



# Melhores práticas **HTML & CSS**

Eleve o nível do seu código,  
construindo aplicações  
mais eficientes



**POR MATHEUS BATTISTI**  
**HORA DE CODAR**

## SOBRE O AUTOR

Matheus Battisti é desenvolvedor há mais de 7 anos e ama a tecnologia da informação como um todo.

Está sempre em busca de um aprendizado constante e evolutivo.

Atua como desenvolvedor Full Stack e produtor de conteúdo.

É criador do Hora de Codar, que hoje passa dos 50 mil alunos.

Acredita que HTML e CSS são as linguagens base de qualquer desenvolvedor que deseja trilhar uma carreira web.

Seu objetivo é sempre passar o máximo de conhecimento possível, unindo toda a experiência de seus estudos e também de sua carreira como desenvolvedor.





## INTRODUÇÃO

Este ebook tem como objetivo te apresentar boas práticas ao desenvolver sites com HTML e CSS.

Implementando as técnicas aqui ensinadas fará você criar sites mais robustos.

Elas vão desde a otimização da manutenção do código até a melhorias de SEO através de programação.

É importante que você implemente os conceitos em situações reais, ou seja, o conhecimento obtido aqui deve ser aplicado em seus projetos.

Isso internalizará os conhecimentos, fazendo com que você seja um melhor dev.

Vamos abordar vinte pontos muito importantes neste material, sendo dez de HTML e dez de CSS.

Está pronto? Vamos começar!



## HTML 1: UMA TAG H1 POR PÁGINA

É importante utilizar apenas uma tag h1 em cada página.

Pois ela determina o assunto principal, e leitores de página como o do Google podem te penalizar por ter mais de uma desta tag.

Lembrando que o HTML nunca lhe penalizará por isso, pois não é uma linguagem que faz o projeto "quebrar" por erros.

Aderindo esta prática você estará melhorando o **SEO** da sua página.

Precisa de mais títulos? Utilize outros headings progressivamente (h2, h3, h4).

Veja um exemplo:

```
1 <div>
2   <h1>Título mais importante</h1>
3   <p>Texto sobre o assunto</p>
4 </div>
5 <div>
6   <h2>Outro assunto da página</h2>
7   <p>Texto sobre outro assunto</p>
8 </div>
```





## HTML 2: USE A TAG FIGURE

Quando uma imagem do seu site possui uma descrição você deve utilizar a tag **figcaption**.

Isso vai relacionar o texto abaixo dela com a imagem, dando mais semântica ao seu projeto.

Os mecanismos de busca também verão como boa prática, melhorando o **SEO**.

Além de gerar acessibilidade, o que também é importante.

Pessoas com deficiência visual entenderão melhor o que você está tentando expressar e conseguirão interpretar a imagem.

Veja um exemplo:

```
1 <figure>
2   
3   <figcaption>Esta ponte metálica foi construída para garantir acesso entre as cidades.</figcaption>
4 </figure>
```





## HTML 3: UTILIZE TAGS SEMÂNTICAS

Assim como figure, temos diversas outras tags semânticas que podem dar mais sentido ao seu site.

Isso vem sendo cada vez mais cobrado pelos motores de busca.

Além disso, os programadores conseguirão entender melhor o seu código para dar continuidade no projeto.

Um exemplo é o **header**, onde temos o cabeçalho da página.

Que pode conter: logo, campo de pesquisa, links de navegação e outros elementos.

Veja um exemplo:

```
1 <header>
2   <h1>Minha Marca</h1>
3   <nav>
4     <ul>
5       <li><a href="/">Home</a></li>
6       <li><a href="/products">Produtos</a></li>
7       <li><a href="/contact">Contato</a></li>
8     </ul>
9   </nav>
10 </header>
```





## HTML 4: NÃO UTILIZE TAGS PARA ESTILO

Antigamente era comum utilizar a tag `<b>` para deixar algo em negrito.

Mas isso é uma má prática, devemos separar as responsabilidades.

Sempre que algo no documento for adicionado apenas por questões de estilo visual, está errado.

Deixe todo o estilo a cargo do **CSS**.

Como evitar isso? Em vez da tag `<b>` utilize a `<span>`, e coloque a propriedade **font-weight** nela.

Veja um exemplo:



```
1 <p>Isso é <span class="bold">muito legal!</span></p>
```





## HTML 5: SEMPRE UTILIZE A TAG TITLE

A tag **<title>** é uma meta tag da tag **<head>**

Devemos sempre configurar esta tag, pois é como o título do site aparece em mecanismos de busca e também na aba do navegador.

