

MÓDULO II

Introdução ao HTML5 e Estrutura Básica

MÓDULO II

O que são os links HTML

Desde a origem da internet, o que torna a web o que ela é hoje é justamente a possibilidade de vincular um documento a qualquer outro documento ou recurso. Essa função é chamada de hiperlink ou link. Mas afinal de contas, o que são os links HTML?

O link HTML, nada mais é que uma função do HTML que permite inserir os hiperlinks em diversos elementos, como textos e imagens. Um link precisa sempre apontar para uma URL (endereço) existente em seu site. Caso contrário, você poderá obter uma mensagem de **erro 404**.

A tag link do HTML está presente desde a primeira versão, criada por Tim Berners-Lee.

MÓDULO II

O surgimento dos links HTML

Em **1980** o Físico Britânico *Tim Berners-Lee* iniciou um projeto baseado no conceito de **hipertexto** denominado Enquire, desenvolvido inicialmente em linguagem **Pascal**. Anos mais tarde, *Tim Berners-Lee* e *Robert Cailliau* conseguiram implementar a primeira comunicação entre um cliente e um servidor através da internet, o protocolo **HTTP**. Em **1989**, surgia então a **World Wide Web** (www).

Em conjunto à implementação da **World Wide Web**, *Tim Berners-Lee* lançou a linguagem **HTML** (*HyperText Markup Language*). A versão inicial do **HTML** foi baseada na **SGML**, uma linguagem de estruturação de documentos. Foi dele que o **HTML** herdou diversas tags como as de título (**<h1>** a **<h6>**), de parágrafos (**<p>**) e a de cabeçalho (**<head>**). A maior diferença entre essas duas linguagens de marcação é que o **HTML** implementava a tag **<a>**, surgindo então o link **HTML**, que permitia a ligação de uma página a outra.

Certamente esse conceito de ligação entre um documento a outro é o grande diferencial e principal conceito que define a base do funcionamento da web.

MÓDULO II

Como fazer um link em HTML

No **HTML**, os links são definidos pela tag `<a>`. Dentro dessa tag incluímos o atributo **href** (*Hypertext Reference*), que é o endereço de destino do link. Dentro do conteúdo da tag `<a>`, incluímos então o texto ou elemento que servirá como redirecionador, ou seja, que ao ser clicado, executará a função de redirecionar para o endereço dentro do atributo **href**.

Dessa forma, a sintaxe **básica do HTML** link é:

```
<a href="url">Exemplo</a>
```

MÓDULO II

A tag `<a>` pode ser utilizada dentro ou fora dos demais **elementos (tags) do HTML**, como no exemplo abaixo onde criamos um parágrafo, e dentro dele, apenas a palavra HomeHost contém um hiperlink para a página inicial da **HomeHost**:

```
<p>Seja redirecionado à página da  
<a href="https://www.homehost.com.br/">HomeHost</a></p>
```

Como resultado, teremos:

Seja redirecionado a página da **HomeHost**

MÓDULO II

Estilizando o link

Por padrão, a tag `<a>` traz consigo o **estilo** próprio com o texto sublinhado e na cor *azul*, para links ainda não visitados, *roxo* para links visitados e *vermelho* para links ativos. Porém, podemos estilizar diretamente os links através dos estilos inline ou dentro do elemento `<style>`.

Dessa forma, podemos reiniciar o estilo da tag apenas aplicando “**text-decoration:none**”.

Também podemos utilizar as pseudo classes do **CSS**.

Vejamos um exemplo de estilização CSS:

MÓDULO II

```
<style>
/*Link não visitado */
a:link {
    color: green;
    text-decoration: none;
}

/*Link já visitado*/
a:visited {
    color: blue;
    text-decoration: none;
}
```

```
/*Quando o mouse passa por cima*/
a:hover {
    color: pink;
    text-decoration: none;
}

/*Link ativo/selecionado*/
a:active {
    color: yellow;
    text-decoration: none;
}
</style>
```

MÓDULO II

Também poderíamos atribuir um estilo único para o elemento `<a>`, fazendo com que, independente da condição, ele receberia o mesmo estilo. Veja o exemplo a seguir:

```
<!DOCTYPE html>
<html>
<head>
<style>
  a{
    text-decoration: none;
    background-color: yellow;
    color: red;
    border: 1px solid blue
    padding:3px 5px;
  }
</style>
</head>

<body>
  <a href="">
    Meu link personalizado
  </a>
</body>
</html>
```

MÓDULO II

Com o código anterior, aplicamos uma borda *azul*, um fundo na cor *amarela* e a cor *vermelha* para o texto. Aplicamos também um *padding* (espaçamento). Portanto, perceba que poderíamos aplicar qualquer propriedade para a tag `<a>`. Isso acontece pois o mesmo tem como padrão de display bloco. Portanto, vejamos resultado do código anterior:



Meu link personalizado

MÓDULO II

O atributo href

Através do atributo principal da tag `<a>`, o **href=""**, podemos redirecionar o usuário a outro documento ou recurso. Existem três diferentes tipos de links utilizados para redirecionamento dentro do href. Um link pode ser:

- interno: redireciona para um elemento existente dentro da mesma página;
- local: utilizados para páginas contendo o mesmo domínio, ou seja, entre páginas do mesmo site;
- global: utilizados para páginas de outros domínios, ou seja, fora do site.

MÓDULO II

Veja abaixo alguns exemplos de como podem ser utilizados:

Interno - `Contato`

Redirecionará ao elemento âncora contato.

Local - `Pagina 2`

Redirecionará ao arquivo pagina2.html pertencente à pasta pages.

Global - `Google`

Redirecionará à pagina inicial do Google

MÓDULO II

O atributo target

Além do atributo href, também podemos incluir o atributo target dentro da tag `<a>`. Esse atributo informa ao navegador como o redirecionamento deverá ocorrer, abrindo a página na mesma janela/aba do navegador ou abrindo uma nova janela/aba.

Os atributos target são:

- **_blank**: abre a página em uma nova janela/aba;
- **_self**: abre a página na mesma janela;
- **_parent**: abre a página na mesma janela do link;
- **_top**: cancela todos os demais frames e abre a nova página no mesmo navegador.

