HTML5

Definição e estrutura básica

Em 1991 Tim Berners-Lee criou essa linguagem de marcação para melhorar a comunicação entre ele e seus colegas de trabalho no CERN, desde então já surgiram 5 versões e o HTML se tornou a base da web.

Com o HTML definimos o significado e a estrutura do conteúdo da web e, além de texto, nossas páginas precisam de imagens, vídeos e vários outros formatos e para isso temos os elementos HTML.

Um elemento HTML é formado pela tag de abertura e seus atributos, o conteúdo e uma tag de fechamento. E mais a frente veremos que existem elementos que não tem tag de fechamento.

Com esses elementos podemos agrupar tipos de conteúdo, alterar tamanho e forma de fontes e adicionar diferentes mídias ao nossa página na web.

E agora podemos ver como é a estrutura básica de um arquivo HTML.

A primeira linha do documento deve ser o <!DOCTYPE html>, apesar de parecer um elemento HTML ela apenas diz ao navegador que ele está lidando com um arquivo do tipo HTML5. Os elementos HTML vem logo abaixo.

<html>

A tag html é a raiz do seu documento, todos os elementos HTML devem estar dentro dela. E nela nós informamos ao navegador qual é o idioma desse nosso documento, através do atributo lang, para o português brasileiro usamos pt-BR.

<head>

A tag head contém elementos que serão lidos pelo navegador, como os metadados - um exemplo é o charset, que é a codificação de caracteres e a mais comum é a UTF-8, o JavaScript com a tag script, o CSS através das tags style e link - veremos a diferença quando falarmos sobre CSS - e o título da página com a tag title.

<body>

E dentro da tag body colocamos todo o conteúdo visível ao usuário: textos, imagens, vídeos.

Prática

Como exercício para esse curso iremos construir um site pessoal, e precisamos começar com a estrutura básica que acabamos de ver.

Vamos criar um arquivo index.html e adicionar o doctype e os elementos html, head e body.

Depois adicionaremos os elementos meta e title, no primeiro adicionamos o atributo charset com o valor UTF-8 para dizer ao navegador qual é a codificação dos caracteres e no segundo podemos colocar nosso nome.

E por último escreveremos nosso nome dentro do elemento body apenas para enxergarmos isso no navegador.

Semântica

Durante muitos anos o elemento padrão no HTML era a div, construíamos nosso conteúdo todo baseado nela, e assim nascia a sopa de divs.

Mas em 2014 saiu a quinta versão do HTML, e com ela vieram vários mudanças importantes, como performance e acessibilidade, mas nesse curso introdutório vamos focar na semântica.

A semântica nos permite descrever mais precisamente o nosso conteúdo, agora um bloco de texto não é apenas uma div, agora é um article e tem mais significado assim. E temos vários elementos para ressignificar as divs:

<section>

Representa uma seção genérica de conteúdo quando não houver um elemento mais específico para isso.

<header> Metadados, CSS e Links para arquivos externos.

É o cabeçalho da página ou de uma seção da página e normalmente contém logotipos, menus, campos de busca.

<article>

Representa um conteúdo independente e de maior relevância dentro de uma página, como um post de blog, uma notícia em uma barra lateral ou um bloco de comentários. Um article pode conter outros elementos, como header, cabeçalhos, parágrafos e imagens.

<aside>

É uma seção que engloba conteúdos relacionados ao conteúdo principal, como artigos relacionados, biografia do autor e publicidade. Normalmente são representadas como barras laterais.

<footer>

Esse elemento representa o rodapé do conteúdo ou de parte dele, pois ele é aceito dentro de vários elementos, como article e section e até do body. Exemplos de conteúdo de um <footer> são informações de autor e links relacionados.

<h1>-<h6> Eles não foram criados na versão 5 do HTML

Eles não foram criados na versão 5 do HTML e nem são específicos para semântica, mas servem para esse propósito. São utilizados para marcar a importância dos títulos, sendo <h1> o mais importante e <h6> o menos. Uma dica: use apenas um <h1> por página, pois ele representa o objetivo da sua página.

Prática

Dando continuidade ao nosso site iremos montar sua estrutura. Pensei em adicionarmos um cabeçalho com nosso nome, uma lista de posts (como um blog) e um rodapé para nossos contatos.

Vamos abrir nosso arquivo index.html e começar pelo cabeçalho: criamos um <header> logo abaixo do <body> e colocamos o título da nossa página dentro de um <h1>.

Depois criaremos a lista de postagens: abrimos um elemento section e dentro dele adicionamos outro <header> contendo um <h2>. Notem que eu posso ter mais de um <header> na página.

Para criar nossa postagem adicionamos um <article> com um <header> e um <h3>.

O último passo desta etapa é criar um rodapé para nossas informações de contato: crie um elemento footer antes de fechar o </body>.

Não se preocupe com o layout e com conteúdo da página, nós vamos tratar isso mais a frente.

Textos e links

A criação do HTML foi motivada pela necessidade de compartilhar textos e documentos, e mesmo depois de quase 30 anos, com toda a evolução da web, isso ainda representa uma boa parte do conteúdo da web.

Já falamos anteriormente sobre os elementos h1-h6 e, eles são essenciais para nos indicar visualmente a importância e localização de seções de texto na página, mas para textos maiores e mais densos usamos o elemento p.

O <p> representa um parágrafo, mas ele não suporta apenas texto, podemos adicionar imagens, código, vídeos e vários outros tipos de conteúdo dentro dele.

Um outro elemento interessante e extremamente necessário na web é o <a> - que significa anchor/âncora, ele representa um hyperlink, é ele que interliga vários conteúdos e páginas na web.

O elemento a tem vários atributos, mas vamos focar em dois, o href e o target.

O href representa o hyperlink para onde sua âncora aponta, pode ser uma página do seu ou de outro site, um e-mail e até mesmo um telefone, os dois últimos precisam dos prefixos mailto: e tel:, respectivamente.

O target neste momento vai servir para nos ajudar a abrir nossos links em outra aba do navegador usando o valor _blank.

Prática

Vamos adicionar um texto fictício a nossa postagem: logo após o fechamento do </header> vamos adicionar um elemento p e inserir um texto que vamos retirar do site lipsum.com

E em alguma parte deste texto vamos adicionar um hiperlink para outra página e um para nosso e-mail.

Criarei um hyperlink para meu perfil no LinkedIn: adicione o hyperlink no atributo href e o valor _blank no atributo target, assim o link será aberto em outra aba. E em algum outro lugar do texto adicionarei meu e-mail e um link para ele, desta forma: lucas@vilaboim.com

Imagens

A web também é feita de imagens e para representá-las temos o elemento <img>, ele é um daqueles elementos sem tag de fechamento.

O elemento img é bem simples, contendo apenas 2 atributos próprios, o src e o alt.

O src é obrigatório e guarda o caminho para a imagem que você quer mostrar na página.

O alt não é obrigatório mas é altamente recomendado por melhorar a acessibilidade, ele mostra a descrição da imagem caso ela não carregue e leitores de tela usam esse atributo para descrever a imagem para o usuário saber o que ela significa.

Prática

Vamos adicionar uma imagem ao cabeçalho da página e uma imagem a postagem.

Primeiro vamos colocar as imagens na pasta do nosso projeto. Para a imagem do cabeçalho eu escolhi uma foto minha com 100 pixels de largura e 100 pixels de altura e para a imagem da postagem eu procurei por html code no site Unsplash, escolhi uma das imagens e deixei ela com 960 pixels de largura por 322 pixels de altura.

Dentro do primeiro <header> da página e antes do <h1> iremos adicionar um elemento img e no atributo src colocamos o caminho para a nossa foto, /lucas-vilaboim.jpg, e o atributo alt deve conter um significado para a imagem, como no meu caso é uma ilustração, colocarei Ilustração do rosto de Lucas Vilaboim.

E dentro do <header> do <article> vamos fazer a mesma coisa, mas agora depois do <h3>, e no atributo alt colocaremos Editor de texto mostrando códigos HTML.

Listas

Os últimos elementos que veremos neste módulo são os relacionados a listas: <ul>, <ol> e <li>.

Listas servem para agrupar uma coleção de itens, como uma lista de ingredientes ou, como será no nosso caso, uma lista com contatos.

O elemento ul cria uma lista não ordenada, onde a ordem dos elementos não é importante, e é representada com pontos, círculos ou quadrados.

O <ol> serve para criar lista ordenadas, nessas a ordem importa, portanto elas são representadas com números, algarismos romanos ou letras.

E o elemento li é um item dentro de uma dessas listas. Um <li> pode conter vários tipos de conteúdos, como parágrafos, imagens e até outras listas.

Prática

Adicionaremos uma lista de contatos ao rodapé da nossa página, e para isso usaremos também o elemento a que vimos anteriormente.

Crie um elemento ul e dentro dele adicione um <li> com um elemento a, no atributo href adicione o link de alguma rede social que você mantenha e, no conteúdo da âncora coloque o nome dessa rede.

CSS 3

Definição e seletores

Após a criação do HTML a necessidade de formatar as páginas ficou evidente, assim, em 1996, foi criada a linguagem de estilo que conhecemos por CSS.

A sintaxe é bem simples e pode ser explicada com a frase "você cria regras de estilo para elementos ou grupos de elementos".

Vamos usar um elemento HTML que vimos anteriormente, a âncora <a>, para exemplificar.

