



Prof. MSc. Marcos Alexandruk

E-mail: alexandruk@uni9.pro.br

<https://github.com/alexandruk/desenvolvimentoparainternet>

# **Introdução para Programação à Internet**

## **aula 1**

## O que é e para que serve a HTML 5

HTML (Hyper Text Markup Language) é uma Linguagem de Marcação de Hipertexto utilizada para criação de páginas da **www** na Internet. Essa linguagem é o que o seu navegador (browser, em inglês) precisa para exibir as páginas.

Desde a invenção da web por Tim Berners-Lee, a **HTML** evoluiu e, atualmente, está na **versão 5**.

Com HTML você define a estrutura de uma página, estabelecendo o que é título, texto, lista, subtítulo, local das imagens, link etc.

Para definir a estrutura da página, a HTML faz uso de **tags**, que são marcações de formatação. Aquilo que você vê quando abre uma página na internet é a interpretação que seu navegador faz do HTML, ou seja, o navegador interpreta todas as marcações, que são sinalizadas por meio de tags, que envolvem o texto, transformando-as em documento capaz de ser lido e entendido pelo usuário. A essa interpretação das marcações chamamos de **renderização**.

Uma das principais características da HTML é a facilidade de manipulação de textos e objetos como figuras, sons, fotos, animações e que tem por objetivo não apenas criar textos, mas hipertextos. Esses textos caracterizam-se por serem rápidos e pequenos, facilitando o acesso dos usuários da web.

## HTML

- HTML é uma linguagem para descrever páginas Web.
- HTML é a sigla de HyperText Markup Language.
- HTML não é uma linguagem de programação e sim uma linguagem de marcação.
- A linguagem de marcação apresenta um conjunto de tags.
- HTML utiliza as tags de marcação para descrever as páginas da web.

## Elementos HTML

Elemento HTML é a menor unidade de marcação. No W3C (World Wide Web Consortium) encontram-se todas as recomendações sobre a utilização dos elementos HTML, bem como os que estão em desuso.

Cada elemento tem uma especificação e o seu uso deve ser de acordo com a finalidade do elemento.

Temos, por exemplo, elementos destinados à marcação de títulos, elementos que têm a finalidade de marcar parágrafos e, assim por diante.

## Tags HTML

Os comandos HTML são chamados de tags HTML.

Os elementos de marcação dos diferentes conteúdos de um documento são representados por tags.

Assim como em outras linguagens, os comandos têm uma sintaxe própria, e seguem algumas regras:

As tags HTML são palavras que estão entre o sinal de "menor que" (<) e o sinal de "maior que" (>), por exemplo: **<p>**.

As tags HTML normalmente são utilizadas em pares, por exemplo: **<p>** e **</p>**.

A primeira tag do par é o início e a segunda é a finalização.

O início e o final das tags também são chamados, respectivamente, de **tags de abertura** e **tags de fechamento**.

O fechamento de uma tag é indicado por uma **"/**": **</p>**.

Algumas tags não necessitam de fechamento, como é o caso da tag **<br />**.

## Página HTML (aula01a.html)

```
<html>
  <head>
    <title>Título da página</title>
  </head>
  <body>
    <h1>Cabeçalho da seção</h1>
    <p>Parágrafo</p>
    <p>Quedra <br> de linha</p>
    <ul>
      <li>Primeiro elemento da lista</li>
      <li>Segundo elemento da lista</li>
    </ul>
  </body>
</html>
```

## Página HTML (aula01b.html)

```
<!DOCTYPE html>
<html lang="pt-br">
  <head>
    <meta charset="utf-8" />
    <title>Título da página</title>
  </head>
  <body>
    <h1>Cabeçalho da seção</h1>
    <p>Parágrafo</p>
    <p>Quedra <br /> de linha</p>
    <ul>
      <li>Primeiro elemento da lista</li>
      <li>Segundo elemento da lista</li>
    </ul>
  </body>
</html>
```



## Linhas de comentários

Linhas de comentários, têm a finalidade de facilitar o entendimento e manutenção de código.

Os comentários são pequenos trechos de textos que você pode utilizar para explicar o seu código.

Para inserir um comentário em HTML, utilizamos o marcador `<!--` para iniciar o comentário e `-->` para fechar o comentário.

Todo texto que estiver entre as marcações de comentário são ignorados pelo navegador, ou seja, esse texto não será renderizado.

Veja exemplo a seguir:

```
<!-- Isto é um comentário -->
```

```
<p>Texto de um parágrafo.</p>
```

## HTML e XHTML

HTML não faz diferença entre maiúsculas e minúsculas (não é "case sensitive"). Portanto, a notação <title> é equivalente a <TITLE> ou <TiTlE>, mas XHTML é case sensitive, isto é, faz diferença entre maiúsculas e minúsculas, portanto, **habitue-se desde já a escrever todas as tags em letras minúsculas**, assim você poderá utilizar tanto HTML quanto XHTML.

**XHTML** é a sigla em inglês para **eXtensible Hyper Text Markup Language**, que em português significa Linguagem Extensível para Marcação de Hipertexto. Trata-se de uma linguagem que utiliza regras de sintaxe muito mais rígidas que as regras para a HTML.

## Atributos

Os atributos são utilizados para fornecer informações adicionais para um elemento HTML e são declarados dentro da tag de abertura do elemento.

Observe a declaração:

```
<h1 style="color:blue">Cabeçalho azul</h1>
```

A sintaxe para se escrever um atributo é: nome do atributo seguido por um valor, que deve ser colocado entre aspas (simples ou duplas) e separado por um sinal de igual.

O texto em negrito acima informa ao navegador que as letras do cabeçalho deverão ser azul (blue).

## O que é e para que serve o CSS

Quando o World Wide Web Consortium (W3C) lançou a versão 4.0 da HTML, incorporou a ela as CSS (Cascading Style Sheets — Folhas de Estilos em Cascata), que têm a finalidade de dar uma forma de apresentação ao conteúdo do documento HTML. Passou a ser possível, portanto, **separar o conteúdo do documento de sua formatação**.

Antes de existirem as folhas de estilo, recorria-se a elementos e atributos da própria HTML para isso. Por exemplo, o elemento <font> definia a fonte, o tamanho e a cor do texto.

Os elementos e atributos específicos de formatação foram acrescentados gradativamente à especificação original da HTML pelas empresas criadoras de browsers.

As CSS oferecem uma maneira de descrever como o conteúdo deve ser apresentado, isto é, formatam o conteúdo, e controlam a apresentação do HTML.

## Benefícios do uso de CSS

CSS é uma revolução no mundo do web design. Os benefícios do uso de CSS incluem:

- Controle do layout de vários documentos a partir de uma simples folha de estilos.
- Maior precisão no controle do layout.
- Aplicação de diferentes layouts para servir diferentes mídias (tela, impressora etc.).
- Emprego de variadas, sofisticadas e avançadas técnicas de desenvolvimento.

## Como o CSS funciona

Suponha que desejemos uma cor de fundo verde para a página web:

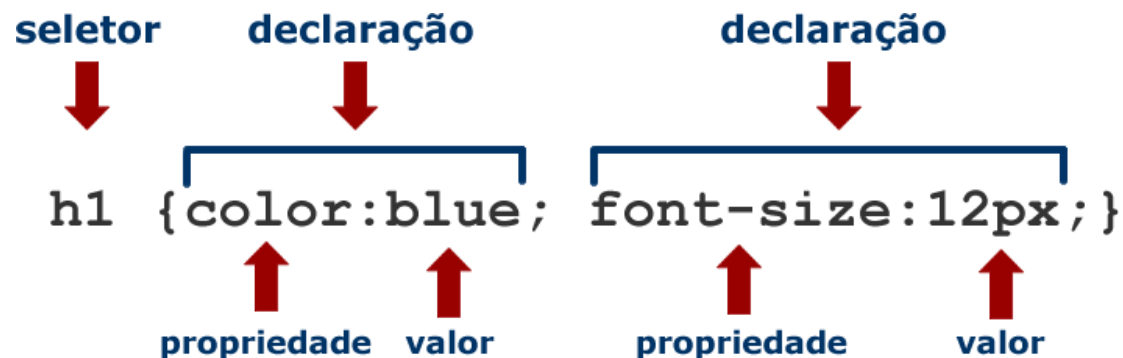
Usando HTML, podemos fazer assim:

```
<body bgcolor="#00ff00">
```

Com CSS, o mesmo resultado será obtido assim:

```
body {background-color: #00ff00;}
```

A sintaxe para CSS tem duas partes principais: um seletor e uma ou mais declarações.



## O que é para que serve o JavaScript

JavaScript é uma linguagem de programação criada pela Netscape em 1995, que a princípio se chamava LiveScript, para atender, principalmente, as seguintes necessidades:

- Validação de formulários no lado cliente (no navegador).
- Interação do usuário com a página.

Com JavaScript é possível incluir funções e aplicações online básicas em páginas da Web. O código JavaScript, pode ser incluído em uma página da Web juntamente ao código HTML.

Com JavaScript podemos criar efeitos especiais nas páginas e definir interatividade com o usuário. O navegador do cliente é o encarregado de interpretar as instruções JavaScript e executá-las.

JavaScript é uma linguagem de programação bastante simples e pensada para fazer as coisas com rapidez e leveza.

Pessoas que não tenham experiência prévia em programação poderão aprender esta linguagem com facilidade e utilizá-la com somente um pouco de prática.

## O que é o W3C

O W3C (World Wide Web Consortium), criado em outubro de 1994, é um consórcio internacional formado por empresas, instituições, pesquisadores, desenvolvedores e público em geral, com a finalidade de desenvolver a web a seu potencial máximo, criando normas e especificações que se aplicam aos mais diversos segmentos e setores da web, desde tecnologia e softwares até fabricantes e fornecedores.

O W3C tem como missão conduzir a World Wide Web para que atinja todo seu potencial, desenvolvendo protocolos e diretrizes que garantam seu crescimento a longo prazo.

No site <https://www.w3c.br>, você encontra um vasto material sobre especificações, padrões e normas, bem como guias de referências e tutoriais para o desenvolvimento e criação de documentos web.







Prof. MSc. Marcos Alexandruk

E-mail: alexandruk@uni9.pro.br

<https://github.com/alexandruk/desenvolvimentoparainternet>

# **Introdução para Programação à Internet**

## **aula 2**

## Estrutura de uma página HTML

Todo documento HTML, de acordo com os padrões do W3C, tem sua marcação mínima estruturada em três partes distintas:

1. Seção no documento para declarar a identificação como sendo um documento HTML.
2. Seção head definida como cabeça do documento HTML.
3. Seção body definida como corpo do documento HTML.

1	<code>&lt;html&gt;</code>
2	<code>&lt;head&gt;</code> <code>  &lt;title&gt; &lt;/title&gt;</code> <code>&lt;/head&gt;</code>
3	<code>&lt;body&gt;</code>  <code>&lt;/body&gt;</code>
1	<code>&lt;/html&gt;</code>

## Estrutura de uma página HTML

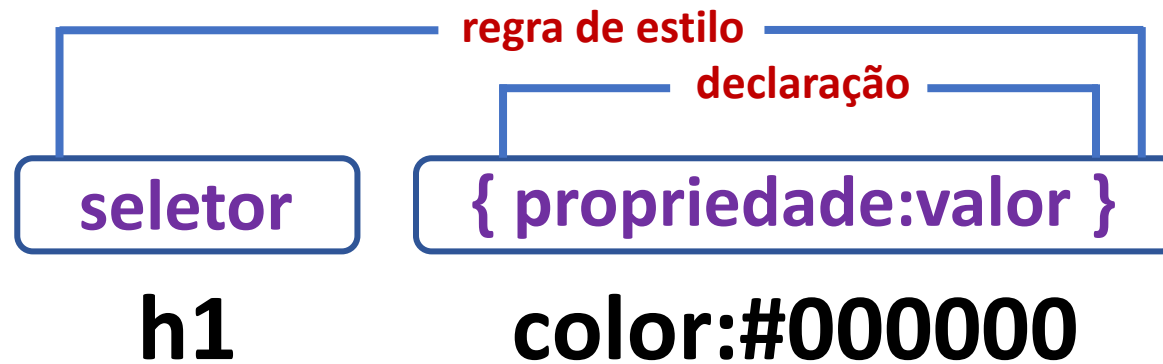
```
1.  <!--Informa ao navegador que é o início do html. É a tag de abertura-->
2.  <html>
3.    <!--Inicia a página. Aqui é determinada a cabeça da página-->
4.    <head>
5.      <!--Título da página que aparece na barra de título da janela do navegador.-->
6.      <!--A tag do elemento título está dentro da tag do elemento head-->
7.      <title>Programação para Internet</title>
8.    <!--Aqui finaliza a cabeça da página-->
9.    </head>
10.  <!--Tag de abertura do elemento body, ou seja o corpo da página-->
11.  <body>
12.    <!--Tudo que estiver entre as tags <body> e </body> é visível no navegador-->
13.    <!--Informa ao navegador onde começa e termina o cabeçalho de nível 1-->
14.    <h1>Bem-vindo(a) ao Curso de Programação para Internet</h1>
15.    <!--Informa ao navegador onde começa e termina o parágrafo-->
16.    <p>Esta é a minha primeira página Web.</p>
17.  <!--Tag de fechamento do elemento body-->
18.  </body>
19. <!--Informa o término da página. É a tag de fechamento-->
20. </html>
```

## CSS – Folha de Estilo em Cascata

Lembre-se:

- HTML é utilizado para estruturar conteúdos
- CSS é utilizado para formatar conteúdos

A regra de estilo ou marcação CSS é composta por três partes distintas: um seletor, uma propriedade e um valor para a propriedade.