MÓDULO II

Com isso, veja o exemplo a seguir onde ao clicar no link, abrirá uma nova aba do navegador com a página inicial da HomeHost:

```
<a href="https://www.homehost.com.br/" target="_blank">  
Página inicial da HomeHost</a>
```

Nesse exemplo temos o resultado:

Página inicial da HomeHost

MÓDULO II

O atributo title

O atributo title permite escrever um texto que aparecerá apenas quando passarmos o mouse por cima do link. Portanto, é um atributo importante que nos permite digitar informações úteis suplementares sobre o link, como o tipo de informação que a página contém ou avisos.

O atributo title de um link **HTML** é usado para fornecer um texto descritivo adicional que é exibido quando o cursor do mouse é colocado sobre o link. Esse texto é conhecido como dica de ferramenta (tooltip) e fornece informações adicionais sobre o destino do link ou uma descrição breve do conteúdo associado.

Ao definir o atributo title em um link HTML, o texto fornecido será exibido em um pequeno balão ou caixa de dica quando o usuário posicionar o cursor do mouse sobre o link. Essa dica de ferramenta ajuda a fornecer informações contextuais ou esclarecer a finalidade do link antes que o usuário clique nele.

MÓDULO II

Veja um exemplo de como podemos utilizar esse atributo.

```
<a href="" title="Exemplo de como funciona o atributo title">  
Confira como funciona o atributo title!  
</a>
```

Com esse código, nosso resultado será:

Confira como funciona o atributo title!

O uso do atributo title é opcional e nem todos os navegadores exibem a dica de ferramenta por padrão.

Além disso, o estilo e o comportamento exato da dica de ferramenta podem variar dependendo do navegador e do dispositivo usado. É importante considerar que o texto fornecido no atributo title deve ser breve e relevante para fornecer um contexto útil para os usuários.

MÓDULO II

Criando âncoras com os links HTML

Conforme o explicado no tópico “O atributo **href**”, uma das possibilidades de utilizar o link HTML é através do redirecionamento interno, processo que também é conhecido como âncora.

Para isso, utilizamos a tag `<a>` para linkar duas seções da mesma página. Além disso, podemos nomear a seção ou atribuir um **ID** a um determinado elemento, e assim, através da âncora, acontecerá o redirecionamento ao elemento.

Um ótimo exemplo de utilização desse processo, se encontra no início desse artigo: ao clicar nos links de qualquer um dos tópicos, você é redirecionado para essa posição! Este recurso também é muito utilizado dentro de menus de páginas únicas e landing pages.

MÓDULO II

Confira o exemplo a seguir, utilizando como base o seguinte código:

```
<h1>Texto 1 de exemplo<a name="text1"></a></h1>  
<h2>Texto 2 de exemplo<a name="text2"></a></h2>  
<h2>Texto 3 de exemplo<a name="text3"></a></h2>
```

Agora vamos criar os links que ao serem clicados realizarão o redirecionamento para a posição do **text1**, do **text2** e do **text3**:

```
<a href="#text1">Esse link leva ao Texto 1 de exemplo</a>  
<a href="#text2">Esse link leva ao Texto 2 de exemplo</a>  
<a href="#text3">Esse link leva ao Texto 3 de exemplo</a>
```

MÓDULO II

Link HTML de e-mail

Atualmente, o HTML é capaz de criar um link que redireciona para o envio de e-mail.

Para criar essa função, basta colocar dentro do atributo href o “mailto:” e separados por uma interrogação “?”, o “subject=”. O “mailto” deve conter o endereço ao qual será enviado o e-mail, o “subject=” nada mais é que o assunto. Vejamos então o exemplo abaixo:

```
<p><a href="mailto:exemplo@exemplo.com?subject=Assunto" >Clique aqui</a>  
para enviar um e-mail.</p>
```

O resultado dessa linha de código é:

Clique aqui para enviar um e-mail.

MÓDULO II

Posteriormente, ainda podemos acrescentar um texto pré-estabelecido para o corpo do e-mail.

Para isso acrescentamos ao código o “**body=**”

```
<p><a href="mailto:exemplo@exemplo.com?subject=Assunto&body=Digite sua mensagem">
Clique aqui</a> para enviar um e-mail.</p>
```

Dessa forma, o resultado fica da seguinte forma:

Clique aqui para enviar um e-mail.

No exemplo utilizado anteriormente, incluímos a função de link em texto, porém, essa função é muito bem aproveitada quando **inserida em imagens** ou ícones. Atenção, para que essa forma de link funcione, é necessário ter um **software de e-mail configurado** em seu PC ou celular.

MÓDULO II

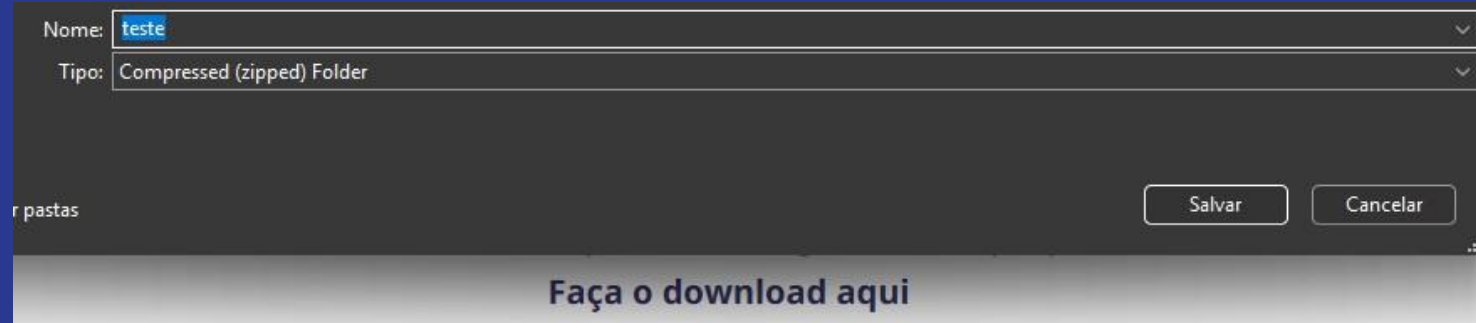
Link HTML e o atributo download

Anteriormente, vimos que é possível utilizar dentro do href uma url global, e com isso temos diversas possibilidades de utilizar os **hiperlinks**. Uma dessas possibilidades é utilizar uma **url** que redirecione para o download de um arquivo. Apenas com o atributo **href**, o download irá abrir normalmente, porém é muito recomendável utilizar o atributo **download**. Com esse atributo, podemos definir um nome padrão para o arquivo que será baixado.

