

Introdução ao HTML

Prof. Jair José Ferronato



Índice de conteúdos

Introdução ao HTML5	3
Requisição e resposta de uma página HTML	7
Nomenclatura de arquivos HTML	8
Estrutura dos arquivos para projetos web locais	9
Codificação e testes de arquivos HTML	12
Estrutura básica de um documento HTML	16
Tags HTML estruturais	17
Tags HTML de conteúdo	23
Trabalhando com links em documentos HTML	43
Trabalhando com listas em documentos HTML	52
Trabalhando com tabelas em documentos HTML	60
Trabalhando com formulários em documentos HTML	66
Campos de entrada do usuário	74
Botões	95

Tutorial básico de HTML

Professor Jair José Ferronato

Introdução ao HTML5

Criada pelo britânico Tim Berners-Lee, o acrônimo HTML significa "HyperText Markup Language", traduzindo ao português "Linguagem de Marcação de Hipertexto". O HTML é o componente básico da web, ele permite inserir o conteúdo e estabelecer a estrutura básica de um website. Portanto, ele serve para dar significado e organizar as informações de uma página na web. Sem isso, o navegador não saberia exibir textos como elementos ou carregar imagens e outros conteúdos.

O HTML consiste, então, em uma série de elementos que você utiliza para delimitar ou agrupar diferentes partes do conteúdo, influenciando como elas aparecem ou se comportam. Cada elemento é delimitado por tags, que são usadas para aplicar diferentes estilos e funcionalidades ao conteúdo. Por exemplo, veja a seguinte linha de conteúdo:

"Este é um curso de HTML!"

Se quisermos que essa linha seja tratada como um parágrafo em um documento HTML, podemos usar a tag de parágrafo, tal como demonstrado no quadro abaixo:

```
<p>Este é um curso de HTML!</p>
```

Vamos explorar esse parágrafo mais profundamente. Logo, observe na figura abaixo as principais partes de um elemento HTML:

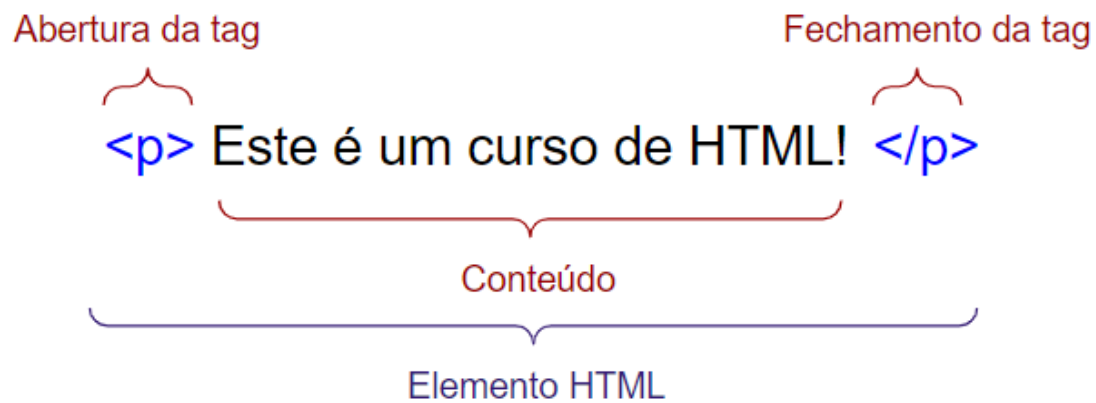


Figura 1 - Principais partes de um elemento HTML

Descrição da imagem: Demonstração de um elemento HTML com o seguinte conteúdo:

Este é um curso de HTML!

Neste contexto, as principais partes de um elemento são:

Tag de abertura: consiste no nome do elemento (por exemplo, p para parágrafo) delimitado por colchetes angulares (`<>`) de abertura e fechamento. Indica o início do elemento e onde seu efeito começa - neste caso, onde é o começo do parágrafo.

Tag de fechamento: similar à tag de abertura, mas inclui uma barra (`/`) antes do nome do elemento. Indica o fim do elemento - neste caso, onde é o fim do parágrafo.

Conteúdo: esse é o conteúdo do elemento, que nesse caso é apenas texto.

Elemento: a tag de abertura, a de fechamento, e o conteúdo formam o elemento.


Algumas tags HTML podem ser abertas ou autocontidas. Isso significa que elas não precisam ser fechadas com `</ >`. Essas tags são normalmente usadas para metadados ou quebras de linha, conforme consta no quadro abaixo:

```
<img>
<br>
<hr>
<input>
<meta>
<link>
```

As tags HTML não têm diferença entre maiúsculas e minúsculas, ou seja, tanto faz escrever <p>, <P>, <p>, <P> , <p>, etc.

Os elementos HTML também podem ter atributos. Os atributos fornecem informações adicionais sobre um elemento HTML que não aparecem diretamente no conteúdo visível. Eles são utilizados para definir propriedades ou comportamentos específicos dos elementos. Observe na figura abaixo um exemplo de atributo em um elemento HTML:

Atributo



`<p class="info" > Este é um curso de HTML! </p>`

Figura 2 - Atributo de um elemento HTML

Descrição: Demonstração de um atributo de um elemento HTML contendo o seguinte texto:

Este é um curso de HTML!

Neste exemplo, "class" é o nome do atributo e "info" é o valor do atributo. O atributo **class** permite atribuir uma ou mais classes ao elemento, que podem ser usadas posteriormente para aplicar estilos CSS, manipulações JavaScript e outras funcionalidades.

A maioria das tags tem seus próprios atributos. Contudo, existem alguns atributos genéricos que podem ser utilizados na maioria das tags HTML, tais como:

- class="..." – Atribui uma classe ao elemento (uma classe pode ser utilizada para um ou mais elementos);

- `id="..."` – Atribui um id ao elemento (um id deve ser único, ou seja atribuído a um único elemento);
- `style="..."` – Permite incluir elementos CSS (estilos) dentro da tag;
- `lang="..."` – Define o idioma principal do elemento;
- `title="..."` – Define o título do elemento;
- `alt="..."` – Define um texto alternativo e, por isso, é muito utilizado em imagens;
- `align="..."` – Permite definir o padrão de alinhamento desse elemento, como por exemplo: right, center, left e justify;
- `width="..."` – Define uma largura para o elemento;
- `height="..."` – Define uma altura para o elemento.

Em HTML, é possível colocar elementos dentro de outros elementos, uma prática conhecida como aninhamento. Essa técnica permite uma maior flexibilidade e riqueza na formatação do conteúdo. Em nosso exemplo anterior, se quisermos enfatizar a palavra "HTML" da nossa frase, podemos envolver a palavra "HTML" com o elemento , **que indica ênfase forte. Observe, no quadro abaixo, este exemplo de aninhamento de elementos HTML.**

```
<p>Este é um curso de <strong>HTML</strong>!</p>
```

Você precisa, no entanto, certificar-se de que seus elementos estejam adequadamente alinhados. No exemplo acima, abrimos primeiro o elemento `<p>` , depois o elemento ``. Portanto, temos que fechar primeiro o elemento `` , depois o elemento `<p>`.

Os hipertextos, por sua vez, são conjuntos de elementos conectados. Esses podem ser palavras, imagens, vídeos, documentos etc. Quando conectados, formam uma rede de informações que permite a comunicação de dados, organizando conhecimentos e guardando informações. Neste contexto, um texto pode conter um link para um vídeo relacionado ou uma imagem pode ser um link para outro documento.

Ao visitar uma página na web, você pode perceber que existem diferentes distribuições e tamanhos para títulos, parágrafos, imagens, vídeos e qualquer outro elemento. Essa estrutura é estabelecida através do HTML. Podemos, portanto, associar o HTML como sendo o " esqueleto " da sua página.

Requisição e resposta de uma página HTML

Um documento HTML é um arquivo de texto comum (ASCII), que tem como extensão .htm ou .html. Para que ele seja visualizado, basta que seja aberto em qualquer navegador, pois o HTML se trata de uma linguagem interpretada. Neste contexto, quando você acessa uma página da web, ocorre uma série de processos que envolvem a comunicação entre o seu navegador e o servidor onde a página está hospedada. Esse processo pode ser descrito em um fluxo de requisição e resposta, que inclui várias etapas desde a solicitação inicial até a exibição da página no navegador.

Logo, para que se possa visualizar um documento HTML, é necessário entender uma série de passos, conforme descritos a seguir:

Passo 1: Com um computador conectado à internet, é necessário ter, entre seus aplicativos instalados, um browser (navegador) para visualizar as páginas HTML

Passo 2: O navegador é direcionado a um servidor da web; então ele solicita uma página - faz uma requisição usando o protocolo HTTP.

Passo 3: O servidor da internet, responde à solicitação do browser (navegador) e envia a página de volta a ele.

Passo 4: A página HTML, então, é visualizada no navegador do computador conectado à internet.

A figura abaixo mostra um fluxograma sobre o processo de requisição e resposta de uma página HTML.

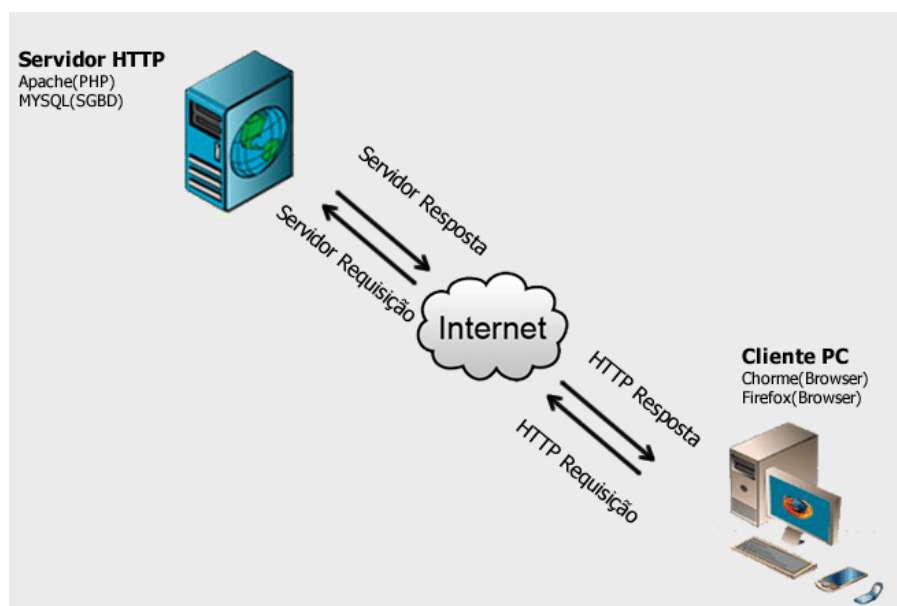


Figura 1 - Fluxograma da requisição e resposta de uma página HTML

Fonte: Banco de Imagens Google, 2024

Descrição da imagem: Ilustração demonstrando um Cliente PC fazendo uma requisição HTTP a um servidor Web, que processa a solicitação e envia uma resposta HTTP de volta ao cliente, exibindo a página web no navegador.

Geralmente, um site é composto por diversas páginas HTML, como por exemplo: um website que contenha três páginas (uma homepage, uma página de contato e uma página de produtos) receberá ao menos três documentos .html distintos, sendo uma para cada página do website.

Cada página consiste em uma série de tags (etiquetas), que podem ser consideradas como sendo os blocos de construção das páginas. Portanto, esses blocos são a maneira com a qual o HTML faz a marcação dos conteúdos, criando a hierarquia e a estrutura do mesmo, dividido entre seções, parágrafos, cabeçalhos, e outros.

Nomenclatura de arquivos HTML

A nomenclatura de arquivos HTML é uma prática crucial para garantir a organização, a manutenção e a acessibilidade dos projetos web. Seguir convenções de nomenclatura claras e consistentes melhora a legibilidade do código, facilita a colaboração e evita erros comuns.

Vejamos abaixo alguns princípios gerais para a nomenclatura de arquivos HTML:

- Utilize sempre letras minúsculas para evitar problemas de sensibilidade a maiúsculas/minúsculas em servidores web.

Exemplo correto: index.html

Exemplo incorreto: Index.HTML

- Não use espaços nos nomes dos arquivos. Espaços podem ser problemáticos para servidores e URLs.

Exemplo correto: meu-arquivo.html

Exemplo incorreto: meu arquivo.html

- Use traços (-) para separar palavras em vez de espaços ou sublinhados.

Exemplo correto: sobre-nos.html

Exemplo incorreto: sobre_nos.html

- Use nomes que descrevam claramente o conteúdo ou a função do arquivo.

Exemplo correto: contato.html

Exemplo incorreto: pagina1.html

- Mantenha um padrão consistente para a nomeação de arquivos em todo o projeto.

Exemplo consistente: servicos.html, produtos.html

Exemplo inconsistente: services.html, produtos.HTML

- Evite o uso de caracteres especiais como !, @, #, \$, etc., que podem causar problemas em URLs e servidores.

Exemplo correto: politica-privacidade.html

Exemplo incorreto: política@privacidade.html

Seguir essas diretrizes de nomenclatura de arquivos HTML contribui para um desenvolvimento web mais organizado, eficiente e compatível com os padrões da indústria.

Estrutura dos arquivos para projetos web locais

Quando você está desenvolvendo um site localmente em seu computador, é importante organizar todos os arquivos do projeto de forma consistente e acessível. Abaixo, veremos algumas diretrizes para manter a estrutura de arquivos organizada.

Localização da pasta do projeto: Um site é composto por diversos tipos de arquivos, incluindo conteúdo em texto, código, folhas de estilo, e mídias como imagens e vídeos. Para criar um site de maneira eficiente, é essencial organizar esses arquivos em uma estrutura bem definida em seu computador local. Você pode escolher qualquer lugar conveniente em seu computador para armazenar seus projetos de sites, como o Desktop, sua pasta Home ou a raiz do seu disco rígido.

Criação da pasta principal: Escolha um local e crie uma pasta principal para todos os seus projetos de sites. Um nome sugestivo pode ser **projetos-web** ou algo semelhante.

Organização de projetos individuais: Dentro da pasta **projetos-web**, crie uma subpasta específica para cada projeto de site. Por exemplo, para seu primeiro site, você pode criar uma pasta chamada **meu-projeto-html** ou um nome mais criativo que reflita o propósito do site.

Estrutura interna do projeto: Dentro de cada pasta de projeto, organize os arquivos de maneira que reflita a estrutura do site que será publicado em um servidor. Isso pode incluir subpastas para imagens, css, js, entre outras.

Vamos explorar agora, a estrutura essencial que seu site de teste deve ter. Nos projetos de sites, geralmente precisamos de um arquivo HTML principal e pastas específicas para imagens, estilos e scripts. Vamos criar essa estrutura passo a passo, considerando o uso do editor de texto Visual Studio Code:

Utilizando seu software gerenciador de arquivos (tal como o Windows Explorer), escolha um local acessível em seu computador (por exemplo, Desktop ou pasta Home).

Crie uma nova pasta com um nome descritivo, como **projetos-web**. Esta será a pasta onde vamos adicionar todos os projetos que vamos utilizar no decorrer do curso.

Agora, abra o editor de texto Visual Studio Code. A figura abaixo ilustra a tela inicial do Visual Studio Code.

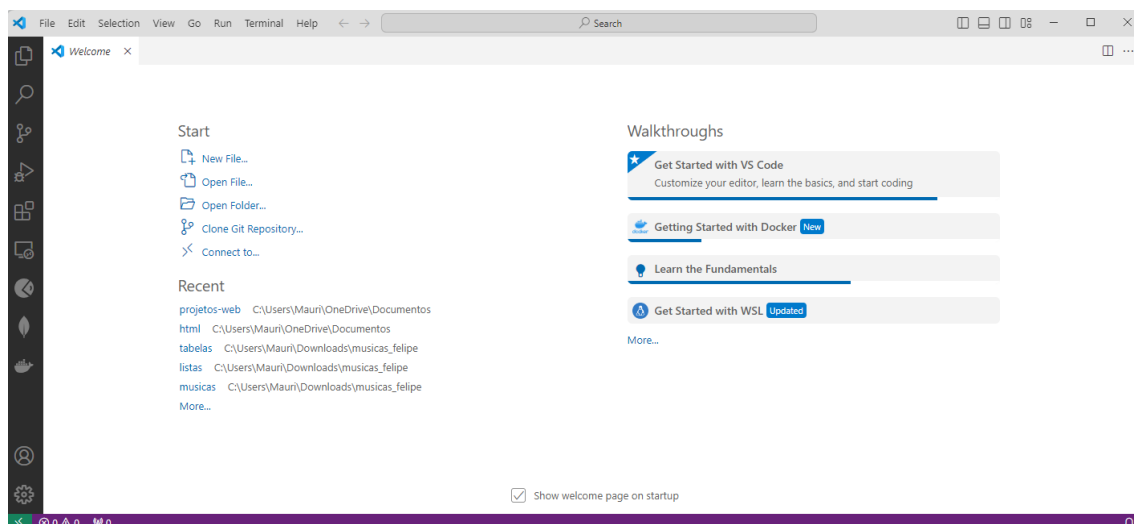


Figura 1 - Tela inicial do Visual Studio Code

Descrição da imagem: A tela inicial do Visual Studio Code inclui um painel de boas-vindas, oferecendo atalhos para criar novos arquivos, abrir pastas e acessar documentos recentes, além de opções para personalizar o layout e explorar mais funcionalidades do VS Code.

Em seguida, vamos abrir a pasta geral dos nossos projetos no Visual Studio Code. Assim, clique em **File (Arquivo)** na barra de menu e selecione **Open Folder... (Abrir Pasta...)**. Navegue até a pasta **projetos-web** e clique em **Select Folder (Selecionar Pasta)**. Pronto, abaixo vamos criar as pastas para cada projeto que formos testar neste curso.

Neste primeiro exemplo, vamos criar uma nova subpasta chamada de **meu-projeto-html**. No **Explorer (Explorador)** do Visual Studio Code, clique com o botão direito na pasta **projetos-web** e selecione **New Folder (Nova Pasta)**. Digite **meu-projeto-html** e pressione **Enter**.

Agora, com a pasta do projeto **meu-projeto-html** criado, vamos adicionar um arquivo para testar um código HTML. No **Explorer (Explorador)** do Visual Studio Code, clique com o botão direito na pasta **meu-projeto-html** e selecione **New File (Novo Arquivo)**. Agora, digite **index.html** e pressione **Enter**.