## Formas de declarar folhas de estilo

Existem três formas de declarar as folhas de estilo:

- **inline:** as regras de estilo são escritas diretamente dentro da tag de abertura do elemento.
- **incorporado:** as regras de estilo são escritas dentro da seção head do documento e utilizando o elemento style. A folha de estilo fica dentro das tags `<style>` `</style>`.
- **externo:** as regras de estilo são escritas em um arquivo externo de texto com a extensão .css. Esse arquivo externo é "linkado" ao arquivo HTML por meio de uma declaração contida no HTML.

## Folhas de estilo inline

As regras de estilo são escritas diretamente dentro da tag de abertura do elemento:

```
<html>
  <head></head>
  <body>
    <p style="font-size: large; color: green;">Parágrafo estilizado - CSS inline</p>
  </body>
</html>
```



## Folhas de estilo **incorporado**

As regras de estilo são escritas dentro da seção head do documento e utilizando o elemento style. A folha de estilo fica dentro das tags <style> </style>.:

```
<html>
  <head>
    <style type="text/css">
      p {color:blue; font-family:Georgia; font-size:25px;}
    </style>
  </head>
  <body>
    <p>Parágrafo estilizado - CSS incorporado</p>
  </body>
</html>
```

## Folhas de estilo externo

As regras de estilo são escritas em um arquivo externo de texto com a extensão .css. Esse arquivo externo é "linkado" ao arquivo HTML por meio de uma declaração contida no HTML :

```
/*style.css*/  
p {  
    color: red;  
    font-family: Georgia;  
    font-size: 20px;  
}
```

```
<html>  
  <head>  
    <link rel="stylesheet" type="text/css" href="style.css" />  
  </head>  
  <body>  
    <p>Parágrafo estilizado - CSS externo</p>  
  </body>  
</html>
```

## Folhas de estilo **importadas**

Nesse método, declara-se a folha de estilo externa utilizando a diretiva **@import** dentro do elemento *style* na seção **head** do documento HTML.

A diretiva **@import**, ela deverá vir antes de todas as declarações de folha estilo para o documento.

```
/*fontes.css*/
p {
    color: green;
    font-family: Georgia;
    font-size: 30px;
}

/*main.css*/
@charset utf-8;
@import url("fontes.css");
body {
    margin: 50;
    background: #00ffff;
}

<html>
  <head>
    <style rel="stylesheet" type="text/css">
      @import url("main.css");
    </style>
  </head>
  <body>
    <p>Parágrafo estilizado - CSS importado</p>
  </body>
</html>
```

## Referências

MARCONDES, C. A. HTML 4.0 Fundamental: A Base da Programação para Web. São Paulo: Editora Érica, 2005.

OLIVIERO, C. A. Faça um site JavaScript - Orientado por Projeto: scripts baseados em objetos. São Paulo: Editora Érica, 2001.

SILVA, M. S. Criando Sites com HTML - Sites de Alta Qualidade com HTML e CSS. São Paulo: Novatec Editora, 2008.





Prof. MSc. Marcos Alexandruk

E-mail: alexandruk@uni9.pro.br

<https://github.com/alexandruk/desenvolvimentoparainternet>

## JavaScript e tags meta

### aula 3

## Iniciando com JavaScript

JavaScript é uma linguagem de programação utilizada com o HTML, com a finalidade de tornar as páginas Web dinâmicas e que possibilitem uma interatividade com o usuário.

O HTML fornece a estrutura da página, o CSS adiciona o estilo, e o JavaScript torna a página dinâmica, podendo exibir mensagens para o usuário, validar campos em formulários, manipular datas, detectar o navegador, entre outras coisas.

Com JavaScript podemos criar pequenos programas denominados scripts, que são inseridos em um documento HTML.

Para inserir um script em um documento HTML precisamos do par da tag **<script> </script>**.

Um script, escrito na linguagem JavaScript, pode ser inserido em duas áreas de um documento HTML, entre as tags **<head> </head>**, ou entre as tags **<body> </body>**.

Seja qual for o local escolhido para inserir o script, ele deve sempre iniciar com **<script language="JavaScript">** e terminar com **</script>**.



## Iniciando com JavaScript

```
<html>
  <head>
    <title>Primeira pagina com JavaScript</title>
  </head>
  <body>
    <script language="JavaScript">
      document.write("Texto impresso com uma função JavaScript");
    </script>
  </body>
</html>
```

## Iniciando com JavaScript

O código JavaScript, inicia na tag `<script language="JavaScript">` e informa ao navegador, que neste ponto começa o script utilizando a linguagem JavaScript. A linha seguinte contém o objeto **document** seguido pelo método **write( )**, cuja função é escrever a frase "Exemplo da utilização de JavaScript". A tag `</script>` finaliza o código.

No JavaScript cada instrução é finalizada com ponto-e-vírgula (;).

JavaScript é case-sensitive, ou seja, é sensível a letras maiúsculas e minúsculas: "A" é diferente de "a".

A sintaxe a seguir demonstra a inclusão de linhas ou blocos de comentários em JavaScript.

```
// Este é um comentário de linha
```

```
/*Aqui estamos utilizando um bloco de comentário  
   em mais de uma linha */
```

## Como inserir arquivo JavaScript externo em um documento HTML

JavaScript também pode ser escrito em um arquivo separado do HTML com a extensão **.js**, por exemplo, **meujavascript.js**, que contém o código que poderá ser utilizado em diferentes páginas Web.

O arquivo externo é chamado no HTML, utilizando o atributo **"src"** na tag `script`.

O atributo **"src"** indica qual o arquivo JavaScript deverá ser utilizado.

```
<head>  
  <script type="text/javascript" src="meujavascript.js">  
  </script>  
</head>
```

O arquivo **.js** deve conter somente instruções JavaScript, não podendo constar dele nenhum código HTML, nem mesmo as tags `<script>` e `</script>`.

## Como inserir arquivo JavaScript externo em um documento HTML

```
/*meujavascript.js*/
document.write("Exemplo da utilizacao de JavaScript ");

<!--aula03b.html-->
<html>
  <head>
    <title>Segunda pagina com JavaScript</title>
  </head>
  <body>
    <script type="text/javascript" src="meujavascript.js"></script>
  </body>
</html>
```

## Elemento **meta**

O elemento meta tem como principal objetivo fornecer informações adicionais sobre o documento HTML.

Para cada tipo de informação utilizamos um atributo que pode ser inserido na seção **head** da página.

Alguns são de uso generalizado e servem para qualquer tipo de página, outros são bastante específicos.

O elemento **meta** destina-se a **fornecer informações sobre o documento**. As informações contidas nele são chamadas de **metadados**.

Metadados incorporados ao código HTML são, na verdade, estruturas de dados sobre os próprios dados, uma breve descrição do conteúdo da página, seu autor, data de criação, linguagem e outras informações relevantes.

Alguns sistemas de busca dão aos conteúdos das tags meta uma forte ênfase no ranking dos sites, a maioria deles indexa os dados das meta tags **description** e **keywords** como sumários da página.

Se essas tags forem usadas correta e racionalmente, elas podem aumentar a relevância nos resultados de busca, o que é vantajoso tanto para o proprietário do site quanto para os usuários.

## Sintaxe e exemplos do elemento **meta**

A sintaxe para escrever um elemento meta consiste em duas partes: **name** ou **http-equiv** define um nome para o metadado e **content** define o conteúdo.

```
meta name="author" content="Fulano de Tal" /
```

Define o autor do documento.

```
meta http-equiv="Content-Type" content="text/html; charset=UTF-8" /
```

Informa que o tipo de conteúdo do documento é texto HTML e a codificação de caracteres é a UTF-8.

```
meta name="language" content="pt-br" /
```

Informa que o idioma principal do documento é o português do Brasil.

```
meta name="description" content="Aulas de HTML, CSS e JavaScript" /
```

Descreve o conteúdo principal do documento. É utilizada pelos mecanismos de busca para descrever o link para a página no resultado de uma busca.

```
name="description" content="none" /
```

Usada para deixar por conta do mecanismo de busca a escolha da frase que irá aparecer no resultado da busca.

```
meta name="keywords" content="css, html, xhtml, javascript" /
```

Keywords usadas por alguns motores de busca para indexar os documentos com informações encontradas em title e body. As frases ou palavras devem ser separadas por vírgulas.

```
meta http-equiv="pragma" content="no-cache" /
```

Faz com que o navegador não armazene a página em cache.

```
meta http-equiv="Content-Script-Type" content="text/javascript" /
```

Informa que a linguagem para scripts inline é a JavaScript.

```
meta http-equiv="Content-Style-Type" content="text/css" /
```

Informa que a linguagem utilizada para estilos inline é CSS.

## Robots

Especifica informações de indexação para os robôs de busca, suporta os seguintes valores:

**all:** valor default, sem restrições para a indexação, o robô de busca não recebe nenhuma informação.

**index:** os robôs de busca podem incluir a página normalmente.

**follow:** robôs podem indexar a página e ainda seguir os links para outras páginas que ela contém.

**noindex:** os links podem ser seguidos, mas a página não é indexada.

**nofollow:** a página é indexada, mas os links não são seguidos.

**none:** os robôs podem ignorar a página.

**noarchive:** (apenas Google): a página não é arquivada.

## Alguns exemplos utilizando robots

Os valores "**index**" e "**noindex**" se referem ao tratamento da página inicial: se o buscador deve ou não incluí-la nos resultados, respectivamente. Já os valores "**follow**" e "**nofollow**" se referem aos links da página inicial, se eles devem ser visitados e indexados ou não.

```
<meta name="robots" content="all" />
```

Não há restrições para a indexação. É o valor padrão e não terá efeito se for listada explicitamente.

```
<meta name="robots" content="index,follow">
```

Indexa a página inicial e todas as páginas nela referenciadas.

```
<meta name="robots" content="noindex,follow">
```

Não indexa a página inicial, mas indexa as páginas nela referenciadas.

```
<meta name="robots" content="index,nofollow">
```

Indexa a página inicial, mas nenhum link que ela contenha.

```
<meta name="robots" content="noindex,nofollow">
```

Não indexa nem a página inicial e nem seus links.

```
<meta name="robots" content="noarchive">
```

Evita que os sites de busca indexem o site.





Prof. MSc. Marcos Alexandruk

E-mail: alexandruk@uni9.pro.br

<https://github.com/alexandruk/desenvolvimentoparainternet>

## Utilização de cores, dos elementos div e span

### aula 4

## Cores

As cores possuem um papel muito importante na composição de páginas web.

A linguagem de marcação (X)HTML tem uma sintaxe própria para defini-las.

Cores podem ser declaradas com uso de atributos ou com folhas de estilo.

As cores são indicadas em valores RGB, ou seja, para conseguir uma cor qualquer misturaremos quantidades de vermelho (Red), verde (Green) e azul (Blue).

Para declarar as cores podemos utilizar a sintaxe hexadecimal, sintaxe RGB ou sintaxe por palavra-chave.

## Sintaxe hexadecimal

A utilização da sintaxe hexadecimal é muito utilizada pelos desenvolvedores.

Os valores RGB são indicados em numeração hexadecimal, em base 16, ou seja, os dígitos vão de 0 até 16. Como não existem tantos dígitos numéricos, utilizam-se as letras de A a F. Veja no quadro, a seguir, a numeração no sistema hexadecimal correspondente ao sistema decimal.

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F

## Sintaxe hexadecimal

Na sintaxe hexadecimal a cor é definida por uma sequência de seis caracteres hexadecimais precedidos por #.

A sintaxe não é case sensitive, ou seja, não faz diferença entre letras maiúsculas ou minúsculas.

Para conseguir uma cor misturamos os valores na forma: #RRGGBB, em que cada valor pode variar de 00 até FF.

Quando temos todas as cores (R,G,B) obtemos a cor branca, que é representada por **#FFFFFF**.

A ausência de cores, o preto, é representado por **#000000**.

Para a cor vermelha usamos **#FF0000**.

Para o verde usamos **#00FF00**.

Para o azul usamos **#0000FF**.