MÓDULO II

Vejamos o exemplo a seguir:

```
<a href="urldownload"  
  download="nomedoarquivo.exe">  
  Faça o download do Firefox 39 para Windows  
</a>
```



MÓDULO II

Links HTML em imagens

Já ficou claro que a tag `<a>` permite sua utilização de diversas formas. Uma das formas mais exploradas atualmente é a de links em blocos. Ou seja: uma vez que o elemento `<a>` não possui nenhum impedimento com a inserção de blocos dentro dele, podemos inserir um link numa **imagem em HTML**, para toda uma `<div>` ou até mesmo para uma seção inteira de um site.

Portanto, vejamos o exemplo a seguir, onde vamos inserir uma imagem para um link:

```
<a href="https://www.homehost.com.br/" target="_blank">  
    
</a>
```

MÓDULO II

Teremos como resultado o seguinte:



Perceba que no exemplo anterior, utilizamos os atributos para alterar o tamanho da imagem normalmente. Portanto, a utilização da tag link não impede a estilização do elemento. A mesma regra vale para elementos de blocos, como no exemplo a seguir:

```
<a href="https://www.homehost.com.br/" target="_blank">
  <div height="100" style="background-color: blue;line-height: 100px;text-align: center;">
    <p style="color: white">Meu link em uma Div</p>
  </div>
</a>
```

MÓDULO II

Da mesma forma que utilizamos atributos de estilo inline normalmente, estamos agora aplicando um link para uma div inteira. Portanto, teste abaixo clicar em qualquer parte do bloco azul:

Meu link em uma Div

Além disso, há infinitas possibilidades de estilização e utilização do elemento link.

MÓDULO II

Para que serve uma Div no HTML?

O elemento **HTML Div** define uma divisão ou seção em um documento **HTML**. O elemento **div** é frequentemente usado como um contêiner para outros elementos, o que facilita na estilização de blocos. Também é muito útil para inclusão de interações com **Scripts (JavaScript)**. Portanto, ele é definido com um contêiner genérico para conteúdo de fluxo.

Diante disso, a **HTML Div** é muito utilizada quando precisamos agrupar elementos sem a necessidade do uso de um elemento semântico do **HTML5**. Isso acontece, pois o elemento **div** não possui um valor semântico. Portanto, não representa nada para o navegador e para os **mecanismos de buscas**.

Como é muito utilizado para agrupar elementos, acaba sendo um facilitador para organizar informações em um layout. Dessa forma, é possível formatar e manipular os elementos, inclusive a própria div, através do **CSS** de uma forma organizada. Geralmente é acompanhado de atributos de **ID** e **classe**, para poder facilitar essa organização e formatação.

MÓDULO II

Como utilizar a HTML Div: a tag <div>

Agora que já conhecemos a definição do elemento de divisão do **HTML**, vamos aprender como utilizá-la.

A sintaxe do elemento **HTML Div** é através da **tag** de abertura **<div>** e de fechamento **</div>**. Por padrão, a **div** gera uma line-break (**quebra de linha**) automática ao ser utilizada. Isso ocorre, pois possui como display padrão um elemento de bloco, ou seja, display **block**. Portanto, para utilizar a **div**, basta colocar os elementos que quiser dentro da tag **<div></div>** como na sintaxe abaixo:

```
<div>
    ...
</div>
```

MÓDULO II

Dessa forma, vamos utilizar algumas divisões para poder separar os parágrafos no nosso exemplo abaixo:

```
<div>
  <p>Divisão 01</p>
</div>
<div>
  <p>Divisão 02</p>
</div>
<div>
  <p>Divisão 03</p>
</div>
```

Claro que podemos incluir qualquer outro elemento HTML, ou seja, outras tags, dentro de uma div HTML.

MÓDULO II

Atributos da Div HTML

O Elemento HTML Div costuma ter entre seus principais atributos:

- **id** – Define uma ID;
- **class** – Define uma classe;
- **title** – Define um título;
- **height** – Define uma altura;
- **width** – Define uma largura.

Porém, com exceção do **ID**, da **class** e do **title**, todos os demais atributos já são considerados depreciados. Portanto, dessa forma, podemos definir a altura, a largura e a cor de fundo através do próprio **estilo CSS**, assim como diversas outras características.

O **HTML Div** também suporta receber todos os **atributos globais**, assim como também os **atributos eventuais**.

MÓDULO II

Alterando a cor de fundo/background da HTML Div

Antes da versão 5 do **HTML** era possível utilizar o atributo **bgcolor** para poder escolher uma cor de background para a div. Porém, a partir da versão 5, o **HTML** passou a não ter mais suporte a este atributo.

Portanto, para poder alterar a cor de fundo de uma div, devemos estilizar através do CSS. Para poder modificar a cor de fundo, basta utilizar a propriedade background ou background-color do CSS. Podemos utilizar o estilo inline, através do atributo **style="..."**, através do **CSS** dentro das tags **<style>** ou ainda através de um folha de estilo externa.

Para utilizar a propriedade do background ou background-color, basta declarar um valor da cor. Para isso, pode ser utilizado um código hexadecimal de cores, o nome da cor, ou ainda declarar as cores através do código **rgb**, **rgba** ou **hsl**. Portanto, vejamos então abaixo as diversas formas de declarar uma cor de fundo:

MÓDULO II

```
<div style="background-color: red"></div>
```

```
<div style="background-color: #FF0000"></div>
```

```
<div style="background-color: rgb(255,0,0);"></div>
```

```
<div style="background: red"></div>
```

```
<div style="background: #FF0000"></div>
```

```
<div style="background: rgb(255,0,0);"></div>
```

MÓDULO II

Independente de qual dos códigos for utilizado, o resultado será o mesmo, uma div contendo um fundo na cor vermelha.