Veja na figura abaixo um exemplo desta estrutura básica do nosso projeto web no Visual Studio Code:

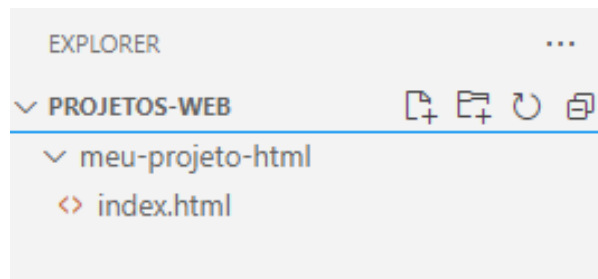


Figura 2 - Estrutura de diretórios do projeto "meu-projeto-html"

Descrição da imagem: A pasta "projetos-web" contém a subpasta "meu-projeto-html", que, por sua vez, abriga o arquivo "index.html".

OBS: Recomendamos que você nomeie suas pastas e arquivos usando apenas letras minúsculas e sem espaços. O motivo é que muitos servidores web são case-sensitive, ou seja, diferenciam maiúsculas de minúsculas. Esses servidores web também podem ter dificuldade em lidar com espaços nos nomes dos arquivos. Assim, substitua espaços por traços para garantir a compatibilidade com todos os sistemas. Neste contexto, sugerimos que você nomeie seus arquivos como sendo "**meu-arquivo.html**" em vez de "Meu Arquivo.html" e nomeie suas pastas como sendo "**site-teste/imagens**" em vez de "site-teste/Imagens do Site".

Com o desenvolvimento de seus estudos, você perceberá a necessidade de separar seu código HTML dos códigos CSS, JavaScript e das imagens. Assim, você já pode começar a pensar nesta organização de arquivo. Segue abaixo uma sugestão de estrutura mais complexa de projeto:

- Dentro da pasta **meu-projeto-html**, crie uma nova pasta chamada **imagens**. Esta pasta armazena todas as imagens que serão usadas no site.
- Em seguida, dentro da pasta **meu-projeto-html**, crie uma nova pasta chamada **estilos**. Esta pasta contém os arquivos CSS que definem a aparência e o estilo do seu site, como cores de texto e de fundo.
- Após, dentro da pasta **meu-projeto-html**, crie uma nova pasta chamada **scripts**. Esta pasta armazena todos os arquivos JavaScript, que adicionam funcionalidades interativas ao seu site, como botões que carregam dados quando clicados.

Veja na figura abaixo um exemplo desta estrutura mais completa do nosso projeto web no Visual Studio Code:

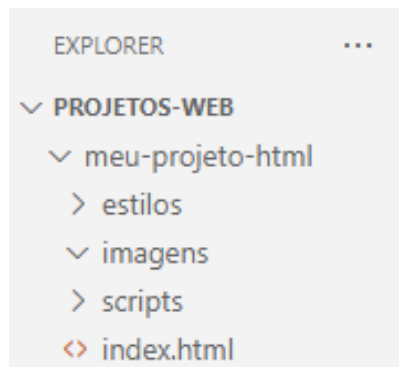


Figura 3 - Estrutura final de diretórios do projeto "meu-projeto-html"

Descrição da imagem: Foram adicionadas à pasta "meu-projeto-html" as subpastas "estilos", "imagens" e "scripts".

Neste momento, é importante que você crie esta estrutura de diretórios e arquivos em seu computador. Precisaremos desta configuração para que você possa fazer os testes com os códigos que iremos aprender no decorrer do curso.

Codificação e testes de arquivos HTML

Executar um arquivo HTML é uma etapa essencial para visualizar e testar o conteúdo de uma página web. Existem várias maneiras de fazer isso, dependendo das necessidades do projeto e do ambiente de desenvolvimento. No decorrer deste texto, detalharemos algumas das principais formas de executar um arquivo HTML.

Entretanto, primeiramente, vamos criar um código HTML simples para praticar nossas habilidades em codificação e teste de páginas web. Não se preocupe se você não entender completamente o código neste momento; o objetivo é garantir que você consiga copiar o código-fonte e testá-lo corretamente. Neste contexto, abra o arquivo **index.html** e insira o seguinte código apresentado no quadro abaixo:

```
<!DOCTYPE html>

<html lang="pt-br">

<head>

<meta charset="UTF-8">

<meta name="viewport" content="width=device-width, initial-scale=1.0">

<title>Meu Projeto HTML</title>

</head>

<body>

<h1>Bem-vindo ao Meu Projeto HTML</h1>

<p>Este é um parágrafo de exemplo.</p>

</body>

</html>
```

Para executar e testar este arquivo HTML, podemos fazer das seguintes maneiras: abrir diretamente no navegador, arrastar e soltar no navegador, usar um editor de texto com live preview e usar um serviço de hospedagem online.

Abrir diretamente no navegador

Este é o método mais simples, onde você abre o arquivo HTML diretamente em um navegador web. Neste contexto, siga os passos abaixo:

Primeiro, certifique-se de que seu arquivo HTML está salvo em um local acessível no seu computador (por exemplo, Desktop, Documentos).

Em seguida, abra o gerenciador de arquivos (tal como o Windows Explorer) e navegue até a pasta onde seu arquivo HTML (index.html) está salvo.

Clique com o botão direito do mouse no arquivo index.html e selecione "Abrir com" e escolha o navegador de sua preferência (por exemplo, Chrome, Firefox, Edge, Safari).

Veja abaixo nossa página **index.html** em execução no navegador:

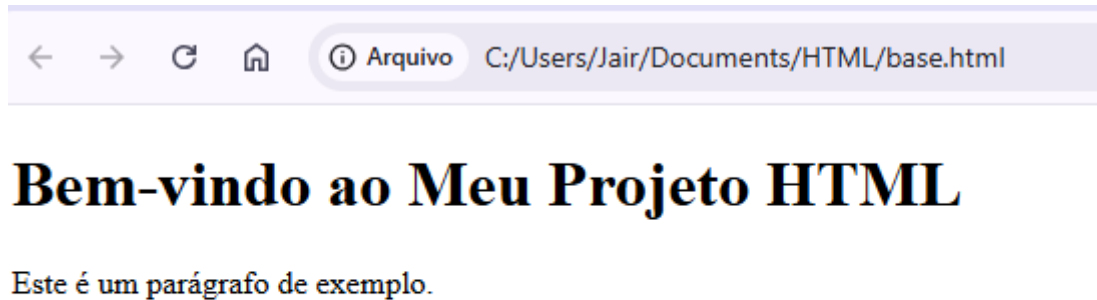


Figura 1 - Página index.html em execução no navegador

Descrição da imagem: Na tela do navegador, o título <h1> aparece em uma fonte maior e em negrito, destacando-se mais que o texto do parágrafo <p>, que é menor e não está em negrito.

Este método tem como vantagens a simplicidade e rapidez nos testes. Entretanto, ele não suporta execução de scripts do lado do servidor (ex.: PHP).

Arrastar e soltar no navegador

Neste método, basta você arraste o arquivo HTML diretamente para a janela aberta do navegador. Neste contexto, siga os passos abaixo:

Abra o navegador de sua escolha.

Localize o arquivo HTML usando o gerenciador de arquivos (tal como o Windows Explorer).

Arraste o arquivo e solte-o na janela do navegador.

Este método também é rápido e simples de ser executado. Entretanto, ele também não suporta execução de scripts do lado do servidor (ex.: PHP).

Usar um editor de texto com live preview

Muitos editores de texto modernos têm recursos de visualização ao vivo. Consulte a documentação do seu editor específico para instruções sobre como ativar essa funcionalidade. No Visual Studio Code, podemos adicionar a **extensão Live Server**. Para isso, siga os passos abaixo:

Primeiro, abra o Visual Studio Code.

Clique no ícone de Extensões na barra lateral esquerda (ícone de quadrado com pequenos quadrados dentro) .

Pesquise por "Live Server" e clique em "Instalar". A figura abaixo ilustra a tela de instalação da extensão Live Server do Visual Studio Code

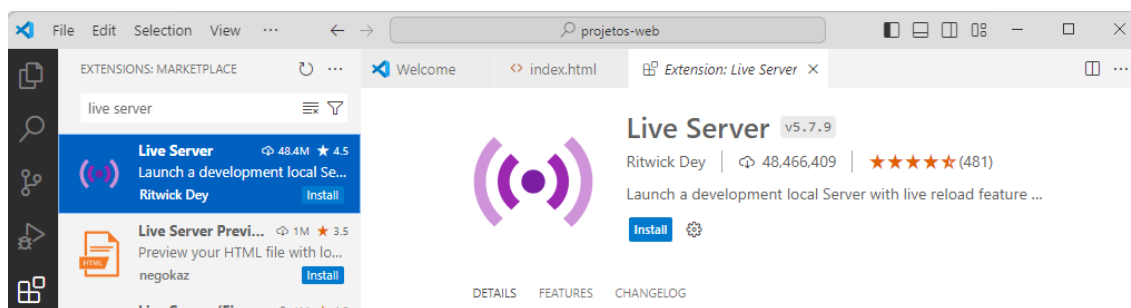


Figura 2 - Tela de instalação da extensão Live Server do Visual Studio Code

Descrição: A tela de instalação da extensão Live Server do Visual Studio Code mostra opções para instalar e configurar a extensão, facilitando a visualização em tempo real das alterações feitas no código.

Em seguida, clique com o botão direito do mouse no arquivo **index.html** e selecione "Open with Live Server". O navegador abrirá automaticamente uma nova aba exibindo sua página HTML.

Veja abaixo nossa página "index.html" em execução no navegador usando a extensão Live Server do Visual Studio Code:



Bem-vindo ao Meu Projeto HTML

Este é um parágrafo de exemplo.

Figura 3 - Execução do arquivo index.html usando a extensão Live Server do VS Code

Descrição da imagem: A execução do arquivo index.html usando a extensão Live Server do Visual Studio Code abre o arquivo no navegador e atualiza automaticamente as mudanças feitas no código, permitindo ver as alterações em tempo real.

A extensão **Live Server** permite visualizar sua página ao vivo no navegador. Além disso, ela oferece atualização em tempo real, o que significa que, ao modificar o código, as alterações são imediatamente refletidas no navegador.

Estrutura básica de um documento HTML

Um documento HTML é a base de qualquer página web e possui uma estrutura definida que organiza o conteúdo e os metadados de forma lógica e acessível. Esta estrutura básica é composta por uma série de elementos que devem ser organizados corretamente para garantir que a página seja exibida corretamente pelos navegadores.

Observe no quadro abaixo um exemplo demonstrando a estrutura básica de um documento HTML.

```
<!DOCTYPE html>

<html>

<head>

<meta charset="UTF-8">

<title>Título do Documento</title>

</head>

<body>

<p>Corpo do Documento</p>

</body>

</html>
```


Vamos agora entender para que serve cada uma dessas tags:

- `<!DOCTYPE html>` – Esta tag informa ao navegador a versão do HTML que está sendo utilizada no documento. Por exemplo: no HTML5, basta incluir `<!DOCTYPE html>`, e assim o navegador já saberá que se trata de um documento na versão HTML5;
- `<html>` – Esta tag é o elemento básico da estrutura do HTML. Assim sendo, ela conterá todos os elementos do seu documento. Todo documento HTML deve iniciar e finalizar com essa tag;
- `<head>` – Essa tag delimita o cabeçalho do documento. Não possui nenhum valor visível, porém é capaz de transmitir ao navegador diversas informações muito úteis e essenciais a uma boa apresentação do seu documento HTML;
- `<meta>` – Essa tag permite inserir metadados ao seu documento, no caso acima, a informação `charset="UTF-8"`, que garante a compatibilidade do código com os caracteres de padrão latino americano;
- `<title>` – Essa tag define o título da sua página, o nome que aparecerá na sua aba, janela ou guia.
- `<body>` – Finalmente, a tag que representa o corpo do documento. Em síntese, é nessa tag que todos os elementos visíveis do seu site devem ser inseridos.

Agora que já conhecemos as tags HTML que formam a estrutura básica de uma página, podemos então estudar as diversas tags que contemplarão o documento.

Tags HTML estruturais

Quando estamos desenvolvendo um web site, percebemos que as páginas da web podem e serão muito diferentes umas das outras, mas todas tendem a compartilhar componentes padrão semelhantes. Geralmente, uma página desenvolvida em HTML possui as seguintes partes:

cabeçalho (header): representa, normalmente, uma grande faixa na parte superior com um grande título e/ou logotipo. É aí que, geralmente, ficam as principais informações comuns sobre um site, sendo mostrado em diversas páginas do site.

barra de navegação: contém links para as principais seções do site; geralmente representado por botões de menu, links ou guias. Como o cabeçalho, esse conteúdo geralmente permanece consistente de uma página para outra. É uma parte muito importante em nosso site, pois uma navegação inconsistente em seu site levará a usuários confusos e frustrados.

conteúdo principal: representa uma grande área no centro do site que contém a maior parte do conteúdo exclusivo de uma determinada página web, por exemplo, o vídeo que você deseja assistir ou a história principal que você está lendo ou o mapa que deseja visualizar ou as manchetes de notícias, etc. Esta é a única parte do site que, definitivamente, irá variar de página para página!

barra lateral (sidebar): contém algumas informações periféricas, links, cotações, anúncios etc. Geralmente, isso é contextual ao conteúdo principal (por exemplo, em uma página de um artigo de notícias, a barra lateral pode conter a biografia do autor ou links para artigos relacionados), mas há também casos em que você encontrará alguns elementos recorrentes como um sistema de navegação secundário.

rodapé (footer): representa uma faixa na parte inferior da página que, geralmente, contém avisos de direitos autorais ou informações de contato. É um lugar para colocar informações comuns (como o cabeçalho). O rodapé também fornece links para acesso rápido a conteúdo popular.

Na figura abaixo, podemos verificar um site e identificar suas diferentes partes.



Figura 1 - Exemplo de um site e suas partes

Fonte: <https://ifpr.edu.br/>

Descrição da imagem: O site é composto por várias partes: o cabeçalho, que fica no topo e contém o título e o logotipo; a barra lateral, que contém o menu de navegação; a parte principal, onde são exibidas as notícias ou o conteúdo principal; e o rodapé, que fica na parte inferior e contém informações adicionais, como contatos ou links úteis.

O exemplo acima permite ilustrar um exemplo típico de layout de website. Em seu código HTML, você pode marcar seções de conteúdo com base em sua funcionalidade. Um dos principais objetivos do HTML é dar estrutura de texto e significado, também conhecido como semântica, para que um navegador possa exibi-lo corretamente.

Para implementar essa marcação semântica, o HTML fornece tags dedicadas que você pode usar para representar essas seções, por exemplo:

cabeçalho: `<header>`.

barra de navegação: `<nav>`.

conteúdo principal: `<main>`, com várias subseções de conteúdo representadas por `<section>`, `<article>` e elementos `<div>`.

rodapé: `<footer>`.

Veja na figura abaixo um exemplo gráfico da disposição das seções de um documento HTML:

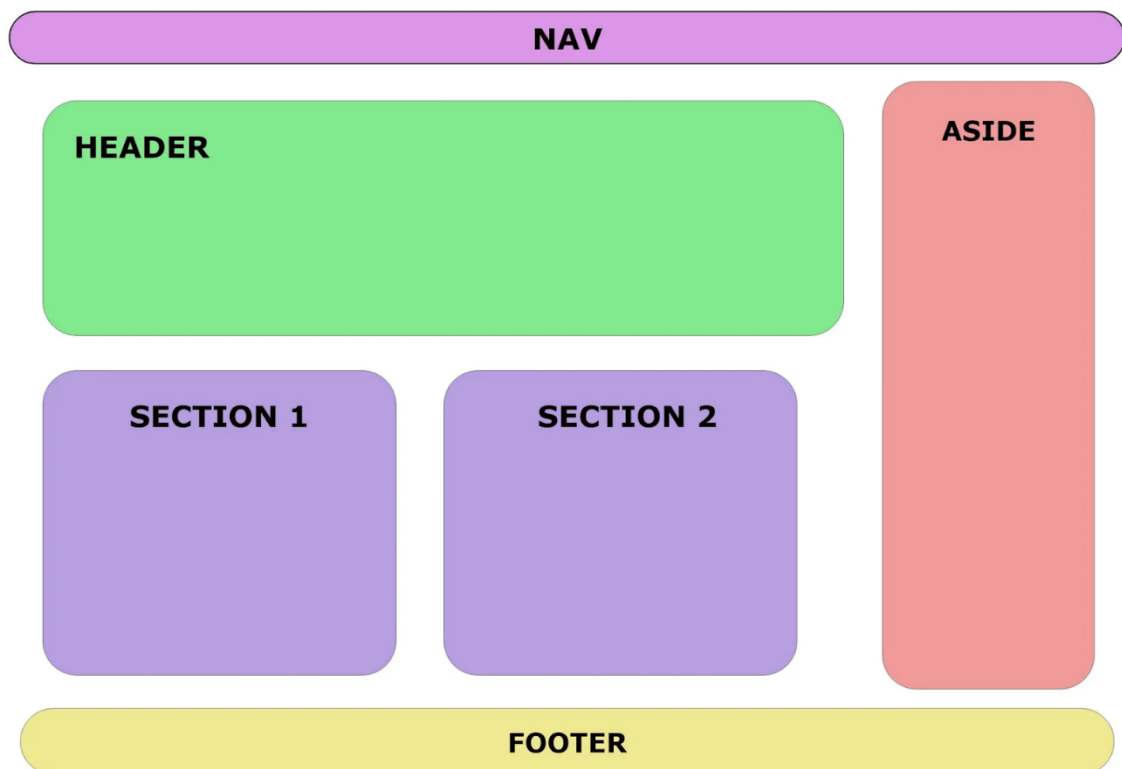


Figura 2 - Exemplo gráfico da disposição das seções de um documento HTML.

Fonte: Banco de Imagens Google, 2024

Descrição da imagem: Um documento HTML é dividido em várias seções: o header (cabeçalho) e o nav (navegação), seguidos pelos section 1 e section 2, que contêm o conteúdo principal. O aside é uma barra lateral com informações adicionais, e o footer (rodapé) está na parte inferior com detalhes complementares.

Essas são as tags principais do HTML e, com elas, conseguimos definir uma estrutura básica para o nosso site e dentro dos padrões estabelecidos pelos principais navegadores de Internet.