Uma regra CSS é representada por um seletor ou um grupo de seletores, no nosso caso é o <a>, então dentro de um par de chaves adicionamos as declarações, no exemplo acima estamos alterando cor e tamanho da fonte dessa âncora, as declarações são formadas por uma propriedade e um valor.

Percebam que podemos colocar vários seletores em uma regra separando-os por vírgula.

E há um último detalhe nesse exemplo: a pseudo-classe. Elementos HTML sofrem alterações causadas pela interação do usuário, como mover o mouse por cima ou clicar nesse elemento.

O a:hover do exemplo significa que a âncora também terá essa aparência quando o usuário passar o mouse por cima de um hyperlink.

ID x Classe

No exemplo anterior criamos uma regra que altera um elemento HTML diretamente, mas isso significa que todos os elementos <a> ficarão com aquela aparência, e normalmente temos sites mais complexos que precisam de várias regras diferentes para elementos iguais.

Para ficar mais tangível vamos relembrar um pouco o site que começamos a fazer no módulo passado, ele tinha vários elementos header, mas não vamos querer que o header principal tenha a mesma formatação que o header de uma postagem, é aí que entram os IDs e Classes.

O seletor que vimos no primeiro exemplo é um seletor de tipo, pois ele representa um elemento HTML, e com IDs e Classes podemos representar qualquer tipo de elemento mas há algumas diferenças entre eles:

ID: é representado pelo símbolo # (hash) seguido de um nome para esse ID.

Classe: a classe é representada de forma parecida do ID, mas é precedida por um ponto em vez do hash.

E a diferença mais importante entre eles é a forma como devem ser usados: o ID só pode ser usado uma vez em uma página HTML enquanto a classe não tem restrições.

Exercício

Vamos adicionar algumas classes no nosso site e alterar alguns elementos, mas antes precisamos adicionar um arquivo CSS a nossa página.

No módulo de HTML descobrimos que podemos adicionar CSS de duas formas, com o elemento style, e assim suas regras ficarão no arquivo HTML, ou podemos criar um arquivo CSS e adicioná-lo na página através do elemento link, e é essa forma que usaremos.

Crie um elemento link dentro do head do seu arquivo e adicione os atributos rel="stylesheet" e href="style.css", o rel denota o tipo de arquivo que estamos incluindo na página e o href é o caminho para o arquivo. E na mesma pasta do arquivo HTML crie um arquivo chamado style.css.

Agora sim vamos ao CSS, adicione um ID #title ao h1 da página, pois queremos que ele seja único, e depois adicione as classes .subtitle e .post_title ao h2 e h3, respectivamente.

No arquivo CSS vamos mudar a cor desses três títulos, e depois alterar o tamanho da fonte do título da postagem.

Box-model

Quando estamos criando o layout de um site o navegador representa cada elemento HTML como uma caixa retangular, isso é o box-model. E com CSS nós alteramos a aparência dessa caixa (largura, altura, cor de fundo, etc.). Essa caixa é composta por 4 áreas: o conteúdo, o padding, a borda e a margem.

As margens (margin) são espaçamentos entre elementos;

As bordas (border) ;

O padding é um espaçamento entre as bordas e o conteúdo, a diferença para as margens é que declarações de imagem

de fundo funcionam nele;

O conteúdo (content) é o que o seu bloco representa, um texto, uma imagem, um vídeo;

Exercício

Para enxergamos o box-model vamos adicionar cores e bordas a alguns elementos.

Primeiro adicionaremos uma cor de fundo para a visualização ficar mais fácil, usaremos a propriedade background com o valor #fcfcfc no elemento body.

Depois vamos adicionar uma classe ao <article>, pode ser .post, e então vamos colocar a cor branca de fundo com a propriedade background e o valor #FFF. Agora conseguimos enxergar o content do box-model.

Vamos adicionar um padding de 10 pixels neste mesmo article. Perceberam o espaçamento que surgiu em volta do nosso

conteúdo?

Agora adicionamos um borda mais escura a ele com a propriedade border. Vou falar mais detalhadamente sobre border

mais a frente, mas por enquanto vamos deixar essa borda com 3 pixels de largura, o contorno sólido e a cor azul.

E por último vamos adicionar uma margem do lado de fora do post com a propriedade margin e o valor 10 pixels.

E agora inspecionando o nosso elemento conseguimos todas aquelas camadas citadas antes: o conteúdo em azul, o padding

em verde, as bordas em marrom e as margens em laranja.

E já que começamos a falar sobre bordas e cor de fundo, no próximo vídeo vamos nos aprofundar nessas propriedades.

Estilizando elementos

Agora que entendemos o box-model podemos focar em deixar nosso site mais bonito, então vamos repassar pelas propriedades já citadas:

Padding e Margin

Anteriormente usamos o padding e o margin da forma mais básica, com apenas um valor, mas eles são mais poderosos que isso.

Se quisermos atribuir tamanhos diferentes para cada lado do box nós podemos, e vamos ver três formas de fazer isso.

A primeira é colocando um valor para as partes superior e inferior e depois para os lados esquerdo e direito.

O valor de 10 pixels se refere ao eixo Y, ou partes superior e inferior, e os 5 pixels se referem aos lados esquerdo e direito.

A segunda forma é dando valores para cada lado do box.

Então começamos pelo topo com 15 pixels, passamos o lado direito com 10 pixels, depois para a parte inferior com 5 pixels e

por último o lado esquerdo com 0, e sempre nessa ordem.

Uma boa dica também é que quando o valor for 0 não precisamos não precisamos colocar a unidade.

A terceira forma é com as propriedades específicas para cada lado, até agora tínhamos visto atalhos para essas propriedades.

Essa opção é mais usada quando temos o mesmo valor para 3 lados, e o quarto precisa ter um valor diferente, então usamos o

padding com apenas um valor e uma dessas opções para representar o lado diferente.

Background

A propriedade background também é um atalho para várias propriedades, mas isso vocês podem absorver aos poucos, e uma

boa opção de leitura é a documentação do MDN.

Por enquanto veremos apenas como mudar a cor de fundo.

E aqui temos 3 formas de colocar uma cor de fundo, e ainda existem outras.

A primeira é pelo nome da cor em inglês, a segunda é pelo código hexadecimal e a terceira é usando apenas o atalho background.

Border

Vimos que a propriedade border pode ter 3 valores: a largura, a cor e o estilo, mas existem algumas particularidades nisso.

A largura pode ser usada com várias unidades, como px, em e mm. A cor pode ser atribuída pelo nome ou por um código

hexadecimal, assim como fizemos com o background, e o estilo é representada por palavras-chave, vamos ver algumas delas:

solid: mostra uma borda simples e reta;

dotted: são bolinhas com um pequeno espaçamento entre elas;

dashed: forma uma linha tracejada.

E aproveitando que mostrei esse código temos que falar sobre como separar a estilização dos lados de uma borda.

E se você não quiser usar a propriedade border existem as propriedade específicas para cada aspecto de uma borda, são elas border-width para a largura, border-color para a cor e border-style para o estilo.

Aqui temos o mesmo código anterior de duas formas diferentes, a primeira com o atalho border e a segunda com cada propriedade específica.

E depois disso podemos juntar os lados com os aspectos de uma borda e criar uma regra mais específica ainda.

Border-radius

E a última propriedade é o border-radius, ele permite arredondar os cantos de um elemento. Podemos usar várias unidades, mas as mais comuns são os pixels e a porcentagem.

Colocando apenas um valor mudamos todos os cantos do elemento, mas seguindo aquela mesma ordem que vimos no padding e margin - topo, direita, inferior e esquerda - conseguimos alterar cada canto separadamente.

Exercício

Neste exercício vamos deixar o nosso site um pouco mais bonito usando as propriedades que acabamos de ver.

Vamos aumentar o padding para 15 pixels e colocar uma margem de também de 15 pixels só na parte de baixo do post.

Quando olhamos para os textos percebemos que os espaçamentos estão diferentes do restante do post, então vamos padronizar isso.

No título do post vamos retirar todas as margens para depois colocar apenas uma margem inferior de 15 pixels. E no corpo do post precisamos adicionar uma classe e remover todas as margens para depois adicionar uma margem superior de 15 pixels.

Podemos manter o background branco, mas vamos diminuir a largura das bordas para 2 pixels e mudar a cor para a mesma do texto - #505050 - e por último adicionaremos um border-radius, 5 pixels são suficientes. Podemos adicionar esse mesmo de valor de border-radius na imagem, para isso vamos acrescentar uma class a imagem antes.

Estilizando textos

Já sabemos que podemos mudar cor e tamanho de algumas fontes, e agora vamos nos aprofundar nisso.

font-family

Com o font-family podemos alterar a fonte dos nossos textos, como uma fonte da internet ou uma que esteja instalada no nosso computador, mas vamos nos ater às fontes seguras, chamadas de web safe fonts.

Essas fontes são chamadas assim pois são encontradas em quases todos os sistemas e podem ser usadas sem preocupação.

font-size

O font-size nos ajuda a mudar o tamanho do texto, existem algumas unidades de medida para ele mas por enquanto os pixels são suficientes para nós.

font-style

Usamos o font-style para tornar um texto itálico, na maioria das vezes você usará apenas o valor italic para ele, mas se precisar tirar o itálico de um texto você pode usar o valor normal.

HTML Introdução

HTML é a linguagem de marcação padrão para criar páginas da Web.

O que é HTML?