Podemos também, quando possível, utilizar a sintaxe abreviada para as cores declarando somente três valores:

**#FFF** equivale a **#FFFFFF**.

**#0F0** equivale a **#00FF00**.

**#F6C** equivale a **#FF66CC**.

## Sintaxe RGB

Essa sintaxe, menos utilizada, define a cor por uma lista de três números colocados entre parênteses e separados por vírgula, precedida da sigla rgb. Existem duas formas de escrever a lista dos três números.

A primeira utiliza números entre **0** e **255**:

**rgb(125, 220, 90)**

**rgb(0, 75, 250)**

A segunda utiliza valores em porcentagem de **0%** a **100%**:

**rgb( 10%, 90%, 45%)**

**rgb(0, 35%, 82%)**

Não é permitido misturar números com porcentagem para compor a cor.

## Sintaxe por palavra-chave

Podemos também utilizar o nome da cor em inglês, porém, de acordo com o W3C, na CSS2 somente 16 cores eram válidas para efeito de marcação (conforme tabela abaixo). Na CSS3, a lista de cores suportadas foi estendida para 147 palavras-chave (disponível em: [https://www.w3schools.com/colors/colors\\_names.asp](https://www.w3schools.com/colors/colors_names.asp))

Black	Purple
White	Fuchsia
Red	Lime
Green	Olive
Blue	Yellow
Silver	Navy
Gray	Teal
Maroon	Aqua

## Cores compatíveis em todos os sistemas

Como as páginas web têm de ser vistas por todos os usuários, e os sistemas que eles utilizam são diferentes, é recomendável utilizar cores compatíveis com a paleta de cores de todos eles.

Uma forma de conseguir isso é limitar a combinação de cores utilizando os valores da seguinte forma:

Utilizar os valores
00
33
66
99
CC
FF

Podemos utilizar, por exemplo, as seguintes combinações: **#3366FF** - **#FF9900** - **#666666**



## Atributo **id**

O atributo **id** é um identificador único e destina-se a atribuir um nome identificador para o elemento. Em um mesmo documento, o nome usado para identificar um elemento deve ser único, ou seja, não podemos atribuir o mesmo nome de um id para outros elementos.

O valor do atributo **id** é um nome que você pode escolher. Devemos respeitar algumas regras para a escolha do nome: não é válido começar o nome de um atributo com um número ou hífen. Já o caractere underscore ( \_ ) é válido. Na prática utilizam-se nomes que começam com letras minúsculas.

A utilização do atributo **id** é útil em consequência dos programas, scripts e folhas de estilo terem a capacidade de entender essa marcação no HTML, conseguindo selecionar os elementos pelo seu identificador.

## Atributo id

A declaração é feita na seção **head**, os elementos encontram-se no **body** e utilizam o **id** declarado.

```
<html>
  <head>
    <title>Atributo Id</title>
    <style type="text/css">
      /*Declaração dos atributos id*/
      p#destaque{
        font-family: sans-serif; /*indica qual a fonte utilizada*/
        font-style: italic; /*indica o estilo da fonte: itálico*/
        font-size: 30px; /*tamanho da fonte: 30 pixels*/
        text-align: left; /*alinhamento do texto: esquerda*/
      }
    </style>
  </head>
  <body>
    <p id="destaque">Paragrafo com atributo id "destaque"</p>
  </body>
</html>
```

## Atributo **class**

O atributo **class** destina-se a atribuir uma classe identificadora para o elemento.

Também é um atributo identificador, porém a diferença dele e do atributo **id** é que com o atributo **class** podemos, em um mesmo documento, utilizar o nome da classe para identificar várias instâncias de um mesmo elemento e também elementos diferentes.

Uma folha de estilo é capaz de identificar os diferentes elementos identificados com a mesma classe e estilizar todos eles de acordo com a declaração dela.

## Atributo **class**

Veja a sintaxe e como declarar um atributo class.

```
<html>
  <head>
    <title>Atributo class</title>
    <style type="text/css">
      /*Declaração do atributo class*/
      p.fonte1{
        font-size:20px; /*tamanho da fonte: 20 pixels*/
        color: #ff0000; /*cor do texto: verde*/
      }
      p.fonte2{
        font-size:30px; /*tamanho da fonte: 30 pixels*/
        color: #00ff00; /*cor do texto: verde*/
      }
    </style>
  </head>
  <body>
    <p class="fonte1">Texto do parágrafo marcado com class fonte1.</p>
    <p class="fonte2">Texto do parágrafo marcado com class fonte2.</p>
  </body>
</html>
```

## Elementos **span** e **div**

Os elementos **div** e **span**, em conjunto com os atributos **id** e **class**, tornam-se um poderoso mecanismo para estruturar e estilizar conteúdos.

Esses elementos definem conteúdo para ser utilizado **inline** (**span**) ou em nível de **bloco** (**div**).

Devem ser utilizadas com folhas de estilo em cascata (CSS).

## Elemento **span**

Utiliza-se o elemento **span** com o atributo **class** para marcar conteúdos **em linha** genericamente.

```
<html>
  <head>
    <title>Utilizando span</title>
    <style type="text/css">
      span.azul{
        color: #0000FF; /*cor do texto: azul*/
      }
    </style>
  </head>
  <body>
    <p>A Terra é <span class="azul">azul</span>, disse Gagarin.</p>
  </body>
</html>
```

## Elemento **div**

Utiliza-se o elemento **div** com o atributo **class** para estruturar e estilizar blocos de conteúdos.

```
<html>
  <head>
    <style>
      .azulBox {
        background-color: #0000ff; /*cor do fundo: azul*/
        text-align: center; /*alinhamento do texto: centro*/
        font-size: 30px; /*tamanho da fonte: 20 pixels*/
        color: #ffffff; /*cor da fonte: branco*/
      }
    </style>
  </head>
  <body>
    <div class="azulBox">
      <p>Parágrafo dentro da div azulBox</p>
    </div>
    <p>Parágrafo fora da div azulBox</p>
  </body>
</html>
```



Prof. MSc. Marcos Alexandruk

E-mail: alexandruk@uni9.pro.br

<https://github.com/alexandruk/desenvolvimentoparainternet>



## **Padrões de medidas e formatação de conteúdos com o atributo style**

### **aula 5**

## Box model

Podemos entender como boxes os blocos formados por um dos elementos HTML quando renderizados em uma tela.

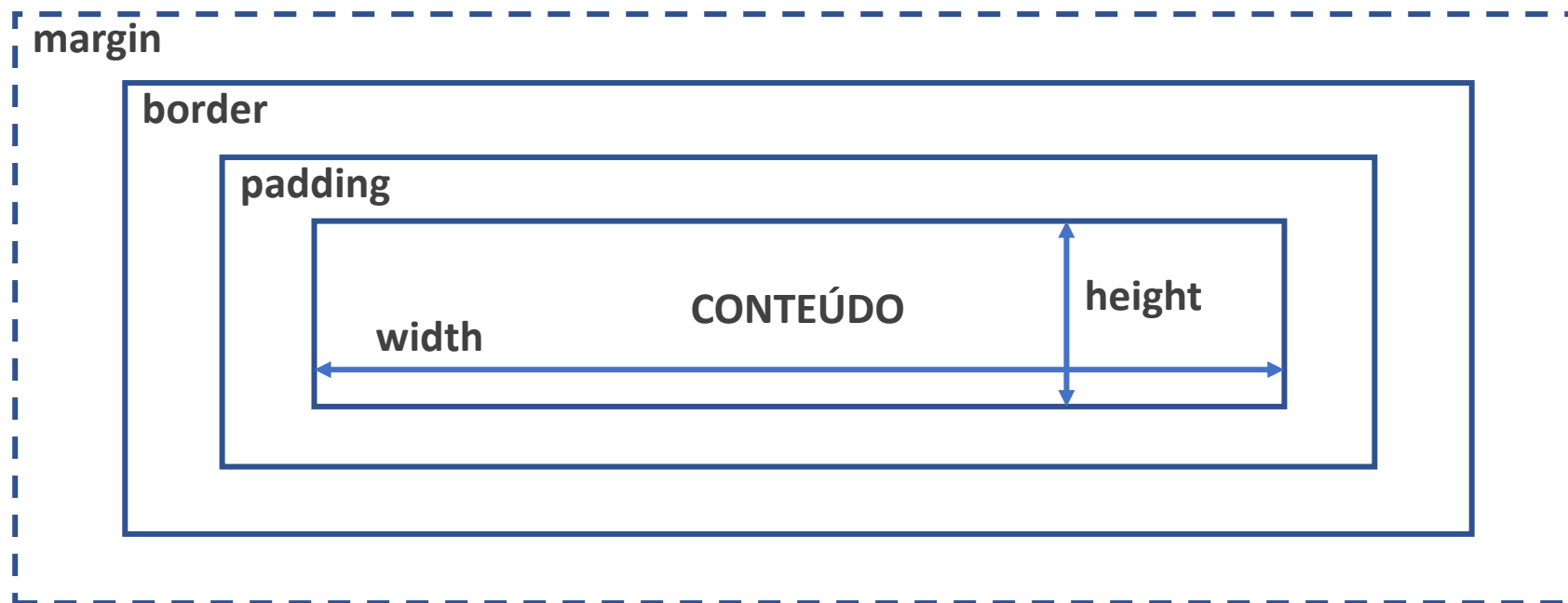
Por exemplo, um parágrafo contendo um texto ocupa na tela um retângulo com uma largura e uma altura bem definida, que é fácil de visualizar.

Box Model é um modelo-padrão de renderização ou apresentação visual de um box de acordo com a formatação CSS.

## Box model

Observe na figura abaixo que existe uma área denominada de conteúdo e suas dimensões são definidas pelas propriedades CSS **width** e **height**. Depois temos a área de enchimento, em que a espessura é definida pela propriedade **padding**. Ainda, em volta do enchimento, temos uma borda em que sua espessura e cor são definidas na propriedade **border**. Por último, existe um espaço chamado de margem e sua espessura é definida na propriedade **margin**. A área da margem é sempre transparente.

A propriedade **background** define o fundo a ser aplicado nas áreas de conteúdos, de enchimento e da borda.



## Box model

Para definir corretamente a altura e largura de um elemento precisamos saber como o box model trabalha.

Note que quando você especificar as propriedades de altura e largura de um elemento com CSS, você está apenas definindo a altura e largura da área do conteúdo. Para saber o tamanho total do elemento, você deverá adicionar o enchimento, a borda e a margem.

O total do width de um elemento sempre deverá ser calculado da seguinte forma:

**width total do elemento = width + left padding + right padding + left border + right border + left margin + right margin**

O total do height de um elemento sempre deverá ser calculado da seguinte forma:

**height total do elemento = height + top padding + bottom padding + top border + bottom border + top margin + bottom margin**

Veja as seguintes definições de um determinado elemento:

**width:250px; padding:10px; border:5px solid gray; margin:10px;**

250px do width;

+ 20px do padding (10px para a direita e 10px para a esquerda);

+ 10px de border (5px para a direita e 5px para a esquerda);

+ 20 px de margin (10px para a direita e 10px para a esquerda).

**300px** comprimento total do elemento

## Unidades de medida em CSS

As unidades de medida são utilizadas em propriedade CSS que controlam dimensões em geral. Normalmente se utiliza em CSS a unidade de medida em pixel (px) ou o centímetro (cm), que raramente é utilizado. Há unidades de medida absolutas e relativas.

sigla	unidade absolutas
in	Polegadas
cm	Centímetros
mm	Milímetros
pt	Ponto. Unidade de medida tipográfica (1 pt é igual a 1/72 polegadas)
pc	Paica. Unidade de medida tipográfica (1 pc é igual a 12 pontos)
sigla	unidade relativas
em	1 em é igual ao tamanho da fonte definido para o elemento. O em é uma unidade muito útil em CSS, uma vez que pode se adaptar ao tipo de fonte utilizada.
ex	1 ex é igual à altura da letra x minúscula da fonte definida para o elemento. Não é comum a utilização dessa medida.
px	Pixel. Um ponto na tela do computador.
%	Porcentagem. Calculada em relação a um valor preexistente, normalmente, uma unidade de medida.

## Propriedade margin (margem)

A propriedade **margin** define uma margem em volta dos boxes. O valor declarado pode ser qualquer unidade CSS de medida.

Propriedade	Descrição	Declaração
margin	Define todas as margens na declaração	margin: 20px 30px 5px 10px;
margin-top	Define a margem superior	margin-top: 20px;
margin-right	Define a margem direita	margin-right: 30px;
margin-bottom	Define a margem inferior	margin-bottom: 5px;
margin-left	Define a margem esquerda	margin-left: 10px;

## Propriedade margin (margem)

As propriedades CSS padding, font, background, margin, border, border-width, borderstyle, border-color e outline admitem a sintaxe abreviada, que consiste em declarar uma lista de valores separados por espaço, como apresentado no quadro (margin: 20px 30px 5px 10px;).

A ordem em que os valores são escritos na lista corresponde aos lados superior, direito, inferior e esquerdo, respectivamente.