Vejamos então outro exemplo, agora utilizando a tag **<style>** para declarar o **CSS**. Vamos estilizar todas as **divs** com uma altura de **50px** e largura de **100px**. Posteriormente, iremos definir uma cor para cada classe, de maneiras diferentes (nome da cor, código **hexadecimal**, valor **RGB**). Por fim, vamos incluir no documento três elementos **HTML Div**, cada um com uma classe:

MÓDULO II

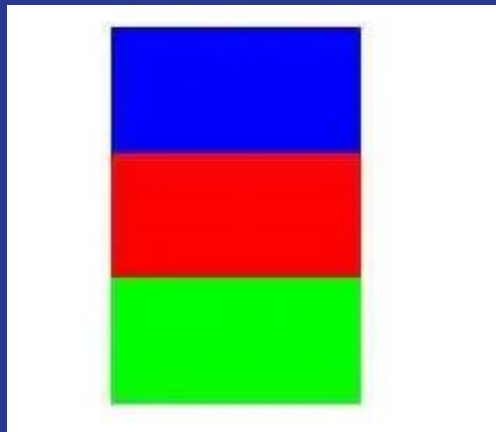
```
<!DOCTYPE html>
<html>
<head>
  <meta charset="utf-8">
  <title>Cores de fundo em HTML Div</title>
  <style type="text/css">
    div{
      width: 100px;
      height: 50px;
    }
    .azul{
      background-color: blue;
    }
    .vermelho{
      background: #FF0000;
    }
    .verde{
      background: rgb(0,255,0);
    }
  </style>
</head>
<body>
  <div class="azul"></div>

  <div class="vermelho"></div>

  <div class="verde"></div>
</body>
</html>
```

MÓDULO II

Com este exemplo, o resultado será conforme a imagem abaixo:



Você pode utilizar uma **tabela de cores HTML** para poder experimentar mais a respeito.

MÓDULO II

ID e classe

Conforme explicado anteriormente, o **HTML Div** comporta receber os atributos **ID** e **Class**. Ou seja, é possível atribuir a uma div um **ID** único ou uma ou mais classes. Isso nos permite formatar um ou mais elementos de forma mais específica e facilitada.

Um **ID** é uma configuração única, ou seja, só podemos utilizar um **ID** em apenas um único elemento. Já uma classe é uma configuração que pode ser utilizada em mais de um elemento.

MÓDULO II

Portanto, vejamos o exemplo abaixo:

```
<div id="cabecalho">
  <h2>Meu documento sobre Div HTML</h2>
</div>
<div id="corpo">
  <div class="sessao_conteudo">
    <h3>Meu conteúdo 01</h3>
    <p>Meu paragrafo contendo texto</p>
    <p>Meu paragrafo contendo texto</p>
  </div>
  <div class="sessao_conteudo">
    <h3>Meu conteúdo 02</h3>
    <p>Meu paragrafo contendo texto</p>
    <p>Meu paragrafo contendo texto</p>
  </div>
</div>
<div id="rodape">
  <p>Todos os direitos reservados</p>
</div>
```

MÓDULO II

Observe que nosso documento **HTML** foi dividido com três **divs** pais, cada uma contendo um **ID** única (**cabecalho**, **corpo** e **rodape**). Dentro da nossa **div** com a **id corpo**, recebemos duas divisões para o conteúdo. Observe que essas divisões receberam uma mesma classe: **sessao_conteudo**.

Portanto, podemos estilizar as **divs** através de suas **ids**, e ainda podemos estilizar todas as **divs** que contêm a mesma classe de uma só vez. Para **IDs**, referenciamos no **CSS** através do “#”, por exemplo: **#cabecalho**. Já para classes, reconhecemos através de um “.”, ou seja, conforme este exemplo:

“**.sessao_conteudo**”.

MÓDULO II

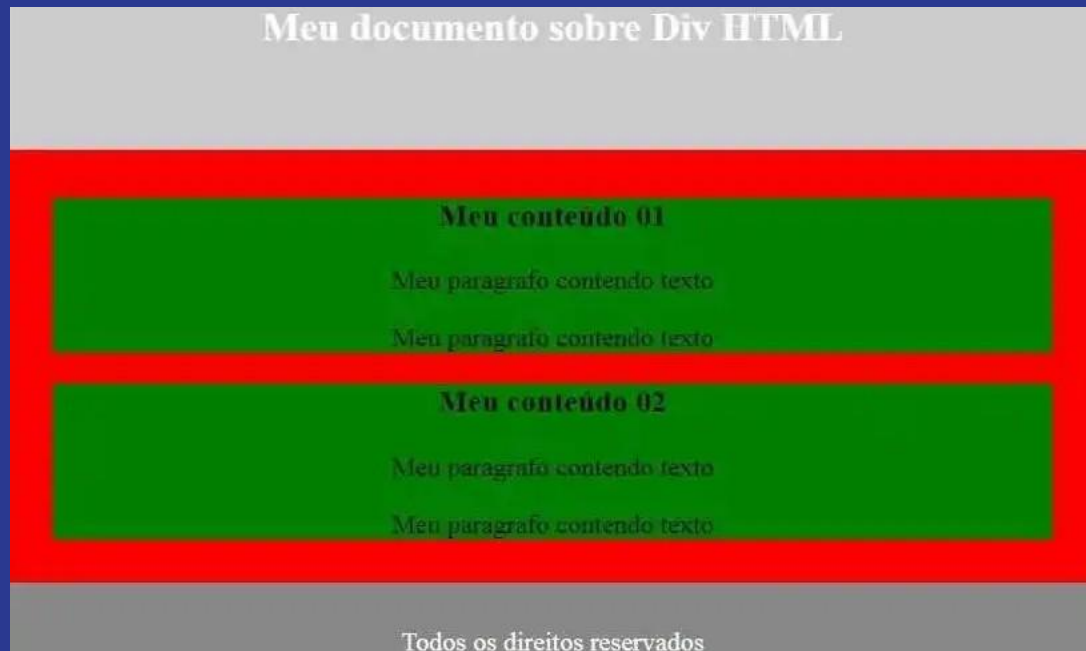
```
<style>
#cabecalho{
    background-color: #cccccc;
    width: 600px;
    padding: 40px 2%;
    color: white;
    text-align: center;
}
#corpo{
    background-color: red;
    width: 600px;
    padding: 10px 2%;
}
#rodape{
    width: 600px;
    padding: 10px 2%;
    background-color: #888888;
    color: white;
    text-align: center;
}
```

Vejamos então o seguinte exemplo de estilo CSS para o documento apresentado no exemplo anterior:

```
.sessao_conteudo{
    text-align: center;
    width: 100%;
    background-color: green;
}
</style>
```

MÓDULO II

Observe então o resultado conforme a imagem abaixo:



Observe que cada **ID** recebeu um estilo diferente. Observe também que ambas as **divs** que contêm a classe **.sessao_conteudo** receberam o mesmo estilo.