Neste sentido, imagine a seguinte estrutura básica para um site, conforme quadro abaixo:

```
<!DOCTYPE html>

<html lang="pt-br">

<head>

<meta charset="UTF-8">

<meta name="viewport" content="width=device-width, initial-scale=1.0">

<title>Aula de Programação Web</title>

</head>

<body>

<header>

<h1>cabeçalho</h1>

</header>

<nav>

<h2>Menu</h2>

</nav>

<main>

<section>

<h2>Conteúdo do site</h2>

<p>Mais conteúdo do site</p>

</section>

<aside>

<h2>Conteúdo relacionado</h2>

</aside>

</main>

<footer>
```

```
<h2>Rodapé</h2>
```

```
</footer>
```

```
</body>
```

```
</html>
```

Veja abaixo a listagem das principais tags HTML estruturais:

- <header> – Essas tags definem um cabeçalho. Portanto, todo conteúdo que estiver dentro dela faz parte de um cabeçalho, podendo ser utilizado dentro de outras sessões. Não confundir com as tags <h1>;
- <main> – Essas tags representam o conteúdo principal do seu corpo, ou seja, o conteúdo relacionado diretamente com o tópico central da página ou com a funcionalidade central da aplicação;
- <footer> – Essas tags definem um rodapé para a página, geralmente utilizadas no final da página;
- <section> – Essas tags definem uma sessão para sua página;
- <article> – Essas tags definem um artigo da sua página. Nesse sentido, são utilizadas para separar o conteúdo da sua página. Muito utilizado principalmente por blogs ou páginas de conteúdo;
- <aside> – Essas tags representam uma seção de uma página cujo conteúdo é tangencialmente relacionado ao conteúdo do seu entorno, que poderia ser considerado separado do conteúdo;
- <nav> – Essa tag define um conteúdo de navegação. Portanto, é muito utilizado em conjunto com listas e na criação de menus;
- <div> – Define uma divisão da página. Desta forma, funciona como um container para conteúdo de fluxo. Uma vez que não possui um valor semântico, é muito utilizado para organizar melhor o conteúdo.
- <!-- --> - São tags de comentários (que é aberta com <!--), ou seja, todos elementos incluídos dentro dela serão apenas comentários, não serão visíveis no navegador.

Tags HTML de conteúdo

Agora que você já conhece as principais tags estruturais, pode organizar o seu conteúdo de forma adequada. Assim, no decorrer deste módulo, vamos listar no decorrer deste curso as principais tags para incluir conteúdo à página, como títulos, parágrafos, imagens, links, etc.

2.3.1 Cabeçalhos

Os cabeçalhos são normalmente utilizados para identificar páginas e seções e possuem aparência diferenciada do restante do texto. No HTML há seis níveis de cabeçalhos que podem ser utilizados por meio das tags `<h1>` até `<h6>`. O quadro abaixo demonstra como você pode usar cada uma delas:

```
<h1>Título de nível 1</h1>
```

```
<h2>Título de nível 2</h2>
```

```
<h3>Título de nível 3</h3>
```

```
<h4>Título de nível 4</h4>
```

```
<h5>Título de nível 5</h5>
```

```
<h6>Título de nível 6</h6>
```

Veja o resultado no navegador, conforme figura abaixo:

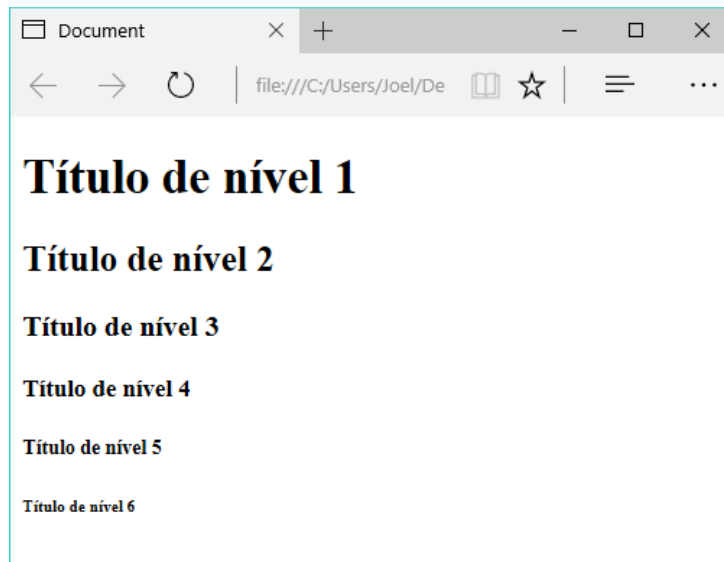


Figura 1 - Níveis de cabeçalhos do HTML

Descrição da imagem: Os níveis de cabeçalhos no HTML vão de `<h1>` a `<h6>`. O `<h1>` é o mais importante e tem a maior fonte, usado para o título principal. Os cabeçalhos subsequentes, `<h2>` a `<h6>`, são usados para subtítulos e têm fontes progressivamente menores, indicando menor importância.

Os cabeçalhos `<h1>` são os mais importantes e geralmente são usados para títulos de página ou seções principais. Os cabeçalhos `<h2>` a `<h6>` são usados para subseções de títulos, com `<h2>` sendo o próximo nível de importância após `<h1>`, e assim por diante. O tamanho e o estilo dos cabeçalhos são definidos pelo navegador padrão, mas podem ser personalizados com CSS.

OBS: De acordo com as regras de SEO (Search Engine Optimization - otimização para mecanismos de busca), é recomendado que uma página possua apenas uma tag `<h1>` que indique seu assunto, pois essa tag informa aos motores de busca qual sua principal palavra-chave.

Parágrafos

Em HTML, você pode criar parágrafos usando a tag `<p>`. Basta envolver o texto que deseja que seja um parágrafo com essa tag. No quadro abaixo está um exemplo simples de parágrafo em HTML:

<p>Primeiro parágrafo do texto.</p>

<p>Segundo parágrafo do texto.</p>

<p>Terceiro parágrafo do texto.</p>

Veja o resultado no navegador, conforme figura abaixo:

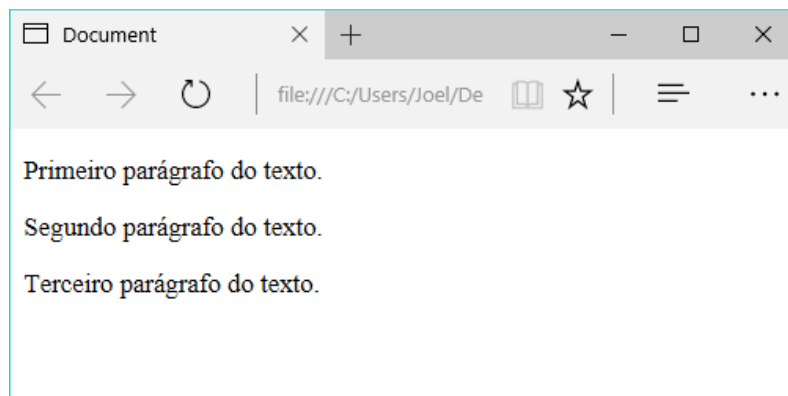


Figura 1 - Demonstração de como os parágrafos são exibidos no navegador.

Descrição da imagem: No navegador, os parágrafos são exibidos como blocos de texto separados, com espaço entre eles, um abaixo do outro, para facilitar a leitura.

Cada <p> representa um parágrafo separado. Quando renderizado em um navegador, cada parágrafo será exibido em uma nova linha e possivelmente com algum espaço entre eles, dependendo do estilo padrão do navegador.

Parte inferior do formulário

Formatação de texto

Embora a maioria das tags HTML sejam usadas para criar elementos, o HTML também fornece tags de formatação de texto para aplicar estilos específicos relacionados ao texto a partes do texto. Este tópico inclui exemplos de formatação de texto HTML, como destaque, negrito, itálico, sublinhado, tachado, sobrescrito e subscrito.

DESTAQUE

O elemento `<mark>` é novo no HTML5 e é utilizado para marcar ou destacar texto em um documento “devido à sua relevância em outro contexto”. No quadro abaixo, mostramos um exemplo de código HTML mostrando os resultados de uma pesquisa onde o usuário inseriu uma consulta de pesquisa e os resultados são mostrados destacando a consulta desejada.

```
<p>Aqui está o conteúdo de um artigo que contém a <mark>consulta pesquisada</mark> que estamos procurando. Destacar o texto tornará mais fácil para o usuário encontrar o que procura.</p>
```

Veja o resultado no navegador, conforme a figura abaixo:

[Image 33a44842 not found in library]

Figura 1 - Demonstra o resultado da utilização da tag `<mark>` no navegador

Descrição da imagem: No navegador, a tag `<mark>` destaca o texto, fazendo com que ele apareça com um fundo amarelo, semelhante a um marcador de texto. Neste caso, fica em destaque "consulta pesquisada"

OBS: A formatação padrão é texto preto sobre fundo amarelo, mas isso pode ser alterado com CSS (Cascading Style Sheets).

TEXTO EM NEGRITO

Para texto em negrito, use as tags `` ou ``. Veja o exemplo demonstrado no quadro abaixo:

```
<strong>Texto em negrito aqui</strong>
```

```
<b>Outro texto em negrito aqui</b>
```

Veja o resultado no navegador, conforme figura abaixo:

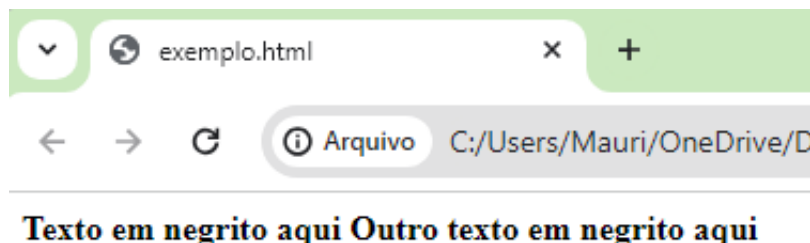


Figura 2 - Demonstra o resultado do uso das tags e no navegador.

Descrição da imagem: No navegador, tanto a tag quanto a tag fazem o texto aparecer em negrito.

Qual é a diferença?

Observamos que ambas tags são usadas para aplicar ênfase ao texto, mas têm diferenças semânticas que afetam como os motores de busca e leitores de tela interpretam o conteúdo. Use quando quiser enfatizar que o texto é semanticamente importante (que tem um significado forte), como avisos, títulos ou palavras-chave. Use apenas para estilizar o texto em negrito, sem atribuir significado semântico adicional.

TEXTO EM ITÁLICO

Para colocar o texto em itálico, use as tags ou <i>. Veja um exemplo, conforme apresentado no quadro abaixo:

```
<em>Texto em itálico aqui</em>
```

```
<i>Texto em itálico aqui</i>
```

Veja o resultado no navegador, conforme a figura abaixo:

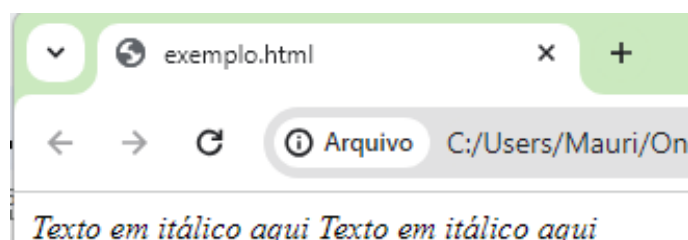


Figura 3 - Demonstra o resultado do uso das tags e <i> no navegador.

Descrição da imagem: No navegador, tanto a tag quanto a tag <i> fazem o texto aparecer em itálico.

Qual é a diferença?

Ambas HTML são usadas para aplicar ênfase ao texto, mas elas têm diferenças semânticas que afetam como os motores de busca e leitores de tela interpretam o conteúdo. Use quando quiser enfatizar que o texto é semanticamente enfatizado, como citações, termos técnicos ou expressões idiomáticas. Use <i> apenas para estilizar o texto em itálico, sem atribuir significado semântico adicional. Por exemplo, se você quiser enfatizar a ação dentro de uma frase, pode fazê-lo enfatizando-a em itálico via : "Você já terminou de estudar?". Mas se você estivesse identificando um livro ou jornal que normalmente colocaria em itálico estilisticamente, você simplesmente usaria <i>: "Fui forçado a ler Romeu e Julieta no ensino médio.".

TEXTO SUBLINHADO

A tag <u> é usada para sublinhar texto. No entanto, o uso desta tag é desencorajado devido à sua falta de semântica clara. Em vez disso, é preferível usar CSS para estilizar o texto, incluindo a adição de sublinhado, pois isso separa a formatação do conteúdo e fornece mais controle sobre a aparência do texto. No quadro abaixo está um exemplo de como você pode usar esta tag:

```
<u>Este texto está sublinhado.</u>
```

Veja o resultado no navegador, conforme a figura abaixo:

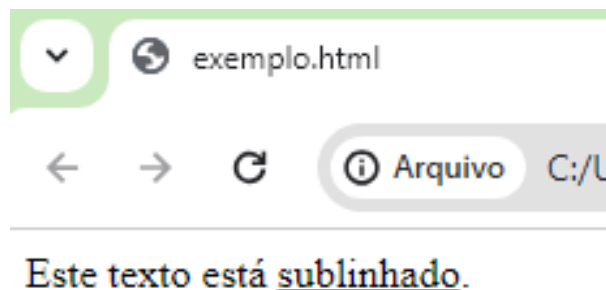


Figura 4 - Demonstra o resultado do uso da tag <u> no navegador.

Descrição da imagem: No navegador, a tag <u> sublinha o texto, destacando-o com uma linha horizontal abaixo das palavras. Neste caso, a palavra "sublinhado" está com este efeito.

TEXTO TACHADO

A tag <s> é usada em HTML para representar texto que foi "riscado" ou "cortado", indicando que o texto não é mais válido ou não é mais preciso. Isso é comumente usado para mostrar informações que foram removidas ou desatualizadas. Por padrão, o texto marcado com <s> é exibido com uma linha horizontal através dele, indicando que foi "cortado". O quadro abaixo apresenta um exemplo de como você pode usar essa tag:

```
<p>Este produto custava <s>R$100</s>, mas agora está em promoção por  
<s>R$80</s> R$60.</p>
```

Veja o resultado no navegador, conforme a figura abaixo:

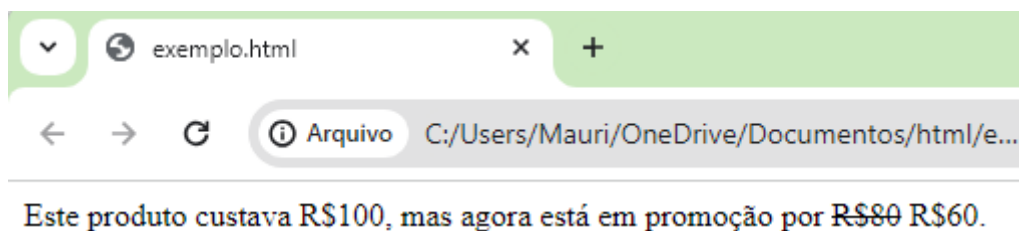


Figura 5 - Demonstra o resultado do uso da tag <s> no navegador.

Descrição da imagem: No navegador, a tag <s> risca o texto com uma linha horizontal, indicando que o texto foi deletado ou está incorreto. Neste caso, "R\$80" está com este efeito.

TEXTO SOBRESCRITO

A tag HTML <sup> é usada para definir texto sobrescrito, ou seja, para colocar o texto em uma posição superior à linha de base normal. Essa tag é comumente utilizada para representar expoentes, notações matemáticas, marcas de registro, entre outros. Veja um exemplo no quadro abaixo:

`<p>2³ representa dois elevado ao cubo.</p>`

`<p>O custo é de R$10⁹⁹.</p>`

Veja o resultado no navegador, conforme a figura abaixo:

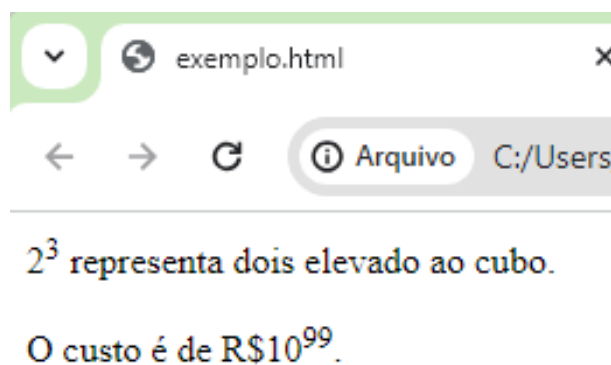


Figura 6 - Demonstra o resultado do uso da tag `<sup>` no navegador.

Descrição da imagem: No navegador, a tag `<sup>` faz o texto aparecer em sobrescrito, elevando-o acima da linha base, como em notas de rodapé ou expoentes. Neste caso, "99" está com este efeito.

TEXTO SUBSCRITO

A tag HTML `<sub>` é usada para definir texto subscrito, ou seja, para colocar o texto em uma posição inferior à linha de base normal. Essa tag é frequentemente utilizada para representar fórmulas químicas, notas de rodapé, números ordinais, entre outros. Veja um código de exemplo no quadro abaixo:

`<p>A fórmula da água é H₂O.</p>`

`<p>Este é o texto da nota de rodapé₁.</p>`

Veja o resultado no navegador, conforme a figura abaixo:

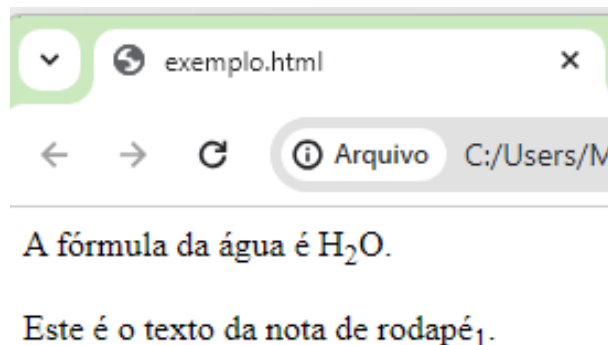


Figura 7 - Demonstra o resultado do uso da tag <sub> no navegador.

Descrição da imagem: No navegador, a tag <sub> faz o texto aparecer em subscrito, posicionando-o abaixo da linha base, como em fórmulas químicas ou notas de rodapé. Neste caso, o "2" em "H₂O" e o "1" em "rodapé₁" tem esse efeito aplicado.

QUEBRA DE LINHA

A tag
 é usada para inserir uma quebra de linha, ou seja, ela cria uma nova linha no local onde é colocada. Ela é útil quando você quer separar visualmente duas linhas de texto em um lugar onde a quebra de linha normal não é realizada automaticamente pelo navegador. Veja no quadro abaixo um exemplo:

```
<p>Este é o primeiro parágrafo.<br>Este é o segundo parágrafo.</p>
```

Veja o resultado no navegador, conforme a figura abaixo:

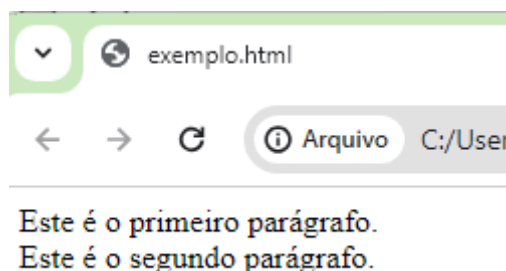


Figura 1 - Demonstra o resultado do uso da tag
 no navegador.