Veja também outras opções para declaração abreviada:

**margin: 20px;** /\* margem de 20px nos quatro lados \*/

**margin: 15px 10px;** /\* margem superior e inferior de 15px e direita e esquerda de 10px \*/

**margin: 20px 10px 15px;** /\* margem superior de 20px, margens direita e esquerda de 10px e margem inferior de 15px \*/

## Propriedade padding (espaçamento interno)

A propriedade **padding** define um espaçamento interno entre o conteúdo do box e seus limites. O valor declarado pode ser qualquer unidade CSS de medida.

Propriedade	Descrição	Declaração
padding	Define todos os espaçamentos na declaração	padding: 10px 5px 25px 12px;
padding-top	Define o espaçamento superior	padding-top: 20px;
padding-right	Define o espaçamento direito	padding-right: 30px;
padding-bottom	Define o espaçamento inferior	padding-bottom: 5px;
padding-left	Define o espaçamento esquerdo	padding-left: 10px;



## Propriedade background (fundo)

A propriedade background define as características de fundo do elemento. Com essa propriedade podemos definir a cor, imagens e seu posicionamento no fundo do elemento.

Veja no quadro as principais definições que podemos utilizar.

Propriedade	Descrição
background	Define todas as propriedades de fundo na declaração
background-attachment	Define se uma imagem ficará fixa ou se irá rolar em relação à área de renderização em que foi colocada
background-color	Define uma cor para o fundo de um elemento
background-image	Define uma imagem de fundo de um elemento
background-position	Define uma posição inicial para a imagem de fundo
background-repeat	Define como a imagem se repetirá no fundo do elemento
background-size	Define o tamanho da imagem de fundo

## Exemplo

```
<p>Parágrafo sem especificação de margem</p>
```

```
<p style="background-color: ffff00; margin-top:100px; margin-bottom:100px; margin-right: 50px; margin-left:50px;"> Parágrafo com especificação de margem</p>
```

```
<p style="border-style= solid; border-color: #f00; padding: 10px ; margin: 20px;">Parágrafo com padding, border e margin</p>
```

```
<p>Parágrafo normal, sem padding</p>
```

```
<p style="padding-left: 2cm;">Parágrafo com padding-left de 2cm</p>
```

```
<p style="padding-left: 50%;">Parágrafo com padding-left de 50 %</p>
```

## Exercício

Aplicar os estilos acima através de folhas de estilo (CSS) externas:

- a) com seletor de classe
- b) com seletor de id

## Seletor de **classe** aplicado a qualquer elemento

Os seletores de **classe** também podem ser definidos sem colocarmos o nome de um elemento no início da definição. Quando isso acontece as definições podem ser aplicadas a qualquer elemento.

```
.a {  
    background-color: #ffff00;  
    margin: 50px;  
    color: #ff0000;  
}  
  
<html>  
    <head>  
        <link rel="stylesheet" type="text/css" href="style05d.css" />  
    </head>  
    <body>  
        <h1>Título sem especificação</h1>  
        <h1 class="a">Título sem especificação</h1>  
        <p>Parágrafo sem especificação</p>  
        <p class="a">Parágrafo com especificação</p>  
    </body>  
</html>
```

## Seletor de **id** aplicado a qualquer elemento

Se quisermos que a regra se aplique a qualquer **elemento** que tenha determinado, para1 basta escrevê-la na forma seguinte:

```
*#a {  
    background-color: #ffff00;  
    margin: 50px;  
    color: #ff0000;  
}  
  
<html>  
    <head>  
        <link rel="stylesheet" type="text/css" href="style05d.css" />  
    </head>  
    <body>  
        <h1>Título sem especificação</h1>  
        <h1 id="a">Título sem especificação</h1>  
        <p>Parágrafo sem especificação</p>  
        <p id="a">Parágrafo com especificação</p>  
    </body>  
</html>
```

## Exemplo



[aula05g.html](#)

## Validador W3C

Existem os padrões recomendados pelo W3C para criação e desenvolvimento de páginas Web, mas como saber se nossa página está de acordo com esses padrões?

É aí que entra o órgão para validar nosso trabalho, o Markup Validation Service: <http://validator.w3.org>.

Essa ferramenta de validação é on-line e nos ajuda a corrigir erros que possam existir na criação da nossa página web.

A cada passo do desenvolvimento de um documento para web, você deve validar sua marcação. Ficará mais fácil corrigir cada etapa do que deixar para validar o site todo de uma só vez.

O validador HTML verifica se a marcação do documento está de acordo com a sintaxe prevista para a versão HTML ou XHTML, devendo constar no documento o DOCTYPE, que contém as informações sobre a versão e sintaxe que será avaliada.

Existem três maneiras de validar um documento:

- **Validate by URI:** informa-se o endereço URL da página para documentos na internet;
- **Validate by File Upload:** para validar documentos que estão no seu HD informando o local do documento;
- **Validate by Direct Input:** Copie e cole a marcação no campo texto.

Adquira o hábito de validar sua marcação, assim sempre facilitará o seu trabalho de desenvolvimento.



Prof. MSc. Marcos Alexandruk

E-mail: alexandruk@uni9.pro.br

<https://github.com/alexandruk/desenvolvimentoparainternet>

## Formatação de fontes, seletores e hierarquia no CSS

### aula 6



## Font

A propriedade **font** da folha de estilo define a família de fontes, tamanho, estilo do texto, ou seja, definimos as características gerais das fontes.

## font-family

A propriedade **font-family** especifica uma família de fontes (como "Times New Roman" ou "Arial").

Exemplo com CSS incorporado e inline ([aula06a.html](#)):

```
</html>
<head>
  <title>Utilizando font-family</title>
  <style type="text/css">
    p { font-family: Arial, 'Lucida Sans', Verdana, Sans-Serif; }
  </style>
</head>
<body>
  <p>Usando font-family - CSS Incorporado</p>
  <p style="font-family: Verdana, Arial;">Usando font-family - CSS Inline</p>
</body>
</html>
```

## font-style

A propriedade **font-style** especifica o estilo da fonte de um texto: normal ou italic.

Exemplo com CSS incorporado ([aula06b.html](#)):

```
</html>
<head>
  <title>Utilizando font-style</title>
  <style type="text/css">
    p.normal {font-style:normal;}
    p.italic {font-style:italic;}
  </style>
</head>
<body>
  <p class="normal">Usando font-style normal</p>
  <p class="italic">Usando font-style italic</p>
</body>
</html>
```

## font-size

A propriedade font-size define o tamanho da fonte para diferentes elementos.

Exemplo com CSS incorporado ([aula06c.html](#)):

```
</html>
<head>
  <title>Utilizando font-size</title>
  <style type="text/css">
    p.large {font-size:large;}
    p.medium {font-size:medium;}
    p.oitenta {font-size:80%;}
  </style>
</head>
<body>
  <p class="large">Usando font-size large</p>
  <p class="medium">Usando font-size medium</p>
  <p class="oitenta">Usando font-size 80%</p>
</body>
</html>
```

Tamanhos absolutos:

medium (default)

xx-small

x-small

small

large

x-large

xx-large

Tamanhos relativos:

smaller

larger

% (100% = medium)

Outros tamanhos:

px

cm

## font-weight

A propriedade font-weight define espessura dos caracteres (normal ou bold).

Exemplo com CSS incorporado ([aula06d.html](#)):

```
</html>
<head>
  <title>Utilizando font-weight</title>
  <style type="text/css">
    p.normal {font-weight:normal;}
    p.bold {font-weight:bold;}
  </style>
</head>
<body>
  <p class="normal">Usando font-weight normal</p>
  <p class="bold">Usando font-weight bold</p>
</body>
</html>
```

## font-family; font-style; font-size; font-weight

A propriedade font-weight define espessura dos caracteres (normal ou bold).

Exemplo com CSS incorporado ([aula06e.html](#)):

```
</html>
<head>
  <title>Utilizando font-weight</title>
  <style type="text/css">
    p.todos {
      font-family: verdana, sans-serif;
      font-style: italic;
      font-size: 20px;
      font-weight: bold;
    }
  </style>
</head>
<body>
  <p class="todos">Usando font family, style, size e weight</p>
</body>
</html>
```

## font-variant

Uma **font-variant** na qual foi atribuído o valor **small-caps**, todas as letras minúsculas são convertidas em maiúsculas. No entanto, as letras convertidas para maiúsculas aparecem em um tamanho de fonte menor do que as letras maiúsculas originais.

Exemplo com CSS incorporado ([aula06f.html](#)):

```
<html>
  <head>
    <style>
      p.normal {font-variant: normal;}
      p.small {font-variant: small-caps;}
    </style>
  </head>
  <body>
    <h1>A propriedade font-variant</h1>
    <p class="normal">Parágrafo normal</p>
    <p class="small">Parágrafo small</p>
  </body>
</html>
```

## !important

Uma declaração de estilo com **!important** ignora qualquer hierarquia e prevalece sobre todas as demais, é a de mais alta prioridade.

Exemplo com CSS incorporado com **!important** ([aula06g.html](#)):

```
</html>
<head>
  <title>Utilizando font-family</title>
  <style type="text/css">
    p { font-family: Arial !important; }
  </style>
</head>
<body>
  <p style="font-family: Verdana;">Usando font-family - CSS Inline</p>
</body>
</html>
```

### Hierarquia CSS

!important

inline

incorporado (no <head> da página)

externo (folha de estilo externa)



## Google Fonts

Google Fonts é uma biblioteca com mais de 800 fontes.

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

Exemplo com CSS externo ([aula06h.html](#)):

```
<html>
  <title>Google Fonts</title>
  <link rel="preconnect" href="https://fonts.gstatic.com">
  <link href="https://fonts.googleapis.com/css2?family=Courgette" rel="stylesheet">
  <style>
    body {font-family: 'Courgette', cursive;}
  </style>
  <body>
    <h1>Título com fonte Courgette</h1>
    <p> Parágrafo com fonte Courgette</p>
  </body>
</html>
```

## @font-face

A diretiva **@font-face** permite importar uma fonte a partir de um arquivo externo e utilizá-la como uma fonte nativa nas páginas HTML. A fonte Ranchers-Regular está disponível em: <https://fonts.google.com/>

Exemplo com a diretiva **@font-face** ([aula06i.html](#)):

```
</html>
<head>
  <title>Utilizando font-weight</title>
  <style type="text/css">
    @font-face {
      font-family: Ranchers-Regular;
      src: url(Ranchers-Regular.ttf)
    }
    p {
      font-family: Ranchers-Regular;
    }
  </style>
</head>
<body>
  <p>Usando font-family Ranchers-Regular</p>
</body>
</html>
```

## Seletor classe

Esse tipo de seletor utiliza o atributo **class** para criar um identificador que possa ser utilizado como seletor e, dessa forma, os elementos que tiverem a classe criada serão estilizados de acordo com as regras criadas.

```
<style>
  h1.inicio {font-family: Verdana, Sans-Serif;}
  p.principal { background:red;}
</style>
...
<body>
  <h1 class="inicio"> A fonte do texto deste cabeçalho será Verdana</h1>
  <p class="principal"> A cor de fundo deste parágrafo será vermelha</p>
</body>
```

O identificador **class** não precisa ser único na marcação de um documento, podemos atribuir a mesma classe para mais de um elemento, inclusive para elementos diferentes dentro de uma mesma página.

```
<style>
  .principal {color: #ff0000;}
</style>
...
<body>
  <h2 class="principal"> O texto deste cabeçalho será na cor vermelha</h2>
  <p class="principal"> O texto deste parágrafo será na cor vermelha</p>
</body>
```

## Seletor id

Esse tipo de seletor utiliza o atributo id para criar um identificar único que possa ser utilizado como seletor e, dessa forma, todos os elementos que possuírem o id criado serão estilizados conforme as regras declaradas.

```
<style>
  h1#inicio {font-family: Verdana, Sans-Serif;}
  p#principal { background:red;}
</style>
...
<body>
  <h1 id="inicio"> A fonte do texto deste cabeçalho será Verdana</h1>
  <p id="principal"> A cor de fundo deste parágrafo será vermelha</p>
</body>
```

O identificador **id** deve ser único, não podendo nem mesmo ser utilizado o mesmo nome dado ao id para mais de um elemento diferente.

```
<style>
  #principal {color: #ff0000;}
</style>
...
<body>
  <h3 id="principal"> A cor de fundo deste cabeçalho será vermelha </h3>
</body>
```

## Exercício

Aplique as diversas opções CSS **font** (font-family; font-style; font-size; font-weight) no poema abaixo (incluir o título e o autor).  
Use um estilo diferente para cada estrofe.

### **Dorme, que a vida é nada**

*Fernando Pessoa*

Dorme, que a vida é nada!  
Dorme, que tudo é vão!  
Se alguém achou a estrada,  
Achou-a em confusão,  
Com a alma enganada.

Não há lugar nem dia  
Para quem quer achar,  
Nem paz nem alegria  
Para quem, por amar,  
Em quem ama confia.

Melhor entre onde os ramos  
Tecem dosséis sem ser  
Ficar como ficamos,  
Sem pensar nem querer.  
Dando o que nunca damos.