MÓDULO II

A diferença entre Div e Span


Uma dúvida muito comum para quem está aprendendo o **HTML** é a diferença entre as tags **<div>** e ****. Ambas as tags representam elementos sem valor semântico e que podem conter outros elementos. Porém, geralmente um **** costuma conter um texto dentro dele. Ainda assim, a principal diferença entre elas é que enquanto a **<div>** tem como padrão o display **block**, a tag **** tem como padrão o display **inline-block**.

Vejamos então o código abaixo, utilizando apenas um texto tanto nas **divs** quanto nos **span**:

```
<span>meu span 1 </span>  
<span>meu span 2 </span>  
<div>minha div 1</div>  
<div>minha div 2</div>
```

MÓDULO II

Dessa forma, observe o resultado conforme a imagem abaixo:



```
meu span 1 meu span 2  
minha div 1  
minha div 2
```

Observe que os span não receberam uma quebra de linha, e por isso ficaram um do lado do outro. Já a HTML Div automaticamente realiza uma quebra de linha.

MÓDULO II

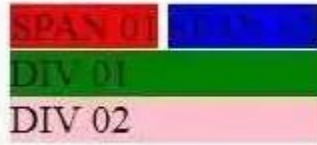
Para efeitos visuais, vamos agora utilizar o seguinte exemplo, incluindo um estilo CSS:

```
<!DOCTYPE html>
<html>
<head>
  <meta charset="utf-8">
  <title>HTML Div vs HTML Span</title>
<style>
  .vermelho{
    background-color: red;
  }
  .azul{
    background-color: blue;
  }
  .verde{
    background-color: green;
  }
  .rosa{
    background-color: pink;
  }
</style>
```

```
</head>
<body>
  <span class="vermelho"></span>
  <span class="azul"></span>
  <div class="verde"></div>
  <div class="rosa"></div>
</body>
</html>
```

MÓDULO II

Com este exemplo, o resultado será conforme a imagem abaixo:



Dessa forma, fica mais evidente a diferença prática entre um elemento **HTML Div** e um elemento **span**.

MÓDULO II

Posicionamento de HTML Div

Através das propriedades do CSS podemos utilizar diversas posições. Vamos explicar aqui algumas delas. Para poder alterar a posição de um elemento HTML através do CSS, podemos modificar a propriedade de posição, assim como utilizando técnicas como margem e flutuação, dentre outras. Dessa forma, vamos entender as propriedades **position**, **margin** e **float** do CSS.

MÓDULO II

CSS Position

A propriedade `position` permite definir o tipo de posição utilizada para o elemento.

Com isso, podemos definir uma *posição relativa*, uma *posição absoluta*, ou ainda uma *posição fixa*. Também há a *posição estática* e a *posição pegajosa*. Vamos estudar as posições *relativa*, *absoluta* e *fixa*:

Posição relativa

Como sugere o nome, especifica uma posição relativa do elemento em relação ao elemento pai ou ao elemento anterior. Esta posição pode ser definida com as propriedades **top**, **bottom**, **left** e **right**.

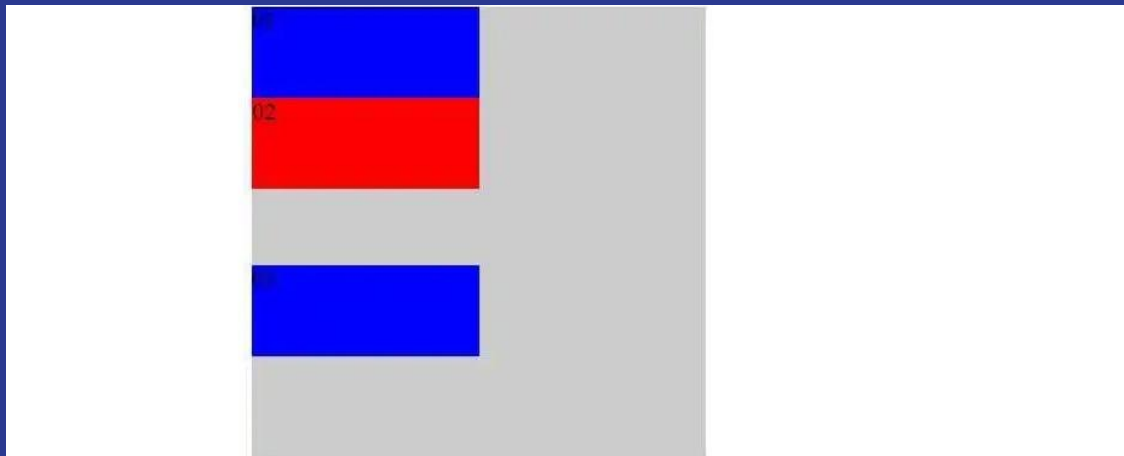
MÓDULO II

```
<!DOCTYPE html>
<html>
<head>
  <meta charset="utf-8">
  <title>HTML Div Position</title>
  <style>
    #elemento_pai{
      width: 300px;
      height: 300px;
      background-color: #ccc;
    }
    .posrelativa{
      position: relative;
      width: 150px;
      height: 60px;
    }
    .vermelho{
      background-color: red;
    }
    .azul{
      background-color: blue;
```

```
      .top50{
        top: 50px;
      }
    </style>
  </head>
  <body>
    <div id="elemento_pai">
      <div class="posrelativa azul">01</div>
      <div class="posrelativa vermelho">02</div>
      <div class="posrelativa azul top50">03</div>
    </div>
  </body>
</html>
```

MÓDULO II

Observe que aplicamos a todas as *divs filhas* uma classe contendo uma posição relativa. Observe também que a *terceira div*, através da classe **.top50**, incluímos um top de **50px**. Como se trata de uma posição relativa, a **div.top50** receberá uma distância de 50px de topo em relação à div anterior. Dessa forma, teremos um resultado conforme o exemplo da imagem abaixo:



MÓDULO II

Position absolute

A posição absoluta representa uma posição que levará em consideração a distância em relação à div pai relativa. Portanto, ela terá uma posição independente dos demais elementos, podendo até mesmo ficar por cima ou por baixo dos demais elementos. Esta posição pode ser definida com as propriedades **top**, **bottom**, **left** e **right**.

