



**i** This page was translated from English by the community. [Learn more](#) and join the MDN Web Docs community.

# Iniciando com HTML

[Menu: Introduction to HTML](#)[Próxima](#)

Neste artigo nós abordamos os princípios básicos do HTML, para você começar. Definimos os elementos, atributos e todos os outros termos importantes que você possa ter ouvido e onde eles se encaixam na linguagem. Também mostramos como um elemento HTML é estruturado, como uma página HTML típica é estruturada e explicamos outras importantes características básicas da linguagem. Ao longo do caminho, nós brincaremos com um pouco de HTML, para despertar seu interesse!

Pré-requisitos:	Básico de informática, <a href="#">software básico instalado</a> e conhecimento básico de como <a href="#">trabalhar com arquivos</a> .
Objetivos:	Obter uma familiaridade básica com a linguagem HTML e adquirir um pouco de prática escrevendo alguns elementos HTML.

## O que é HTML?

[HTML](#) (HyperText Markup Language) não é uma linguagem de programação, é uma *linguagem de marcação* utilizada para dizer ao seu navegador como estruturar a página web que você visita. A página pode ser tanto complicada como simples quanto o desenvolvedor web desejar que seja. HTML consiste em uma série de [elementos](#) que você usa para anexar, envolver ou *marcar* diferentes partes do conteúdo para que apareça ou aja de uma certa maneira. O fechamento das [tags](#) pode transformar uma parte do

conteúdo dentro do elemento em um link para direcionar à uma outra página web, colocar as palavras em *itálico*, e assim por diante. Por exemplo, observe o conteúdo a seguir:

Meu gato é muito mal-humorado.

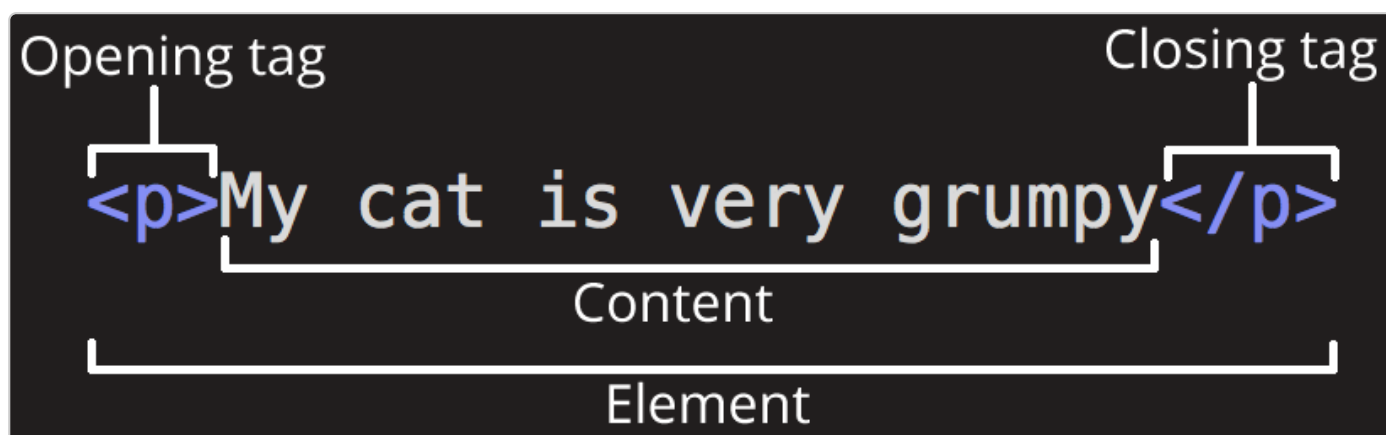
Se nós quisermos que a linha permaneça, nós podemos especificar que é um parágrafo, fechando-a com a elemento de parágrafo( `<p>` ):

```
<p>Meu gato é muito mal-humorado.</p>
```



## Anatomia de um elemento HTML

Vamos explorar nosso elemento parágrafo um pouco mais:



As partes principais do elemento são:

1. **Tag de abertura:** Consiste no nome do elemento ( neste caso: `p` ), envolvido entre **parênteses angulares** de abertura e fechamento. Isso indica onde o elemento começa, ou inicia a produzir efeito — neste caso, onde o parágrafo se inicia.
2. **Tag de fechamento:** É o mesmo que a tag de abertura, exceto que este inclui uma barra diagonal antes do nome do elemento. Indica onde o elemento termina — neste caso, onde fica o fim do parágrafo. Falhar em incluir o fechamento de uma tag é um erro comum para iniciantes e pode levar a resultados estranhos.
3. **O conteúdo:** Este é o conteúdo do elemento, que neste caso é somente texto.
4. **O elemento:** A tag de abertura, mais a tag de fechamento, mais o conteúdo, é igual ao elemento.

# Aprendizado ativo: criando seu primeiro elemento HTML

Edite a linha abaixo na área *Entrada* colocando-a entre as tags `<em>` e `</em>` (adicione o `<em>` antes para abrir o elemento, e `</em>` depois, para fechar o elemento). Isto dará à linha uma ênfase em itálico! Você poderá ver as mudanças efetuadas no momento na área *Saída*.

Caso você cometa um erro, você pode usar o botão *Resetar* para desfazer a ação incorreta. Se você realmente não souber o que fazer, pressione o botão *Mostrar solução* para visualizar a resposta.

## Saída ao vivo

Este é meu texto.

## Código editável

Pressione Esc para afastar o foco da área de código (Tab insere um caractere de tabulação).

```
Este é meu texto.
```

Resetar

Mostrar solução

## Aninhando elementos

Elementos podem ser inseridos dentro de outros elementos — isto é chamado de **aninhamento**. Se nós quisermos dizer que nosso gato é **muito** mal-humorado, nós poderemos envolver a palavra "muito" com o elemento `<strong>`, que significa enfatizar fortemente a palavra:

```
<p>Meu gato é <strong>muito</strong> mal-humorado.</p>
```



No entanto, você precisa garantir que seus elementos estejam adequadamente aninhados: no exemplo acima nós abrimos o elemento `p` primeiro, e então o elemento `strong`,

portanto temos que fechar o elemento `strong` primeiro, depois o `p`. O código a seguir está errado:

```
<p>Meu gato é <strong>muito mal-humorado.</p></strong>
```



Os elementos devem abrir e fechar corretamente para que eles fiquem claramente dentro ou fora do outro. Caso eles se sobreponham, como no exemplo acima, então o seu navegador tentará adivinhar o que você quis dizer, e talvez você obtenha resultados inesperados. Então não faça isso!

## Elementos em bloco versus elementos inline

Há duas categorias importantes no HTML, que você precisa conhecer. Eles são elementos em bloco e elementos inline.

- Elementos em bloco formam um bloco visível na página — eles aparecerão em uma nova linha logo após qualquer elemento que venha antes dele, e qualquer conteúdo depois de um elemento em bloco também aparecerá em uma nova linha. Elementos em bloco geralmente são elementos estruturais na página que representam, por exemplo: parágrafos, listas, menus de navegação, rodapés etc. Um elemento em bloco não seria aninhado dentro de um elemento inline, mas pode ser aninhado dentro de outro elemento em bloco.
- Elementos inline (na linha) são aqueles que estão contidos dentro de elementos em bloco envolvem apenas pequenas partes do conteúdo do documento e não parágrafos inteiros ou agrupamentos de conteúdo. Um elemento inline não fará com que uma nova linha apareça no documento: os elementos inline geralmente aparecem dentro de um parágrafo de texto, por exemplo: um elemento `<a>` (hyperlink) ou elementos de ênfase como `<em>` ou `<strong>`.

Veja o seguinte exemplo:

```
<em>primeiro</em><em>segundo</em><em>terceiro</em>
```

```
<p>quarto</p><p>quinto</p><p>sexto</p>
```



O elemento `<em>` é inline, então como você pode ver abaixo, os três primeiros elementos ficam na mesma linha uns dos outros sem espaço entre eles. O `<p>`, por outro lado, é um elemento em bloco, então cada elemento aparece em uma nova linha, com espaço acima e abaixo de cada um (o espaçamento é devido à [estilização CSS](#) padrão que o browser aplica aos parágrafos).