Prof. MSc. Marcos Alexandruk

E-mail: alexandruk@uni9.pro.br

<https://github.com/alexandruk/desenvolvimentoparainternet>

## Inserção de imagens e links com formatação em CSS

### aula 7

## Objetivo

Nessa aula você aprenderá como utilizar o elemento **img** para inserir imagens em documentos HTML e o elemento âncora **a**, para definir links para documentos. Aprenderá também como aplicar as regras CSS in-line para estilizar as imagens e as regras CSS interna (incorporado) para estilizar os links.



## Inserção de imagens no documento

As imagens dentro de uma página devem estar preferencialmente no formato "gif", "jpg" ou "png". Esses são os formatos mais aceitos universalmente pelos navegadores.

O elemento **img** é um elemento vazio, ou seja, essa tag **contém somente atributos e não possui tag de fechamento**.

O elemento **img** se destina a **inserir uma imagem** no documento e o uso do atributo **src** é **obrigatório**.

O atributo **src** (source) identifica a **URL (Uniform Resource Locator, endereço na web)**, em que está hospedada a imagem.

Utiliza-se a seguinte notação para incluir uma imagem: ``.

## GIF; JPG; PNG

### **GIF (Graphics Interchange Format)**

Lançado em 1987

256 cores (8 bits)

Ajuda a manter o tamanho dos arquivos baixo

GIFs animados longos podem chegar a centenas de megabytes

Não é adequado para fotografias

### **JPG (Joint Photographic Experts Group)**

Lançado em 1992

Utiliza compressão de imagens

16 milhões de cores (24 bits) e aceita transparência

Formato "padrão" para fotografia (quantidade significativa de detalhes em um arquivo pequeno)

Melhor opção para fotos com muitas cores e detalhes

É flexível porque imagens JPG podem ser exportadas em vários níveis de qualidade

### **PNG (Portable Network Graphics)**

Lançado em 1996

16 milhões de cores (24 bits) e aceita transparência

Usado principalmente para logotipos e ícones

É indicado para imagens para as quais se deseja linhas nítidas

## Atividade

Há muitos outros formatos de arquivos. Além dos apresentados, os mais comuns são: **BMP, RAW, TIFF e SVG**.

Quais são as características destes arquivos? Justifique porque cada um deles pode ou não ser indicado para ser usado em páginas Web.

## Inserção de imagens no documento

A tabela a seguir apresenta alguns atributos utilizados no elemento **img**.

Atributos Requeridos		
Atributo	Valor	Descrição
alt	texto	Define um texto alternativo para a imagem
src	URL	Define a URL da imagem
Atributos Opcionais		
Atributo	Valor	Descrição
height	pixels ou %	Define a altura da imagem
width	pixels ou %	Define a largura da imagem

Exemplos:

```

```

```

```

## Exemplo

Uma vez que o atributo **align** está em desuso, a regra de estilo que define o posicionamento usa a declaração **float** e seus valores **left** (esquerda), **right** (direita), **none** (nenhum) e **inherit** (herdado).

```
<html>
  <body>
    <h2>Imagem sem alinhamento</h2>
    <p>Imagem sem alinhamento.</p>
    <h2>Imagem com alinhamento à direita utilizando float</h2>
    <p>Imagem alinhada à direita.</p>
  </body>
</html>
```

[aula07a.html]



[https://www.w3schools.com/css/css\\_float.asp](https://www.w3schools.com/css/css_float.asp)

## Inserção de links no documento

Para criar um link, seja para outro documento ou para o mesmo, utilizamos o elemento âncora **a**.

Esse elemento destina-se a marcar um conteúdo qualquer do documento (texto, imagem, animação etc.) com o qual o usuário poderá interagir ou para "linkar" o documento a outro documento web.

O elemento âncora **a** cria os hipertextos interligando tudo o que estiver disponível na internet.

A tag **<a>** define uma âncora e pode ser utilizada de duas formas:

1. Com o atributo **href** para criar um **link para outro documento**, por exemplo:

```
<a href="https://uninove.br">Visite a Uninove</a>
```

O texto **Visite a Uninove** é o link que aparece no navegador e **https://uninove.br** é o endereço do documento destino.

[aula07b.html]

2. Com o atributo **name** para criar um ponto no documento que poderá ser utilizado como destino de um link, isto é, quando o link é acionado, em vez de abrir um documento novo, o navegador rola o documento atual para um ponto localizado no próprio documento, por exemplo:

```
<a name="topo"> para indicar o ponto do documento que desejamos
```

```
<a href="#topo">Voltar ao topo</a> para criar o link
```

[aula07c.html]

## Inserção de links no documento

Os links, por padrão, aparecerão da seguinte forma nos navegadores:

Um link **não visitado** é sublinhado na cor **azul**.

Um link **visitado** é sublinhado na cor **roxa**.

Um link **ativo** é sublinhado na cor **vermelha**.

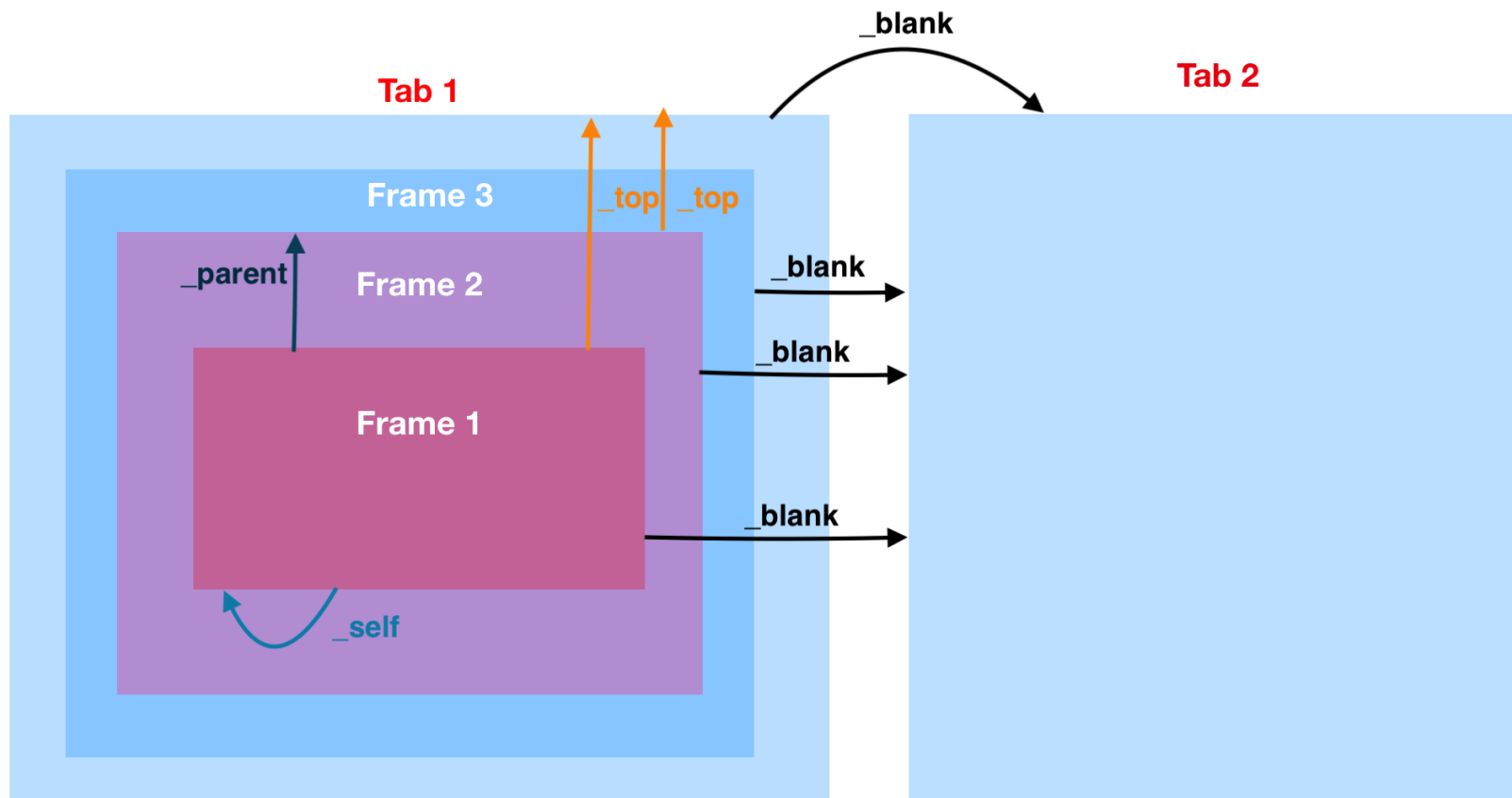
Podemos definir também como o documento destino do link será aberto utilizando o atributo **target**.

Esse atributo possui os seguintes valores:

Valor	Descrição
_blank	O documento destino do link abre em uma nova janela ou aba
_self	O documento destino do link abre na mesma janela do documento atual (padrão)
_parent	O documento destino do link abre na janela pai do documento atual
_top	O documento destino do link abre no corpo da janela do documento atual
framename	O documento destino do link abre em uma das janelas do documento construído com frames

[aula07d.html]

## Inserção de links no documento: atributo **target**



[aula07e.html]



## Definindo uma imagem como um link

Podemos definir uma imagem que, quando selecionada, abrirá a página destino.

Para definir uma imagem como um link, utilizamos o elemento `a` em conjunto com o elemento `img`.

O exemplo a seguir define uma imagem que é um de link para abrir uma página em uma nova janela:

```
<a href="http://www.uninove.br" target="_blank">  
    
</a>
```

[aula07f.html]

## Definindo um link para e-mail

Podemos definir um link que, quando selecionado, abrirá o programa para enviar email (o programa para enviar e-mail deverá estar instalado).

```
<!-- sem preencher o campo "assunto" no email -->
```

```
<a href="mailto:sugestoes@exemplo.com">Enviar email</a>
```

```
<!-- preenchendo automaticamente o campo "assunto" no email -->
```

```
<a href="mailto:sugestoes@exemplo.com?Subject=Sugestões">Enviar email</a>
```

[aula07f.html]

## Formatação de links em CSS

Um link pode ser estilizado com qualquer propriedade CSS (por exemplo, color, fontfamily, background, etc).

Os links podem ser estilizados diferentemente, dependendo do estado em que se encontram.

Existem quatro estados:

Estado	Descrição
a:link	define o estilo do link no estado inicial ou não visitado
a:visited	define o estilo do link visitado
a:hover	define o estilo do link quando passa-se o mouse sobre ele
a:active	define o estilo do link ativo (o que foi "clicado")

As regras CSS são definidas dentro de uma folha de estilos incorporada.

**Nota:** É importante a ordem de definição das regras para os estados dos links:

**a:hover** deverá vir DEPOIS de **a:link** e **a:visited**;

**a:active** deverá vir DEPOIS de **a:hover**.

[aula07h.html]

## Formatação de links em CSS

A propriedade **color** é utilizada para definir a cor dos links, conforme o seu estado:

```
<style type="text/css">
  a:link {color:#FF0000;}      /* link não visitado, cor vermelha */
  a:visited {color:#00FF00;} /* link visitado, cor verde*/
  a:hover {color:#FFFF00;}    /* link quando passa o mouse, cor amarela*/
  a:active {color:#0000FF;}    /* link clicados, cor azul*/
</style>
```

[aula07i.html]

## Formatação de links em CSS

A propriedade **text-decoration** é utilizada para remover o sublinhado dos links:

```
<style type="text/css">
  a:link {text-decoration:underline;}      /* sublinhado */
  a:visited {text-decoration:underline;}
  a:hover {text-decoration:none;}         /* sem sublinhado */
  a:active {text-decoration:none;}
</style>
```

[aula07i.html]

## Formatação de links em CSS

A propriedade **background-color** é utilizada para definir a cor de fundo dos links:

```
<style type="text/css">
  a:link {background-color:#B2FF99;}
  a:visited {background-color:#FFFF85;}
  a:hover {background-color:#FF704D;}
  a:active {background-color:#FF704D;}
</style>
```

[aula07j.html]

## Utilizando o atributo **class** para produzir efeitos nos links

```
<style type="text/css">
  a.one:link {color:#ff0000;}
  a.one:visited {color:#0000ff;}
  a.one:hover {color:#ffcc00;}
  a.two:link {color:#ff0000;}
  a.two:visited {color:#0000ff;}
  a.two:hover {font-size:150%;}
  a.three:link {color:#ff0000;}
  a.three:visited {color:#0000ff;}
  a.three:hover {background:#66ff66;}
  a.four:link {color:#ff0000;}
  a.four:visited {color:#0000ff;}
  a.four:hover {font-family:monospace;}
  a.five:link {color:#ff0000;text-decoration:none;}
  a.five:visited {color:#0000ff;text-decoration:none;}
  a.five:hover {text-decoration:underline;}
</style>
```

[aula07k.html]

## Utilizando iframes

```
<html>
<head>
<style>
iframe { border:none;overflow:auto; }
</style>
</head>
<body>
<div><iframe name="interno" width="800" height="400" src="aula07m.html"></iframe></div>
<p><a href="aula07m.html" title="aula07m" target="interno">aula07m</a></p>
<p><a href="aula07n.html" title="aula07n" target="interno">aula07n</a></p>
<p><a href="aula07o.html" title="aula07o" target="interno">aula07o</a></p>
</body>
</html>
```

A propriedade **overflow** especifica se deve cortar o conteúdo ou adicionar barras de rolagem quando o conteúdo de um elemento é muito grande para caber na área especificada.