Descrição da imagem: No navegador, a tag
 insere uma quebra de linha, movendo o texto seguinte para a linha de baixo, como em uma nova linha em um documento.

Neste exemplo, a quebra de linha (
) separa os dois parágrafos, criando uma nova linha para o segundo parágrafo iniciar.

OBS: É importante notar que a tag
 não possui uma tag de fechamento, já que ela não envolve nenhum conteúdo. Ela é uma tag vazia e não precisa de uma barra de fechamento, tal como
, mas que também é válido em HTML.

SEPARADOR

A tag HTML <hr> é usada para inserir uma linha horizontal, conhecida como "separador" ou "divisor", em uma página HTML. Ela é útil para separar visualmente seções de conteúdo, como partes de um artigo, ou para criar uma divisão entre diferentes tópicos em uma página. Por padrão, a linha horizontal criada pela tag <hr> estende-se pela largura do contêiner pai (normalmente a largura total da página). Veja o exemplo de código no quadro abaixo:

```
<p>Este é um parágrafo.</p>

<hr>

<p>Este é outro parágrafo.</p>
```

Veja o resultado no navegador, conforme a figura abaixo:

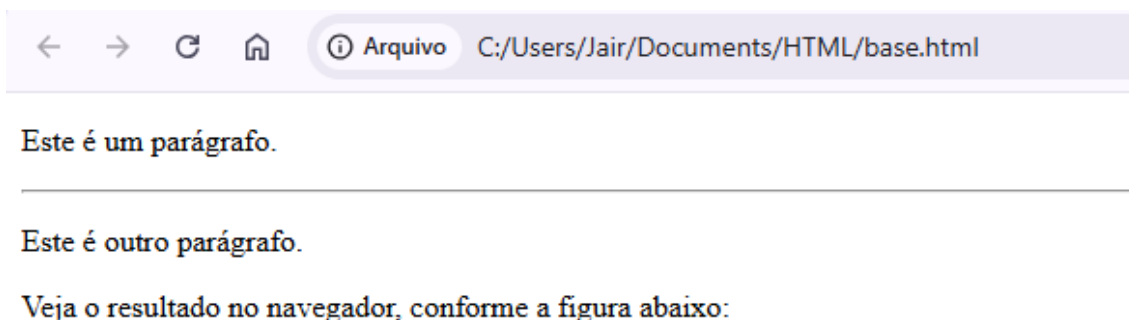


Figura 1 - Demonstra o resultado do uso da tag <hr> no navegador.

Descrição da imagem: No navegador, a tag <hr> insere uma linha horizontal para separar conteúdos, como uma divisão visual entre seções do texto.

No exemplo acima, a tag <hr> é usada para criar uma linha horizontal entre os dois parágrafos, separando-os visualmente.

A tag `<hr>` é uma tag vazia, o que significa que não possui uma tag de fechamento correspondente. Ela pode ser usada sozinha, sem a necessidade de uma tag de fechamento, como `<hr/>`.

Inserindo uma imagem

Para inserir uma imagem em uma página HTML, você pode usar a tag ``. O Quadro abaixo apresenta um exemplo de como você pode usar essa tag:

```

```

Neste exemplo:

A tag `` é usada para inserir a imagem.

O atributo `src` especifica o caminho para a imagem que você deseja exibir.

O atributo `alt` fornece um texto alternativo para a imagem, que é exibido se a imagem não puder ser carregada ou se o usuário estiver usando um leitor de tela.

OBS: A tag `` é uma tag de auto término, o que significa que não há uma tag de fechamento ``.

Para ilustrar o uso de imagens em arquivos HTML, vamos experimentar a utilização da tag `` em nosso projeto de teste, denominado “meu-projeto-html”. Primeiro, baixe o uma imagem (com extensão .PNG) de sua preferência na internet. Renomeie este arquivo para “imagem.png”. Em seguida, salve esta imagem na pasta “imagens” do nosso projeto, conforme demonstrado na figura abaixo:

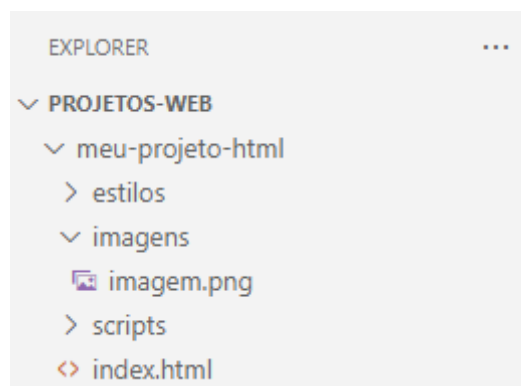


Figura 1 - Estrutura de pastas do projeto "meu-projeto-html"

Descrição da imagem: No VS Code, o arquivo imagem.png foi adicionado à pasta imagens dentro de meu-projeto-html.

Em seguida, altere o arquivo **index.html** para mostrar a imagem no navegador, conforme quadro abaixo.

```
<!DOCTYPE html>

<html lang="pt-br">

<head>

<meta charset="UTF-8">

<meta name="viewport" content="width=device-width, initial-scale=1.0">

<title>Meu Projeto HTML</title>

</head>

<body>

<h1>Bem-vindo ao Meu Projeto HTML</h1>

<p>Este é um parágrafo de exemplo.</p>



<p>Fonte: Elaborado pelo autor.</p>

</body>

</html>
```

Veja o resultado no navegador, conforme a figura abaixo:

Bem-vindo ao Meu Projeto HTML

Este é um parágrafo de exemplo.



Figura 2 - Demonstra o resultado do uso da tag no navegador.

Descrição: Esse código resulta em uma página web com o título "Bem-vindo ao Meu Projeto HTML", seguido por um parágrafo de exemplo. Além disso, há uma imagem do logotipo do IFPR, representado pelo arquivo imagem.png.

Cabe salientar que a tag em HTML possui vários atributos que permitem especificar diferentes propriedades da imagem que está sendo exibida.

ATRIBUTO SRC

O atributo **src** na tag é um atributo obrigatório e representa a origem (source) da imagem que será exibida na página da web. Ele especifica o URL (Uniform Resource Locator) ou o caminho do arquivo da imagem que o navegador deve carregar e exibir. Veja um exemplo da sua sintaxe no quadro abaixo:

```

```

Definir corretamente o caminho de uma imagem em HTML é essencial para garantir que ela seja exibida corretamente no navegador. Existem duas formas principais de especificar esses caminhos: caminhos relativos e caminhos absolutos.

Caminhos Relativos

Um caminho relativo especifica a localização da imagem em relação ao arquivo HTML atual. É amplamente utilizado porque facilita a movimentação do site entre diferentes ambientes (como desenvolvimento e produção) sem a necessidade de alterar os caminhos das imagens. Observe abaixo alguns exemplos de uso de caminhos relativos:

Mesmo diretório: quando a imagem está no mesmo diretório que o arquivo HTML, conforme quadro abaixo.

```

```

Subdiretório: quando a imagem está em um subdiretório do diretório do arquivo HTML, conforme quadro abaixo.

```

```

Diretório Superior: quando a imagem está em um diretório acima do diretório do arquivo HTML, conforme quadro abaixo.

```

```

OBS: Observe que cada .. sobe um nível na hierarquia de diretórios.

Combinação de níveis: combina subdiretórios e diretórios superiores, conforme quadro abaixo.

```

```

Caminhos absolutos

Um caminho absoluto fornece a localização completa da imagem, incluindo o protocolo (http ou https), o domínio e o caminho completo do diretório. É útil para imagens hospedadas em outros servidores ou para recursos compartilhados entre diferentes sites. Observe abaixo alguns exemplos de uso de caminhos absolutos:

URL completa: especifica o URL completo, incluindo protocolo e domínio, conforme quadro abaixo.

```

```

Caminho completo no servidor: indica o caminho completo desde a raiz do servidor, conforme quadro abaixo.

```

```

Escolher entre caminhos relativos e absolutos depende das necessidades do projeto e da localização dos recursos. Entender as diferenças e aplicá-las corretamente é essencial para a eficiência e a organização de qualquer projeto web.

ATRIBUTO ALT

O atributo **alt** na tag é usado para fornecer um texto alternativo para a imagem. Ele é essencial para acessibilidade na web, pois é lido por leitores de tela para usuários com deficiência visual ou quando a imagem não pode ser carregada.

Veja um exemplo de sua sintaxe no quadro abaixo:

```

```

O texto alternativo deve descrever o conteúdo e o propósito da imagem. Isso pode incluir uma breve descrição do que está na imagem ou fornecer informações contextuais sobre a função da imagem na página.

É uma boa prática sempre incluir o atributo **alt** em todas as tags ``, mesmo que a imagem seja puramente decorativa. Para imagens puramente decorativas, você pode usar um texto alternativo vazio (`alt=""`) para indicar aos leitores de tela que a imagem não tem significado semântico, conforme quadro abaixo.

```

```

O texto alternativo é importante para garantir que todos os usuários possam compreender o conteúdo da página, independentemente de suas capacidades visuais ou das condições de carregamento da imagem.

ATRIBUTO WIDTH

O atributo **width** na tag `` é usado para especificar a largura da imagem em pixels. Ele permite controlar o tamanho da imagem na página da web. Veja um exemplo de sua sintaxe no quadro abaixo:

```

```

Neste exemplo, "300" é o valor da largura da imagem em pixels.

Ao definir a largura da imagem usando o atributo **width**, você pode controlar o tamanho da imagem na página sem afetar sua proporção de aspecto. No entanto, é importante usar esse atributo com cautela para garantir que a imagem não seja distorcida ou pixelizada.

OBS: Lembre-se de que definir explicitamente a largura da imagem pode não ser a melhor prática em todos os casos, especialmente para imagens responsivas. Em vez disso, você pode preferir usar CSS ou técnicas de dimensionamento responsivo para controlar o tamanho da imagem de forma mais flexível, adaptando-a ao tamanho da tela do dispositivo ou ao layout da página.

ATRIBUTO HEIGHT

O atributo **height** na tag `` é usado para especificar a altura da imagem em pixels. Ele permite controlar o tamanho vertical da imagem na página da web. Veja um exemplo de sua sintaxe no quadro abaixo:

```

```

Neste exemplo, "200" é o valor da altura da imagem em pixels.

Ao definir a altura da imagem usando o atributo **height**, você pode controlar o tamanho vertical da imagem na página sem afetar sua proporção de aspecto. No entanto, é importante usar esse atributo com cautela para garantir que a imagem não seja distorcida ou pixelizada.

OBS: Assim como com o atributo **width**, é importante lembrar que definir explicitamente a altura da imagem pode não ser a melhor prática em todos os casos, especialmente para imagens responsivas. Em vez disso, você pode preferir usar CSS ou técnicas de dimensionamento responsivo para controlar o tamanho da imagem de forma mais flexível, adaptando-a ao tamanho da tela do dispositivo ou ao layout da página.

IMPORTANTE: Se você informar apenas o valor do atributo **height** ou apenas o valor do atributo **width**, a outra dimensão será ajustada automaticamente para manter a proporção original da imagem.

ATRIBUTO TITLE

O atributo **title** na tag `` é opcional e é usado para fornecer uma dica de ferramenta (tooltip) quando o usuário passa o mouse sobre a imagem. Essa dica de ferramenta pode fornecer informações adicionais sobre a imagem. Veja um exemplo de sua sintaxe no quadro abaixo:

```

```

Veja o resultado no navegador, conforme figura abaixo:

Bem-vindo ao Meu Projeto HTML

Este é um parágrafo de exemplo.



Figura 3 - Demonstra o resultado do uso do atributo title da tag no navegador.

Descrição da imagem: No navegador, o atributo title na tag exibe um texto quando o cursor passa sobre a imagem, fornecendo informações adicionais como uma dica. Neste caso, exibe o texto "Título da Imagem" quando o usuário passa o mouse sobre a imagem

ATRIBUTO STYLE

O atributo **style** na tag é usado para aplicar estilos CSS diretamente à imagem. Isso permite controlar vários aspectos visuais da imagem, como tamanho, cor, margens, entre outros. Veja um exemplo de sua sintaxe no quadro abaixo:

```

```

Veja o resultado no navegador, conforme figura abaixo:

Bem-vindo ao Meu Projeto HTML

Este é um parágrafo de exemplo.



Figura 4 - Demonstra o resultado do uso do atributo style na tag no navegador.

Descrição da imagem: No navegador, o uso do atributo style na tag permite aplicar estilos diretamente à imagem. Neste exemplo, a imagem tem uma largura de 300 pixels, altura de 200 pixels e uma borda preta de 1 pixel ao redor.

O atributo **style** permite especificar um conjunto de regras de estilo diretamente no HTML. No entanto, o uso excessivo de estilos inline pode tornar o HTML menos legível e mais difícil de manter. Portanto, em situações mais complexas, é preferível usar classes ou IDs CSS externos para estilizar elementos HTML.

3.1.2 Especificando múltiplas fontes da imagem

A tag <picture> em HTML foi projetada para oferecer maior controle sobre como as imagens são exibidas em diferentes dispositivos e resoluções de tela. Ela permite ao desenvolvedor especificar múltiplas fontes de imagem e instruir o navegador a escolher a imagem mais apropriada com base em critérios como dimensões da viewport, resolução da tela, e outros fatores. Isso é especialmente útil para implementar designs responsivos e para otimizar o carregamento de imagens, melhorando tanto a performance quanto a experiência do usuário em dispositivos variados.

OBS: A **viewport** é a área visível da página web no navegador de um usuário. Em outras palavras, é a parte da página que está atualmente visível sem a necessidade de rolar.

A tag <picture> funciona como um contêiner para um ou mais elementos <source> e um elemento de fallback. Cada <source> pode ter atributos que especificam a mídia ou as condições sob as quais essa fonte de imagem deve ser usada. Se nenhum <source> corresponder às condições atuais, o navegador usará a imagem especificada no elemento .

OBS: O **fallback** para um elemento <picture> é uma imagem de substituição que será exibida caso nenhum dos arquivos de imagem especificados nas tags <source> seja suportado pelo navegador do usuário.

Vejamos alguns dos seus principais atributos:

srcset: Especifica o caminho da imagem. Pode incluir uma lista de imagens com diferentes tamanhos e a largura da viewport para cada uma.

media: Define condições de mídia CSS, como a largura da viewport, para que o navegador escolha a imagem baseada nessas condições.

type: Permite especificar o tipo MIME da imagem, o que pode ser usado para fornecer formatos de imagem diferentes (como WebP) para navegadores que suportam esses formatos.

Veja abaixo um exemplo de uso da tag <picture>:

```
<picture>

<source media="(min-width: 800px)" srcset="image-large.jpg">

<source media="(min-width: 450px)" srcset="image-medium.jpg">



</picture>
```

Neste exemplo, **image-large.jpg** é exibido para viewports com largura mínima de 800 pixels, **image-medium.jpg** para viewports com largura mínima de 450 pixels, e **image-small.jpg** é usado como fallback para viewports menores ou quando os formatos fornecidos nos elementos <source> não são suportados pelo navegador.

Trabalhando com links em documentos HTML

Os "links", ou hiperlinks, são elementos interativos em uma página da web que permitem aos usuários navegar entre diferentes recursos da web, como outras páginas da web, arquivos, documentos, imagens, vídeos, endereços de e-mail e muito mais. Eles são fundamentais para a experiência de navegação na web, pois conectam diferentes partes da internet, permitindo que os usuários explorem e acessem informações relevantes. Os links são criados usando a tag <a> em HTML (abreviação de âncora), que veremos a seguir neste curso.

A tag <a> (abreviação de "anchor", que significa âncora) é usada em HTML para criar links entre diferentes recursos da web, como outras páginas da web, imagens, arquivos de áudio, vídeos, endereços de e-mail etc.

No quadro abaixo está a estrutura básica da tag <a>:

```
<a href="URL_do_destino">Texto do Link</a>
```

Onde:

href: O atributo obrigatório que especifica o destino do link. Pode ser uma URL, um endereço de e-mail ou um identificador dentro do documento.

Texto do Link: O texto ou conteúdo que será exibido como o link. Quando o usuário clica nesse texto, ele é direcionado para o destino especificado no atributo **href**.

Para realizarmos os testes sobre o uso da tag <a>, sugerimos que você utilize o arquivo **index.html** do nosso projeto de testes. Neste contexto, altere esse arquivo para que contenha o seguinte código no corpo do documento (na tag <body>), conforme quadro abaixo:

```
<a href="https://ifpr.edu.br/">Visite o site do IFPR</a>
```

Este link direcionará os usuários para o site <https://ifpr.edu.br/> quando clicado e exibirá o texto "Visite o site Exemplo" como o link.

Veja o resultado no navegador, conforme figura abaixo:

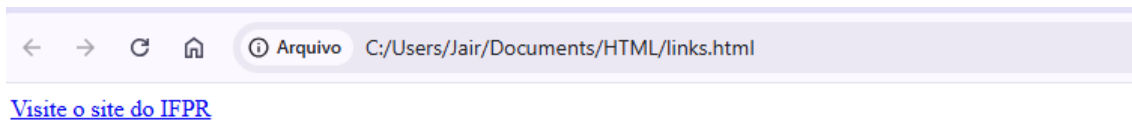


Figura 1 - Demonstra o resultado do uso da tag <a> no navegador.

Descrição da imagem: No navegador, a tag <a> cria um link que pode ser clicado para navegar para outra página ou seção. Neste caso, quando você clica no texto "Visite o site do IFPR" dentro da tag <a>, você é redirecionado para o destino especificado pelo atributo href da tag <a>.

Ao criar links em HTML, é essencial entender a diferença entre caminhos relativos e absolutos para garantir a navegação adequada entre páginas e recursos. Ambos os tipos de caminho têm seus usos específicos e vantagens, dependendo do contexto do projeto.

CAMINHOS RELATIVOS

Um caminho relativo define a localização do destino do link em relação ao documento HTML atual. É especialmente útil para sites com uma estrutura de diretórios bem definida e para desenvolvimento local. Observe abaixo alguns exemplos de caminhos relativos para links:

Mesmo diretório: o arquivo de destino está no mesmo diretório que o HTML atual, conforme quadro abaixo.

```
<a href="pagina.html">Ir para Página</a>
```

Subdiretório: o arquivo de destino está em um subdiretório do diretório do HTML atual, conforme quadro abaixo.

```
<a href="subdiretorio/pagina.html">Ir para Página no Subdiretório</a>
```

Diretório superior: o arquivo de destino está em um diretório acima do diretório do HTML atual, conforme quadro abaixo.

```
<a href="../subdiretorio/pagina.html">Ir para Página em Múltiplos Níveis</a>
```

OBS: Observe que cada .. sobe um nível na hierarquia de diretórios.