- HTML significa Hyper Text Markup Language
- HTML é a linguagem de marcação padrão para criar páginas da Web
- HTML descreve a estrutura de uma página da Web
- HTML consiste em uma série de elementos
- Elementos HTML informam ao navegador como exibir o conteúdo
- Os elementos HTML rotulam partes do conteúdo como "este é um título", "este é um parágrafo", "este é um link", etc.

Um documento HTML simples

Exemplo

```
<!DOCTYPE html>
<html>
<head>
<title>Page Title</title>
</head>
<body>

<h1>My First Heading</h1>
<p>My first paragraph.</p>

</body>
</html>
```

Exemplo explicado

- o `<!DOCTYPE html>`declaração define que este documento é um documento HTML5
- o `<html>`elemento é o elemento raiz de um HTML página
- o `<head>`elemento contém meta-informações sobre o página HTML
- o `<title>`elemento especifica um título para o Página HTML (que é mostrada na barra de título do navegador ou na aba da página)
- o `<body>`elemento define o corpo do documento, e é um recipiente para todo o conteúdo visível, como títulos, parágrafos, imagens, hiperlinks, tabelas, listas, etc.
- o `<h1>`elemento define um título grande
- o `<p>`elemento define um parágrafo

O que é um elemento HTML?

Um elemento HTML é definido por uma tag inicial, algum conteúdo e uma tag final:

< tagname > O conteúdo vai aqui... < /tagname >

HTML **elemento** é tudo, desde a tag inicial até a tag final:

< h1 > Meu Primeiro título < /h1 >

< p > Meu primeiro parágrafo. < /p >

Start tag	Element content	End tag
<h1>	My First Heading	</h1>
	My first paragraph.	

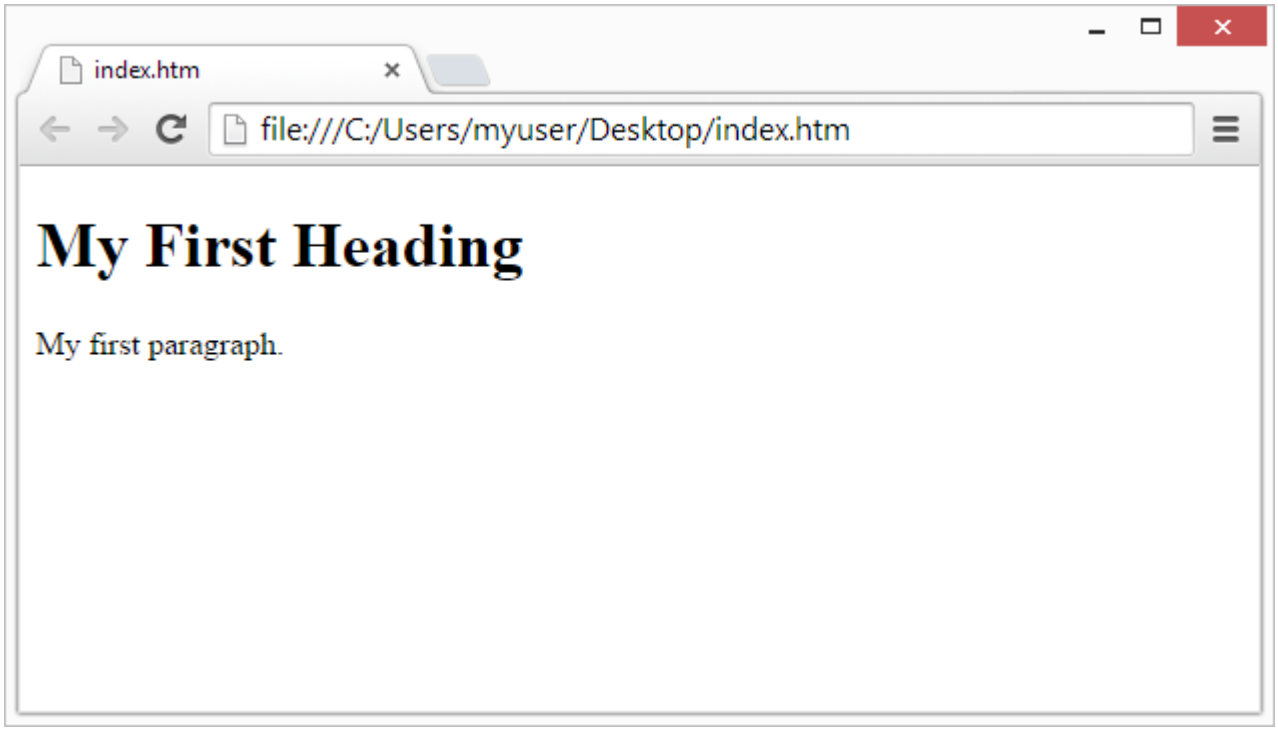
 	none	*none*

Nota: Alguns elementos HTML não têm conteúdo (como o
 elemento). Esses elementos são chamados de elementos vazios. Elementos vazios não possuem uma tag final!

Navegadores da Web

O objetivo de um navegador da Web (Chrome, Edge, Firefox, Safari) é ler documentos HTML e exibi-los corretamente.

Um navegador não exibe as tags HTML, mas as usa para determinar como exibir o documento:



Estrutura da página HTML

Abaixo está uma visualização de uma estrutura de página HTML:

<html>
<cabeça>
<title>Título da página</title>

</head>

<corpo>

<h1>Este é um título</h1>

Este é um parágrafo.

Este é outro parágrafo.

</body>

</html>

Nota: O conteúdo dentro da seção <body> (a área branca acima) será exibido em um navegador. O conteúdo dentro do elemento <title> será mostrado na barra de título do navegador ou na guia da página.

Histórico HTML

Desde os primeiros dias da World Wide Web, tem havido muitas versões de HTML:

Year	Version
1989	Tim Berners-Lee invented www
1991	Tim Berners-Lee invented HTML
1993	Dave Raggett drafted HTML+
1995	HTML Working Group defined HTML 2.0
1997	W3C Recommendation: HTML 3.2
1999	W3C Recommendation: HTML 4.01
2000	W3C Recommendation: XHTML 1.0
2008	WHATWG HTML5 First Public Draft
2012	WHATWG HTML5 Living Standard
2014	W3C Recommendation: HTML5
2016	W3C Candidate Recommendation: HTML 5.1
2017	W3C Recommendation: HTML5.1 2nd Edition
2017	W3C Recommendation: HTML5.2

Este tutorial segue o padrão HTML5 mais recente.

HTML Editores de

Um editor de texto simples é tudo que você precisa para aprender HTML.

Aprenda HTML usando o Bloco de Notas ou TextEdit

As páginas da Web podem ser criadas e modificadas usando editores profissionais de HTML.

No entanto, para aprender HTML, recomendamos um editor de texto simples como o Bloco de Notas (PC) ou o TextEdit (Mac).

Acreditamos que usar um editor de texto simples é uma boa maneira de aprender HTML.

Siga as etapas abaixo para criar sua primeira página da Web com o Bloco de Notas ou o TextEdit.

Passo 1: Abra o Bloco de Notas (PC)

Windows 8 ou posterior:

Abra a **tela inicial** (o símbolo da janela no canto inferior esquerdo da tela). Digite **Bloco** .

Windows 7 ou anterior:

Abra **Iniciar** > **Programas > Acessórios > Bloco de notas**

Passo 1: Abra o TextEdit (Mac)

Abra o **Finder > Aplicativos > TextEdit**

Altere também algumas preferências para que o aplicativo salve os arquivos corretamente. Em **Preferências > Formato >** escolha **"Texto Simples"**

Em seguida, em "Abrir e salvar", marque a caixa que diz "Exibir arquivos HTML como código HTML em vez de texto formatado".

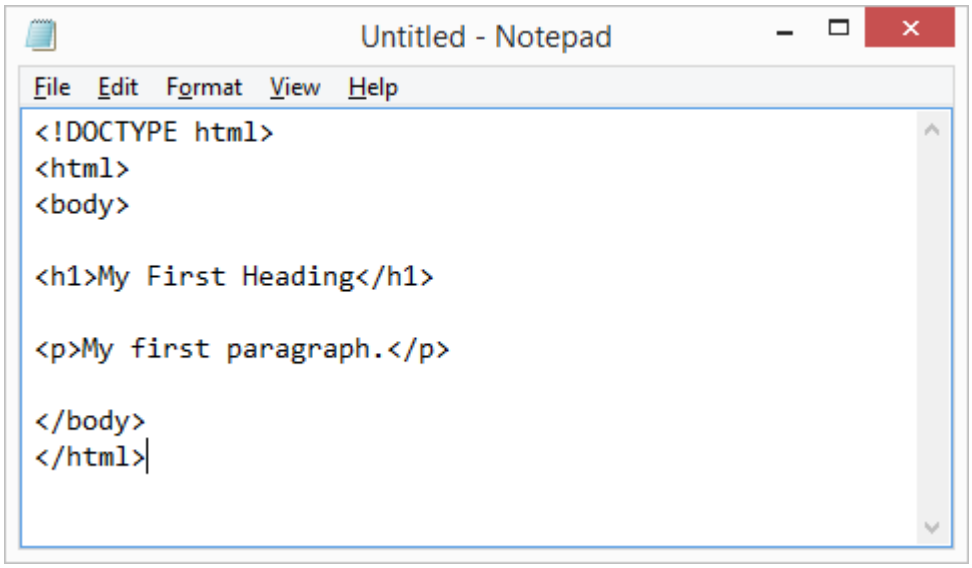
Em seguida, abra um novo documento para colocar o código.