O valor **auto** adiciona barras de rolagem apenas quando necessário.



[https://www.w3schools.com/css/css\\_overflow.asp](https://www.w3schools.com/css/css_overflow.asp)

[aula07l.html] => [aula07m.html] [aula07n.html] [aula07o.html]



## Dica do dia

<https://tutorialehtml.com/pt/html-guia-completo-tutorial-html/>

Tutoriais HTML em português.

Ideal para quem está começando ou quando surge aquela dúvida sobre algum recurso da HTML.

**CUIDADO:** Alguns recursos envolvendo estilo, apresentados no site, devem, preferencialmente, ser implementados em folhas de estilo (CSS) e não internamente às tags HTML.



Prof. MSc. Marcos Alexandruk

E-mail: alexandruk@uni9.pro.br

<https://github.com/alexandruk/desenvolvimentoparainternet>

## Inserção de tabelas com formatação em CSS

### aula 8

## Objetivo

Nessa aula você aprenderá como usar o elemento **table**, que tem a finalidade de inserir tabelas em documentos HTML e aplicar as regras CSS inline e interna (incorporado) para estilizar as tabelas.

## Inserção de tabelas no documento

As tabelas são definidas com as tags `<table>` e `</table>`.

Uma tabela é dividida em linhas e colunas.

Cada **linha** de uma tabela deve sempre aparecer entre as marcações `<tr>` `</tr>` (table row) e é dividida em **células** de dados com o par de tags `<td>` `</td>` (table data).

A tag `<td>` pode conter texto, links, imagens, listas, formulários, outras tabelas etc.

Como padrão, o texto nas células é alinhado à esquerda.

## Criando uma tabela

Exemplo: Tabela com duas linhas e duas colunas que utiliza o atributo **border** para criar uma borda em volta dos elementos e da tabela.

```
<!--Define a tabela-->  
<table border="1"> <!--Define a borda da tabela-->  
<tr> <!--Define primeira linha -->  
<td>linha 1, célula 1</td> <!--Define primeira linha e primeira célula-->  
<td>linha 1, célula 2</td> <!--Define primeira linha e segunda célula-->  
</tr> <!--fim da primeira linha-->  
<tr> <!--Define segunda linha -->  
<td>linha 2, célula 1</td> <!--Define segunda linha e segunda célula-->  
<td>linha 2, célula 2</td> <!--Define segunda linha e segunda célula-->  
</tr> <!--fim da segunda linha-->  
</table> <!--fim da tabela-->
```

Se não especificarmos o atributo border, a tabela será apresentada sem bordas.

[\[aula08a.html\]](#)

## Inserindo cabeçalho na tabela

Para inserir um cabeçalho (título) na célula da tabela utilize o par de tags <th> </th> (th = table header). A maioria dos navegadores apresenta o texto da tag th em negrito e centralizado.

```
<!--Define a tabela-->
<table border="1"> <!--Define a borda da tabela-->
<tr> <!--Define a linha do cabeçalho-->
<th>Cabeçalho 1</th> <!--Define o título da primeira célula-->
<th>Cabeçalho 2</th> <!--Define o título da segunda célula-->
</tr> <!--fim da linha do cabeçalho -->
<tr> <!--Define primeira linha dos dados-->
<td>linha 1, célula 1</td> <!--Define primeira linha e primeira célula-->
<td>linha 1, célula 2</td> <!--Define primeira linha e segunda célula-->
</tr> <!--fim da primeira linha-->
<tr> <!--Define segunda linha dos dados-->
<td>linha 2, célula 1</td> <!--Define segunda linha e segunda célula -->
<td>linha 2, célula 2</td> <!--Define segunda linha e segunda célula -->
</tr> <!--fim da segunda linha -->
</table> <!--fim da tabela -->
```

[aula08b.html]

## Mesclado colunas na tabela

Para mesclar colunas utilizamos o atributo **colspan**. Esse atributo define quantas **colunas** uma célula poderá abranger.

```
<h4>Mesclando duas colunas:</h4>
<table border="1">
  <tr>
    <th>Nome</th> <!--Título-->
    <th colspan="2">Telefone</th> <!--mesclando duas colunas-->
  </tr>
  <tr>
    <td>Fulano</td>
    <td>2222-3333</td>
    <td>4444-5555</td>
  </tr>
</table>
```

[aula08c.html]



## Mesclado linhas na tabela

Para mesclar as **linhas** utilizamos o atributo **rowspan**. Da mesma forma que para as colunas, ele define quantas linhas uma mesma célula pode abranger.

```
<h4>Mesclando duas linhas:</h4>
<table border="1">
  <tr>
    <th>Nome:</th> <!--Título-->
    <td>Fulano</td>
  </tr>
  <tr>
    <th rowspan="2">Telefone:</th> <!--Título; mesclando duas linhas-->
    <td>2222-3333</td>
  </tr>
  <tr>
    <td>4444-5555</td>
  </tr>
</table>
```

[aula08d.html]

## Tags utilizadas no elemento table

tag	Definição
<code>&lt;table&gt;</code>	Define uma tabela
<code>&lt;th&gt;</code>	Define o cabeçalho da tabela
<code>&lt;tr&gt;</code>	Define uma linha da tabela
<code>&lt;td&gt;</code>	Define uma célula da tabela
<code>&lt;caption&gt;</code>	Define uma legenda para tabela
<code>&lt;colgroup&gt;</code>	Define um grupo de colunas na tabela para formatação
<code>&lt;col /&gt;</code>	Define os valores dos atributos para uma ou mais colunas na tabela
<code>&lt;thead&gt;</code>	Define um grupo de células que constituem os cabeçalhos da tabela
<code>&lt;tbody&gt;</code>	Define um grupo de células que constituem o corpo da tabela
<code>&lt;tfoot&gt;</code>	Define um grupo de células que constituem o rodapé da tabela

## Formatação de tabelas com CSS

```
<html>
  <head>
    <title>Formatação de tabela com CSS</title>
    <style>
      thead { font-family: Verdana; }
      tbody { font-family: "Arial"; }
      tfoot { font-family: "Courier New"; }
      table { border: 2px solid green; }
      th { border: 1px solid red; }
      td { border: 1px solid blue; }
    </style>
  </head>
  ...
```

```
...
<html>
  <body>
    <table border="1">
      <caption>Legenda da Tabela</caption>
      <thead>
        <tr>
          <th>Cabeçalho 1</th>
          <th>Cabeçalho 2</th>
        </tr>
      </thead>
      <tbody>
        <tr>
          <td>linha 1, célula 1</td>
          <td>linha 1, célula 2</td>
        </tr>
      </tbody>
      <tfoot>
        <tr>
          <td>Rodape 1</td>
          <td>Rodape 2</td>
        </tr>
      </tfoot>
    </table>
  </body>
</html>
```

[aula08e.html]

## Formatação de tabelas com border-collapse

Para apresentar a tabela com borda simples, utilizamos a propriedade border-collapse, que define se as bordas das células de uma tabela devem colidir. Os valores que essa propriedade pode assumir são:

valor	Definição
<b>collapse</b>	Toda tabela compartilha uma mesma borda. Não há bordas internas como as que são exibidas em bordas normais.
<b>separate</b>	Define que a tabela terá uma borda interna e uma borda externa.

```
<table style="border: 1px solid blue; border-collapse: collapse">  
  <tr>  
    <td style="padding: 5px; border: 1px solid blue">Célula 1</td>  
    <td style="padding: 5px; border: 1px solid blue">Célula 2</td>  
  </tr>  
</table>
```

[\[aula08f.html\]](#)

## Definindo a largura e a altura da tabela

Para definir a **largura** e a **altura** da tabela, utilizamos os atributos **width** e **height**, respectivamente.

```
<style>
  table { width: 100%; }
  th { height: 50px; }
</style>

<table border="1">
  <tr>
    <th>Cabeçalho 1</th>
    <th>Cabeçalho 2</th>
  </tr>
  <tr>
    <td>linha 1, célula 1</td>
    <td>linha 1, célula 2</td>
  </tr>
</table>
```

[aula08g.html]

## Definindo as cores das bordas e do fundo da tabela

O exemplo abaixo define as **cores das bordas** e do **fundo** da tabela.

```
<style>
  table, td, th { border: 1px solid blue; }
  th { background-color: blue; color: white; }
</style>

<table>
  <tr>
    <th>Cabeçalho 1</th>
    <th>Cabeçalho 2</th>
  </tr>
  <tr>
    <td>linha 1, célula 1</td>
    <td>linha 1, célula 2</td>
  </tr>
</table>
```

[\[aula08h.html\]](#)

## Definindo o espaçamento das bordas das células da tabela

O atributo **padding** especifica o espaço entre o conteúdo da célula e suas bordas.

```
<style>
  table, td, th { border: 1px solid blue; }
  th, td { padding: 15px; }
</style>
```

```
<table>
  <tr>
    <th>Cabeçalho 1</th>
    <th>Cabeçalho 2</th>
  </tr>
  <tr>
    <td>linha 1, célula 1</td>
    <td>linha 1, célula 2</td>
  </tr>
</table>
```

[aula08i.html]

## Definindo o espaçamento entre as células da tabela

O atributo **border-spacing** especifica o espaço entre as células.

```
<style>
  table, td, th { border: 1px solid blue; }
  table { border-spacing: 15px; }
</style>
```

```
<table>
  <tr>
    <th>Cabeçalho 1</th>
    <th>Cabeçalho 2</th>
  </tr>
  <tr>
    <td>linha 1, célula 1</td>
    <td>linha 1, célula 2</td>
  </tr>
</table>
```

[aula08j.html]



## Definindo um estilo diferente para cada tabela

O atributo **border-spacing** especifica o espaço entre as células.

```
<style>

  table, td, th {
    border: 1px solid blue;
  }

  #t01 {
    width: 100%;
    background-color: #00ff00;
  }

  #t02 {
    width: 50%;
    background-color: #ffffff;
  }

</style>
```

[aula08k.html]

```
<table id="t01">
  <tr>
    <th>Cabeçalho 1</th>
    <th>Cabeçalho 2</th>
  </tr>
  <tr>
    <td>linha 1, célula 1</td>
    <td>linha 1, célula 2</td>
  </tr>
</table>

<table id="t02">
  <tr>
    <th>Cabeçalho 1</th>
    <th>Cabeçalho 2</th>
  </tr>
  <tr>
    <td>linha 1, célula 1</td>
    <td>linha 1, célula 2</td>
  </tr>
</table>
```

## Alternando a cor de fundo das linhas com nth-child( )

Usando **even** e **odd** especificamos duas cores de fundo diferentes para linhas **ímpares** e **pares**.

```
<style>

#t01 th {
    background-color: black;
    color: white;
}
#t01 tr:nth-child(even) { // linhas ímpares
    background-color: #c0c0c0;
}
#t01 tr:nth-child(odd) { // linhas pares
    background-color: #ffffff;
}

</style>
```

```
<table id="t01">
  <tr>
    <th>Cabeçalho 1</th>
    <th>Cabeçalho 2</th>
  </tr>
  <tr>
    <td>linha 1, célula 1</td>
    <td>linha 1, célula 2</td>
  </tr>
  <tr>
    <td>linha 2, célula 1</td>
    <td>linha 2, célula 2</td>
  </tr>
</table>
```

## Alternando a cor de fundo das linhas com nth-child( )

Usando uma fórmula (  $an + b$  ). Onde: **a** representa o tamanho de ciclo, **n** é o contador (começa em 0) e **b** é o valor de deslocamento.

```
<style>

#t01 th {
    background-color: black;
    color: white;
}
#t01 tr:nth-child(2n+0) {
    background-color: #c0c0c0;
}
#t01 tr:nth-child(odd) {
    background-color: #ffffff;
}

</style>
```

```
<table id="t01">
  <tr>
    <th>Cabeçalho 1</th>
    <th>Cabeçalho 2</th>
  </tr>
  <tr>
    <td>linha 1, célula 1</td>
    <td>linha 1, célula 2</td>
  </tr>
  <tr>
    <td>linha 2, célula 1</td>
    <td>linha 2, célula 2</td>
  </tr>
  <tr>
    <td>linha 3, célula 1</td>
    <td>linha 3, célula 2</td>
  </tr>
</table>
```

[\[aula08m.html\]](#)

## Tipos de bordas

Pesquisar:

[https://www.w3schools.com/cssref/pr\\_border-bottom\\_style.asp](https://www.w3schools.com/cssref/pr_border-bottom_style.asp)



Prof. MSc. Marcos Alexandruk

E-mail: alexandruk@uni9.pro.br

<https://github.com/alexandruk/desenvolvimentoparainternet>

## Inserção de iframes com formatação em CSS

### aula 9

## Objetivo

Nessa aula você aprenderá como usar o elemento **iframe**, que tem a finalidade de inserir um quadro com conteúdos em documentos HTML e aplicar as regras CSS inline e interna (incorporado) para estilizar o iframe.