Múltiplos níveis: combina subdiretórios e diretórios superiores, conforme quadro abaixo.

```
<a href="../subdiretorio/pagina.html">Ir para Página em Múltiplos Níveis</a>
```

CAMINHOS ABSOLUTOS

Um caminho absoluto define a localização completa do destino do link, incluindo o protocolo (http ou https), o domínio e o caminho completo do diretório. É ideal para links para recursos externos ou compartilhados. Veja abaixo alguns exemplos de caminho absoluto para links:

URL completa: inclui o protocolo, domínio e o caminho completo, conforme quadro abaixo.

```
<a href="https://www.exemplo.com/pagina.html">Ir para Página Externa</a>
```

Caminho completo no servidor: indica o caminho completo desde a raiz do servidor, conforme quadro abaixo.

```
<a href="/subdiretorio/pagina.html">Ir para Página no Caminho Completo</a>
```

Entender a diferença entre caminhos relativos e absolutos para links em HTML é essencial para criar uma navegação eficaz e manter a integridade do site. Caminhos relativos são ideais para links internos e desenvolvimento local, enquanto caminhos absolutos são melhores para links externos e recursos compartilhados. A escolha correta garante uma experiência de usuário fluida e uma manutenção simplificada do site.

ATRIBUTO HREF

O atributo **href** na tag <a> é usado para especificar o destino do link. Ele define para onde o usuário será redirecionado quando o link for clicado. O valor do atributo **href** pode ser uma URL, um endereço de e-mail ou um identificador dentro do próprio documento HTML.

Abaixo estão alguns exemplos de como usar o atributo **href**:

Para criar um link para uma página da web externa, conforme quadro abaixo.

```
<a href="https://www.exemplo.com">Visite o site Exemplo</a>
```

Neste exemplo, o link "Visite o site Exemplo" leva os usuários para o site <https://www.exemplo.com>.

Para criar um link para uma página dentro do mesmo site, conforme quadro abaixo.

```
<a href="pagina_interna.html">Página Interna</a>
```

Neste caso, suponha que **pagina_interna.html** seja uma página dentro do mesmo diretório do arquivo HTML atual (index.html). Quando o link "Página Interna" é clicado, os usuários são redirecionados para a página **pagina_interna.html**.

Para criar um link interno (também conhecido como âncora) em uma mesma página. Neste exemplo, altere a página **index.html** com a estrutura definida no quadro abaixo:

```
<!DOCTYPE html>

<html lang="pt-br">

<head>

<meta charset="UTF-8">

<meta name="viewport" content="width=device-width, initial-scale=1.0">

<title>Exemplo de Link Interno</title>

</head>

<body>

<h1>Exemplo de Link Interno</h1>

<p>Este é um exemplo de link interno em HTML. Clique no link abaixo para rolar até a seção correspondente.</p>

<p><a href="#secao2">Ir para a Seção 2</a></p>

<p>Lorem ipsum dolor sit amet. Sit nulla ducimus ut voluptatem dolor a beatae deserunt vel perferendis exercitationem. Aut doloribus consequuntur et obcaecati voluptas sit molestiae obcaecati ad dicta fugit.</p>

<p>Est quos quibusdam quo excepturi dolores et corporis autem aut sunt sunt et accusamus nemo qui accusantium minus. Sit dolorem galisum non quia eligendi rem veniam omnis cum dolores omnis vel voluptatem facilis et autem reprehenderit ea quibusdam autem. Et numquam nihil eum tenetur facere aut blanditiis voluptatem id maiores deleniti ut quidem perferendis sit tempora amet et excepturi odio.</p>

<p>Est omnis esse eos eveniet omnis qui fugiat porro sit eveniet velit est deleniti quod qui iste eaque qui enim iste? Ab itaque architecto sed temporibus molestias At harum inventore. Et galisum perferendis id recusandae voluptatem est maiores quibusdam aut debitis odio id Quis quisquam. Vel aspernatur expedita vel quos nesciunt ea ratione recusandae eum mollitia unde eos iste voluptas?</p>

<h2 id="secao2">Seção 2</h2>

<p>Esta é a Seção 2 do documento.</p>
```

```
</body>

</html>
```

Neste exemplo:

O texto "**Ir para a Seção 2**" é um link interno que leva o usuário para a seção correspondente do documento.

O href do link interno é **#secao2**, que corresponde ao identificador id="secao2" na tag <h2>.

Quando o link é clicado, o navegador rola automaticamente até a seção que possui o identificador secao2.

Isso é útil para criar navegação dentro de uma única página da web, permitindo que os usuários acessem facilmente diferentes partes do conteúdo. Certifique-se de que o valor do atributo href corresponda ao identificador correto dentro do documento HTML.

Para criar um link para um endereço de e-mail, conforme quadro abaixo.

```
<a href="mailto:email@example.com">Enviar e-mail</a>
```

Neste exemplo, ao clicar no link "Enviar e-mail", o cliente de e-mail padrão do usuário será aberto com o endereço de e-mail **email@example.com** preenchido no campo "Para". Veja esse exemplo em execução na figura abaixo:

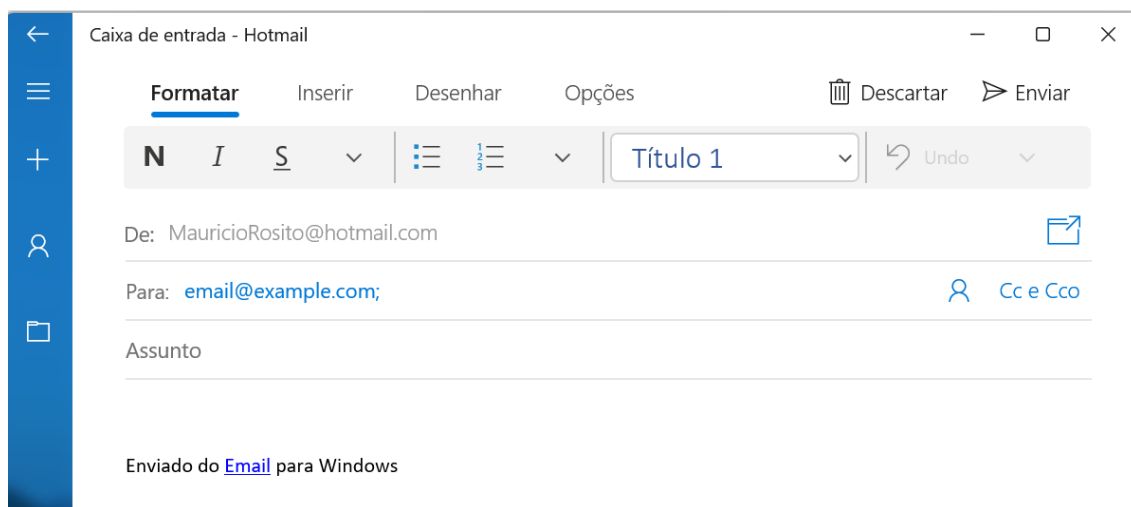


Figura 2 - Exemplo de uso de um link para um endereço de e-mail, que abre o software padrão de e-mail do computador para compor uma nova mensagem.

Descrição da imagem: Um link para um endereço de e-mail, quando clicado, abre o programa de e-mail padrão do computador e inicia uma nova mensagem com o endereço de e-mail do remetente já preenchido.

Esses são apenas alguns exemplos do uso do atributo href na tag <a>. Ele é fundamental para criar links navegáveis em páginas da web.

ATRIBUTO TARGET

Além do atributo **href**, a tag <a> também pode ter outros atributos, tais como o **target**: Este atributo define onde abrir o destino do link. Os valores comuns são estes a seguir:

_self: link é aberto no mesmo quadro ou janela em que o link foi clicado (este é o comportamento padrão se o atributo target não for especificado). Veja um exemplo de uso no quadro abaixo.

```
<a href="https://www.example.com" target="_self">Abrir na mesma janela</a>
```

_blank: O link é aberto em uma nova janela ou aba do navegador. Veja um exemplo de uso no quadro abaixo.

```
<a href="https://www.example.com" target="_blank">Abrir em nova janela/  
aba</a>
```

_parent: O link é aberto no quadro (frame) pai do quadro em que o link foi clicado, se houver um. Veja um exemplo de uso no quadro abaixo.

```
<a href="https://www.example.com" target="_parent">Abrir no quadro pai</a>
```

_top: O link é aberto no topo da janela atual, ignorando qualquer quadro aninhado. Veja um exemplo de uso no quadro abaixo.

```
<a href="https://www.example.com" target="_top">Abrir no topo da janela</a>
```

name: Você também pode especificar o nome de uma janela ou quadro HTML específico onde o link será aberto. Se a janela especificada não existir, uma nova janela será aberta. Veja no quadro abaixo um exemplo de sua sintaxe:

```
<a href="https://www.example.com" target="minhaJanela">Abrir em uma janela chamada minhaJanela</a>
```

Neste contexto, no quadro abaixo temos um exemplo de seu uso:

```
<a href="https://www.example.com" target="_blank">Abrir em Nova Aba</a>
```

O link “<https://www.example.com>” é aberto em uma nova janela ou aba do navegador.

OBS: O uso do atributo **target** permite um controle mais preciso sobre o comportamento de abertura de links em uma página da web.

ATRIBUTO DOWNLOAD

O atributo **download** instrui o navegador a fazer o download do recurso especificado quando o link é clicado, em vez de navegá-lo. Veja no quadro abaixo um exemplo:

```
<a href="documento.pdf" download>Download do Documento</a>
```

Neste exemplo, quando o usuário clica no link "Download do Documento", o arquivo **documento.pdf** será baixado em vez de ser aberto no navegador.

É importante observar que este atributo só funciona para links que apontam para recursos que podem ser baixados, como arquivos PDF, imagens, vídeos, arquivos de áudio, etc. Se o recurso estiver em uma origem diferente do site onde o link está sendo hospedado, pode haver restrições de segurança relacionadas à política de mesma origem (CORS - Cross-Origin Resource Sharing).

ATRIBUTO TITLE

O atributo **title** na tag <a> é usado para fornecer uma dica (tooltip) quando o usuário passa o cursor sobre o link. Essa dica de ferramenta é útil para fornecer informações adicionais sobre o destino do link, como uma breve descrição do conteúdo vinculado ou instruções adicionais.

Veja no quadro abaixo um exemplo de como usar o atributo **title**:

```
<a href="https://www.example.com" title="Visite o site exemplo">Link Externo</a>
```

Neste exemplo, quando o usuário passa o cursor sobre o link "Link Externo", uma dica de ferramenta será exibida com o texto "Visite o site exemplo", conforme figura abaixo.

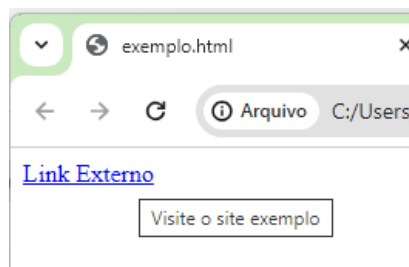


Figura 3 - Demonstra o resultado do uso do atributo title da tag <a> no navegador.

Descrição da imagem: Nesta figura, quando o usuário passa o cursor sobre o link "Link Externo", uma dica de ferramenta será exibida com o texto.

O texto fornecido no atributo **title** pode ser personalizado conforme necessário para fornecer informações relevantes para os usuários. É importante notar que as dicas de ferramentas podem variar de tamanho dependendo do navegador e da configuração do usuário, portanto, é melhor manter o texto conciso e informativo.

Usando links em uma imagem

Para criar um link em uma imagem em HTML, você precisa envolver as tags <a> ou . No quadro abaixo está um exemplo.

```
<a href="pagina_destino.html">  
  
</a>
```

Neste exemplo:

A tag **<a>** envolve a tag ****, criando um link em torno da imagem.

O atributo **href** na tag **<a>** especifica a URL para a qual o link deve redirecionar quando a imagem for clicada. A tag **** representa a imagem que será exibida e funciona como o conteúdo do link.

Assim, quando um usuário clicar na imagem, ele será redirecionado para a URL especificada no atributo **href** da tag **<a>**.

Trabalhando com listas em documentos HTML

As listas em HTML são estruturas que permitem agrupar e organizar conteúdo de forma ordenada ou não ordenada. Elas são amplamente usadas para criar menus de navegação, listas de itens, definições, entre outros. Existem três tipos principais de listas em HTML:

- listas não ordenadas;
- listas ordenadas; e
- listas de definições.

As listas ordenadas utilizam sequências ordinais para indicar a ordem dos elementos da lista. As listas não ordenadas usam um símbolo definido, como um marcador, para listar elementos sem ordem designada. As listas de descrição usam recuos para listar elementos pais com seus filhos.

Veremos mais sobre como podemos trabalhar com listas em HTML no decorrer deste curso.

Listas ordenadas

Para criar uma lista ordenada em HTML, você pode usar a tag `` (ordered list) e aninhar cada item da lista dentro da tag `` (list item).

O quadro abaixo apresenta um exemplo simples de lista ordenada:

```
<ol>

<li>Primeiro item</li>

<li>Segundo item</li>

<li>Terceiro item</li>

</ol>
```

Neste exemplo, criamos uma lista ordenada com três itens numerados de forma sequencial. Você pode adicionar quantos itens quiser seguindo o mesmo padrão. O navegador renderizará a lista com os números automaticamente, conforme figura abaixo.

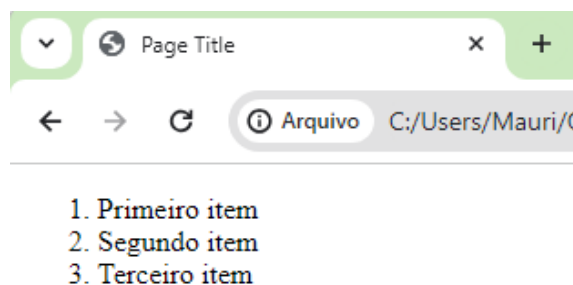


Figura 1 - Demonstra o resultado do uso da tag `` no navegador.

Descrição da imagem: Esse código cria uma lista ordenada com três itens. Cada item é marcado com um número, indicando a ordem sequencial na lista. O conteúdo dos itens é "Primeiro item", "Segundo item" e "Terceiro item", respectivamente.

A tag `` pode aceitar vários atributos para personalizar sua aparência e comportamento. Veremos abaixo mais informações sobre estes atributos.

ATRIBUTO START

O atributo **start** da tag (ordered list) é usado para especificar o valor inicial da contagem numérica de uma lista ordenada. Isso pode ser útil quando você deseja iniciar a contagem em um número diferente de 1.

No quadro abaixo está um exemplo de como usar o atributo **start**:

```
<ol start="5">  
  
<li>Item 5</li>  
  
<li>Item 6</li>  
  
<li>Item 7</li>  
  
</ol>
```

Neste exemplo, a lista ordenada inicia a contagem a partir do número 5, como especificado pelo atributo **start**. Assim, o primeiro item da lista será "Item 5", o segundo será "Item 6" e assim por diante. O navegador renderizará a lista de acordo com os valores fornecidos, conforme figura abaixo.

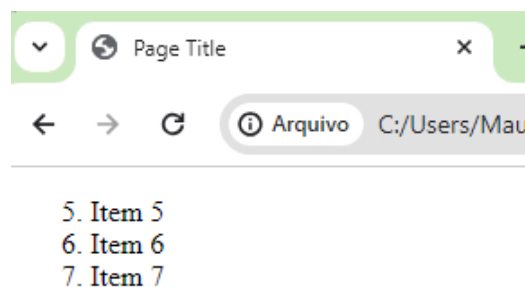


Figura 2 - Demonstra o resultado do uso do atributo start da tag no navegador.

Descrição da imagem: No navegador, esse código cria uma lista ordenada com três itens. O primeiro item é marcado como "Item 5", o segundo como "Item 6" e o terceiro como "Item 7". Isso acontece porque o atributo "start" define o valor inicial da contagem na lista ordenada, neste caso, começando em 5.

ATRIBUTO TYPE

O atributo **type** da tag é usado para especificar o tipo de marcador usado para os itens da lista ordenada.

Os valores possíveis para esse atributo são:

type="1": Usa números (1, 2, 3, ...) como marcadores. Este é o valor padrão se nenhum for especificado.

type="A": Usa letras maiúsculas (A, B, C, ...).

type="a": Usa letras minúsculas (a, b, c, ...).

type="I": Usa algarismos romanos maiúsculos (I, II, III, ...).

type="i": Usa algarismos romanos minúsculos (i, ii, iii, ...).

No quadro abaixo está um exemplo de como usar o atributo **type**:

```
<ol type="A">  
  
<li>Item A</li>  
  
<li>Item B</li>  
  
<li>Item C</li>  
  
</ol>
```

Neste exemplo, a lista ordenada usa letras maiúsculas como marcadores. Veja o resultado no navegador, conforme figura abaixo.

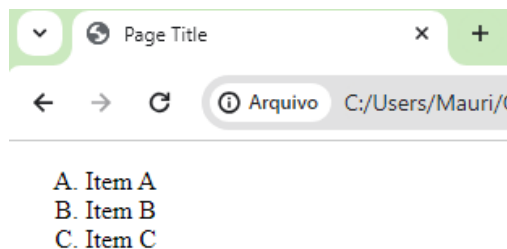


Figura 3 - Demonstra o resultado do uso do atributo type da tag no navegador.

Descrição da imagem: No navegador, esse código cria uma lista ordenada com três itens, onde cada item é marcado com letras maiúsculas do alfabeto. O primeiro item é marcado como "A", o segundo como "B" e o terceiro como "C". Isso acontece porque o atributo "type" define o tipo de marcação utilizada na lista ordenada.

ATRIBUTO REVERSED

O atributo **reversed** da tag é usado para indicar que a lista ordenada deve ser exibida em ordem decrescente, em vez da ordem padrão crescente. Esse atributo não requer um valor, basta incluí-lo para indicar que a lista deve ser revertida.

No quadro abaixo está um exemplo de como usar o atributo **reversed**:

```
<ol reversed>
```

```
<li>Item 3</li>
```

```
<li>Item 2</li>
```

```
<li>Item 1</li>
```

```
</ol>
```

Neste exemplo, a lista ordenada será exibida em ordem decrescente, começando pelo último item especificado até o primeiro. O marcador será decrementado em vez de incrementado conforme os itens são listados.

Veja na figura abaixo o resultado de sua execução no navegador.

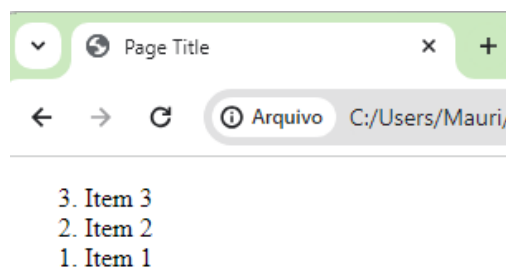


Figura 4 - Demonstra o resultado do uso do atributo reversed da tag no navegador.