Etapa 2: escreva um pouco de HTML

Escreva ou copie o seguinte código HTML no Bloco de Notas:

```
<!DOCTYPE html>
<html>
<body>

<h1>My First Heading</h1>

<p>My first paragraph.</p>
```

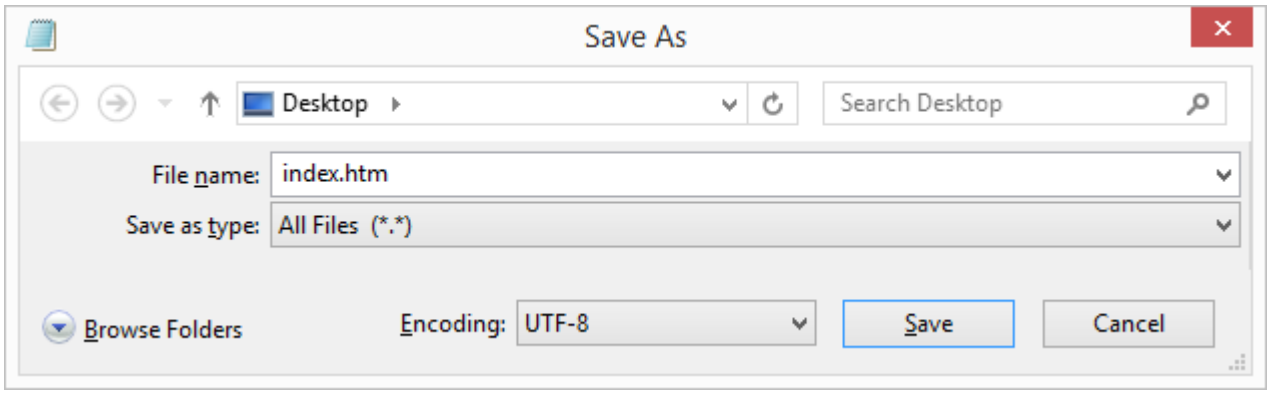
```
</body>
</html>
```



Etapa 3: salve a página HTML

Salve o arquivo em seu computador. Selecione **Arquivo > Salvar como** no menu do Bloco de Notas.

Nomeie o arquivo **"index.htm"** e defina a codificação para **UTF-8** (que é a codificação preferida para arquivos HTML).

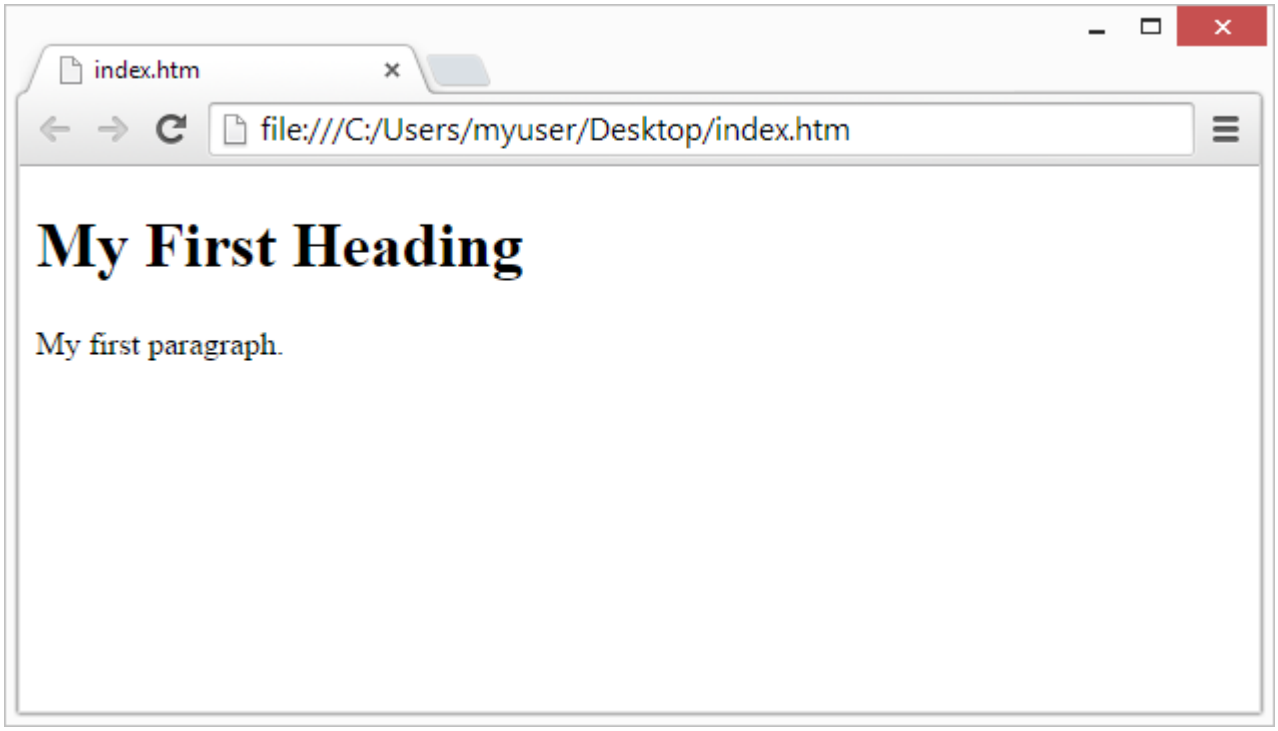


Dica: Você pode usar .htm ou .html como extensão de arquivo. Não há diferença, depende de você.

Etapa 4: visualize a página HTML em seu navegador

Abra o arquivo HTML salvo em seu navegador favorito (clique duas vezes no arquivo, ou clique com o botão direito do mouse - e escolha "Abrir com").

O resultado será muito parecido com isso:



Editor online do W3Schools - "Experimente você mesmo"

Com nosso editor online gratuito, você pode editar o código HTML e visualizar o resultado em seu navegador.

É a ferramenta perfeita quando você deseja **testar** o código rapidamente. Também tem cor codificação e a capacidade de salvar e compartilhar código com outras pessoas:

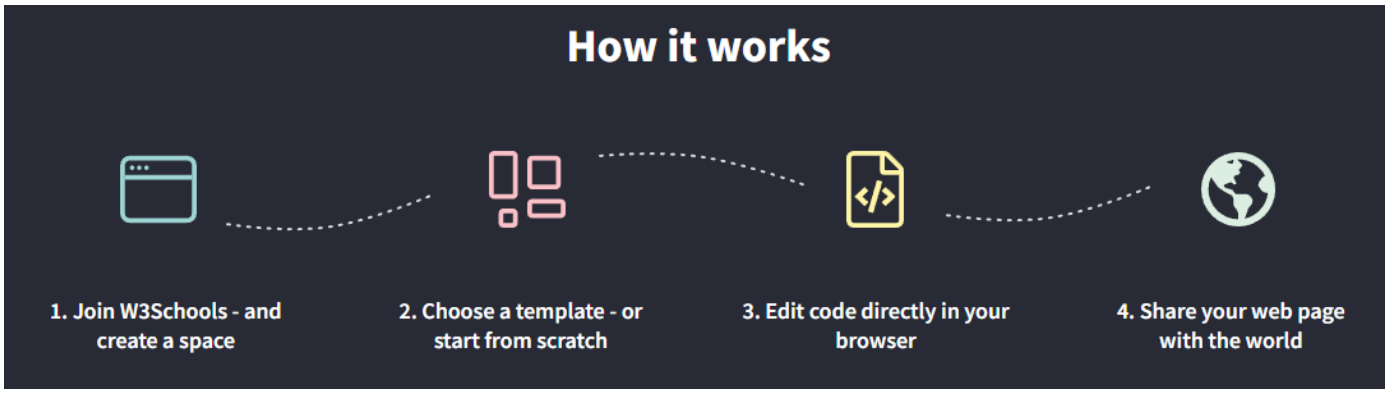
Exemplo

```
<!DOCTYPE html>
<html>
<head>
<title>Page Title</title>
</head>
<body>

<h1>This is a Heading</h1>
<p>This is a paragraph.</p>

</body>
</html>
```

Clique no botão "Experimente você mesmo" para ver como funciona.

Espaços W3Schools

Se você deseja criar seu próprio site e salvar seu código online, experimente nosso **construtor de sites gratuito** , chamado **W3schools Spaces** :



HTML Exemplos básicos de

Neste capítulo mostraremos alguns exemplos básicos de HTML.

Não se preocupe se usarmos tags que você ainda não conhece.

Documentos HTML

Todos os documentos HTML devem começar com uma declaração de tipo de documento: `<!DOCTYPE html>`.

O próprio documento HTML começa com `<html>`e termina com `</html>`.

A parte visível do documento HTML está entre `<body>`e `</body>`.

Exemplo

```
<!DOCTYPE html>
<html>
<body>

<h1>My First Heading</h1>
<p>My first paragraph.</p>

</body>
</html>
```

A Declaração <!DOCTYPE>

o `<!DOCTYPE>`declaração representa o tipo de documento e ajuda os navegadores a exibir as páginas da Web corretamente.