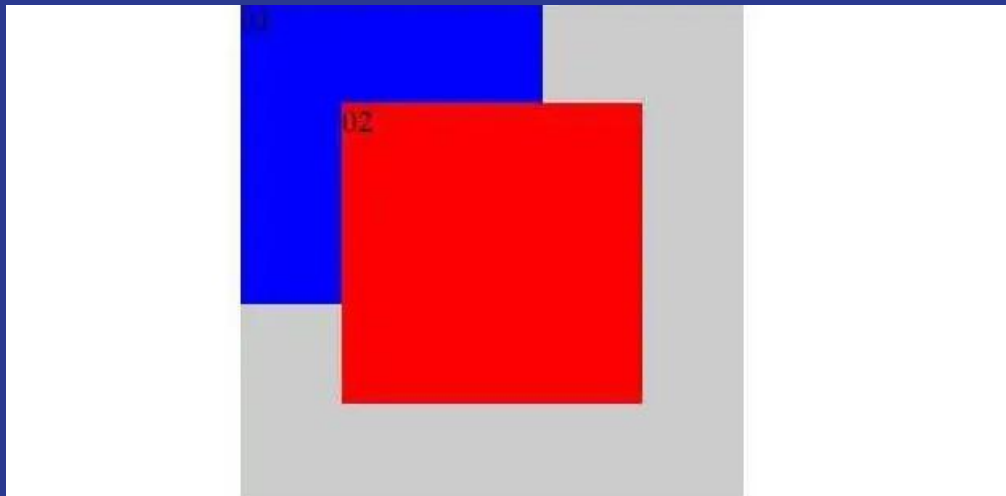
MÓDULO II

```
<!DOCTYPE html>
<html>
<head>
  <meta charset="utf-8">
  <title>HTML Div Position</title>
  <style>
    #elemento_pai{
      width: 250px;
      height: 250px;
      background-color: #ccc;
      position: relative;
    }
    .posrelativa{
      position: relative;
      width: 150px;
      height: 150px;
    }
  </style>
</head>
<body>
  <div id="elemento_pai">
    <div class="posrelativa">01</div>
    <div class="posabsoluta">02</div>
  </div>
</body>
</html>
```

```
.posabsoluta{
  position: absolute;
  top: 50px;
  left: 50px;
  width: 150px;
  height: 150px;
}
.azul{
  background-color: blue;
}
.vermelho{
  background-color: red;
}
</style>
</head>
<body>
  <div id="elemento_pai">
    <div class="posrelativa azul">01</div>
    <div class="posabsoluta vermelho">02</div>
  </div>
</body>
</html>
```

MÓDULO II

Dessa forma, o resultado do exemplo será conforme na imagem abaixo:



Observe que utilizamos uma posição absoluta ao bloco vermelho. Como o elemento pai (**div#elemento_pai**) possui uma posição relativa, a div com posição absoluta levará em consideração apenas a posição do elemento pai. Como definimos um valor de distância do topo e distância da esquerda de 50px, chegamos ao exemplo acima. Se não tivéssemos atribuído às propriedades **top** e **left**, a mesma ficaria por cima do bloco azul.

MÓDULO II

Position fixed

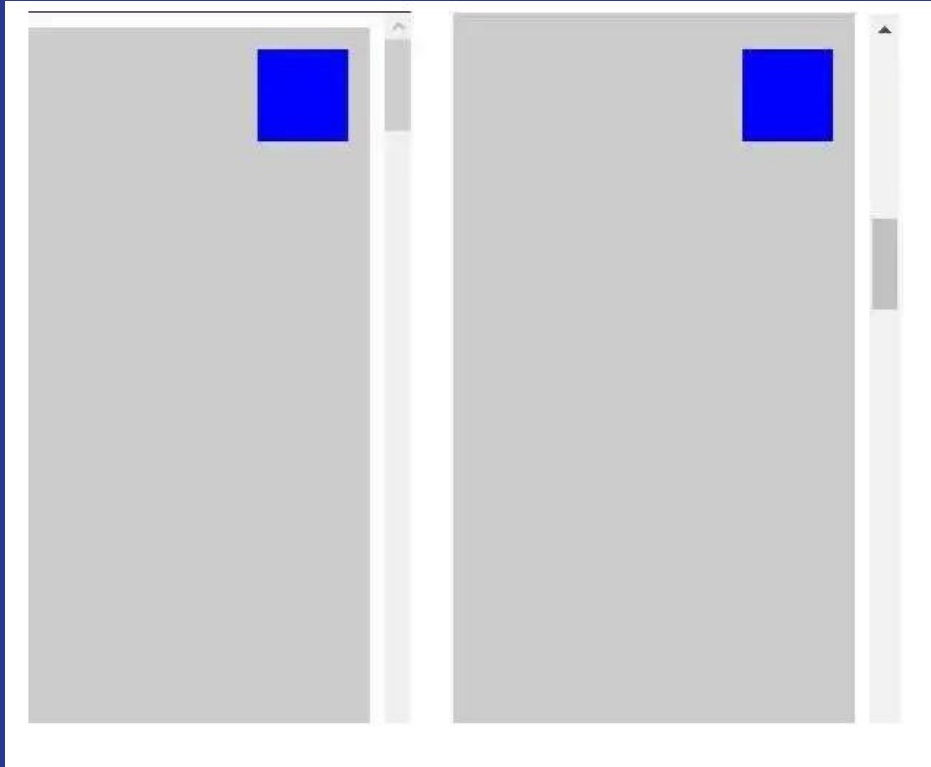
A posição fixa é muito parecida com a posição absoluta. Porém, ela se mantém fixa, ou seja, independente da posição da barra de rolagem (**scroll**), este elemento se encontrará na mesma posição da tela.

Esta posição pode ser definida com as propriedades **top**, **bottom**, **left** e **right**.