Descrição da imagem: No navegador, esse código cria uma lista ordenada com três itens, onde cada item é numerado de forma decrescente. O último item é marcado como "1", o segundo como "2" e o primeiro como "3". Isso acontece porque o atributo "reversed" inverte a contagem normal dos itens na lista ordenada.

Listas não ordenadas

Para criar uma lista não ordenada em HTML5, você utiliza a tag `` (unordered list) e aninha cada item da lista dentro da tag `` (list item).

O quadro abaixo apresenta um exemplo simples de uso desta tag:

```
<ul>

<li>Item 1</li>

<li>Item 2</li>

<li>Item 3</li>

</ul>
```

Veja o resultado no navegador, conforme figura abaixo.

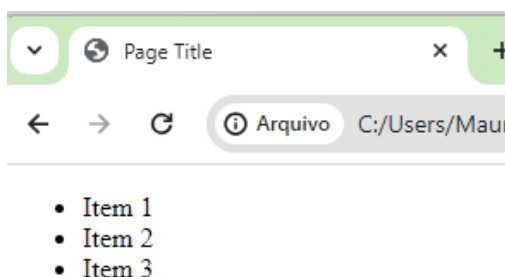


Figura 1 - Demonstra o resultado do uso da tag `` no navegador.

Descrição da imagem: No navegador, esse código cria uma lista não ordenada com três itens. Cada item é representado por um marcador padrão, como um ponto ou um círculo, dependendo do navegador. Os itens são "Item 1", "Item 2" e "Item 3".

Você pode adicionar quantos itens quiser seguindo o mesmo padrão. O navegador renderizará a lista com os marcadores automaticamente.

Cabe observar que esta tag aceita apenas um atributo padrão, o **type**. Veremos mais sobre este atributo a seguir.

ATRIBUTO TYPE

O atributo **type** especifica o tipo de marcador que será usado para os itens da lista. No entanto, ele é pouco utilizado, uma vez que os navegadores geralmente definem automaticamente o tipo de marcador para listas não ordenadas.

Os valores possíveis para o atributo type são:

- "disc" (círculo preenchido) - é o padrão
- "circle" (círculo não preenchido)
- "square" (quadrado)

Veja no quadro abaixo um exemplo de como usar o atributo **type**:

```
<ul type="circle">  
  
<li>Item 1</li>  
  
<li>Item 2</li>  
  
<li>Item 3</li>  
  
</ul>
```

Neste exemplo, os itens da lista serão marcados com círculos. Veja na figura abaixo o resultado no navegador.

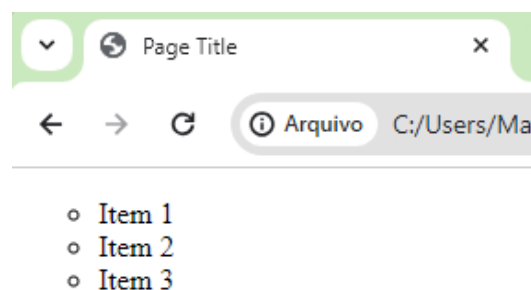


Figura 2 - Demonstra o resultado do uso do atributo type da tag no navegador.

Descrição da imagem: No navegador, esse código cria uma lista não ordenada com três itens. Cada item é representado por um marcador em forma de círculo. Os itens são "Item 1", "Item 2" e "Item 3".

OBS: É importante notar que o uso do atributo **type** na tag não é tão comum, pois os navegadores geralmente definem automaticamente o estilo dos marcadores. O atributo **type** pode ser mais útil em elementos (ordered list), onde você pode especificar o tipo de numeração.

Listas de descrição

Em HTML, você pode criar listas de descrição usando as tags <dl> (description list), <dt> (description term) e <dd> (description details).

No quadro abaixo está um exemplo de como criar uma lista de descrição:

```
<dl>
<dt>Título 1</dt>
<dd>Descrição detalhada do título 1.</dd>
<dt>Título 2</dt>
<dd>Descrição detalhada do título 2.</dd>
<dt>Título 3</dt>
<dd>Descrição detalhada do título 3.</dd>
</dl>
```

Neste exemplo, a tag <dl> é usada para envolver toda a lista de descrição. Cada par de <dt> e <dd> representa um item na lista. O <dt> é usado para o termo ou título, enquanto <dd> é usado para a descrição detalhada correspondente. O navegador renderizará isso como uma lista onde cada título é seguido pela sua descrição, conforme figura abaixo.

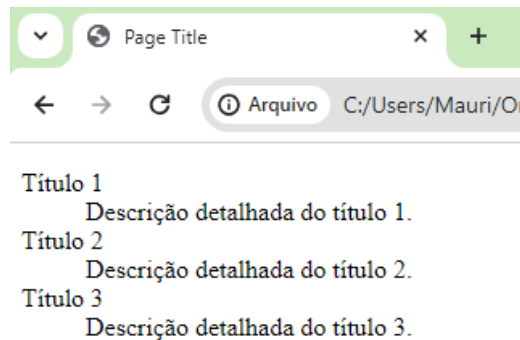


Figura 1 - Demonstra o resultado do uso da tag <dl> no navegador.

Descrição da imagem: No navegador, esse código cria uma lista de definição, onde cada termo (representado por <dt>) é seguido por sua descrição correspondente (representada por <dd>). Cada título é seguido por uma descrição detalhada. Neste caso, "Título 1" tem uma descrição "Descrição detalhada do título 1", e assim por diante para os outros itens da lista.

Trabalhando com tabelas em documentos HTML

As tabelas em HTML5 são usadas para organizar e exibir dados em formato de grade, facilitando a apresentação de informações de maneira clara e estruturada. Originalmente, as tabelas eram frequentemente utilizadas para layout de página na web, mas essa prática é agora desaconselhada em favor de CSS para design de layout.

No HTML5, neste contexto, as tabelas são recomendadas principalmente para exibir dados relacionados, como:

Dados Tabulares: Qualquer informação que se encaixe naturalmente em uma matriz, como horários, listas de preços, especificações de produtos, resultados de pesquisas, comparações, e muito mais.

Relatórios e Análises: Apresentação de dados de pesquisa, resultados de análises financeiras, estatísticas, e outros tipos de relatórios que requerem comparação de números ou textos lado a lado.

Agendas e Horários: Programações de eventos, horários de aulas, planos de projeto, ou qualquer outro tipo de informação que se beneficie de uma representação em linhas e colunas.

Listas e Inventários: Tabelas podem ser usadas para listar itens de forma organizada, mostrando propriedades ou características relevantes lado a lado para facilitar a consulta.

Veremos no decorrer deste curso como podemos criar tabelas em HTML5.

Elementos fundamentais de uma tabela

Para criar uma tabela em HTML, é essencial conhecer os elementos-chave que constituem a estrutura de uma tabela.

Tag <table>

Criar uma tabela no HTML5 envolve o uso da tag **<table>** e algumas tags relacionadas para definir linhas, células e, opcionalmente, cabeçalhos de colunas. Logo, podemos dizer que a tag **<table>** é o container que define a tabela. Todos os outros elementos de tabela são inseridos dentro deste elemento. Esta tag tem um conjunto de atributos que ajudam na sua formatação, os quais veremos em breve neste documento.

Tag <tr>

Dentro da tag **<table>**, use a tag **<tr>** para iniciar e terminar cada linha da tabela. Pode conter células de cabeçalho (**<th>**) da tabela ou células de dados (**<td>**).

Tag <th>

Dentro de cada tag **<tr>**, você pode usar a tag **<th>** para definir células de cabeçalho (Table Header) da tabela, que são usadas para definir títulos para colunas ou linhas da tabela. As células **<th>** são, por padrão, formatadas em negrito e centralizadas.

Tag <td>

Dentro de cada tag **<tr>**, você também pode usar a tag **<td>** para definir células de dados (Data Cell) da tabela, ou seja, será o lugar você coloca o conteúdo das informações da tabela.

Veamos no quadro abaixo um exemplo básico de como você pode criar uma tabela em HTML5:

NOME	IDADE	PROFISSÃO
MARIA	28	Designer

NOME	IDADE	PROFISSÃO
JOÃO	32	Desenvolvedor

Onde:

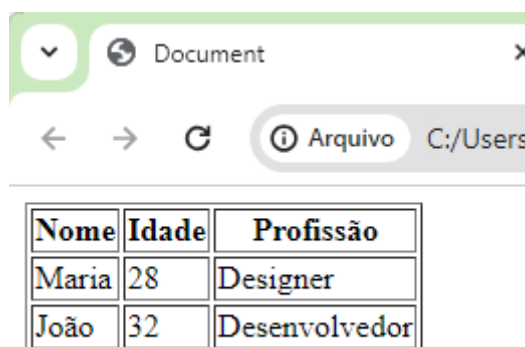
<table border="1"> Cria a tabela e, opcionalmente, define a largura da borda da tabela como 1. No HTML5, o atributo **border** é considerado obsoleto, e é recomendado usar CSS para estilizar as bordas, mas ainda é comum ver o uso para exemplos simples para aprendizado.

<tr> Define uma linha na tabela.

<th> Define uma célula de cabeçalho na linha. Células **<th>** são usadas para cabeçalhos de coluna e são, por padrão, negrito e centralizado.

<td> Define uma célula padrão (data cell) na linha, usada para dados da tabela.

Veja o resultado no navegador, conforme figura abaixo.



Nome	Idade	Profissão
Maria	28	Designer
João	32	Desenvolvedor

Figura 1 - Exemplo de uma tabela HTML exibida no navegador.

Descrição da imagem: No navegador, este código cria uma tabela com bordas visíveis. A tabela tem três colunas: "Nome", "Idade" e "Profissão". Na primeira linha de dados, os valores são "Maria", 28, e Designer. Na segunda linha de dados, os valores são "João", 32, e Desenvolvedor.

Melhorando a semântica e acessibilidade das tabelas

Melhorar a semântica e a acessibilidade de uma tabela HTML é essencial para garantir que todos os usuários, incluindo aqueles que usam tecnologias assistivas como leitores de tela, possam entender e navegar pelos dados apresentados.

Neste documento estão algumas sugestões para você criar tabelas mais acessíveis e semanticamente corretas.

Use <th> para cabeçalhos de coluna e linha

Utilize a tag **<th>** (table header) em vez de **<td>** (table data) para marcar cabeçalhos de colunas e linhas. Isso ajuda os leitores de tela a identificar a função das células e a fornecer um contexto adequado aos usuários. Se necessário, para maior clareza, você pode usar na tag **<th>** os atributos **scope="col"** para cabeçalhos de colunas e **scope="row"** para cabeçalhos de linhas.

OBS: Leitores de tela são softwares assistivos que permitem que pessoas com deficiência visual ou cegas acessem e interajam com conteúdo em computadores, smartphones e outros dispositivos eletrônicos. Eles funcionam convertendo texto e elementos gráficos em fala ou em braille (através de uma linha braille), permitindo que o usuário ouça ou leia o conteúdo da tela.

Veja um exemplo no quadro abaixo:

NOME	IDADE
JOÃO	30
ANA	25

Neste exemplo, os cabeçalhos "Nome" e "Idade" são especificados como cabeçalhos de coluna (**scope="col"**), enquanto "João" e "Ana" são especificados como cabeçalhos de linha (**scope="row"**). Isso ajuda a indicar a função desses cabeçalhos na organização dos dados da tabela, tornando-a mais acessível para tecnologias assistivas.

Nome	Idade
João	30
Ana	25

Figura 1- Exemplo de uso da tag <th> exibido no navegador.

Descrição: No navegador, este código cria uma tabela com bordas visíveis. A primeira linha tem dois cabeçalhos de coluna: "Nome" e "Idade". As linhas seguintes têm cabeçalhos de linha: "João" com a idade "30" na segunda coluna, e "Ana" com a idade "25" na segunda coluna. Os cabeçalhos de linha estão em negrito e identificam cada linha.

Use <caption> para descrever a tabela

O elemento **<caption>** em HTML é usado para fornecer um título ou legenda para uma tabela. Ele é colocado imediatamente após a tag de abertura **<table>**, antes de quaisquer linhas (**<tr>**), cabeçalhos de coluna (**<th>**), etc. O propósito do **<caption>** é oferecer um contexto ou uma descrição geral da tabela, melhorando sua acessibilidade e compreensão.

Vejamos um exemplo no quadro abaixo:

Horário de Aulas

DIA	MATÉRIA	PROFESSOR(A)
SEGUNDA-FEIRA	Matemática	Prof. Carlos
TERÇA-FEIRA	Português	Profª. Maria

Neste exemplo, a **<caption>** "Horário de Aulas" fornece um entendimento imediato do propósito da tabela para o usuário, bem como para tecnologias assistivas, como leitores de tela.

Veja o resultado no navegador, conforme figura abaixo.

Horário de Aulas

Dia	Matéria	Professor(a)
Segunda-feira	Matemática	Prof. Carlos
Terça-feira	Português	Profª. Maria

Figura 2 - Exemplo de uso da tag <caption> exibido no navegador.

Descrição: No navegador, este código cria uma tabela com bordas visíveis e o título "Horário de Aulas" acima da tabela. A primeira linha contém os cabeçalhos "Dia", "Matéria" e "Professor(a)". A segunda linha tem os dados "Segunda-feira", "Matemática" e "Prof. Carlos". A terceira linha tem os dados "Terça-feira", "Português" e "Profª. Maria".

Agrupe cabeçalhos e corpo da tabela com <thead>, <tbody> e <tfoot>

Agrupar cabeçalhos, corpo e rodapé de uma tabela usando **<thead>**, **<tbody>** e **<tfoot>** no HTML é uma prática recomendada para criar tabelas estruturadas, semânticas e acessíveis. Esses elementos ajudam a definir explicitamente diferentes seções de uma tabela, melhorando a legibilidade do código e permitindo que navegadores e tecnologias assistivas, como leitores de tela, interpretem e apresentem a tabela de maneira mais eficaz.

Vejamos abaixo mais informações sobre estas tags:

- **<thead>**: é usada para agrupar um ou mais elementos **<tr>** que contêm células de cabeçalho da tabela. Os cabeçalhos de coluna localizados dentro de **<thead>** ajudam a descrever o tipo de informação contida nas colunas do **<tbody>**.
- **<tbody>**: é utilizada para agrupar o conteúdo principal da tabela, contendo uma ou mais linhas (**<tr>**) de células de dados (**<td>**). Uma tabela pode ter múltiplos elementos **<tbody>** para agrupar logicamente diferentes seções de dados dentro da mesma tabela.
- **<tfoot>**: contém um ou mais elementos **<tr>** que resumem ou concluem as informações apresentadas nas colunas do **<tbody>**. O **<tfoot>** deve ser colocado dentro do elemento **<table>** e, idealmente, antes do **<tbody>**, apesar de ser visualmente renderizado na parte inferior da tabela. Essa prática é recomendada porque permite que navegadores processem e exibam o rodapé antes mesmo de todo o corpo da tabela ser carregado, o que pode ser especialmente útil para tabelas muito grandes com muitas linhas de dados.

Venhamos um exemplo, conforme quadro abaixo:

Horário Semanal de Aulas

DIA DA SEMANA	MATÉRIA	HORÁRIO
SEGUNDA-FEIRA	Matemática	08:00 - 10:00
HORÁRIOS SUJEITOS A ALTERAÇÕES		

Vejamos o resultado no navegador, conforme figura abaixo.

Horário Semanal de Aulas		
Dia da Semana	Matéria	Horário
Segunda-feira	Matemática	08:00 - 10:00
Horários sujeitos a alterações		

Figura 3 - Exemplo de uso das tags <thead>, <tfoot> e <tbody> no navegador.

Descrição: No navegador, este código cria uma tabela com bordas visíveis e o título "Horário Semanal de Aulas" acima da tabela. A primeira linha no cabeçalho (<thead>) contém os cabeçalhos "Dia da Semana", "Matéria" e "Horário". No rodapé (<tfoot>), há uma linha que abrange todas as três colunas com a mensagem "Horários sujeitos a alterações". No corpo da tabela (<tbody>), há uma linha com os dados "Segunda-feira", "Matemática" e "08:00 - 10:00".

A organização clara das tabelas usando <thead>, <tbody> e <tfoot> contribui significativamente para a usabilidade e acessibilidade da web, tornando o conteúdo mais acessível e a experiência do usuário mais inclusiva.

Trabalhando com formulários em documentos HTML

Os formulários HTML são uma parte essencial da web, permitindo a interação entre usuários e sites. Eles são usados para coletar informações dos usuários, como nomes, endereços de e-mail, entre outras. Essas informações podem ser enviadas para um servidor web para processamento ou armazenamento, facilitando uma ampla gama de funcionalidades, desde pesquisas e registros até sistemas de login.

Veremos neste curso como podemos criar formulário em HTML5.

Criando um formulário básico

A tag <form> serve como um contêiner para diferentes tipos de inputs e controles de formulário, como caixas de texto, botões de opção, caixas de seleção, botões de envio, entre outros.

Esta tag tem um conjunto de atributos que ajudam na sua formatação, de maneira que veremos abaixo os três atributos mais usados.

Atributo action

O atributo **action** da tag **<form>** especifica o URL do servidor ao qual os dados do formulário serão enviados para processamento. Esse atributo define a localização (endereço) onde os dados coletados pelo formulário devem ser encaminhados após o usuário submeter o formulário.

Veja no quadro abaixo um exemplo:

```
<form action="submit.php" method="post">
  <!-- Campos do formulário aqui -->
  <input type="submit" value="Enviar">
</form>
```

Neste exemplo, **action** aponta para **submit.php**, indicando que os dados serão enviados para este script no servidor.

Observe que quando um formulário é submetido, os dados inseridos pelo usuário são empacotados e enviados ao servidor especificado no atributo **action**. O servidor, então, pode processar esses dados conforme necessário, o que pode incluir salvar as informações em um banco de dados, enviar e-mails, autenticar usuários, entre outras ações.

OBS: Quando não usados o atributo **action**, os dados são enviados para a própria página que inclui o formulário.

Atributo method

O atributo **method** da tag **<form>** define como os dados do formulário devem ser enviados ao servidor via protocolo HTTP. Ele pode ter dois valores principais: **GET** ou **POST**, cada um indicando um método HTTP diferente para a transmissão dos dados.

OBS: O HTTP (Hypertext Transfer Protocol) é um protocolo de comunicação fundamental da World Wide Web, utilizado para transferir dados entre um navegador web (cliente) e um servidor web.

Para compreender a diferença entre esses dois métodos HTTP, é crucial explorar seu funcionamento básico. Sempre que um recurso na web é solicitado, seu navegador envia uma requisição a um determinado URL. Essa requisição HTTP se divide em dois componentes principais: o cabeçalho, que carrega metadados detalhando as capacidades do navegador, e o corpo, que pode incluir dados específicos requeridos pelo servidor para processar adequadamente a requisição.