Ele deve aparecer apenas uma vez, no topo da página (antes de qualquer tag HTML).

o `<!DOCTYPE>`declaração não diferencia maiúsculas de minúsculas.

o `<!DOCTYPE>`declaração para HTML5 é:

```
<!DOCTYPE html>
```

Títulos HTML

Os cabeçalhos HTML são definidos com o `<h1>`para `<h6>`Tag.

`<h1>`define o título mais importante. `<h6>`define o menos importante cabeçalho:

Exemplo

```
<h1>This is heading 1</h1>
<h2>This is heading 2</h2>
<h3>This is heading 3</h3>
```

Parágrafos HTML

Os parágrafos HTML são definidos com o `<p>`marcação:

Exemplo

```
<p>This is a paragraph.</p>
<p>This is another paragraph.</p>
```

Links HTML

Os links HTML são definidos com o `<a>`marcação:

Exemplo

```
<a href="https://www.w3schools.com">This is a link</a>
```

O destino do link é especificado no `href`atributo.

Os atributos são usados para fornecer informações adicionais sobre elementos HTML.

Você aprenderá mais sobre atributos em um capítulo posterior.

Imagens HTML

As imagens HTML são definidas com o `<img>`marcação.

O arquivo de origem (`src`), Texto Alternativo (`alt`), `width`, e `height`são fornecidos como atributos:

Exemplo

```
<img src="w3schools.jpg" alt="W3Schools.com" width="104" height="142">
```

Como visualizar o código-fonte HTML?

Você já viu uma página da Web e se perguntou "Ei! Como eles fizeram isso?"

Veja o código-fonte HTML:

Clique com o botão direito do mouse em uma página HTML e selecione "Exibir código-fonte da página" (em Chrome) ou "View Source" (no Edge) ou similar em outros navegadores. Isso abrirá uma janela contendo o código-fonte HTML da página.

Inspecione um elemento HTML:

Clique com o botão direito do mouse em um elemento (ou em uma área em branco) e escolha "Inspecionar" ou "Inspecionar Elemento" para ver de que elementos são compostos (você verá os dois o HTML e o CSS). Você também pode editar o HTML ou CSS on-the-fly no Painel Elementos ou Estilos que é aberto.

HTML Elementos

Um elemento HTML é definido por uma tag inicial, algum conteúdo e um etiqueta final.

Elementos HTML

HTML **elemento** é tudo, desde a tag inicial até a tag final:

< tagname > O conteúdo vai aqui... < /tagname >

Exemplos de alguns elementos HTML:

< h1 > Meu Primeiro título < /h1 >

< p > Meu primeiro parágrafo. < /p >

Start tag	Element content	End tag
<h1>	My First Heading	</h1>
	My first paragraph.	
 	none	*none*

Nota: Alguns elementos HTML não têm conteúdo (como o
 elemento). Esses elementos são chamados de elementos vazios. Elementos vazios não possuem uma tag final!

Elementos HTML aninhados

Os elementos HTML podem ser aninhados (isso significa que os elementos podem conter outros elementos).

Todos os documentos HTML consistem em elementos HTML aninhados.

O exemplo a seguir contém quatro elementos HTML (`<html>`, `<body>`, `<h1>` e `<p>`):

Exemplo

```
<!DOCTYPE html>
<html>
<body>

<h1>My First Heading</h1>
<p>My first paragraph.</p>
```

```
</body>
</html>
```

Exemplo explicado

o `<html>`elemento é o elemento raiz e define todo o documento HTML.

Tem uma etiqueta de início `<html>`e uma etiqueta final `</html>`.

Então, dentro do `<html>`elemento existe uma `<body>` elemento:

```
<body>

<h1>My First Heading</h1>
<p>My first paragraph.</p>

</body>
```

o `<body>`elemento define o corpo do documento.

Tem uma etiqueta de início `<body>`e uma etiqueta final `</body>`.

Então, dentro do `<body>`elemento lá são dois outros elementos: `<h1>`e `<p>`:

```
<h1>My First Heading</h1>
<p>My first paragraph.</p>
```

o `<h1>`elemento define um título.

Tem uma etiqueta de início `<h1>`e uma etiqueta final `</h1>`:

```
<h1>My First Heading</h1>
```

o `<p>`elemento define um parágrafo.

Tem uma etiqueta de início `<p>`e uma etiqueta final `</p>`:

```
<p>My first paragraph.</p>
```


Nunca pule a tag final

Alguns elementos HTML serão exibidos corretamente, mesmo que você esqueça a tag final:

Exemplo

```
<html>
<body>

<p>This is a paragraph
<p>This is a paragraph

</body>
</html>
```

No entanto, nunca confie nisso! Resultados inesperados e erros podem ocorrer se você esquecer a tag final!

Elementos HTML vazios

Elementos HTML sem conteúdo são chamados de elementos vazios.

o `<br>`tag define uma quebra de linha e é um elemento vazio sem uma tag de fechamento:

Exemplo

```
<p>This is a <br> paragraph with a line break.</p>
```

HTML não diferencia maiúsculas de minúsculas

As tags HTML não diferenciam maiúsculas de minúsculas: `<P>`significa o mesmo que `<p>`.

O padrão HTML não requer tags minúsculas, mas W3C **recomenda** letras minúsculas em HTML e **exige** letras minúsculas para tipos de documentos mais rígidos, como XHTML.

No W3Schools, sempre usamos nomes de tags em minúsculas.

Referência de tags HTML

A referência de tags do W3Schools contém informações adicionais sobre essas tags e seus atributos.

Tag	Description
<html>	Defines the root of an HTML document
<body>	Defines the document's body
<h1> to <h6>	Defines HTML headings

Para obter uma lista completa de todas as tags HTML disponíveis, visite nossa Referência de tags HTML .

HTML Atributos

Os atributos HTML fornecem informações adicionais sobre elementos HTML.

Atributos HTML

- Todos os elementos HTML podem ter **atributos**
- Os atributos fornecem **informações adicionais** sobre os elementos
- Os atributos são sempre especificados na **tag inicial**
- Os atributos geralmente vêm em pares nome/valor como: **name="value"**

O atributo href

o `<a>`tag define um hiperlink. o `href`atributo especifica o URL da página o link vai para:

Exemplo

```
<a href="https://www.w3schools.com">Visit W3Schools</a>
```

Você aprenderá mais sobre links em nossos links HTML capítulo .

O atributo src

o `<img>`A tag é usada para incorporar um imagem em uma página HTML. o `src`atributo especifica o caminho para a imagem a ser exibida:

Exemplo

```
<img src="img_girl.jpg">
```

Há duas maneiras de especificar o URL no `src` atributo:

1. URL absoluto - Links para uma imagem externa hospedada em outro site. Exemplo: src="https://www.w3schools.com/images/img_girl.jpg" .

Notas: Imagens externas podem estar sob direitos autorais. Se você fizer não obter permissão para usá-lo, você pode estar violando as leis de direitos autorais. Dentro além disso, você não pode controlar imagens externas; ele pode ser removido de repente ou mudado.

2. URL Relativo - Links para uma imagem hospedada em o site. Aqui, o URL não inclui o nome de domínio. Se o URL começar sem uma barra, será relativo à página atual. Exemplo: src="img_girl.jpg". Se o URL começar com uma barra, será relativo ao domínio. Exemplo: src="/images/img_girl.jpg".

Dica: quase sempre é melhor usar URLs relativos. Elas não vai quebrar se você mudar de domínio.

Atributos de largura e altura

o `<img>`A tag também deve conter o `width`e `height`atributos, que especifica a largura e altura da imagem (em pixels):

Exemplo

```
<img src="img_girl.jpg" width="500" height="600">
```

O atributo alt

O necessário `alt`atributo para o `<img>`etiqueta especifica um texto alternativo para uma imagem, se a imagem por algum motivo não puder ser exibida. Isso pode ser devido conexão lenta ou um erro no `src`atributo, ou se o usuário usa uma tela leitor.

Exemplo

```
<img src="img_girl.jpg" alt="Girl with a jacket">
```

Exemplo

Veja o que acontece se tentarmos exibir uma imagem que não existe:

```
<img src="img_typo.jpg" alt="Girl with a jacket">
```

Você aprenderá mais sobre imagens em nosso capítulo Imagens HTML .

O atributo de estilo

o `style`atributo é usado para adicionar estilos a um elemento, como cor, fonte, tamanho e muito mais.

Exemplo

```
<p style="color:red;">This is a red paragraph.</p>
```

Você aprenderá mais sobre estilos em nosso capítulo Estilos HTML .

O atributo lang

Você deve sempre incluir o `lang`atributo dentro de `<html>`etiqueta, para declarar o idioma da página da Web. Isto destina-se a ajudar os motores de busca e navegadores.

O exemplo a seguir especifica o inglês como o idioma:

```
<!DOCTYPE html>
<html lang="en">
<body>
...
</body>
</html>
```

Os códigos de país também podem ser adicionados ao código de idioma no `lang`atributo. Assim, os dois primeiros caracteres definem o idioma da página HTML, e os dois últimos caracteres definem o país.

O exemplo a seguir especifica inglês como idioma e Estados Unidos como o país:

```
<!DOCTYPE html>
<html lang="en-US">
<body>
...
</body>
</html>
```

Você pode ver todos os códigos de idioma em nosso Referência de código de linguagem HTML .

O atributo título

o `title`atributo define algum extra informações sobre um elemento.