## O elemento **iframe**

O elemento `iframe` destina-se a colocar um quadro, com conteúdos em um documento html.

Os conteúdos de cada `iframe` podem ser blocos de texto ou outros documentos html independentes.

Utiliza-se o atributo `src` para definir o documento que será inserido no `iframe`.

A sintaxe para se adicionar um `iframe` é: **`<iframe src="URL"></iframe>`**.

Navegadores mais antigos não suportam iframes. Portanto, é recomendável incluir um texto alternativo para informar os usuários destes navegadores mais antigos.



## Atributos opcionais para **iframe**

Atributo	Valor	Descrição
frameborder	1 ou 0	Define se deve ou não existir uma borda em torno do frame
height	pixels ou %	Define a altura do frame
longdesc	URL	Define a pagina que contém um descrição longa do conteúdo do frame.
marginheight	pixels	Define a margem superior (top) e inferior (bottom) do frame
marginwidth	pixels	Define a margem esquerda (left) e direita (right) do frame
name	name	Define um nome para o frame
scrolling	yes, no ou auto	Define se deve ou não existir barra de rolagem
src	URL	Define a URL do documento que será apresentado no frame
width	pixels ou %	Define a largura do frame

## Inserindo um **iframe** no documento

```
<html>
  <head>
    <title>IFrame</title>
  </head>
  <body>
    <iframe src = "http://uninove.br" width = "100%" height = "300">
      <!--Caso o navegador não suportar iframe aparecerá a mensagem -->
      <p>Seu navegador não dá suporte ao iframe.</p>
    </iframe>
  </body>
</html>
```

## Removendo a borda do **iframe**

```
<html>
  <head>
    <title>IFrame</title>
  </head>
  <body>
    <iframe src = "http://uninove.br" width = "100%" height = "300" frameborder = "0">
      <!--Caso o navegador não suportar iframe aparecerá a mensagem -->
      <p>Seu navegador não dá suporte ao iframe.</p>
    </iframe>
  </body>
</html>
```

## Removendo a barra de rolagem do **iframe**

```
<html>
  <head>
    <title>IFrame</title>
  </head>
  <body>
    <iframe src = "http://uninove.br" width = "100%" height = "300" scrolling = "no">
      <!--Caso o navegador não suportar iframe aparecerá a mensagem -->
      <p>Seu navegador não dá suporte ao iframe.</p>
    </iframe>
  </body>
</html>
```

**IMPORTANTE:** O atributo **scroolling** não é compatível com HTML5. Deve ser substituído por CSS.

[aula09c.html]

## Abrindo um link em um **iframe**

```
<html>
  <head>
    <title>IFrame</title>
  </head>
  <body>
    <iframe src = "http://uninove.br" width = "100%" height = "300" name="iframe_a">
      <p>Seu navegador não dá suporte ao iframe.</p>
    </iframe>
    <p><a href="https://google.com" target="iframe_a">W3Schools.com</a></p>
  </body>
</html>
```

**IMPORTANTE:** Muito sites não podem ser abertos em um iframe.

## Inserindo um vídeo do YouTube em um **iframe**

```
<html>
  <head>
    <title>IFrame</title>
  </head>
  <body>
    <iframe width="853" height="480" src="https://www.youtube.com/embed/JiqjFEmef9g">
    </iframe>
    <p><a href="https://google.com" target="iframe_a">W3Schools.com</a></p>
  </body>
</html>
```

No YouTube clicar em **COMPARTILHAR** (abaixo do vídeo) e escolher a opção **incorporar**.

## Definindo regra de estilo **inline** para o **iframe**

No exemplo a seguir usamos CSS **inline** para estilizar a borda na cor azul (blue) tracejada (dashed), e o tamanho do iframe é definido nas propriedades **width** e **height**:

```
<html>
<head>
  <title>IFrame</title>
</head>
<body>
  <iframe name='iframe1' src="https://uninove.br"
    style="border: 5px dashed blue; width: 100%; height: 200px;"></iframe>
</iframe>
</body>
</html>
```

[aula09f.html]

## Definindo regra de estilo **incorporada** para o **iframe**

No exemplo a seguir usamos CSS **incorporada** para estilizar a borda na cor azul (blue) tracejada (dashed), e o tamanho do iframe é definido nas propriedades **width** e **height**:

```
<html>
<head>
  <title>IFrame</title>
  <style type="text/css">
    iframe {
      border: 5px dashed blue;
      width: 100%;
      height: 200px;
    }
  </style>
</head>
<body>
  <iframe name='iframe1' src="https://uninove.br"></iframe>
</body>
</html>
```

[aula09g.html]



## Inserindo um **iframe** em uma célula de **tabela**

No exemplo a seguir, um iframe foi definido como conteúdo de uma célula.

```
<html>
  <head>
    <title>IFrame</title>
  </head>
  <body>
    <table border="1" width = 100%>
      <tr>
        <th width = 20%>Cabeçalho 1</th>
        <th width = 60%>Cabeçalho 2</th>
        <th width = 20%>Cabeçalho 3</th>
      </tr>
      <tr>
        <td>linha 1, célula 1</td>
        <td><iframe src="https://uninove.br" width = "100%" height = "300" ></iframe></td>
        <td>linha 1, célula 3</td>
      </tr>
    </table>
  </body>
</html>
```

[aula09h.html]



Prof. MSc. Marcos Alexandruk

E-mail: alexandruk@uni9.pro.br

<https://github.com/alexandruk/desenvolvimentoparainternet>

## Inserção de formulários com formatação em CSS

### aula 10

## Objetivo

Nessa aula você aprenderá como criar formulários em documentos HTML com o elemento **form** e os elementos para entrada de dados. Aprenderá também como aplicar as regras de estilo CSS para formatar formulários.

## Formulários HTML

Os formulários HTML são utilizados para diferentes tipos de entrada de dados.

Os dados que são preenchidos nos formulários são enviados a um servidor.

O elemento **form** destina-se a servir de container para os controles (ou campos) de formulários.

Um **form** pode conter elementos de entrada como caixa de texto, checkboxes, botões de seleção, botões e envio etc.

Um **form** também pode conter listas de seleção, área de texto, legendas, rótulos, entre outros elementos.

Para criar um formulário em HTML utiliza-se o par de tags **<form>** e **</form>**.

## Formulários HTML

Quando preenchemos um formulário, precisamos definir como enviar as informações ao servidor.

Existem dois métodos (**method**): **get** e **post** que são definidos no atributo method do formulário.

A maioria dos documentos HTML é recuperada a partir da requisição de uma única URL ao servidor.

O método **get** envia toda a string de consulta (pares nome/valor) na URL. *(Dados confidenciais nunca devem ser enviados através do método get.)*

Os dados enviados ao servidor com o método **post** são armazenados no corpo da solicitação http.

O atributo **action** informa a URL (endereço na web em que está hospedada a aplicação que irá processar o formulário).

## Formulários HTML

O tipo de controle é definido pelo atributo **type**. Veja, a seguir, os valores possíveis:

Valor	Descrição
<b>text</b>	Cria um campo para entrada de texto em uma linha.
<b>password</b>	Cria um campo para entrada de senha ou dados que devem permanecer ocultos quando digitados.
<b>checkbox</b>	Cria uma caixa de seleção.
radio	Cria um botão de opção.
submit	Cria um botão para envio do formulário.
reset	Cria um botão que restaura o formulário ao seu estado inicial.
file	Cria um campo para entrada de um arquivo local que deva ser enviado ao servidor.
hidden	Cria um campo cujos dados não são visíveis ao usuário.
image	Cria um botão personalizado com uma imagem que funciona como um botão de envio do formulário.
button	Cria um botão destinado a acionar um script para realizar uma determinada ação.

## Formulário: exemplo

No exemplo a seguir, foram utilizados diferentes valores para o atributo **type**:

```
<html>
<head>
  <title>Formulario</title>
</head>
<body>
  <form name="input" action="" method="get">
    Nome: <input type="text" name="nome" /> <br />
    Sobrenome: <input type="text" name="sobrenome" /> <br />
    Senha: <input type="password" name="pwd" /> <br />
    <input type="radio" name="sexo" value="masculino" /> Masculino
    <input type="radio" name="sexo" value="feminino" /> Feminino<br />
    <input type="checkbox" name="veiculo" value="Bicicleta" /> Bicicleta <br />
    <input type="checkbox" name="veiculo" value="Carro" /> Carro <br />
    <input type="submit" value="Enviar" />
    <input type="reset" value="Limpar" />
  </form>
</body>
</html>
```

[aula10a.html]



## Formulário: usando o método **get**

No exemplo a seguir, o **method** foi definido como **get**:

```
<html>
  <head>
    <title>Formulario</title>
  </head>
  <body>
    <form method="get" id="cadastro" action="obrigado.html">
      <p><label for="nome">Nome:</label>
        <input type="text" name="nome" id="nome" placeholder="Digite o seu nome">
      </p>
      <p><label for="email">E-mail:</label>
        <input type="text" name="email" id="email" placeholder="Digite o seu e-mail">
      </p>
      <input type="submit" value="Enviar">
    </form>
  </body>
</html>
```

## Formulário: usando o método **post**

No exemplo a seguir, o **method** foi definido como **post**:

```
<html>
  <head>
    <title>Formulario</title>
  </head>
  <body>
    <form method="post" id="cadastro" action="obrigado.html">
      <p><label for="nome">Nome:</label>
        <input type="text" name="nome" id="nome" placeholder="Digite o seu nome">
      </p>
      <p><label for="email">E-mail:</label>
        <input type="text" name="email" id="email" placeholder="Digite o seu e-mail">
      </p>
      <input type="submit" value="Enviar">
    </form>
  </body>
</html>
```

[aula10c.html]

## Formulário: o elemento **select**

O elemento **select** é utilizado para criar uma **lista de seleção** (drop-down list). As tags **<option>**, dentro do elemento select, cria as opções disponíveis na lista.

```
<html>
<head>
  <title>Formulario</title>
</head>
<body>
  <form method="post" id="cadastro" action="obrigado.html">
    <select name="carros">
      <option value="volvo">Volvo</option>
      <option value="bmw">BMW</option>
      <option value="mercedes">Mercedes</option>
      <option value="audi">Audi</option>
    </select>
    <input type="submit" value="Enviar">
  </form>
</body>
</html>
```

[aula10d.html]

## Formulário: o elemento **textarea**

O elemento **textarea** é utilizado para entrada de texto que contenha mais de uma linha. A área de texto pode conter um número ilimitado de caracteres. O tamanho desse elemento pode ser definido pelos atributos **rows** (linhas) e **cols** (colunas).

```
<html>
  <head>
    <title>Formulario</title>
  </head>
  <body>
    <form method="post" id="cadastro" action="obrigado.html">
      <textarea rows="10" cols="30">
        Digite um texto.
      </textarea>
      <input type="submit" value="Enviar">
    </form>
  </body>
</html>
```

## Estilizando formulários

Podemos definir as regras CSS inline ou incorporada para estilizar os formulários.

Veja, a seguir, as declarações CSS **incorporada** para formatar os elementos do formulário.

```
<style type="text/css">
  form {
    border: 1px solid #666699;
    padding: 5px;
  }
  input {
    background-color: #B0E0E6;
    font: 12px verdana, arial, helvetica, sans-serif;
    color:#003399;
    border:2px solid #000099;
  }
  select {
    background-color: #B0E0E6;
    font:12px verdana, arial, helvetica, sans-serif;
    color:#003399;
  }
  textarea {
    background-color: #B0E0E6;
    font:12px verdana, arial, helvetica, sans-serif;
    color:#003399;
  }
</style>
```

[\[aula10f.html\]](#)

## Estilizando formulários

Podemos também criar as regras CSS inline para estilizar cada elemento ou para modificar a formatação incorporada de um elemento específico.

```
<head>
  <style type="text/css">
    form {
      border: 1px solid #666699;
      padding: 5px;
    }
    input {
      color:#003399;
      border:2px solid #000099;
    }
    ...
  </style>
</head>
...
<form method="post" id="cadastro" action="obrigado.html">
  <p><label for="nome">Nome:</label>
    <input type="text" name="nome" id="nome">
  </p>
  <p><label for="email">E-mail:</label>
    <!--Regra de estilo inline sobrepõe a regra incorporada criada na seção head-->
    <input type="text" name="email" id="email" style="border: 2px dashed #0000FF;" color:#FF0000;>
  </p>
...
[aula10g.html]
```

## Formulários: **fieldset** e **legend**

A tag **fieldset** é usada para agrupar elementos relacionados em um formulário e desenha uma caixa ao redor dos elementos relacionados.