*primeirosegundoterceiro*

quarto

quinto

sexto

**Nota:** o HTML5 redefiniu as categorias de elemento em HTML5: veja [Categorias de conteúdo de elementos](#). Enquanto essas definições são mais precisas e menos ambíguas que as anteriores, elas são muito mais complicadas de entender do que "em bloco" e "inline", então usaremos estas ao longo deste tópico.

**Nota:** Os termos "bloco" e "inline", conforme usados neste tópico, não devem ser confundidos com os [tipos de caixas CSS](#) com os mesmos nomes. Embora eles se correlacionem por padrão, alterar o tipo de exibição CSS não altera a categoria do elemento e não afeta em quais elementos ele pode conter e em quais elementos ele pode estar contido. Um dos motivos pelos quais o HTML5 abandonou esses termos foi evitar essa confusão bastante comum.

**Nota:** Você pode encontrar páginas de referência úteis que incluem uma lista de elementos inline e em bloco — veja [elementos em bloco](#) e [elementos inline](#).

## Elementos vazios

Nem todos os elementos seguem o padrão acima de: tag de abertura, conteúdo, tag de fechamento. Alguns elementos consistem apenas em uma única tag, que é geralmente usada para inserir/incorporar algo no documento no lugar em que ele é incluído. Por

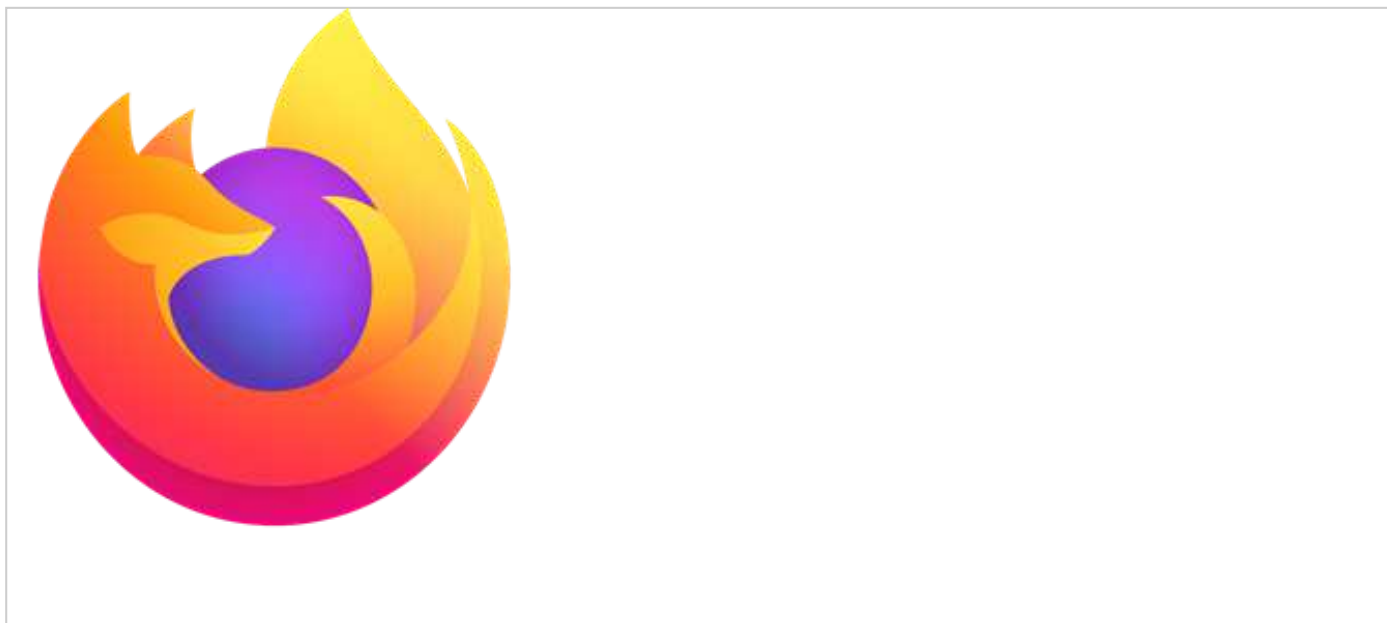
exemplo, o elemento `<img>` insere uma imagem em uma página na posição em que ele é incluído:

```