O valor do atributo title será exibido como uma dica de ferramenta quando você passa o mouse sobre o elemento:

Exemplo

```
<p title="I'm a tooltip">This is a paragraph.</p>
```

Sugerimos: Sempre use atributos em minúsculas

O padrão HTML não requer nomes de atributos em letras minúsculas.

O atributo title (e todos os outros atributos) podem ser escritos com letras maiúsculas ou minúsculas como **título** ou **TÍTULO** .

No entanto, o W3C **recomenda** atributos minúsculos em HTML e **exige** atributos em minúsculas para tipos de documentos mais restritos como XHTML.

No W3Schools, sempre usamos nomes de atributos em letras minúsculas.

Sugerimos: Sempre cite os valores dos atributos

O padrão HTML não exige aspas em torno dos valores de atributo.

No entanto, o W3C **recomenda** cotações em HTML e **exige** cotações para tipos de documentos mais rígidos como XHTML.

Bom:

```
<a href="https://www.w3schools.com/html/">Visit our HTML tutorial</a>
```

Ruim:

```
<a href=https://www.w3schools.com/html/>Visit our HTML tutorial</a>
```

Às vezes você tem que usar aspas. Este exemplo não será exibido o atributo title corretamente, pois contém um espaço:

Exemplo

```
<p title=About W3Schools>
```

Na W3Schools sempre usamos aspas em torno dos valores dos atributos.

Citações simples ou duplas?

Aspas duplas em torno de valores de atributo são as mais comuns em HTML, mas simples citações também podem ser usadas.

Em algumas situações, quando o próprio valor do atributo contém aspas duplas, é necessário usar aspas simples:

```
<p title='John "ShotGun" Nelson'>
```

Ou vice-versa:

```
<p title="John 'ShotGun' Nelson">
```

Resumo do capítulo

- Todos os elementos HTML podem ter **atributos**
- o `href`atributo de `<a>`especifica o URL da página para a qual o link vai
- o `src`atributo de `<img>`especifica o caminho para a imagem a ser exibida
- o `width`e `height`atributos de `<img>`fornecer informações de tamanho para imagens
- o `alt`atributo de `<img>`fornece um texto alternativo para uma imagem
- o `style`atributo é usado para adicionar estilos a um elemento, como cor, fonte, tamanho e mais
- o `lang`atributo do `<html>`etiqueta declara o idioma da página da Web
- o `title`atributo define algum extra informações sobre um elemento

---

Exercícios HTML

Teste-se com exercícios

Exercício:

Adicione uma "dica" ao parágrafo abaixo com o texto "Sobre o W3Schools".

<p ="Sobre o W3Schools">O W3Schools é um site para desenvolvedores da web.</p>

Comece o exercício

Referência de atributo HTML

Uma lista completa de todos os atributos para cada elemento HTML está listada em nosso: Referência de atributo HTML .

HTML Títulos

Cabeçalhos HTML são títulos ou legendas que você deseja exibir em uma página da web.

Exemplo

Heading 1

Heading 2

Heading 3

Heading 4

Heading 5

Heading 6

Títulos HTML

Os cabeçalhos HTML são definidos com o `<h1>`para `<h6>`Tag.

`<h1>`define o título mais importante. `<h6>`define o título menos importante.

Exemplo

```
<h1>Heading 1</h1>
<h2>Heading 2</h2>
<h3>Heading 3</h3>
<h4>Heading 4</h4>
<h5>Heading 5</h5>
<h6>Heading 6</h6>
```

Nota: Os navegadores adicionam automaticamente algum espaço em branco (uma margem) antes e depois de um título.

Os títulos são importantes

Os mecanismos de pesquisa usam os títulos para indexar a estrutura e o conteúdo de suas páginas da web.

Os usuários geralmente percorrem uma página por seus títulos. É importante usar títulos para mostrar a estrutura do documento.

`<h1>`títulos devem ser usados para títulos principais, seguidos por `<h2>`títulos, então o menos importante `<h3>`, e assim por diante.

Nota: Use títulos HTML apenas para títulos. Não use títulos para fazer texto **GRANDE** ou **negrito** .

Títulos maiores

Cada cabeçalho HTML tem um tamanho padrão. No entanto, você pode especificar o tamanho de qualquer título com o `style`atributo, usando o CSS `font-size`propriedade:

Exemplo

```
<h1 style="font-size:60px;">Heading 1</h1>
```


Exercícios HTML

Teste-se com exercícios

Exercício:

Use a tag HTML correta para adicionar um título com o texto "Londres".

Londres é a capital da Inglaterra. É a cidade mais populosa do Reino Unido, com uma área metropolitana de mais de 13 milhões de habitantes.

Comece o exercício

Referência de tags HTML

A referência de tags do W3Schools contém informações adicionais sobre essas tags e seus atributos.

Tag	Description
<html>	Defines the root of an HTML document
<body>	Defines the document's body
<h1> to <h6>	Defines HTML headings

Para obter uma lista completa de todas as tags HTML disponíveis, visite nossa Referência de tags HTML .

HTML Parágrafos

Um parágrafo sempre começa em uma nova linha e geralmente é um bloco de texto.

Parágrafos HTML

O HTML <p>elemento define um parágrafo.

Um parágrafo sempre começa em uma nova linha e os navegadores adicionam automaticamente algum espaço em branco (uma margem) antes e depois de um parágrafo.

Exemplo

```
<p>This is a paragraph.</p>
<p>This is another paragraph.</p>
```

Exibição HTML

Você não pode ter certeza de como o HTML será exibido.

Telas grandes ou pequenas e janelas redimensionadas criarão resultados diferentes.

Com HTML, você não pode alterar a exibição adicionando espaços extras ou linhas extras em seu código HTML.

O navegador removerá automaticamente quaisquer espaços e linhas extras quando a página for exibido:

Exemplo

```
<p>
This paragraph
contains a lot of lines
in the source code,
but the browser
ignores it.
</p>

<p>
This paragraph
```

```
contains        a lot of spaces
in the source        code,
but the        browser
ignores it.
</p>
```

Regras horizontais HTML

o `<hr>`define uma quebra temática em uma página HTML e é mais frequentemente exibido como uma régua horizontal.

o `<hr>`elemento é usado para separar o conteúdo (ou definir uma mudança) em um HTML página:

Exemplo

```
<h1>This is heading 1</h1>
<p>This is some text.</p>
<hr>
<h2>This is heading 2</h2>
<p>This is some other text.</p>
<hr>
```

o `<hr>`tag é uma tag vazia, o que significa que não tem tag final.

Quebras de linha HTML

O HTML `<br>`elemento define uma quebra de linha.

Usar `<br>`se você quiser uma quebra de linha (uma nova linha) sem iniciar um novo parágrafo:

Exemplo

```
<p>This is<br>a paragraph<br>with line breaks.</p>
```

o `<br>`tag é uma tag vazia, o que significa que não tem tag final.

O problema do poema

Este poema será exibido em uma única linha:

Exemplo

```
<p>
  My Bonnie lies over the ocean.

  My Bonnie lies over the sea.

  My Bonnie lies over the ocean.

  Oh, bring back my Bonnie to me.
</p>
```

Solução - O elemento HTML <pre>

O HTML `<pre>`elemento define o texto pré-formatado.

O texto dentro de um `<pre>`elemento é exibido em uma fonte de largura fixa (geralmente Courier), e preserva espaços e quebras de linha:

Exemplo

```
<pre>
  My Bonnie lies over the ocean.

  My Bonnie lies over the sea.

  My Bonnie lies over the ocean.

  Oh, bring back my Bonnie to me.
</pre>
```

Exercícios HTML

Teste-se com exercícios

Exercício:

Use a tag HTML correta para adicionar um parágrafo com o texto "Hello World!".

```
<html>
<corpo>
```

```
</body>
</html>
```

Comece o exercício

Referência de tags HTML

A referência de tag do W3Schools contém informações adicionais sobre elementos HTML e seus atributos.

Tag	Description
<p>	Defines a paragraph
<hr>	Defines a thematic change in the content
 	Inserts a single line break
<pre>	Defines pre-formatted text

Para obter uma lista completa de todas as tags HTML disponíveis, visite nossa Referência de tags HTML .

HTML Estilos

O HTML `style`O atributo é usado para adicionar estilos a um elemento, como cor, fonte, tamanho e muito mais.

Exemplo

eu sou vermelho

eu sou azul

Eu sou grande

O atributo de estilo HTML

Definir o estilo de um elemento HTML, pode ser feito com o `style`atributo.

O HTML `style`atributo tem a seguinte sintaxe:

<*tagname* style="*property*:*value*;">

A ***propriedade*** é uma propriedade CSS. O ***valor*** é um valor CSS.

Você aprenderá mais sobre CSS posteriormente neste tutorial.

Cor de fundo

O CSS `background-color`propriedade define a cor de fundo para um elemento HTML.