Ou seja, é um dos principais elementos para sermos encontrados corretamente através de busca orgânica.

Veja um exemplo:



```
1 <head>
2   <title>Aprenda tudo sobre HTML e CSS</title>
3 </head>
```







## HTML 6: ESCREVA SEMPRE EM LOWERCASE

Por mais que você precise escrever os textos do seu site em caixa alta, escreva os de forma normal.

Assim como uma redação, mantenha o padrão de letras iniciais no começo de cada frase maiúscula.

As demais devem estar em caixa baixa, a não ser que sejam o nome de alguém, por exemplo.

Utilize a propriedade **text-transform** do CSS para modificar o comportamento.

Veja um exemplo:



```
1  <!-- Errado -->
2  <h1>MEU TÍTULO CHAMATIVO</h1>
3
4  <!-- Correto -->
5  <h1>Meu título chamativo</h1>
```





## HTML 7: UTILIZE AS VALIDAÇÕES DO HTML

Ao trabalhar com formulários tente ao máximo prever todos os casos com validações.

Você pode utilizar tanto em JavaScript, como as próprias do HTML.

Elas são feitas por meio de atributos, como: **required** e **type**.

Veja um exemplo:



```
1 <input type="email" required placeholder="Digite o seu e-mail">
```





## HTML 8: NÃO UTILIZE INLINE STYLE

Novamente um ponto que é trabalhado em conjunto com o CSS.

O HTML possibilita estilizar as tags pelo atributo **style**.

Porém isso a longo prazo pode causar um caos ao seu código.

Sempre externalize o seu CSS para um arquivo **.css**, que é o local correto.

Veja um exemplo:



```
1 <p style="color: red;">Não faça isso</p>
```





## HTML 9: CUIDADO COM COMENTÁRIOS

Os comentários devem ser inseridos com cautela no seu HTML.

Utilize para separação de seções no seu código, por exemplo.

Mas nunca para transmitir informações sensíveis, qualquer um pode ver os comentários do HTML.

E também não coloque comentários em excesso, isso pode tornar o seu arquivo mais pesado, interferindo na performance da página.

Veja um exemplo:

```
1  <!-- Seção principal -->
2  <section>
3    ...
4  </section>
5
6  <!-- Banner lateral -->
7  <aside>
8    ...
9  </aside>
```





## HTML 10: ATRIBUTO ALT PARA IMAGENS

Toda imagem deve conter o atributo **alt**.

Seguindo a mesma linha da tag **figure**, este atributo ajuda na acessibilidade e **SEO**.

Basicamente precisamos descrever o que é exibido na imagem.

Veja um exemplo:



```
1 
```





## CSS 1: TOME CUIDADO COM A ESPECIFICIDADE

Especificidade é importante para aplicar regras em elementos que já possuem outras regras.

Porém a '**super-especificação**' pode acabar poluindo o código.

Tente mudar a regra ou adicionar uma **classe** ou **id** no elemento que precisa ser estilizado

Veja um exemplo:

```
1  /* Ruim */
2  div.container a.link {
3
4  }
5
6  /* Bom */
7  .container .link {
8
9  }
```





## CSS 2: NÃO UTILIZE !IMPORTANT

Você deve evitar ao máximo o uso de **!important**.

A regra parece resolver o problema de início, mas pode facilmente gerar uma guerra de importants.

Pois o CSS ficará difícil de ser sobrescrito por causa das regras anteriores com o recurso de **!important**

Veja um exemplo:

```
1  a {  
2      color: red !important;  
3  }  
4  
5  /* Não sobrescreve o important */  
6  .link {  
7      color: blue;  
8  }
```





## CSS 3: UTILIZE OS SHORTHANDS

O recurso de **shorthand** permite inserir valores para uma propriedade em vários aspectos.

Por exemplo: em vez de utilizar **padding-top** e **padding-bottom**, utilize apenas **padding**.

Você pode economizar até três regras de CSS, pois normalmente os valores se repetem.

E mesmo que os quatro valores sejam diferentes, você também consegue inserir com a **shorthand**.