MÉTODO GET

O **método GET** é utilizado por padrão para o envio de dados quando, na criação de um formulário, não se especifica explicitamente um método através do atributo `method` na tag **<form>**. Com esse método, os dados submetidos são anexados à URL da página destinada a processá-los, acompanhando o nome da página na própria URL. Assim, a URL incluirá os nomes e valores dos campos do formulário, visíveis para qualquer pessoa.

Veja um exemplo no quadro abaixo:

```
<form action="submit.php" method="get">
  <label for="search">Pesquisar:</label>
  <input type="text" id="search" name="query">
  <input type="submit" value="Enviar">
</form>
```

Veja como fica o formulário no navegador, conforme figura abaixo:

Pesquisar:

Figura 1 - Exemplo de uso da tag `<form>` no navegador.

Descrição: No navegador, este código cria um formulário de pesquisa. Ele contém uma etiqueta "Pesquisar:" seguida de um campo de texto onde o usuário pode digitar sua consulta. Abaixo do campo de texto, há um botão "Enviar" para enviar a consulta.

Quando o usuário preenche o campo de pesquisa e clica em "Enviar", os dados do formulário são enviados para o servidor web. Se o usuário digitar, por exemplo, "IFPR" no campo de pesquisa, a URL final para onde o navegador redireciona parecerá algo como mostrado no quadro abaixo:

```
submit.php?query=IFPR
```

Assim, o servidor, ao receber essa URL, processa os parâmetros (neste caso, **query=IFPR**) para realizar a ação desejada, como realizar uma busca no site com base na entrada do usuário.

Como os dados são dados submetidos são anexados à URL da página destinada a processá-los, há limitações de tamanho para a quantidade de dados que podem ser enviados (depende do navegador e do servidor). Este modelo é adequado, portanto, para buscas e qualquer situação em que os dados enviados não sejam sensíveis.

MÉTODO POST

O método **POST**, entretanto, é usado para enviar dados do cliente (normalmente, um navegador) para o servidor de uma maneira que não expõe as informações na URL, uma vez que transmite os dados no corpo da requisição HTTP. Dessa forma, esse método é adequado para transferir dados sensíveis ou grandes quantidades de informação.

Veja um exemplo, conforme quadro abaixo:

```
<form action="submit.php" method="post">
  <label for="nome">Nome:</label>
  <input type="text" id="nome" name="nome" required><br><br>
  <label for="email">E-mail:</label>
  <input type="email" id="email" name="email" required><br><br>
  <label for="senha">Senha:</label>
  <input type="password" id="senha" name="senha" required><br><br>
  <input type="submit" value="Enviar">
</form>
```

Veja na figura abaixo como fica o formulário o navegador.

Nome: E-mail: Senha:

Figura 2 - Exemplo de uso do atributo post da tag <form> exibido no navegador.

Descrição: No navegador, este código cria um formulário com campos para o usuário preencher. O formulário inclui: 1) Um campo de texto para o nome, identificado pela etiqueta "Nome: "; 2) Um campo de e-mail, identificado pela etiqueta "E-mail: "; 3) Um campo de senha, identificado pela etiqueta "Senha: "; 4) Um botão "Enviar" para submeter os dados. Todos os campos são obrigatórios.

Neste caso, quando o usuário preenche os campos do formulário e clica em "Enviar", os dados são enviados para o servidor web através de uma requisição HTTP do tipo **POST**. Os dados submetidos não são visíveis na URL, oferecendo uma camada adicional de privacidade em comparação ao método **GET**. O servidor web, ao receber os dados, pode processá-los conforme necessário — por exemplo, salvar as informações em um banco de dados ou autenticar o usuário com base no e-mail e senha fornecidos.

Atributo `enctype`

O atributo **enctype** da tag `<form>` especifica como os dados do formulário devem ser codificados ao serem enviados ao servidor. Este atributo é particularmente importante quando o formulário inclui operações de envio de arquivos.

Existem três valores possíveis para **enctype**:

- **application/x-www-form-urlencoded:** Este é o valor padrão se o atributo não for especificado. Com esse método, os caracteres são codificados de forma que se ajustem aos padrões da URL, com espaços convertidos em + e caracteres especiais convertidos em valores ASCII HEX.
- **multipart/form-data:** Este valor é necessário quando você está fazendo upload de arquivos através do formulário.
- **text/plain:** Com esse método, os dados são enviados sem nenhuma codificação. Isso é raramente utilizado e não é recomendado para a maioria das aplicações web, pois os dados podem ser enviados de uma forma que é difícil de interpretar pelo servidor web.

Veja no quadro abaixo um exemplo:

```
<form action="upload.php" method="post" enctype="multipart/form-data">
  <label for="file">Selecione um arquivo:</label>
  <input type="file" id="file" name="arquivo">
  <input type="submit" value="Upload">
</form>
```

Veremos outros exemplos do uso deste atributo mais adiante neste documento.

Atributo autocomplete

O atributo **autocomplete**, quando usado em um elemento **<form>**, controla a capacidade do navegador de preencher automaticamente os campos de formulário com dados previamente inseridos pelo usuário. Esse recurso é útil para formulários que requerem informações repetidas, como endereço, e-mail ou informações de pagamento, pois pode economizar tempo para o usuário e reduzir erros de digitação.

Os valores para **autocomplete** em um elemento **<form>** são os seguintes:

- **on**: ativa o preenchimento automático para todos os campos do formulário, a menos que o atributo **autocomplete** seja especificamente definido como **off** em um elemento **<input>** individual. Isso permite que o navegador preencha automaticamente os campos com dados correspondentes salvos anteriormente pelo usuário.
- **off**: Desativa a auto-completação para todos os campos do formulário. Isso pode ser útil em formulários que lidam com informações sensíveis ou únicas que não devem ser reutilizadas, como formulários de pagamento ou de alteração de senha.

Vejamos no quadro abaixo um exemplo ativando o preenchimento automático dos campos do formulário:

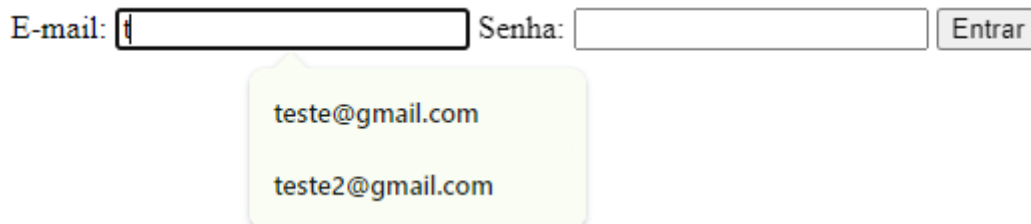
```
<form action="/login" method="post" autocomplete="on">
  <label for="email">E-mail:</label>
  <input type="email" id="email" name="email"><br><br>

  <label for="senha">Senha:</label>
  <input type="password" id="senha" name="senha"><br><br>

  <input type="submit" value="Entrar">
</form>
```

Neste exemplo, o atributo **autocomplete="on"** no elemento **<form>** ativa o preenchimento automático para o formulário, permitindo que o navegador preencha automaticamente os campos de e-mail e senha, se informações correspondentes estiverem disponíveis.

Veja na figura abaixo o resultado no navegador.



The image shows a web form with two input fields. The first field is labeled "E-mail:" and contains the text "teste@gmail.com". The second field is labeled "Senha:" and is empty. To the right of the "Senha:" field is a button labeled "Entrar". Below the "E-mail:" field, a yellow tooltip box displays two suggestions: "teste@gmail.com" and "teste2@gmail.com".

Figura 3 - Exemplo de uso do atributo `autocomplete="on"` da tag `<form>` exibido no navegador.

Descrição: No navegador, este código cria um formulário de login com dois campos: 1) Um campo de e-mail identificado pela etiqueta "E-mail: "; 2) Um campo de senha identificado pela etiqueta "Senha:". Há também um botão "Entrar" para submeter o formulário. O atributo `autocomplete="on"` permite que o navegador sugira automaticamente valores previamente usados para os campos.

Vejamos no quadro abaixo um exemplo desativando o preenchimento automático dos campos do formulário:

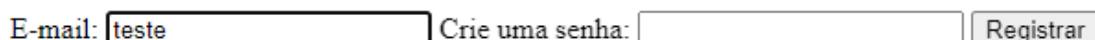
```
<form action="/register" method="post" autocomplete="off">
  <label for="email">E-mail:</label>
  <input type="email" id="email" name="email"><br><br>

  <label for="new-password">Crie uma senha:</label>
  <input type="password" id="new-password" name="new_password"
    autocomplete="new-password"><br><br>

  <input type="submit" value="Registrar">
</form>
```

Neste exemplo, o atributo **`autocomplete="off"`** no elemento **`<form>`** desativa o preenchimento automático para o formulário de registro, impedindo que o navegador preencha automaticamente os campos de e-mail e senha.

Veja na figura abaixo o resultado no navegador.



The image shows a web form with two input fields. The first field is labeled "E-mail:" and contains the text "teste". The second field is labeled "Crie uma senha:" and is empty. To the right of the "Crie uma senha:" field is a button labeled "Registrar".

Figura 4 - Exemplo de uso do atributo `autocomplete="off"` da tag `<form>` exibido no navegador.

Descrição: No navegador, este código cria um formulário de registro com dois campos: 1) Um campo de e-mail identificado pela etiqueta "E-mail:"; 2) Um campo de senha identificado pela etiqueta "Crie uma senha:". Há também um botão "Registrar" para enviar o formulário. O atributo `autocomplete="off"` impede que o navegador sugira automaticamente valores previamente usados para os campos.

OBS: Os usuários podem ter configurado seus navegadores para ignorar as sugestões de preenchimento automático, afetando a eficácia desse atributo.

Atributo `novalidate`

O atributo **`novalidate`** na tag `<form>` é usado para desativar a validação do lado do cliente fornecida pelo HTML5 para o formulário em que é aplicado. Quando presente, impede que o navegador realize qualquer validação automática dos campos do formulário, como verificar se os campos obrigatórios (**`required`**) estão preenchidos, se o formato do email está correto (**`type="email"`**), entre outros.

O **`novalidate`** é útil em situações como:

- Quando você está desenvolvendo ou testando um formulário e não quer que as validações interfiram.
- Quando você prefere usar suas próprias funções de validação com JavaScript, oferecendo feedback customizado ou complexo que não pode ser conseguido com a validação automática do HTML5.

Vejamos no quadro abaixo um exemplo:

```
<form action="/submit-data" method="post" novalidate>
  <label for="email">E-mail:</label>
<input type="email" id="email" name="email" required><br><br>

  <label for="age">Idade:</label>
<input type="number" id="age" name="age" min="18"><br><br>

  <input type="submit" value="Enviar">
</form>
```

Neste exemplo, mesmo que o campo de e-mail esteja vazio ou não contenha um endereço de e-mail válido, e a idade seja menor que 18 (violando a regra **min="18"**), o navegador não exibirá mensagens de erro de validação ao tentar enviar o formulário.

OBS: Desativar a validação do lado do cliente não elimina a necessidade de validar os dados no lado do servidor. Independentemente das validações do cliente, sempre valide e sane os dados do formulário no servidor para evitar injeções de código malicioso e garantir a integridade dos dados.

Campos de entrada do usuário

A tag **<input>** é um dos elementos mais versáteis e fundamentais em formulários HTML, utilizada para criar campos onde os usuários podem inserir dados. Esta é uma tag autocontida, ou seja, não possui uma tag de fechamento, e é altamente versátil devido aos seus múltiplos tipos de entrada.

Veja um exemplo, conforme quadro abaixo:

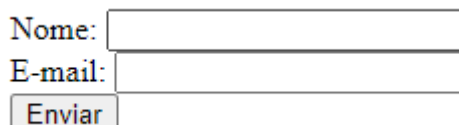
```
<form action="submit.php" method="post">
  <label for="nome">Nome:</label>
  <input type="text" id="nome" name="nome"><br><br>

  <label for="email">E-mail:</label>
  <input type="email" id="email" name="email"><br><br>

  <input type="submit" value="Enviar">
</form>
```

Neste exemplo, criamos um formulário com um campo para entrada de dados para nome e e-mail, além de um botão para submeter o formulário. Ao clicar neste botão, os dados serão enviados através do método HTTP POST para o script em **submit.php**.

Veja o resultado no navegador, conforme figura abaixo.



Nome:

E-mail:

Figura 1 - Exemplo de uso da tag <input> no navegador.

Descrição: No navegador, este código cria um formulário com dois campos: 1) Um campo de texto para "Nome" identificado pela etiqueta "Nome:"; 2) Um campo de e-mail identificado pela etiqueta "E-mail:". Há também um botão "Enviar" para submeter o formulário. Quando o botão é clicado, os dados são enviados para "submit.php" usando o método POST.

A tag <input> suporta diversos tipos de dados através do atributo **type**, permitindo a coleta de informações variadas, como texto, seleções, arquivos e mais. Aqui estão alguns dos principais usos e atributos desta tag <input>:

- **text:** Campo de texto padrão para entrada de texto simples.
- **password:** Campo de entrada que oculta o texto digitado, utilizado para senhas.
- **number:** Campo para entrada de números, com suporte para limites mínimo e máximo.
- **email:** Campo para inserção de e-mail, com validação de formato básica.
- **url:** Campo para inserção de URLs, com validação de formato básica.
- **radio:** Botões de opção que permitem ao usuário selecionar uma única opção de um conjunto.
- **checkbox:** Caixas que podem ser marcadas ou desmarcadas, permitindo múltiplas seleções.
- **submit:** Botão para enviar o formulário ao servidor.
- **reset:** Botão para limpar todos os campos do formulário.
- **file:** Campo para o usuário carregar um arquivo.
- **date, time, datetime-local:** Campos para seleção de data e/ou hora.
- **range:** Um controle deslizante para selecionar um valor dentro de um intervalo especificado.
- **color:** Um seletor de cor.
- **hidden:** Campo invisível ao usuário, utilizado para enviar informações que não precisam ser alteradas ou vistas.

Vejamos os principais elementos e seus atributos relacionados no decorrer deste documento.

Campo de entrada de texto simples

O elemento **<input type="text">** é usado em formulários HTML para criar um campo onde os usuários podem inserir uma linha de texto. É um dos tipos de entrada mais comuns e versáteis, adequado para coletar uma variedade de informações, como nomes, endereços de e-mail (embora para e-mails, o tipo **email** seja mais apropriado), comentários curtos ou qualquer outro tipo de dado textual de curta a média extensão.

Os atributos básicos deste elemento são os seguintes:

- **id:** Uma identificação única para o campo, que pode ser usada para referenciá-lo com CSS ou JavaScript.
- **name:** Define o nome do campo, que é enviado junto com o valor do campo no formulário para o servidor.
- **value:** Especifica o valor inicial do campo, que pode ser alterado pelo usuário.
- **placeholder:** Fornece um texto de orientação dentro do campo, que desaparece quando o usuário começa a digitar.
- **maxlength:** Limita o número máximo de caracteres que o usuário pode inserir no campo.
- **readonly:** Torna o campo somente leitura, impedindo que o usuário altere o valor, mas ainda permite que o valor seja enviado com o formulário.
- **disabled:** Desabilita o campo, impedindo que o usuário interaja com ele e que seu valor seja enviado.
- **required:** Indica que o campo deve ser preenchido antes que o formulário possa ser enviado.
- **size:** Define a largura visual do campo em número de caracteres.
- **autocomplete:** Controla se o navegador deve ou não preencher automaticamente o campo. Este atributo apresenta os seguintes valores:
 - **autocomplete="on":** Permite que o navegador preencha automaticamente o campo de senha com uma senha previamente salva pelo usuário. Útil para formulários de login.
 - **autocomplete="off":** Desativa a auto-completação do navegador para esse campo, forçando o usuário a digitar manualmente a senha. Pode ser útil em formulários de cadastro ou alteração de senha, onde a reutilização de senhas anteriores não é desejada.
 - **autocomplete="new-password":** Uma diretriz para o navegador que o campo é destinado à criação de uma nova senha. Isso pode incentivar o navegador a oferecer a geração de senha automática, se suportado.

Veja no quadro abaixo um exemplo:

```
<form action="submit.php" method="post">
  <label for="username">Nome de Usuário:</label>
  <input type="text" id="username" name="username"
    placeholder="Digite seu nome de usuário"
    maxlength="20" size="25" required><br><br>
  <input type="submit" value="Enviar">
</form>
```

Neste exemplo, um campo de entrada de texto é criado para que o usuário insira seu nome de usuário. O campo é obrigatório (**required**), o que significa que o formulário não pode ser enviado sem preencher este campo. O atributo **maxlength** limita o usuário a inserir no máximo 20 caracteres, e o **size** ajusta a largura visual do campo para acomodar 25 caracteres.

A imagem abaixo apresenta o resultado no navegador.



Figura 2 - Exemplo de uso do atributo `type="text"` da tag `<input>` exibido no navegador.

Descrição: No navegador, este código cria um formulário com um campo para "Nome de Usuário" e um botão "Enviar". O campo de nome de usuário tem um espaço reservado com o texto "Digite seu nome de usuário", é obrigatório, permite até 20 caracteres e exibe até 25 caracteres de largura. Quando o botão "Enviar" é clicado, os dados são enviados para "submit.php" usando o método POST.

Campo de entrada de senhas

O elemento `<input type="password">` é usado em formulários HTML para criar um campo destinado à entrada de senhas ou qualquer outro tipo de informação sensível que não deve ser visível na tela. Quando o usuário digita neste campo, os caracteres inseridos são substituídos por símbolos de ocultação (geralmente pontos ou asteriscos) para proteger a informação.

OBS: Este elemento contém muitos dos atributos básicos do elemento `<input>`, tais como: **id**, **name**, **value**, **maxlength**, **readonly**, **disabled**, **required**, **size** e **autocomplete**.

Cabe lembrar que o atributo **autocomplete** em um elemento **<input type="password">** é usado para controlar se os navegadores devem permitir a auto-completação para campos de senha. Esse recurso pode melhorar a experiência do usuário permitindo que ele reutilize senhas salvas anteriormente, mas também deve ser usado com cautela por questões de segurança e privacidade.

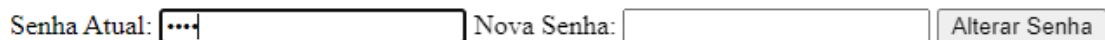
Veja no quadro abaixo um exemplo:

```
<form action="/change-password" method="post">
  <label for="current-password">Senha Atual:</label>
  <input type="password" id="current-password" name="current_password"
    autocomplete="current-password"><br><br>

  <label for="new-password">Nova Senha:</label>
  <input type="password" id="new-password" name="new_password"
    autocomplete="new-password"><br><br>

  <input type="submit" value="Alterar Senha">
</form>
```

Veja o resultado no navegador, conforme imagem abaixo.