Exemplo

Defina a cor de fundo de uma página para azul-pó:

```
<body style="background-color:powderblue;">

<h1>This is a heading</h1>
<p>This is a paragraph.</p>

</body>
```

Exemplo

Defina a cor de fundo para dois elementos diferentes:

```
<body>

<h1 style="background-color:powderblue;">This is a heading</h1>
<p style="background-color:tomato;">This is a paragraph.</p>

</body>
```

Cor do texto

O CSS `color`propriedade define a cor do texto para um elemento HTML:

Exemplo

```
<h1 style="color:blue;">This is a heading</h1>
<p style="color:red;">This is a paragraph.</p>
```

Fontes

O CSS `font-family`propriedade define a fonte a ser usada para um elemento HTML:

Exemplo

```
<h1 style="font-family:verdana;">This is a heading</h1>
<p style="font-family:courier;">This is a paragraph.</p>
```

Tamanho do texto

O CSS `font-size`propriedade define o tamanho do texto para um elemento HTML:

Exemplo

```
<h1 style="font-size:300%;">This is a heading</h1>
<p style="font-size:160%;">This is a paragraph.</p>
```

Alinhamento de texto

O CSS `text-align`A propriedade define o alinhamento horizontal do texto para um elemento HTML:

Exemplo

```
<h1 style="text-align:center;">Centered Heading</h1>
<p style="text-align:center;">Centered paragraph.</p>
```

Resumo do capítulo

- Use o `style`atributo para estilizar elementos HTML
- Usar `background-color`para cor de fundo
- Usar `color`para cores de texto
- Usar `font-family`para fontes de texto
- Usar `font-size`para tamanhos de texto
- Usar `text-align`para alinhamento de texto

Exercícios HTML

Teste-se com exercícios

Exercício:

Use o atributo HTML correto e o CSS para definir a cor do parágrafo como "azul".

<p [ ]=" [ ];">Este é um parágrafo.</p>

Comece o exercício

HTML Formatação de texto

HTML contém vários elementos para definir o texto com um significado especial.

Exemplo

This text is bold

This text is italic

This is subscript and superscript

Elementos de formatação HTML

Os elementos de formatação foram projetados para exibir tipos especiais de texto:

- `<b>`- Texto em negrito
- `<strong>`- Texto importante
- `<i>`- Texto em itálico
- `<em>`- Texto em destaque
- `<mark>`- Texto marcado
- `<small>`- Texto menor
- `<del>`- Texto excluído
- `<ins>`- Texto inserido
- `<sub>`- Texto subscrito
- `<sup>`- Texto sobrescrito

Elementos HTML <b> e

O HTML `<b>`elemento define texto em negrito, sem qualquer importância extra.

Exemplo

```
<b>This text is bold</b>
```

O HTML `<strong>`elemento define o texto com forte importância. O conteúdo interno geralmente é exibido em negrito.

Exemplo

```
<strong>This text is important!</strong>
```


Elementos HTML <i> e

O HTML `<i>`elemento define uma parte de texto em uma voz ou humor alternativo. O conteúdo dentro é normalmente exibido em itálico.

Dica: O `<i>`tag é frequentemente usada para indicar um termo técnico, uma frase de outra língua, um pensamento, um nome de navio, etc.

Exemplo

```
<i>This text is italic</i>
```

O HTML `<em>`elemento define texto enfatizado. O conteúdo interno geralmente é exibido em itálico.

Dica: Um leitor de tela pronunciará as palavras em `<em>` com ênfase, usando estresse verbal.

Exemplo

```
<em>This text is emphasized</em>
```

Elemento HTML <pequeno>

O HTML `<small>`elemento define texto menor:

Exemplo

```
<small>This is some smaller text.</small>
```

Elemento HTML <mark>

O HTML `<mark>`elemento define o texto que devem ser marcados ou destacados:

Exemplo

```
<p>Do not forget to buy <mark>milk</mark> today.</p>
```

Elemento HTML <del>

O HTML `<del>`elemento define o texto que foi excluído de um documento. Os navegadores geralmente atingem uma linha através do texto excluído:

Exemplo

```
<p>My favorite color is <del>blue</del> red.</p>
```

Elemento HTML <ins>

O HTML `<ins>`elemento define um texto que foi inserido em um documento. Os navegadores geralmente sublinham inseridos texto:

Exemplo

```
<p>My favorite color is <del>blue</del> <ins>red</ins>.</p>
```

Elemento HTML <sub>

O HTML `<sub>`elemento define texto subscrito. O texto subscrito aparece meio caractere abaixo da linha normal, e às vezes é renderizado em uma fonte menor. O texto subscrito pode ser usado para fórmulas químicas, como H_2O:

Exemplo

```
<p>This is <sub>subscripted</sub> text.</p>
```

Elemento HTML <sup>

O HTML `<sup>`elemento define texto sobrescrito. O texto sobrescrito aparece meio caractere acima do normal linha, e às vezes é renderizado em uma fonte menor. O texto sobrescrito pode ser usado para notas de rodapé, como WWW [1] :

Exemplo

```
<p>This is <sup>superscripted</sup> text.</p>
```

Exercícios HTML

Teste-se com exercícios

Exercício:

Adicione importância extra à palavra "degradação" no parágrafo abaixo.

A missão do WWF é parar o degradação nosso ambiente natural do planeta.

Comece o exercício

Elementos de formatação de texto HTML

Tag	Description
<b>	Defines bold text
	Defines emphasized text
<i>	Defines a part of text in an alternate voice or mood
<small>	Defines smaller text
	Defines important text
<sub>	Defines subscripted text
<sup>	Defines superscripted text
<ins>	Defines inserted text
<del>	Defines deleted text
<mark>	Defines marked/highlighted text

Para obter uma lista completa de todas as tags HTML disponíveis, visite nossa Referência de tags HTML .

HTML e Elementos de Citação

Neste capítulo vamos percorrer o `<blockquote>`, `<q>`, `<abbr>`, `<address>`, `<cite>`, e `<bdo>`Elementos HTML.

Exemplo

Here is a quote from WWF's website:

For nearly 60 years, WWF has been protecting the future of nature. The world's leading conservation organization, WWF works in 100 countries and is supported by more than one million members in the United States and close to five million globally.

HTML <blockquote> para Cotações

O HTML `<blockquote>`elemento define uma seção que é citado de outra fonte.

Os navegadores geralmente recuam `<blockquote>`elementos.

Exemplo

```
<p>Here is a quote from WWF's website:</p>
<blockquote cite="http://www.worldwildlife.org/who/index.html">
For 50 years, WWF has been protecting the future of nature.
The world's leading conservation organization,
WWF works in 100 countries and is supported by
1.2 million members in the United States and
close to 5 million globally.
</blockquote>
```

HTML <q> para citações curtas

O HTML `<q>`tag define uma cotação curta.

Os navegadores normalmente inserem aspas ao redor da cotação.

Exemplo

```
<p>WWF's goal is to: <q>Build a future where people live in harmony with nature.</q></p>
```

HTML <abbr> para abreviações

O HTML `<abbr>`tag define uma abreviação ou um acrônimo, como "HTML", "CSS", "Sr.", "Dr.", "ASAP", "ATM".

Marcar abreviaturas pode fornecer informações úteis para navegadores, tradução sistemas e motores de busca.

Dica: Use o atributo global title para mostre a descrição do abreviação/acrônimo quando você passa o mouse sobre o elemento.

Exemplo

```
<p>The <abbr title="World Health Organization">WHO</abbr> was founded in 1948.</p>
```

HTML <endereço> para informações de contato

O HTML `<address>`tag define as informações de contato do autor/proprietário de um documento ou um artigo.

As informações de contato podem ser um endereço de e-mail, URL, endereço físico, telefone número, identificador de mídia social, etc.

O texto no `<address>`elemento geralmente é renderizado em *itálico*, e os navegadores sempre adicione uma quebra de linha antes e depois do `<address>`elemento.

Exemplo

```
<address>
Written by John Doe.<br>
Visit us at:<br>
Example.com<br>
Box 564, Disneyland<br>
USA
</address>
```

HTML <cite> para título do trabalho

O HTML `<cite>`tag define o título de um trabalho criativo (por exemplo, um livro, um poema, uma música, um filme, uma pintura, uma escultura, etc.).

Nota: O nome de uma pessoa não é o título de uma obra.

O texto no `<cite>`elemento geralmente é renderizado em *itálico* .

Exemplo

```
<p><cite>The Scream</cite> by Edvard Munch. Painted in 1893.</p>
```

HTML <bdo> para substituição bidirecional

BDO significa substituição bidirecional.

O HTML `<bdo>`tag é usada para substituir a direção do texto atual:

Exemplo

```
<bdo dir="rtl">This text will be written from right to left</bdo>
```

Exercícios HTML

Teste-se com exercícios

Exercício:

Use um elemento HTML para adicionar aspas ao redor das letras "cool".

```
<p>
eu sou tão  legal .
</p>
```

Comece o exercício

Cotação HTML e Elementos de Citação

Tag	Description
<abbr>	Defines an abbreviation or acronym
<address>	Defines contact information for the author/owner of a document
<bdo>	Defines the text direction
<blockquote>	Defines a section that is quoted from another source
<cite>	Defines the title of a work
<q>	Defines a short inline quotation

Para obter uma lista completa de todas as tags HTML disponíveis, visite nossa Referência de tags HTML .

Flex Container

Flex Item

Guia Flexbox

Flex Container

O Flex Container é a tag que envolve os itens flex, ao indicar display: flex, essa tag passa a ser um Flex Container.

1 • display

Define o elemento como um flex container, tornando os seus filhos flex-itens.

- display: *flex;* // Torna o elemento um flex container automaticamente transformando todos os seus filhos diretos em flex itens.

2 • flex-direction

Define a direção dos flex itens. Por padrão ele é row (linha), por isso quando o display: flex; é adicionado, os elementos ficam em linha, um do lado do outro.