Veja um exemplo:

```
1  /* Ruim */
2  p {
3      padding-left: 10px;
4      padding-right: 10px;
5      padding-top: 10px;
6      padding-bottom: 10px;
7  }
8
9  /* Bom */
10 p {
11     padding: 10px;
12 }
```







## CSS 4: UTILIZE UNIDADES RELATIVAS

Hoje em dia a maioria das visitas a um site é feito por dispositivo móvel.

Sendo assim seu site deve ser otimizado para isso.

Então em vez de **px**, prefira utilizar unidades como: **em**, **rem**, **vh** e **vw**

Elas vão aumentar e diminuir o elemento proporcionalmente, facilitando a adaptação para dispositivos móveis.

Veja um exemplo:

```
1  p {  
2      font-size: 1rem;  
3  }
```





## CSS 5: COMENTÁRIOS NO CSS

Você pode aplicar os comentários no CSS assim como os do HTML.

Os dois possuem os mesmos pontos negativos e positivos.

Não utilize para informações sensíveis e nem excesso.

Utilize para delimitar seções do código.

Veja um exemplo:



```
1  /* Estilos gerais da aplicação */
2  * {
3      padding: 0;
4      margin: 0;
5  }
```





## CSS 6: MINIFICAR CSS

Minificar CSS é uma opção muito interessante quando você busca performance.

Utilize uma ferramenta para isso, como a **Minify**

Seu CSS ficará comprimido, fazendo com que o documento carregue mais rápido para os usuários.

Melhorar performance melhorar também o **SEO**.





## CSS 7: USE VARIÁVEIS

As variáveis de CSS já são uma realidade!

Você pode definir as cores principais e tamanhos de fonte, por exemplo.

E depois atribuir aos elementos.

Se a regra for alterada, todos os elementos serão atualizados também.

Veja um exemplo:

```
1  :root {  
2      --primary-color: #333  
3  }  
4  
5  p {  
6      color: var(--primary-color);  
7  }
```





## CSS 8: BOX SIZING

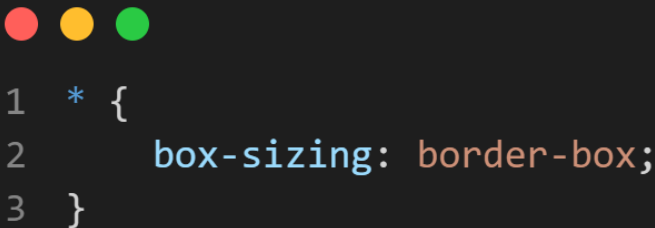
Alguns elementos, como inputs, não respeitam a largura máxima dos elementos pais.

Podemos resolver isso com a propriedade **box-sizing**.

É um padrão comum aplicá-la a todos os elementos.

Isso faz com que eles fiquem mais previsíveis.

Veja um exemplo:



```
1 * {  
2     box-sizing: border-box;  
3 }
```





## CSS 9: CONSISTÊNCIA

Procure manter um padrão ao longo de todo o projeto.

Por exemplo: separe nome de classes ou **underline** ou traço, nunca misture os dois.

Utilize nomes declarativos nas **classes** e **ids**, não coloque nome de tags ou nomes muito específicos.

Veja um exemplo:

```
1  .main_heading {
2      /* ... */
3  }
4
5  .left_container {
6      /* ... */
7  }
8
9  .footer_links {
10     /* ... */
11 }
```





## CSS 10: ORGANIZAÇÃO

Organize as suas folhas de estilo igual ao fluxo do HTML.

Ou seja, estilize primeiramente os elementos superiores (**navbar**, **banners**) e depois os inferiores (**footer**).

Isso fará seu CSS consistente, outros devs conseguirão entender mais fácil seu código e também dar manutenção.

Se você embaralhar as regras de CSS até você futuramente terá problemas em entender como o código foi desenvolvido.

Então sempre espelhe o seu HTML no seu CSS.





## CURSO DOS FUNDAMENTOS

Possuo dois cursos que podem te ajudar a ir mais longe em HTML e CSS.

**Tudo de forma gratuita**, no meu canal de YouTube

Fundamentos do HTML: <https://youtu.be/SV7TL0hxmlQ>

Fundamentos do CSS: <https://youtu.be/vwbegraDXD8>

Ambos com projetos!

Não esqueça de ***deixar seu like*** e se ***inscrever no canal***, me ajuda bastante = )







# Obrigado

Enquanto houver pessoas dispostas  
a aprender, eu estarei criando  
conteúdo para enriquecer ainda  
mais os seus conhecimentos

## Matheus Battisti

Hora de Codar