A tag **legend** é usada para definir uma legenda para o elemento fieldset.

```
<style type="text/css">
...
    fieldset {
        margin-top: 5px;
        background-color: #eeeeee;
    }
    legend {
        background-color: gray;
        color: white;
        padding: 5px 10px;
    }
    ...
</style>
...
<form method="post" id="cadastro" action="obrigado.html">
    <fieldset>
        <legend>Dados Pessoais</legend>
        ...
    </fieldset>
```

[\[aula10h.html\]](#)



Prof. MSc. Marcos Alexandruk

E-mail: alexandruk@uni9.pro.br

<https://github.com/alexandruk/desenvolvimentoparainternet>



## Listas

### aula 11

## Objetivo

Nessa aula você aprenderá a criar e aplicar estilos a listas ordenadas e não ordenadas utilizando HTML e CSS.

## Listas ordenadas

Listas ordenadas são aquelas que os itens recebem uma marca de ordenação sequencial, que pode ser alfabética ou numérica.

Listas não ordenadas são aquelas que os itens recebem uma marca ou bullet do lado esquerdo.

O elemento ol (ordered list) é utilizado para criar a lista ordenada e o elemento ul (unordered list) para criar a lista não ordenada.

### **Lista ordenada:**

1. Primeiro item
2. Segundo item
3. Terceiro item

### **Lista não ordenada:**

- Primeiro item
- Segundo item
- Terceiro item

## Listas ordenadas

```
<html>
  <head>
    <title>Listas Ordenadas</title>
  </head>
  <body>
    <ol>
      <li>Paris</li>
      <li>Rio de Janeiro</li>
      <li>Roma</li>
    </ol>
  </body>
</html>
```

## Listas não ordenadas

```
<html>
  <head>
    <title>Listas Não Ordenadas</title>
  </head>
  <body>
    <ul>
      <li>Paris</li>
      <li>Rio de Janeiro</li>
      <li>Roma</li>
    </ul>
  </body>
</html>
```

## Listas de definição

Para criar listas de definição utilizamos o elemento **dl** (definition lists) em conjunto com os elementos **dt** e **dd**.

```
<html>
  <head>
    <title>Listas Não Ordenadas</title>
  </head>
  <body>
    <dl>
      <dt>Paris</dt>
      <dd>Cidade luz</dd>
      <dt>Rio de Janeiro</dt>
      <dd>Cidade maravilhosa</dd>
      <dt>Roma</dt>
      <dd>Cidade eterna</dd>
    </dl>
  </body>
</html>
```

[aula11c.html]

## Intercalando listas

As listas podem ser encadeadas ou intercaladas arbitrariamente, produzindo resultados bastante interessantes. Pode-se inclusive ter um parágrafo, intercalado com uma lista que contenha um único item.

```
<ul>
  <li>Região Sul: </li>
<ul>
  <li>Rio Grande do Sul</li>
  <li>Santa Catarina</li>
  <li>Paraná</li>
</ul>
  <li>Região Sudeste: </li>
<ul>
  <li>São Paulo</li>
  <li>Minas Gerais</li>
  <li>Rio de Janeiro</li>
  <li>Espírito Santo</li>
</ul>
</ul>
```

[\[aula11d.html\]](#)

## Estilizando listas

A propriedade `list-style-type` define a aparência do marcador de uma lista que pode ser de três tipos: **glifo**, **numérico** e **alfabético**. Veja alguns valores possíveis:

Valor	Aparência
Tipo Glifo	
disc	círculo preenchido (padrão)
circle	círculo sem preenchimento
square	quadrado
Tipo Numérico	
decimal	numeração decimal começando em 1
lower-roman	algarismos romanos minúsculos
upper-roman	algarismos romanos maiúsculos
Tipo Alfabético	
lower-alpha ou lower-latin	letras minúsculas
upper-alpha ou upper-latin	letras maiúsculas



## Estilizando listas

O exemplo a seguir apresenta as regras de estilo para lista.

```
<style type="text/css">
ol.a {list-style-type: upper-roman;}
ul.b {list-style-type: square;}
</style>
<ol class="a">
  <li>Paris</li>
  <li>Rio de Janeiro</li>
  <li>Roma</li>
</ol>
<ul class="b">
  <li>Paris</li>
  <li>Rio de Janeiro</li>
  <li>Roma</li>
</ul>
```

[aula11e.html]

## Estilizando listas

O exemplo a seguir apresenta outro exemplo de estilo para lista.

```
<style type="text/css">
  ul {
    border-left: 5px solid blue;
    background-color: #f1f1f1;
    list-style-type: none;
    padding: 10px 20px;
  }
</style>
...
<ul>
  <li>Paris</li>
  <li>Rio de Janeiro</li>
  <li>Roma</li>
</ul>
```

[aula11f.html]

## Estilizando listas

O exemplo a seguir apresenta um exemplo com zeros à esquerda do marcador.

```
<style type="text/css">
  ol {
    list-style-type: decimal-leading-zero;
  }
</style>
...
<ol>
  <li>Paris</li>
  <li>Rio de Janeiro</li>
  <li>Roma</li>
</ol>
```



Prof. MSc. Marcos Alexandruk

E-mail: alexandruk@uni9.pro.br

<https://github.com/alexandruk/desenvolvimentoparainternet>

## Criação de layouts em camadas

### aula 12

## Objetivo

Nessa aula você aprenderá a criar layouts em camadas utilizando o elemento **div**, e a usar os modelos de formatação visual em CSS com os esquemas de posicionamento **static**, **relative**, **fixed**, **absolute** e **sticky**, aplicados na propriedade **position**.

## A propriedade **position**

A propriedade **position** especifica o tipo de método de posicionamento usado para um elemento:

- **static** (estático)
- **relative** (relativo)
- **fixed** (fixo)
- **absolute** (absoluto)
- **sticky** (aderente)

Os elementos são posicionados usando as propriedades **top** (superior), **bottom** (inferior), **left** (esquerda) e **right** (direita). No entanto, essas propriedades não funcionarão a menos que a propriedade **position** seja definida primeiro. Eles também funcionam de forma diferente dependendo do valor da posição.

## A posição **static**

Os elementos HTML são posicionados estáticos por padrão.

Os elementos posicionados estáticos não são afetados pelas propriedades **top**, **bottom**, **left** e **right**.

Um elemento com **position: static**; não é posicionado de nenhuma maneira especial; está sempre posicionado de acordo com o fluxo normal da página.

Os elementos são posicionados usando as propriedades **top** (superior), **bottom** (inferior), **left** (esquerda) e **right** (direita). No entanto, essas propriedades não funcionarão a menos que a propriedade **position** seja definida primeiro. Eles também funcionam de forma diferente dependendo do valor da posição.



## A posição **static**

```
<html>
  <head>
    <style>
      div.static {
        position: static;
        border: 3px solid #00ff00;
      }
    </style>
  </head>
  <body>
    <h2>position: static;</h2>
    <p>Um elemento com position: static; é posionado conforme o fluxo normal da página:</p>
    <div class="static">
      Esse div foi definido como position: static;
    </div>
  </body>
</html>
```

[aula12a.html]

## A posição **relative**

Um elemento com **position: relative;** está posicionado em relação à sua posição normal.

Definir as propriedades top, bottom, left e right de um elemento relativamente posicionado fará com que ele se distancie da sua posição normal. Outro conteúdo não será ajustado para caber em qualquer lacuna deixada pelo elemento.

## A posição **relative**

```
<html>
  <head>
    <style>
      div.relative {
        position: relative;
        left: 30px;
        border: 3px solid #00ff00;
      }
    </style>
  </head>
  <body>
    <h2>position: relative;</h2>
    <p>Um elemento com position: relative; está posicionado em relação à sua posição normal:</p>
    <div class="relative">
      Esse div foi definido como <b>position: relative</b>;
    </div>
  </body>
</html>
```

[aula12b.html]

## A posição **fixed**

Um elemento com **position: fixed**; está posicionado em relação à janela de visualização, o que significa que sempre permanece no mesmo lugar, mesmo se a página for rolada.

As propriedades top, bottom, left e right são usadas para posicionar o elemento.

Um elemento fixo não deixa uma lacuna na página onde normalmente estaria localizado.

## A posição **fixed**

```
<html>
  <head>
    <style>
      div.fixed {
        position: fixed;
        bottom: 0;
        right: 0;
        width: 300px;
        border: 3px solid #00ff00;
      }
    </style>
  </head>
  <body>
    <h2>position: fixed;</h2>
    <p>Um elemento com position: fixed; está posicionado em relação à janela de visualização, <br />
    o que significa que sempre permanece no mesmo lugar, mesmo se a página for rolada.</p>
    <p>Observe o elemento fixo no canto inferior direito da página.</p>
    <div class="fixed">
      Esse div foi definido como position: fixed;
    </div>
  </body>
</html>
```

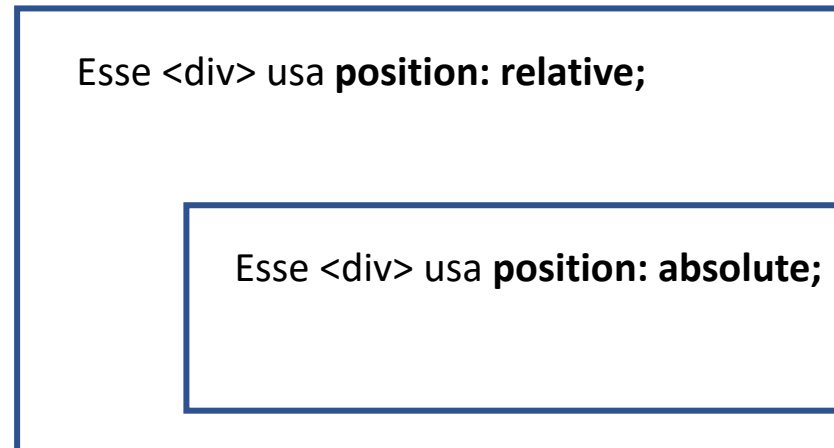
[aula12c.html]

## A posição **absolute**

Um elemento com **position: absolute;** é posicionado em relação ao ancestral posicionado mais próximo (em vez de posicionado em relação à janela de visualização, como fixo).

Contudo; se um elemento posicionado de forma absoluta não tiver ancestrais posicionados, ele usará o corpo do documento e se moverá junto com a rolagem da página.

Nota: Um elemento "posicionado" é aquele cuja posição é qualquer coisa, exceto estática.



## A posição **absolute**

```
<html>
  <head>
    <style>
      div.relative {
        position: relative; width: 600px; height: 300px; border: 3px solid #00ff00;
      }
      div.absolute {
        position: absolute; top: 80px; right: 0; width: 400px; height: 200px; border: 3px solid #00ff00;
      }
    </style>
  </head>
  <body>
    <h2>position: absolute;</h2>
    <p>Um elemento com position: absolute; é posicionado em relação ao ancestral mais próximo <br />
    (em vez de posicionado em relação à janela de visualização, como fixo):</p>
    <div class="relative">Esse div usa position: relative;
      <div class="absolute">Esse div usa position: absolute;</div>
    </div>
  </body>
</html>
```

[aula12d.html]

## A posição **sticky**

Um elemento com **position: sticky;** é posicionado com base na posição de rolagem do usuário.

Um elemento **sticky** alterna entre **relative** e **fixed**, dependendo da posição de rolagem. Ele é posicionado relative até que uma determinada posição de deslocamento seja encontrada na janela de exibição - então, ele "se fixa" no lugar (como a posição: **fixed**).

É preciso especificar pelo menos um valor para uma das seguintes posições: **top**, **bottom**, **left** ou **right** para que o posicionamento fixo funcione.

Observação: O Internet Explorer não oferece suporte para posicionamento fixo. O Safari requer um prefixo -webkit-sticky (veja exemplo a seguir).



## A posição **sticky**

```
<html>
  <head>
    <style>
      div.sticky {
        position: -webkit-sticky; position: sticky; top: 0; padding: 5px; border: 2px solid #00ff00;
      }
    </style>
  </head>
  <body>
    <p>Use o <b>scroll</b> para entender como position: sticky funciona.</p>
    <div class="sticky">Isso é um sticky!</div>
    <div style="padding-bottom:2000px">
      <p>Neste exemplo, o elemento sticky adere ao topo da página (topo: 0) <br />
      quando você atinge a posição de rolagem.</p>
      <p>Role para cima para remover o sticky.</p>
      <p>Algum texto para habilitar o scrolling...</p>
      <p>Algum texto para habilitar o scrolling...</p>
      <p>Algum texto para habilitar o scrolling...</p>
      <p>Algum texto para habilitar o scrolling...</p>
    </div>
  </body>
</html>
```

[aula12e.html]