A mudança de row para column geralmente acontece quando estamos definindo os estilos em media queries para o mobile. Assim você garante que o conteúdo seja apresentado em coluna única.

- flex-direction: *row;* // Os itens ficam em linha
- flex-direction: *row-reverse;* // Os itens ficam em linha reversa, ou seja 3, 2, 1.
- flex-direction: *column;* // Os itens ficam em uma única coluna, um embaixo do outro.

- flex-direction: *column-reverse;* // Os itens ficam em uma única coluna, um embaixo do outro, porém em ordem reversa: 3, 2 e 1.

3 • flex-wrap

Define se os itens devem quebrar ou não a linha. Por padrão eles não quebram linha, isso faz com que os flex itens sejam compactados além do limite do conteúdo.

Essa é geralmente uma propriedade que é quase sempre definida como flex-wrap: wrap; Pois assim quando um dos flex itens atinge o limite do conteúdo, o último item passa para a coluna debaixo e assim por diante.

- flex-wrap: *nowrap;* // Valor padrão, não permite a quebra de linha.
- flex-wrap: *wrap;* // Quebra a linha assim que um dos flex itens não puder mais ser compactado.
- flex-wrap: *wrap-reverse;* // Quebra a linha assim que um dos flex itens não puder mais ser compactado. A quebra é na direção contrária, ou seja para a linha acima.

4 • flex-flow

O flex-flow é um atalho para as propriedades flex-direction e flex-wrap. Você não verá muito o seu uso, pois geralmente quando mudamos o flex-direction para column, mantemos o padrão do flex-wrap que é nowrap.

E quando mudamos o flex-wrap para wrap, mantemos o padrão do flex-direction que é row.

- flex-flow: *row nowrap;* // Coloca o conteúdo em linha e não permite a quebra de linha.
- flex-flow: *row wrap;* // Coloca o conteúdo em linha e permite a quebra de linha.
- flex-flow: *column nowrap;* // Coloca o conteúdo em coluna e não permite a quebra de linha.

5 • justify-content

Alinha os itens flex no container de acordo com a direção. A propriedade só funciona se os itens atuais não ocuparem todo o container. Isso significa que ao definir flex: 1; ou algo similar nos itens, a propriedade não terá mais função

Excelente propriedade para ser usada em casos que você deseja alinhar um item na ponta esquerda e outro na direita, como em um simples header com marca e navegação.

- justify-content: *flex-start;* // Alinha os itens ao início do container.
- justify-content: *flex-end;* // Alinha os itens ao final do container.
- justify-content: *center;* // Alinha os itens ao centro do container.
- justify-content: *space-between;* // Cria um espaçamento igual entre os elementos. Mantendo o primeiro grudado no início e o último no final.
- justify-content: *space-around;* // Cria um espaçamento entre os elementos. Os espaçamentos do meio são duas vezes maiores que o inicial e final.

6 • align-items

O align-items alinha os flex itens de acordo com o eixo do container. O alinhamento é diferente para quando os itens estão em colunas ou linhas.

Essa propriedade permite o tão sonhado alinhamento central no eixo vertical, algo que antes só era possível com diferentes hacks.

- align-items: *stretch;* // Valor padrão, ele que faz com que os flex itens cresçam igualmente.
- align-items: *flex-start;* // Alinha os itens ao início.
- align-items: *flex-end;* // Alinha os itens ao final.
- align-items: *center;* // Alinha os itens ao centro.
- align-items: *baseline;* // Alinha os itens de acordo com a linha base da tipografia.

7 • align-content

Alinha as linhas do container em relação ao eixo vertical. A propriedade só funciona se existir mais de uma linha de flex-itens. Para isso o flex-wrap precisa ser wrap.

Além disso o efeito dela apenas será visualizado caso o container seja maior que a soma das linhas dos itens. Isso significa que se você não definir height para o container, a propriedade não influencia no layout.

- align-content: *stretch;* // Valor padrão, ele que faz com que os flex itens cresçam igualmente na vertical.
- align-content: *flex-start;* // Alinha todas as linhas de itens ao início.
- align-content: *flex-end;* // Alinha todas as linhas de itens ao final.
- align-content: *center;* // Alinha todas as linhas de itens ao centro.
- align-content: *space-between;* // Cria um espaçamento igual entre as linhas. Mantendo a primeira grudada no topo e a última no bottom.
- align-content: *space-around;* // Cria um espaçamento entre as linhas. Os espaçamentos do meio são duas vezes maiores que o top e bottom.

Flex Item

Os Flex Itens são os filhos diretos do Flex Container, lembrado que uma tag se torna um flex container a partir do momento que você definir display: flex.

É possível que um Flex Item seja também um Flex Container, basta definir display: flex nele. Assim os filhos desse item também serão flex itens.

1 • flex-grow

Define a habilidade de um flex item crescer. Por padrão o valor é zero, assim os flex itens ocupam um tamanho máximo relacionado o conteúdo interno deles ou ao width definido.

Ao definir 1 para todos os Flex Itens, eles tentarão ter a mesma largura e vão ocupar 100% do container. Digo tentarão pois caso um elemento possua um conteúdo muito largo, ele irá respeitar o mesmo.

Se você tiver uma linha com quatro itens, onde três são flex-grow: 1 e um flex-grow: 2, o flex-grow: 2 tentará ocupar 2 vezes mais espaço extra do que os outros elementos.

OBS: justify-content não funciona em items com flex-grow definido.

- flex-grow: *número;* // Basta definir um número
- flex-grow: *0;* // Obedece o width do elemento ou o flex-basis.

2 • flex-basis

Indica o tamanho inicial do flex item antes da distribuição do espaço restante.

Quando definimos o flex-grow: 1; e possuímos auto no basis, o valor restante para ocupar o container é distribuído ao redor do conteúdo do flex-item.

- flex-basis: *auto;* // Esse é o padrão, ele faz com que a largura da base seja igual a do item. Se o item não tiver tamanho especificado, o tamanho será de acordo com o conteúdo.
- flex-basis: *unidade;* // Pode ser em %, em, px e etc.
- flex-basis: *0;* // Se o grow for igual ou maior que 1, ele irá tentar manter todos os elementos com a mesma largura, independente do conteúdo (por isso 0 é o valor mais comum do flex-basis). Caso contrário o item terá a largura do seu conteúdo.

3 • flex-shrink

Define a capacidade de redução de tamanho do item.

- flex-shrink: *1;* // Valor padrão, permite que os itens tenham os seus tamanhos (seja esse tamanho definido a partir de width ou flex-basis) reduzidos para caber no container.
- flex-shrink: *0;* // Não permite a diminuição dos itens, assim um item com flex-basis: 300px; nunca diminuirá menos do que 300px, mesmo que o conteúdo não ocupe todo esse espaço.
- flex-shrink: *número;* // Um item com shrink: 3 diminuirá 3 vezes mais que um item com 1.

4 • flex

Atalho para as propriedades flex-grow, flex-shrink e flex-basis. Geralmente você verá a propriedade flex nos flex itens ao invés de cada um dos valores separados.

Para melhor consistência entre os browsers, é recomendado utilizar a propriedade flex ao invés de cada propriedade separada.

No exemplo é possível ver as mesmas configurações do exemplo do flex-basis porém agora utilizando apenas a propriedade flex.

- flex: *1;* // Define flex-grow: 1; flex-shrink: 1; e flex-basis: 0; (em alguns browsers define como 0%, pois estes ignoram valores sem unidades, porém a função de 0 e 0% é a mesma.)
- flex: *0 1 auto;* // Esse é o padrão, se você não definir nenhum valor de flex ou para as outras propriedades separadas, o normal será flex-grow: 0, flex-shrink: 1 e flex-basis: auto.
- flex: *2;* // Define exatamente da mesma forma que o flex: 1; porém neste caso o flex-grow será de 2, o flex-shrink continuará 1 e o flex-basis 0.
- flex: *3 2 300px;* // flex-grow: 3, flex-shrink: 2 e flex-basis: 300px;

5 • order

Modifica a ordem dos flex itens. Sempre do menor para o maior, assim order: 1, aparece na frente de order: 5.

- order: *número;* // Número para modificar a ordem padrão. Pode ser negativo.
- order: *0;* // 0 é o valor padrão e isso significa que a ordem dos itens será a ordem apresentada no HTML. Se você quiser colocar um item do meio da lista no início da mesma, sem modificar os demais, o ideal é utilizar um valor negativo para este item, já que todos os outros são 0.

6 • align-self

O align-self serve para definirmos o alinhamento específico de um único flex item dentro do nosso container. Caso um valor seja atribuído, ele passara por cima do que for atribuído no align-items do container.

Vale lembrar que o alinhamento acontece tanto em linha quanto em colunas. Por exemplo o flex-start quando os itens estão em linhas, alinha o item ao topo da sua linha. Quando em colunas, alinha o item ao início (esquerda) da coluna.

- align-self: *auto;* // Valor inicial padrão. Vai respeitar o que for definido pelo align-items no flex-container.
- align-self: *flex-start;* // Alinha o item ao início.
- align-self: *flex-end;* // Alinha o item ao final.
- align-self: *center;* // Alinha o item ao centro.
- align-self: *baseline;* // Alinha o item a linha de base.
- align-self: *stretch;* // Estica o item.

Praia de Botafogo, 300, 5º andar - Botafogo - Rio de Janeiro - RJ - 22250-040.

Principais referências utilizadas para a criação deste guia: CSS Tricks e MDN