MÓDULO II

```
<!DOCTYPE html>
<html>
<head>
  <meta charset="utf-8">
  <title>HTML Div Position</title>
  <style>
    #corpo{
      height: 10000px;
      width: 100%;
      background-color: #ccc;
    }
    #fixo{
      position: fixed;
      top: 20px;
      right: 20px;
      width: 50px;
      height: 50px;
      background-color: blue;
    }
  </style>
</head>
<body>
  <div id="corpo"></div>
  <div id="fixo"></div>
</body>
</html>
```

MÓDULO II



Observe então nas **imagens abaixo a mesma página**, porém com posição da barra de rolagem diferentes. Mesmo assim, a **div#fixo** se mantém na mesma posição em relação à tela:

MÓDULO II

CSS margin

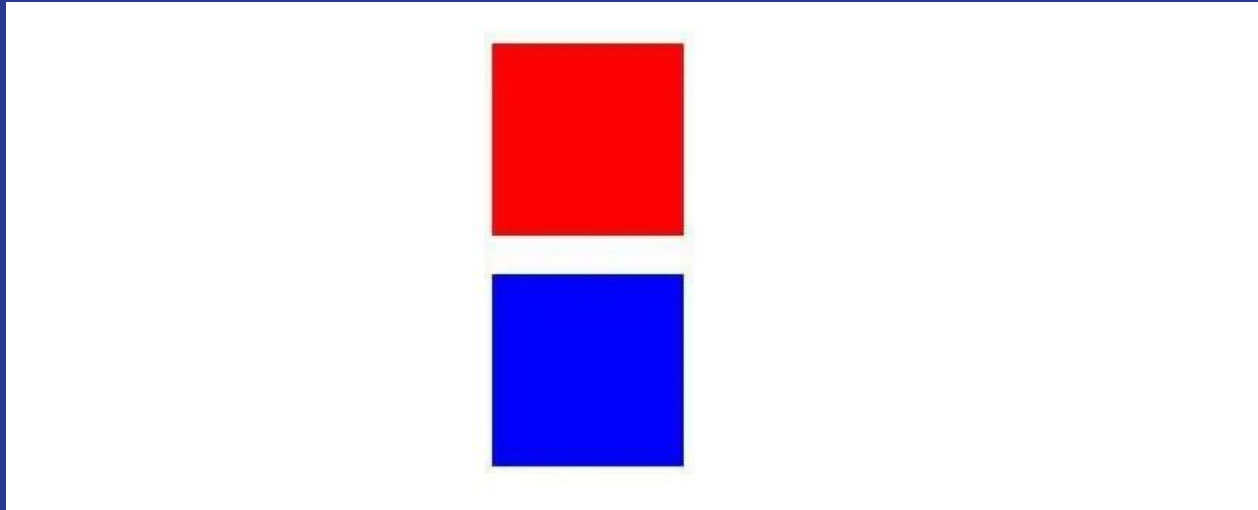
Através de uma margem, é possível manipular a posição de um elemento HTML. Com isso podemos posicionar um elemento **HTML Div** através da **margin**. Existem quatro tipos de margin: **margin-top**, **margin-right**, **margin-bottom**, **margin-left**. Ou seja, margem superior, margem à direita, margem inferior e margem à esquerda. Também podemos utilizar a propriedade **margin**, contendo um, dois ou quatro valores. Caso utilize apenas um valor, esse valor será aplicado a todas as margens. Caso utilize dois valores, como, por exemplo, **margin:10px 5px**, o primeiro valor será para as margens superior e inferior, já o segundo valor será para as margens direita e esquerda. Por fim, utilizando quatro valores, o primeiro representa a margem superior, o segundo margem a direita, o terceiro a margem inferior e o quarto a margem a esquerda.

MÓDULO II

```
<!DOCTYPE html>
<html>
<head>
  <meta charset="utf-8">
  <title>HTML Div Margin</title>
  <style>
    div{
      width: 100px;
      height: 100px;
    }
    .div_1{
      background-color: red;
      margin-bottom: 20px;
    }
    .div_2{
      background-color: blue;
    }
  </style>
</head>
<body>
  <div class="div_1"></div>
  <div class="div_2"></div>
</body>
</html>
```

MÓDULO II

Dessa forma, nosso elemento `div.div_1` fica com uma margem inferior de `20px`, portanto o resultado será como na imagem abaixo:



MÓDULO II

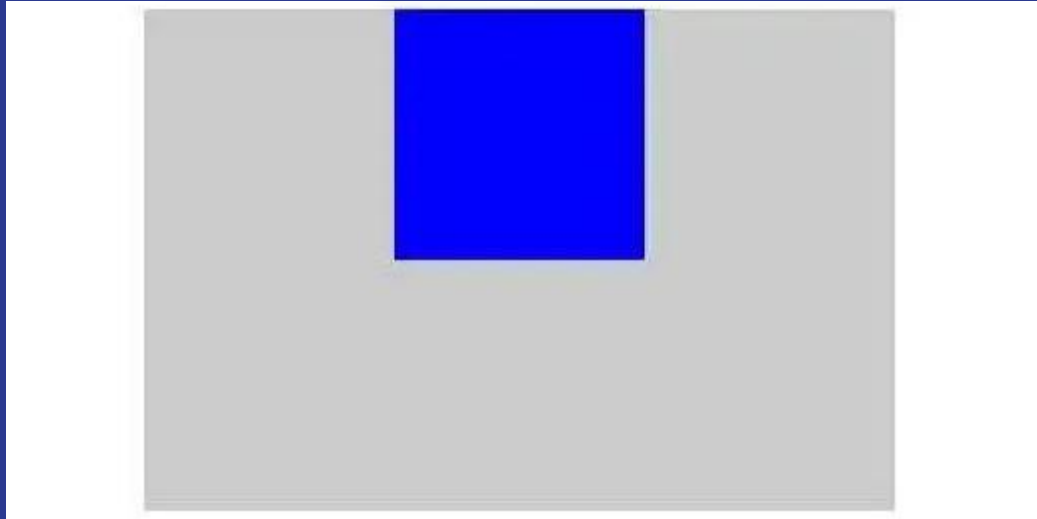
Podemos ainda utilizar a margem para centralizar uma HTML div. Para isso, basta definir que as margens à direita e à esquerda fiquem no modo automático. Dessa forma, a margem será definida igualmente para ambos os lados e, portanto, o elemento ficará centralizado. Vejamos o exemplo abaixo:

MÓDULO II

```
<!DOCTYPE html>
<html>
<head>
  <meta charset="utf-8">
  <title>HTML Div Margin</title>
  <style>
    div#pai{
      width: 300px;
      height: 200px;
      background-color: #ccc;
    }
    div#filha{
      margin-right: auto;
      margin-left: auto;
      width: 100px;
      height: 100px;
      background-color: blue;
    }
  </style>
</head>
<body>
  <div id="pai">
    <div id="filha"></div>
  </div>
</body>
</html>
```

MÓDULO II

Observe que o elemento ficou centralizado. Poderíamos também ter utilizado a propriedade de margem dessa forma: **margin: 0 auto;**.



