ordem de declaração dos espaçamentos é: **superior, direito, inferior e esquerdo**.

Há quatro maneiras de se declarar abreviadamente os espaçamentos:

*padding*: valor1;

os 4 espaçamentos terão valor1

*padding*: valor1 valor2;

espaçamento superior e inferior terão valor1

espaçamento direito e esquerdo terão valor2

*padding*: valor1 valor2 valor3;

espaçamento superior terá valor1

espaçamento direito e esquerdo terão valor2

espaçamento inferior terá valor3

*padding*: valor1 valor2 valor3 valor4;

espaçamentos superior, direito, inferior e esquerdo nesta ordem.

## HTML

<p>

Um espaçamento superior de 20px, um espaçamento à direita de 40px,  
um espaçamento inferior de 80px e um espaçamento à esquerda de 15px

</p>

## CSS

.p {padding: 20px 40px 80px 15px;}

A renderização do código acima é conforme mostrado a seguir:

Um espaçamento superior de 20px, um espaçamento à direita de 40px, um espaçamento inferior de 80px e um espaçamento à esquerda de 15px

## Boxes responsivos

Dissemos que valores de *padding* declarados com uso de percentagem são calculados, **para os quatro lados do box**, tomando-se a percentagem em relação à

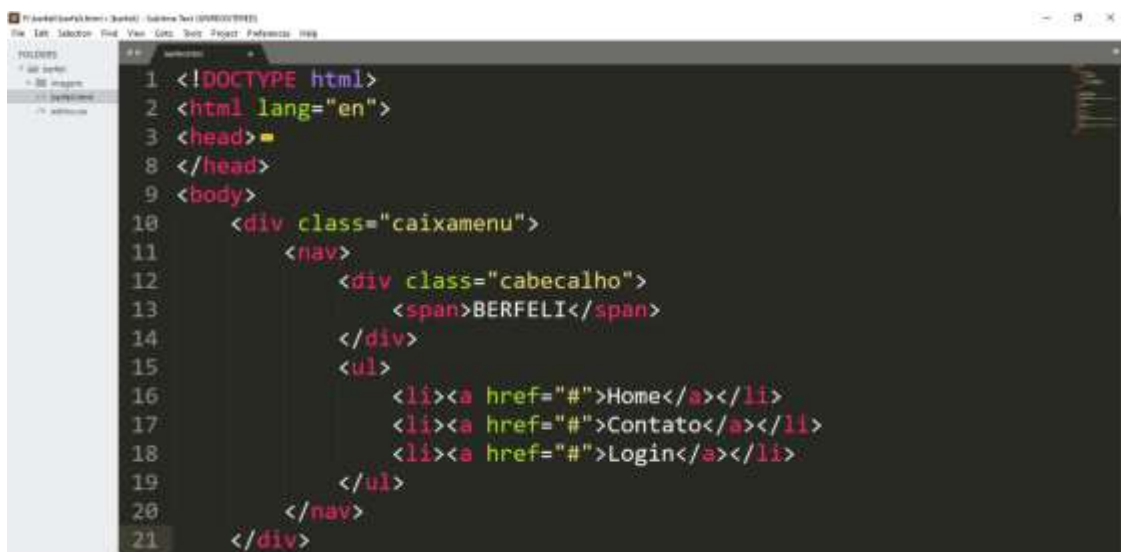
largura (width) do box. Este comportamento possibilita a criação de boxes responsivos, pois uma vez que a declaração de altura (height) em percentagem é praticamente inviável, mesmo em layouts simples podemos declarar *padding* em percentagem para criar uma box responsivo com razão de especto constante.

## CSS

```
div {  
  
    width: 100%;  
  
    padding-bottom: 56.25%;  
  
    background: brown;
```

## EXERCÍCIO 1 – CRIANDO MENU HORIZONTAL APLICANDO RECURSOS DE CSS

### 1- HTML

A screenshot of a code editor window with a dark theme. The editor shows HTML code for a horizontal menu. The code is as follows:

```
1 <!DOCTYPE html>  
2 <html lang="en">  
3 <head>  
4 </head>  
5 <body>  
6 <div class="caixamenu">  
7 <nav>  
8 <div class="cabecalho">  
9 <span>BERFELI</span>  
10 </div>  
11 <ul>  
12 <li><a href="#">Home</a></li>  
13 <li><a href="#">Contato</a></li>  
14 <li><a href="#">Login</a></li>  
15 </ul>  
16 </nav>  
17 </div>
```

### 2- CSS

```

1  nav{
2    display: flex;
3    color: black;
4    padding: 10px 20px;
5    border-radius: 10px 10px 0px 0px;
6    background-color: chocolate;
7    font-size: 20px;
8    justify-content: space-around;
9    padding-top: 30px;
10 }
11 nav ul{
12
13   display: inline-block;
14 }
15 nav ul li{
16   list-style: none;
17   display: inline-block;
18 }
19 nav ul li a{
20   text-decoration: none;
21   padding: 10px 20px;
22 }