```



Isto exibirá em sua página:



**Nota:** Elementos vazios são também chamados de *void elements*.

## Atributos

Elementos também podem conter atributos, que se apresentam da seguinte forma:

Attribute

```
<p class="editor-note">My cat is very grumpy</p>
```

Atributos contém informação extra sobre o elemento, mas que você não deseja que apareça no conteúdo. Neste caso, o atributo `class` permite que você dê ao elemento um nome de identificação, que pode ser usada mais tarde para direcionar informação de estilo ao elemento e outras coisas.

Um atributo deve conter:

1. Um espaço entre ele e o nome do elemento (ou o atributo anterior, caso o elemento já contenha um ou mais atributos.)
2. O nome do atributo, seguido por um sinal de igual.
3. Um valor de atributo, com aspas de abertura e fechamento em volta dele.

## Aprendizado ativo: Adicionando atributos a um elemento

Outro exemplo de um elemento é `<a>` — isso significa "âncora" e fará com que a parte do texto que ele envolve vire um link. Isso pode ter vários atributos, mas os mais comuns são os seguintes:

- O valor desse atributo especifica o endereço da web para o qual você deseja que o link aponte; onde o navegador irá quando o link for clicado. Por exemplo `href="https://www.mozilla.org/"`.
- `title`: O atributo `title` especifica uma informação extra sobre o link, assim como o assunto da página que está sendo linkada. Por exemplo `title="Homepage da Mozilla"`. Isto será exibido como uma *tooltip* (*dica de contexto*) quando passarmos o mouse sobre o link.

Edite a linha abaixo na área de Entrada para transformá-la em um link para o seu site favorito.

1. Primeiro, adicione o elemento `<a>`.
2. Segundo, adicione o atributo `href` e o atributo `title`.
3. Por último, especifique o atributo `target` para abrir o link em uma nova aba.

Você poderá ver as atualizações das alterações ao vivo na área Saída. Você deve ver um link que, quando passa o mouse sobre ele, exibe o valor do atributo `title` e, quando clicado, navega para o endereço da web no atributo `href`. Lembre-se de que você precisa incluir um espaço entre o nome do elemento e cada atributo.

Caso você cometa um erro, você poderá desfazê-lo usando o botão `_Reset_ar`. Caso você realmente não saiba como fazer, pressione o botão *Mostrar solução* para ver a resposta.

## Saída ao vivo

Um link para o meu site favorito.

## Código editável

Pressione Esc para afastar o foco da área de código (Tab insere um caractere de tabulação).

```
<p>Um link para o meu site favorito.</p>
```

Resetar

Mostrar solução

## Atributos booleanos

Às vezes você verá atributos escritos sem valores — isso é permitido nos chamados atributos booleanos, e eles podem ter somente um valor, que é geralmente o mesmo nome do atributo. Por exemplo, o atributo `disabled` você pode atribuir para os elementos de entrada de formulários, se desejar que estes estejam desativados (acinzentados), para que o usuário não possa inserir nenhum dado neles.

```
<input type="text" disabled="disabled">
```

De forma abreviada, é perfeitamente permitido escrever isso da seguinte maneira (também incluímos um elemento de entrada de formulário não desativado para referência, para dar uma idéia do que está acontecendo):

```
<!-- o uso do atributo disabled impede que o usuário final insira texto na caixa de entrada -->
```

```
<input type="text" disabled>
```

```
<!-- O usuário pode inserir texto na caixa de entrada a seguir, pois não contém o atributo disabled -->
```

```
<input type="text">
```





Ambos resultarão em uma *Saída* da seguinte forma:

## Omitindo as aspas dos valores do atributo

Olhando a World Wide Web (internet), você encontrará todos os tipos de estilos de marcação HTML, incluindo valores de atributos sem as aspas. Isso é permitido em algumas circunstâncias, mas irá quebrar sua marcação em outras. Por exemplo, se voltarmos ao exemplo anterior de link , nós podemos escrever a versão básica deste somente com o atributo `href` , da seguinte forma:

```
<a href=https://www.mozilla.org/>site favorito</a>
```