MÓDULO II

Este recurso é muito utilizado para a criação de contêiner com um limite de largura que deve se manter sempre centralizado na página. Portanto, é importante entender como funciona esta técnica, pois, muito provavelmente, você precisará utilizá-la ou verá em diversas folhas de estilo.

MÓDULO II

CSS float

Existe uma propriedade muito útil do **CSS** chamado de **float**, ou **flutuação**. Desta forma é possível manter um elemento flutuado. Por exemplo, quando queremos manter duas divs em uma mesma linha, podemos flutuar elas. Ou ainda quando queremos manter um texto e uma div flutuante ao lado desse texto. Podemos utilizar o **float:left**; para flutuar o elemento à esquerda ou ainda o **float:right**; para flutuar o elemento à direita. Para limpar uma flutuação podemos usar o **float:none**;. Porém, a flutuação é uma propriedade um pouco imprecisa, portanto, recomenda-se utilizar aliado a propriedades de margem ou padding. Para que não ocorra erros de renderização, também é importante, sempre que utilizar uma flutuação, limpar a mesma com a propriedade clear.

MÓDULO II

```
<!DOCTYPE html>
<html>
<head>
  <meta charset="utf-8">
  <title>HTML Div Float</title>
  <style>
    div.container{
      width: 400px;
      background-color: #ccc;
      height: auto;
    }
    div{
      width: 100px;height: 100px;
    }
    .esquerda{
      background-color: green;
      float: left;
    }
  </style>
</head>
<body>
  <div class="container">
    <div class="esquerda">Div 01</div>
    <div class="direita">Div 02</div>
    <div style="clear: both;"></div>
  </div>
</body>
</html>
```

```
.direita{
  background-color: blue;
  float: right;
}
</style>
</head>
<body>
  <div class="container">
    <div class="esquerda">Div 01</div>
    <div class="direita">Div 02</div>
    <div style="clear: both;"></div>
  </div>
</body>
</html>
```

MÓDULO II

Observe que no exemplo utilizamos uma div pai, com a classe **.container**, para poder conter os elementos dentro de até 400px. Posteriormente, incluímos duas divs, cada um contendo uma classe com flutuação para um dos lados. Por fim, utilizamos uma div com o estilo “**clear:both**” para poder limpar o efeito da flutuação. Vejamos então o resultado abaixo:



MÓDULO II

Caso não utilizássemos a propriedade clear ao final dos elementos flutuados, observe que o elemento pai, que contém uma altura automática, sumiria, portanto, geraria uma falha no layout:

```
<div class="container">  
  <div class="esquerda">Div 01</div>  
  <div class="direita">Div 02</div>  
</div>
```



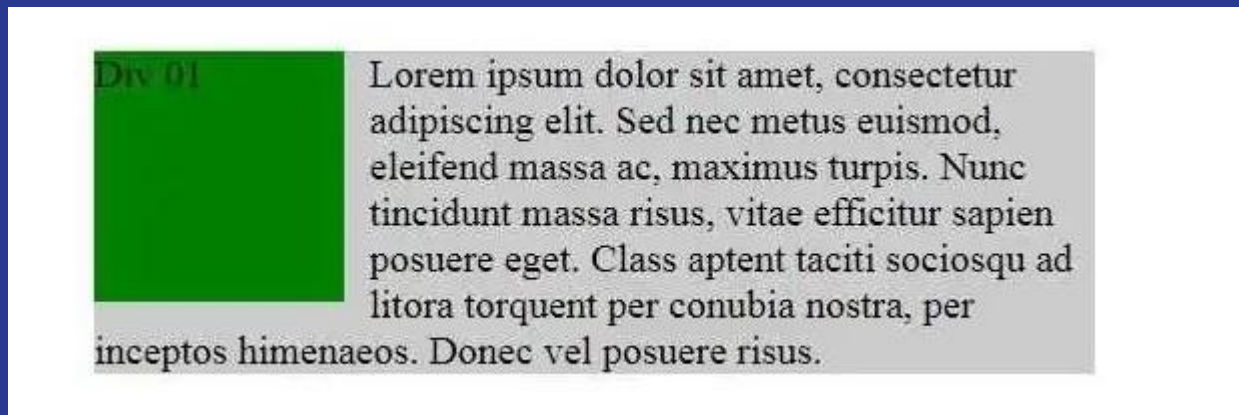
MÓDULO II

Vamos agora ver uma forma de utilizar uma div flutuando ao lado de um texto:

```
<!DOCTYPE html>
<html>
<head>
  <meta charset="utf-8">
  <title>HTML Div Float</title>
  <style>
    div.container{
      width: 400px;
      background-color: #ccc;
      height: auto;
    }
    .flutuante{
      background-color: green;
      float: left;
      width: 100px;height: 100px;
      margin-right: 10px;
      margin-bottom: 10px;
    }
  </style>
</head>
<body>
  <div class="container">
    <div class="flutuante">Div 01</div>
    <p>Lorem ipsum dolor sit amet, consectetur adipiscing elit. Sed nec metus euismod, eleifend massa ac, maximus turpis. Nunc tincidunt massa risus, vitae efficitur sapien posuere eget. Class aptent taciti sociosqu ad litora torquent per conubia nostra, per inceptos himenaeos. Donec vel posuere risus.</p>
    <div style="clear: both;"></div>
  </div>
</body>
</html>
```

MÓDULO II

Dessa forma, obtemos o resultado do exemplo conforme a imagem abaixo:



MÓDULO II

Portanto, podemos utilizar o **float** para diversas situações, principalmente em casos envolvendo texto e **HTML Div**. Isso permite criar diversos elementos dentro do layout de nossas páginas web.

Porém, apesar dos conhecimentos sobre flutuação serem muito importantes, já são considerados ultrapassados para o desenvolvimento completo de layouts. Portanto, recomendamos utilizar a flutuação apenas em casos realmente necessário e não como elemento principal do seu layout.

MÓDULO II

EXERCÍCIOS...

MÓDULO II

FIM...