```

## EXERCÍCIO 2 – CRIANDO MENU VERTICAL APLICANDO RECURSOS DE CSS

### 1- HTML

```

1  <!DOCTYPE html>
2  <html>
3  <head>
4    <meta charset="UTF-8">
5    <meta name="viewport" content="width=device-width, initial-scale=1.0">
6    <link rel="stylesheet" href="estilos1.css">
7    <title>BERFELI-TECNOLOGIAS</title>
8  </head>
9  <body>
10   <div class="menuvertical">
11     <div class="cabeca">
12       <h3>BERFELI</h3>
13     </div>
14     <nav class="navegacao">
15       <ul>
16         <li><a href="#">Home</a></li>
17         <li><a href="#">Cursos</a></li>
18       </ul>
19       <ul>
20         <li><a href="#">Redes</a></li>
21         <li><a href="#">Programação</a></li>
22         <li><a href="#">Base de Dados</a></li>
23       </ul>
24       <li><a href="#">Sobre</a></li>
25     </nav>
26   </div>
27 </body>

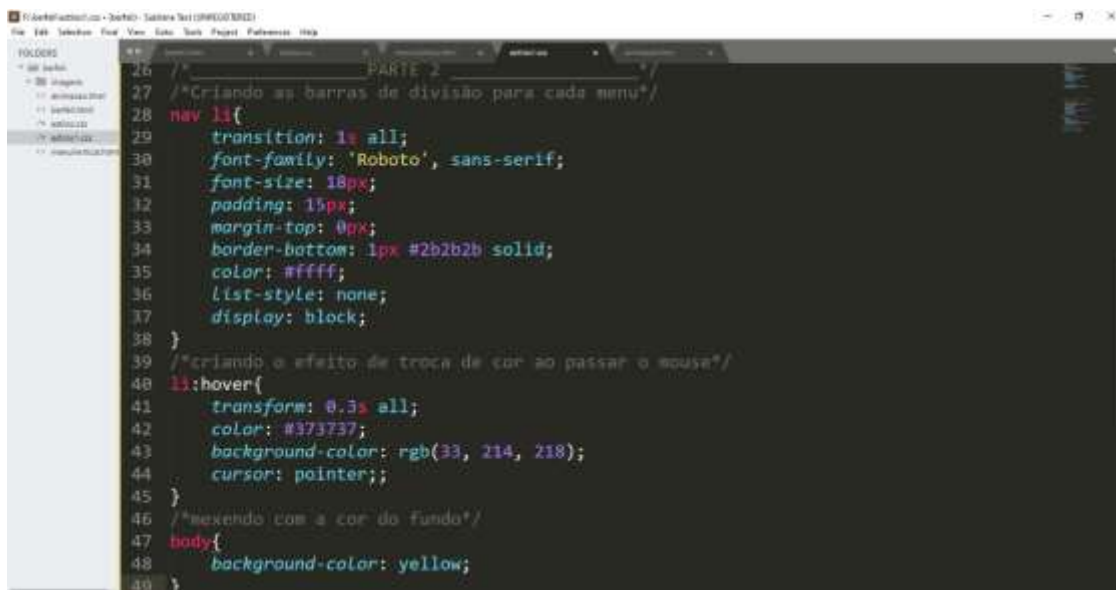
```

### 2- CSS

```

1  /*colocando as bordas*/
2  .cabeca{
3    padding: 10px 20px;
4    border-radius: 10px 10px 0px 0px;
5    background-color: #424242;
6  }
7  .menuvertical{
8    width: 300px;
9    height: auto;
10   background-color: transparent;
11   border-radius: 10px 10px;
12   margin: auto;
13   overflow: hidden;
14 }
15 h3{
16   color: #ffffff;
17   font-family: 'Roboto', sans-serif;
18   margin-left: 1rem;
19 }

```



```
26 /*
27  *Criando as barras de divisão para cada menu*/
28 nav li{
29     transition: 1s all;
30     font-family: 'Roboto', sans-serif;
31     font-size: 18px;
32     padding: 15px;
33     margin-top: 0px;
34     border-bottom: 1px #2b2b2b solid;
35     color: #ffff;
36     list-style: none;
37     display: block;
38 }
39 /*criando o efeito de troca de cor ao passar o mouse*/
40 li:hover{
41     transform: 0.3s all;
42     color: #373737;
43     background-color: rgb(33, 214, 218);
44     cursor: pointer;;
45 }
46 /*mexendo com a cor do fundo*/
47 body{
48     background-color: yellow;
49 }
```

## Criando animação flipping com CSS

### Introdução

Criar animações com CSS é muito divertido. O legal das animações é que com algumas poucas propriedades é possível criar-se praticamente qualquer coisa, desde simples efeitos de esmaecimento (fade) até complexas e estonteantes animações. Um efeito de animação CSS bem legal é o efeito flip (virar). Para obtê-lo se insere em um container dois conteúdos; um na frente e outro no verso e mostra-se um ou outro conteúdo segundo uma animação. Este tutorial se propõe a mostrar como criar esa animação de uma forma a mais simples possível.