No entanto, assim que adicionamos o atributo `title` neste elemento, a marcação resultará em erro:

```
<a href=https://www.mozilla.org/ title=The Mozilla homepage>site favorito</a>
```



Neste ponto, o navegador irá interpretar errado sua marcação, de modo a pensar que o atributo `title` trata-se, na verdade, de três atributos: o atributo `title` com o valor "The", e dois atributos booleanos, `Mozilla` e `homepage` . Esta obviamente não era a intenção e irá causar erros ou um comportamento inesperado no código, assim como visto no exemplo abaixo. Tente colocar o mouse em cima do link para visualizar qual é o título que aparece!

Nossa recomendação é *sempre incluir as aspas nos valores dos atributos* — isto evita inúmeros problemas, além de resultar em um código mais legível.

## Aspas simples ou duplas?

Você pode perceber que os valores dos atributos exemplificados neste artigo estão todos com aspas duplas, mas você poderá ver aspas simples no HTML de algumas pessoas. Esta é puramente uma questão de estilo e você pode se sentir livre para escolher qual prefere. As duas linhas de código a seguir são equivalentes:

```
<a href="http://www.example.com">Um link para o exemplo.</a>
```



```
<a href='http://www.example.com'>Um link para o exemplo.</a>
```

Entretanto, você deve se certificar de não misturar os dois tipos de aspas juntos. O exemplo a seguir está errado!

```
<a href="http://www.exemplo.com">Um link para o exemplo.</a>
```



Se utilizar um tipo de aspas no seu HTML, você pode inserir o outro tipo de aspas no texto, por exemplo, que não ocorrerá erro, desta forma:

```
<a href="http://www.example.com" title="Isn't this fun?">A link to my example.</a>
```



No entanto, se você quiser incluir aspas, dentro de aspas onde ambas as aspas são do mesmo tipo (aspas simples ou aspas duplas), será necessário usar entidades HTML para as aspas. Por exemplo, isso irá quebrar:

```
<a href='http://www.example.com' title='Isn't this fun?'>A link to my example.</a>
```



Então você precisa fazer isso:

```
<a href='http://www.example.com' title='Isn&#39;t this fun?'>A link to my example.  
</a>
```



# Anatomia de um documento HTML

Já vimos os conceitos básicos dos elementos individuais do HTML, mas eles não são muito úteis sozinhos, Vamos aprender como estes elementos individuais são combinados entre si para criar uma página HTML inteira:

```
<!DOCTYPE html>  
<html>  
  <head>  
    <meta charset="utf-8">  
    <title>My test page</title>  
  </head>  
  <body>  
    <p>This is my page</p>  
  </body>  
</html>
```



Neste código nós temos:

1. `<!DOCTYPE html>`: O doctype. Nas névoas do tempo, quando o HTML era recente (por volta de 1991/2), doctypes funcionavam como links para uma série de regras as quais uma página HTML tinha que seguir para ser considerada uma página com um bom HTML, o que poderia significar a verificação automática de erros e outras coisas úteis. Ele costumava ser assim:

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"  
"https://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
```

No entanto, atualmente, ninguém se importa com eles, e eles são realmente apenas um artefato histórico que precisa ser incluído para que tudo funcione corretamente. `<!DOCTYPE html>` é a menor cadeia de caracteres que conta como um doctype válido; é tudo o que você realmente precisa saber.

2. `<html></html>`: O elemento `<html>` envolve o conteúdo da página inteira e é conhecido como o "elemento raiz" da página HTML.

3. `<head></head>` : O elemento `<head>` atua como um container para todo o conteúdo da página HTML que não é visível para os visitantes do site. Isso inclui palavras-chave e a descrição da página que você quer que apareça nos resultados de busca, o CSS para estilizar o conteúdo da página (apesar de ser recomendado fazer isso em um arquivo separado), declaração de conjunto de caracteres, e etc. Você aprenderá mais sobre isso no próximo artigo da série.
4. `<meta charset="utf-8">` : Este elemento define o tipo da codificação dos caracteres que o seu documento deve usar, neste caso, para o tipo UTF-8, que inclui a maioria dos caracteres das línguas humanas escritas. Essencialmente, ele consegue lidar com qualquer tipo de conteúdo textual que você puder colocar no documento. Não há motivos para não indicar essa informação e esse elemento ajuda a evitar vários problemas na sua página.
5. `<title></title>` : O elemento `<title>` . Isto define o título de sua página, que aparecerá na guia do navegador na qual a página está carregada e é usado para descrevê-la quando for salva nos favoritos.
6. `<body></body>` : O elemento `<body>` contém *todo* o conteúdo que você quer mostrar aos usuários quando eles visitarem sua página, como texto, imagens, vídeos, jogos, faixas de áudio reproduzíveis, ou qualquer outra coisa.

## Aprendizado ativo: Adicionando alguns recursos ao documento HTML

Se você quiser experimentar como funciona um documento HTML no seu computador, você pode:

1. Copiar o exemplo de página HTML mostrada acima.
2. Criar um novo documento em seu editor de texto.
3. Colar o código no novo arquivo de texto.
4. Salvar o arquivo com o nome `index.html` .

**Nota:** Você também pode encontrar o template básico de HTML no [MDN Learning Area Github repo](#) [↗](#).

Você pode abrir este arquivo no navegador para ver como o código renderizado se apresenta, e então, editar o código e atualizar a página no navegador para ver o resultado com as mudanças. Inicialmente será exibido assim:



Neste exercício, você pode editar o código localmente em seu computador, com descrito acima, ou você pode editá-lo na janela de exemplo editável abaixo (esta janela editável representa apenas o conteúdo do elemento `<body>` , neste caso). Nós gostaríamos que você seguisse as seguintes etapas:

- Logo abaixo da tag de abertura `<body>` , adicione um título principal para o documento. Isso deve estar dentro da tag de abertura `<h1>` e da tag de fechamento `</h1>` .
- Edite o conteúdo do parágrafo para incluir algum texto sobre algo de seu interesse.
- Faça com que todas as palavras importantes sejam destacadas em negrito, colocando-as dentro da tag de abertura `<strong>` e da tag de fechamento `</strong>` .
- Adicione um link ao seu parágrafo, conforme [explicado anteriormente neste artigo](#).
- conforme explicado anteriormente no artigo. Você receberá pontos de bônus se conseguir vincular a uma imagem diferente (localmente no seu computador ou em outro lugar da web).

Caso você cometa um erro, você poderá desfazê-lo usando o botão *Resetar*. Caso você realmente não saiba como fazer, pressione o botão *Mostrar solução* para ver a resposta.

## Espaços em branco no HTML

Nos exemplos anteriores, você pode ter percebido a presença de espaços em branco nos códigos — isto não é necessário; os dois trechos de códigos a seguir são equivalentes:

```
<p>Dogs are silly.</p>
```



```
<p>Dogs      are  
      silly.</p>
```

Não importa quantos espaços em branco você use (que pode incluir caracteres de espaço, mas também quebras de linha), o analisador de HTML reduz cada um deles em um único espaço ao renderizar o código. Então, por que espaço em branco? A resposta é legibilidade — é muito mais fácil entender o que está acontecendo no seu código, se você o tiver bem formatado, e não apenas agrupado em uma grande confusão. Em nosso

HTML, cada elemento aninhado é indentado em dois espaços a mais do que aquele em que está localizado. Depende de você que estilo de formatação você usa (quantos espaços para cada nível de recuo, por exemplo), mas considere formatá-lo.

## Referências de entidades: incluindo caracteres especiais no HTML

No HTML, os caracteres `<`, `>`, `"`, `'` e o `&` são caracteres especiais. Eles fazem parte da própria sintaxe HTML; portanto, como você inclui um desses caracteres no seu texto? Por exemplo, se você realmente deseja usar um e comercial ou sinal de menor que, e não tenha ele interpretado como código.

Temos que usar referências de caracteres — códigos especiais que representam caracteres e podem ser usados nessas circunstâncias. Cada referência de caractere é iniciada com um e comercial (&) e finalizada por um ponto e vírgula (;).


Caractere literal	Referência de caractere equivalente
<	&lt;
>	&gt;
"	&quot;
'	&apos;
&	&amp;

No exemplo abaixo, você pode ver dois parágrafos, que estão falando sobre tecnologias da web:

```
<p>Em HTML, você define um parágrafo usando o elemento <p>.</p>
```

```
<p>Em HTML, você define um parágrafo usando o elemento &lt;p&gt;.</p>
```

Na saída ao vivo abaixo, você pode ver que o primeiro parágrafo deu errado, porque o navegador pensa que a segunda instância de `<p>` está iniciando um novo parágrafo. O segundo parágrafo parece bom, porque substituímos os parênteses angulares por referências de caracteres.

**i Nota:** A tabela com todas as referências de caracteres disponíveis em HTML pode ser encontrada na Wikipédia: [List of XML and HTML character entity references](#) . Observe que você não precisa usar referências de entidade para outros símbolos, pois os navegadores modernos manipularão os símbolos reais muito bem, desde que a codificação de caracteres do HTML esteja definida como UTF-8.

## Comentários no HTML

Em HTML, assim como na maioria das linguagens de programação, há um mecanismo para escrevermos comentários no código — comentários são ignorados pelo navegador e são invisíveis para o usuário, seu propósito é permitir a inclusão de comentários para descrever como o código funciona, especificar o que cada parte dele faz, e por aí vai. Isso pode ser muito útil se você retornar a uma base de código em que não trabalhou há muito tempo e não se lembrar do que fez — ou se você entregar seu código para outra pessoa trabalhar.

Para transformar uma seção do conteúdo HTML em um comentário, você precisa agrupá-lo nos marcadores especiais `<!--` e `-->`, por exemplo:

```
<p>Eu não estou dentro de um comentário</p>
```



```
<!-- <p>Eu estou!</p> -->
```

Como você pode ver abaixo, o primeiro parágrafo fica visível na saída ao vivo, mas o segundo (que é um comentário) não.



## Sumário

Você chegou ao final do artigo — esperamos que tenha gostado do seu tour pelos princípios básicos do HTML! Nesse ponto, você deve entender como é a linguagem, como ela funciona em um nível básico e ser capaz de escrever alguns elementos e atributos. Este é o lugar perfeito para se estar agora, já que os artigos subsequentes deste módulo abordarão algumas das coisas que você já examinou com mais detalhes e introduzirão alguns novos conceitos da linguagem. Fique ligado!

**i Nota:** Nesse ponto, à medida que você começa a aprender mais sobre HTML, também pode querer explorar os conceitos básicos de Cascading Style Sheets, ou CSS. CSS é a linguagem usada para estilizar suas páginas da web (por exemplo, alterando a fonte ou as cores ou alterando o layout da página). HTML e CSS vão muito bem juntos, como você descobrirá em breve.

## Veja também

- [Aplicando cores a elementos HTML usando CSS](#)

[Menu: Introduction to HTML](#)

[Próxima](#)

## Neste módulo

- [Iniciando com HTML](#)
- [O que está no cabeçalho? Metadados em HTML](#)
- [Fundamentos do texto em HTML](#)
- [Criando links](#)
- [Formatação avançada de texto](#)
- [Estrutura do documento e site](#)

- [Depurando HTML](#)
- [Marcando uma carta](#)
- [Estruturando o conteúdo de uma página](#)

## Found a content problem with this page?

- Edit the page [on Github](#).
- Report the [content issue](#).
- View the source [on GitHub](#).

Want to get more involved? Learn [how to contribute](#).

This page was last modified on 15 de dez. de 2022 by [MDN contributors](#).

**mdn**

Your blueprint for a better internet.



MDN

About

Hacks Blog

Careers

Our communities

MDN Community

MDN Forum

MDN Chat

Support

Product help

Report an issue

Developers

Web Technologies

Learn Web Development

MDN Plus

Visit [Mozilla Corporation's](#) not-for-profit parent, the [Mozilla Foundation](#).

Portions of this content are ©1998–2023 by individual mozilla.org contributors. Content available under [a Creative Commons license](#).