[Ver exemplo](#)

### HTML

A marcação HTML para criar a frente e verso de um container é conforme mostrada a seguir:

```
1 <!DOCTYPE html>
2 <html>
3 <head>
4   <meta charset="UTF-8">
5   <meta name="viewport" content="width, initial-scale=1.0">
6   <link rel="stylesheet" href="estilos.css">
7   <title>BERFELI-TECNOLOGIAS</title>
8 </head>
9 <body>
10  <div class="flip-container">
11    ontouchstart="this.classList.toggle('hover');">
12    <div class="flipper">
13      <div class="front">
14        <p align="center">BERFELI</p>
15      </div>
16      <div class="back">
17        <p align="center">TECNOLOGIAS</p>
18      </div>
19    </div>
20  </div>
21 </body>
22 </html>
```

Existem dois containers *.front* e *.back* como era de se esperar. Esses dois containers estão contidos no container *.flipper* e o container geral é *.flip-container*. A função destes dois containers gerais fica clara quando examinamos as regras CSS criadas para obtenção do efeito. Notar ainda o atributo *ontouchstart* que faz com que a animação seja possível em telas sensíveis ao toque. Obviamente você poderá retirar aquele código JavaScript da marcação HTML e colocá-lo em arquivo separado se assim preferir.

## CSS

Deixando de lado os prefixos proprietários observe como algumas poucas regras CSS são suficientes para obter o efeito:

```
98 /* O container geral define a perspectiva */
99 .flip-container { perspective: 1000; }
100 /* vira os containers frente e verso quando o mouse passa em cima */
101 .flip-container:hover .flipper, .flip-container.hover .flipper {
102   transform: rotateY(180deg);
103 }
104 .flip-container, .front, .back {
105   width: 320px;
106   height: 480px;
107 }
108 /* define a velocidade da transição */
109 .flipper {
110   transition: 0.6s;
111   transform-style: preserve-3d;
112   position: relative;
113 }
```

```

120 /* esconde o verso durante a animação */
121 .front, .back {
122     backface-visibility: hidden;
123     position: absolute;
124     top: 0;
125     left: 0;
126 }
127 /* frente posicionada sobre o verso */
128 .front { z-index: 2; }
129 /* verso inicialmente escondido */
130 .back { transform: rotateY(180deg); }

```

A seguir uma breve explicação do processo de animação:

- O container geral *.flip-container* define a área de animação;
- Os containers *.front* e *.back* são os elementos que se movimentam rotacionando de 180º quando o container *.flipper* recebe o ponteiro do mouse sobre ele. É para esse container que se define também a velocidade da transição. Definindo a rotação para -180º. Inverte-se a direção da animação.
- Os containers para a frente e verso são posicionados de forma absoluta de modo que um fique sobre o outro na mesma posição. **backface-visibility** é definida como *hidden* para que o verso do elemento virado não seja mostrado durante a animação.
- O elemento da frente possui um *z-index* maior do que o elemento do verso possibilitando a que o elemento da frente conste em primeiro na marcação HTML, mas seja renderizada na frente do verso.
- O elemento do verso faz a rotação de 180º de modo a que ele atue efetivamente como se estivesse no verso.

Isso é tudo! Crie a estrutura como mostrado e estiliza cada lado, frente e verso, como bem entender!!

OBS: Every wrong you collect in document, please, let us know, sending e-mail  
to [lukokipaulo@gmail.com](mailto:lukokipaulo@gmail.com).

Thank you!!!



## DADOS DO PROFESSORE

**Nome:** Bernardo Kinavuide Paulo Lukoki

**Formação:** Engenharia Informática

**Especialidade:** Engenharia de Software

## FORMAÇÕES DE PÓS-GRADUAÇÃO

- ✓ Curso a distância de Programação Web, DELTEC, Brasil
  - a. HTML
  - b. CSS
  - c. JavaScript
- ✓ Curso a distância de Programação Web, DELTEC, Brasil
  - a. PHP
  - b. MySql
- ✓ Curso a distância de Administração de Bases de dado, Angola, Cacuaco e Uíge, Nucleo universitário (Kimpa Vita).
- ✓ Curso presencial de Redes de computador e infra-estruturas de backbones, MSQ-Tecnologias, Angola, Belas.
- ✓ Curso de Inglês completo, Boy Dream School e Freedom Inglis School, Angola, Cazenga.
- ✓ Curso de programação web com Python, ECTIC, Angola, Belas, e por correspondência, SybexTechno, Africa do Sul.
- ✓ Curso de gestão de projetos informáticos, Caritas, Angola, Uíge.
- ✓ Formação integral de análise matemática I, II, III, IV e V, albert Einste, Angola, Cacuaco.
- ✓ Formação Física básica, albert Einste, Angola, Cacuaco.
- ✓ Curso de Manutenção e reparação de computadores (Hardware), Angola, Uíge.
- ✓ Curso de Agragação Pedagógica, Faculdade de economia, Angola, Uíge.
- ✓ Curso de Excel Avançado, ECTIC, Angola, Belas.
- ✓ Curso de Power BI (Business Intelligence), ECTIC, Angola, Belas.
- ✓ Curso de IPTV (Televisão sobre Internet), MAMBU-Tecnologias, LDA, Angola, Cacuaco.

# FIM

I WISH YOU BETTER