Senha Atual: Nova Senha:

Figura 3 - Exemplo de uso do atributo `type="password"` da tag `<input>` exibido no navegador.

Descrição: No navegador, este código cria um formulário para alterar a senha com dois campos: 1) Um campo de senha para "Senha Atual", identificado pela etiqueta "Senha Atual:"; 2) Um campo de senha para "Nova Senha", identificado pela etiqueta "Nova Senha:". Ambos os campos têm o atributo `autocomplete` para sugerir automaticamente senhas usadas anteriormente. Há também um botão "Alterar Senha" para submeter o formulário.

A especificação HTML sugere que os navegadores ignorem o atributo **autocomplete="off"** para campos de senha em formulários de login, devido à sua importância para a experiência do usuário. Por isso, o comportamento pode variar entre diferentes navegadores. A escolha de usar ou não a auto-completação deve considerar tanto a conveniência para o usuário quanto as práticas recomendadas de segurança.

Campo de entrada de números

O elemento `<input type="number">` é utilizado em formulários HTML para permitir que os usuários insiram um número. Ele apresenta uma interface de usuário otimizada para a entrada de números, incluindo, em muitos navegadores, pequenos botões de incremento e decremento para ajustar o valor.

Este elemento contém muitos dos atributos básicos do elemento `<input>`, tais como: **id**, **name**, **value**, **maxlength**, **readonly**, **disabled**, **required**, **size** e **autocomplete**. Adicionalmente, é possível utilizar estes outros atributos com o elemento `<input type="number">`:

- **min**: Define o valor mínimo que os usuários podem inserir.
- **max**: Define o valor máximo que os usuários podem inserir.
- **step**: Define o intervalo entre os valores que o usuário pode selecionar, permitindo maior controle sobre os valores válidos.

Veja no quadro abaixo um exemplo:

```
<form action="submit.php" method="post">
  <label for="age">Idade:</label>
  <input type="number" id="age" name="age" min="0" max="130" step="1"
    required><br><br>
  <input type="submit" value="Enviar">
</form>
```

Neste exemplo, um campo de entrada é criado para que os usuários insiram sua idade. O campo é configurado para aceitar apenas números inteiros entre 0 e 130. Os atributos **min** e **max** garantem que apenas valores dentro deste intervalo sejam aceitos, enquanto **step="1"** permite apenas valores inteiros. O campo também é marcado como **required**, o que significa que o formulário não pode ser enviado sem que um número válido seja inserido.

Veja o resultado no navegador, conforme imagem abaixo.



Figura 4 - Exemplo de uso do atributo `type="number"` da tag `<input>` exibido no navegador.

Descrição: No navegador, este código cria um formulário com um campo para "Idade" e um botão "Enviar". O campo de idade aceita apenas números entre 0 e 130 e aumenta ou diminui em passos de 1. Este campo é obrigatório.

O navegador automaticamente valida a entrada do usuário com base nos atributos **min**, **max**, e **step**. Se um valor inválido for inserido, o formulário não será enviado até que um valor válido seja fornecido. Ainda, em dispositivos móveis, campos com **type="number"** geralmente exibem um teclado numérico, facilitando a inserção de números.

Campo de entrada de e-mail

O elemento **<input type="email">** é utilizado em formulários HTML para criar um campo destinado à entrada de endereços de e-mail. Este tipo de input facilita a validação do lado do cliente, assegurando que o usuário insira um endereço de e-mail em um formato válido antes de submeter o formulário.

Este elemento contém muitos dos atributos básicos do elemento **<input>**, tais como: **id**, **name**, **value**, **maxlength**, **readonly**, **disabled**, **required**, **size** e **autocomplete**. Adicionalmente, é possível utilizar estes outros atributos com o elemento **<input type="email">**:

- **multiple:** Permite a inserção de múltiplos endereços de e-mail, separados por vírgulas. Esse atributo não é suportado em todos os navegadores.
- **pattern:** Uma expressão regular que o valor do campo deve corresponder para ser considerado válido. Embora o tipo **email** já faça uma validação básica do formato de e-mail, o atributo **pattern** pode ser usado para regras mais específicas.

Veja no quadro abaixo um exemplo:

```
<form action="submit.php" method="post">
  <label for="user_email">Seu e-mail:</label>
  <input type="email" id="user_email" name="user_email"
    placeholder="exemplo@dominio.com" required><br><br>
  <input type="submit" value="Enviar">
</form>
```


Neste exemplo, o campo de e-mail é obrigatório (**required**), o que significa que o formulário não pode ser enviado sem preencher este campo com um valor que passe na validação de formato de e-mail. O atributo **placeholder** fornece uma dica visual para o usuário sobre o tipo de informação que deve ser inserida.

Veja o resultado no navegador, conforme imagem abaixo.



Figura 5 - Exemplo de uso do atributo `type="email"` da tag `<input>` exibido no navegador.

Descrição: No navegador, este código cria um formulário com um campo para "Seu e-mail" e um botão "Enviar". O campo de e-mail aceita apenas endereços de e-mail válidos e exibe um espaço reservado com o formato típico de um endereço de e-mail. Este campo é obrigatório.

O navegador automaticamente valida o campo para garantir que o texto inserido seja um endereço de e-mail válido (por exemplo, contendo caracteres como @ e um domínio). Se o usuário tentar enviar o formulário com um e-mail em formato incorreto, o navegador exibirá uma mensagem de erro padrão e impedirá o envio do formulário até que um endereço de e-mail válido seja fornecido.

Botões de opção (radio buttons)

O elemento `<input type="radio">` é usado em formulários HTML para criar botões de opção, permitindo que os usuários selecionem uma única opção de um conjunto de escolhas. Cada botão de opção dentro do mesmo grupo deve ter o mesmo nome no atributo **name** para garantir que apenas uma opção possa ser selecionada por vez.

Os atributos básicos deste elemento são os seguintes:

- **id:** Uma identificação única para o campo, que pode ser usada para referenciá-lo com CSS ou JavaScript.
- **name:** Define o nome do grupo ao qual o botão de opção pertence. Todos os botões de opção que compõem um grupo devem ter o mesmo **name**.
- **value:** O valor enviado ao servidor se essa opção for selecionada pelo usuário.
- **checked:** Um atributo booleano que, se presente, indica que o botão de opção está selecionado por padrão quando a página é carregada.
- **disabled:** Um atributo booleano que, se presente, desabilita o botão de opção, impedindo que o usuário o selecione.

Veja no quadro abaixo um exemplo:

```
<form action="submit.php" method="post">
  <p>Escolha sua bebida favorita:</p>
  <input type="radio" id="cafe" name="bebida" value="cafe">
    <label for="cafe">Café</label><br>

  <input type="radio" id="cha" name="bebida" value="cha" checked>
    <label for="cha">Chá</label><br>

  <input type="radio" id="agua" name="bebida" value="agua">
    <label for="agua">Água</label><br><br>

  <input type="submit" value="Enviar">
</form>
```

Observe que:

- Cada **<input type="radio">** deve ter o mesmo valor no atributo **name** para que sejam considerados parte do mesmo grupo. Isso garante que apenas uma opção possa ser selecionada por vez.
- O atributo **value** é usado para indicar o valor enviado ao servidor web quando a opção correspondente é selecionada.
- O atributo **id** em cada **<input>** e o atributo **for** em cada **<label>** são associados, facilitando a seleção da opção ao clicar no rótulo.
- O atributo **checked** em um dos botões de rádio marca essa opção como selecionada por padrão quando o formulário é carregado.

Este exemplo demonstra um formulário de pesquisa que pergunta aos usuários sobre sua bebida preferida, oferecendo três opções e garantindo uma experiência de usuário acessível e fácil de navegar.

Veja o resultado no navegador, conforme imagem abaixo.

Escolha sua bebida favorita:

☐ Café
☒ Chá
☐ Água

Figura 6 - Exemplo de uso do atributo `type="radio"` da tag `<input>` exibido no navegador.

Descrição: No navegador, este código cria um formulário com uma pergunta "Escolha sua bebida favorita:". Abaixo da pergunta, há três opções de escolha representadas por botões de rádio: 1) Café: O usuário pode selecionar esta opção clicando no botão de rádio próximo à palavra "Café"; 2) Chá: Esta opção já está selecionada por padrão (representada pelo atributo "checked"); 3) Água: O usuário pode selecionar esta opção clicando no botão de rádio próximo à palavra "Água". Após selecionar uma das opções, o usuário pode clicar no botão "Enviar" para submeter a escolha.

Essa estrutura de formulário é comumente usada em pesquisas, questionários e qualquer situação onde uma escolha exclusiva dentre várias opções é necessária.

Caixa de seleção (checkbox)

O elemento `<input type="checkbox">` é usado em formulários HTML para criar uma caixa de seleção que o usuário pode marcar ou desmarcar. Ele permite a seleção de uma ou mais opções de um conjunto de escolhas, tornando-o ideal para coletar entradas do usuário onde múltiplas seleções são possíveis e aceitáveis.

Os atributos básicos deste elemento são os seguintes:

- **id:** Uma identificação única para o campo, que pode ser usada para referenciá-lo com CSS ou JavaScript.
- **name:** Define o nome do campo que será enviado com o formulário.
- **value:** O valor enviado ao servidor se a caixa de seleção for marcada.
- **checked:** Um atributo booleano que, se presente, marca a caixa de seleção como selecionada por padrão quando a página é carregada.
- **disabled:** Um atributo booleano que, se presente, desabilita a caixa de seleção, impedindo interações do usuário.

Veja no quadro abaixo um exemplo:

```
<form action="submit.php" method="post">
  <input type="checkbox" id="newsletter" name="newsletter" value="sim"
    checked>
  <label for="newsletter">Inscreva-se para receber nossos newsletters:</
    label><br><br>

  <input type="checkbox" id="terms" name="terms" value="aceito" required>
  <label for="terms">Eu concordo com os termos e condições:</
    label><br><br>

  <input type="submit" value="Enviar">
</form>
```

Neste exemplo, há duas caixas de seleção: uma para se inscrever em newsletters, que está selecionada por padrão (**checked**), e outra para concordar com os termos e condições, que o usuário precisa marcar explicitamente antes de enviar o formulário.

Veja o resultado no navegador, conforme figura abaixo.

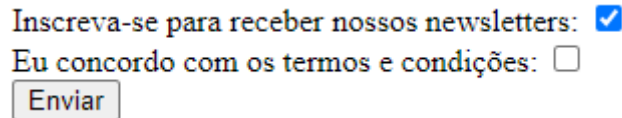


Figura 7 - Exemplo de uso do atributo `type="checkbox"` da tag `<input>` exibido no navegador.

Descrição: No navegador, este código cria um formulário com duas opções de caixa de seleção: 1) "Inscreva-se para receber nossos newsletters": Esta opção já está marcada (representada pelo atributo "checked"). O usuário pode desmarcá-la se não quiser se inscrever; 2) "Eu concordo com os termos e condições": Esta opção está desmarcada inicialmente. O usuário pode marcá-la para indicar que concorda com os termos e condições. Após fazer suas seleções, o usuário pode clicar no botão "Enviar" para submeter suas escolhas.

Sobre o elemento o elemento `<input type="checkbox">`, cabe ressaltar:

- Caixas de seleção sem o atributo **checked** serão desmarcadas por padrão.
- O valor de um `<input type="checkbox">` só é incluído no envio do formulário se ele estiver marcado.
- É recomendável usar a tag `<label>` juntamente com cada `<input type="checkbox">` para melhorar a acessibilidade, permitindo que os usuários cliquem no texto do rótulo para alternar a seleção da caixa.

Botão para submeter um formulário

O elemento `<input type="submit">` é usado em formulários HTML para criar um botão que permite ao usuário enviar o formulário ao servidor. Quando clicado, este botão envia os dados do formulário para o endereço especificado no atributo **action** da tag `<form>`, utilizando o método definido no atributo **method** (geralmente **GET** ou **POST**).

Os atributos básicos deste elemento são os seguintes:

- **id:** Uma identificação única para o campo, que pode ser usada para referenciá-lo com CSS ou JavaScript.

- **name:** Embora menos comum para botões de envio, este atributo pode ser usado para identificar o botão no lado do servidor quando o formulário é submetido.
- **type:** Este atributo deve ser definido como **submit** para indicar que o elemento é um botão de envio de formulário.
- **value:** Define o texto que será exibido no botão de envio. Esse texto é visível para o usuário e pode ser personalizado para se adequar ao contexto do formulário.

Veja no quadro abaixo um exemplo:

```
<form action="submit.php" method="post">
  <label for="nome">Nome:</label>
  <input type="text" id="nome" name="nome"><br><br>

  <label for="email">E-mail:</label>
  <input type="email" id="email" name="email"><br><br>

  <input type="submit" value="Enviar Dados">
</form>
```

Neste exemplo, o botão de envio é rotulado como "Enviar Dados". Quando o usuário clica neste botão, os dados do formulário são enviados para o servidor no endereço **submit.php** usando o método **POST**.

Veja o resultado no navegador, conforme figura abaixo.



Figura 8 - Exemplo de uso do atributo `type="submit"` da tag `<input>` exibido no navegador.

Descrição: No navegador, este código cria um formulário com dois campos: 1) Um campo para "Nome", identificado pela etiqueta "Nome:"; 2) Um campo para "E-mail", identificado pela etiqueta "E-mail:". Abaixo dos campos, há um botão "Enviar Dados".

O botão de envio é um elemento crucial em qualquer formulário, agindo como o ponto final da interação do usuário, onde ele confirma a intenção de enviar os dados inseridos.

Botão para redefinir os valores dos campos do formulário

O elemento `<input type="reset">` cria um botão que permite ao usuário redefinir todos os campos de um formulário aos seus valores iniciais. Esse tipo de botão é útil para formulários longos ou complexos, onde os usuários podem querer começar novamente com as informações padrão depois de fazer várias alterações.

Ao clicar em um botão de redefinição, todos os campos do formulário voltam aos seus valores padrão, ou seja, aos valores que tinham quando a página foi carregada pela primeira vez. Isso inclui limpar qualquer texto inserido, desmarcar checkboxes ou botões de rádio selecionados e reverter seleções em listas suspensas (`<select>`).

Diferentemente do `<input type="submit">`, o `<input type="reset">` não envia o formulário. Sua única função é limpar os campos de entrada.

Veja um exemplo no quadro abaixo:

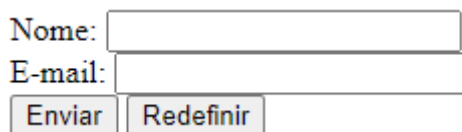
```
<form action="submit.php" method="post">
  <label for="nome">Nome:</label>
  <input type="text" id="nome" name="nome" value="Seu Nome"><br><br>

  <label for="email">E-mail:</label>
  <input type="email" id="email" name="email"
    value="seu.email@exemplo.com"><br><br>

  <input type="submit" value="Enviar">
  <input type="reset" value="Redefinir">
</form>
```

Neste exemplo, além do botão de envio (Enviar), há um botão de redefinição (Redefinir). Se um usuário inserir informações nos campos de texto e, por algum motivo, desejar redefinir todas as suas entradas para o estado inicial, ele pode clicar em "Redefinir" para limpar todas as modificações.

Veja o resultado no navegador, conforme figura abaixo.



Nome:

E-mail:

Figura 9 - Exemplo de uso do atributo `type="reset"` da tag `<input>` exibido no navegador.

Descrição: No navegador, este código cria um formulário com dois campos: 1) Um campo para "Nome", identificado pela etiqueta "Nome:"; 2) Um campo para "E-mail", identificado pela etiqueta "E-mail:". Abaixo dos campos, há dois botões: 1) Um botão "Enviar" que, quando clicado, envia os dados inseridos nos campos para "submit.php" usando o método POST; 2) Um botão "Redefinir" que, quando clicado, limpa os campos do formulário, permitindo que o usuário redefina os dados inseridos.

Campo de seleção de arquivos

O elemento **<input type="file">** é utilizado em formulários HTML para permitir que os usuários selecionem arquivos de seus dispositivos para upload. Esse tipo de campo de entrada cria um botão de seleção de arquivo e uma caixa de texto que exibe o nome do arquivo escolhido. É amplamente utilizado para enviar imagens, documentos e outros arquivos para um servidor.

Os atributos básicos deste elemento são os seguintes:

- **id:** Uma identificação única para o campo, que pode ser usada para referenciá-lo com CSS ou JavaScript.
- **name:** O nome do campo, que será enviado com o formulário.
- **accept:** Permite especificar um ou mais tipos de arquivos que o usuário pode selecionar, usando tipos MIME (como **image/*** para imagens ou **.pdf** para documentos PDF). Isso ajuda a filtrar os arquivos no diálogo de seleção, facilitando para o usuário encontrar o tipo de arquivo desejado.
- **multiple:** Quando presente, permite que os usuários selecionem mais de um arquivo simultaneamente.
- **required:** Indica que o usuário deve selecionar um arquivo antes de enviar o formulário.

OBS: Tipos MIME (Multipurpose Internet Mail Extensions) são um padrão que indica a natureza e o formato de um arquivo, documento ou conjunto de dados, utilizado na Internet para facilitar o intercâmbio e a interpretação de arquivos entre sistemas diferentes.

Veja um exemplo de uso no quadro abaixo:

```
<form action="submit.php" method="post" enctype="multipart/form-data">
    <label for="arquivo">Escolha um arquivo:</label>
    <input type="file" id="arquivo" name="arquivo" accept="image/*"
        multiple><br><br>
    <input type="submit" value="Upload">
</form>
```

Sobre este exemplo, observe:

- Para que o upload de arquivos funcione corretamente, o atributo **enctype** do **<form>** deve ser definido como **multipart/form-data**. Isso garante que os arquivos sejam enviados corretamente como dados binários junto com qualquer outro dado do formulário.
- A especificação do atributo **accept** pode ajudar a reduzir a quantidade de arquivos inválidos ou indesejados enviados, orientando o usuário na seleção de arquivos permitidos, mas o filtro aplicado pelo navegador pode variar. A validação do tipo de arquivo também deve ser realizada no lado do servidor para maior segurança.
- O suporte para o atributo **multiple** pode variar entre navegadores mais antigos. Quando habilitado, permite aos usuários selecionar e enviar vários arquivos em um único campo de entrada.

Veja o resultado no navegador, conforme figura abaixo:



Figura 10 - Exemplo de uso do atributo enctype da tag <form> exibido no navegador.

Descrição: No navegador, este código cria um formulário para fazer upload de arquivos: 1) Um campo "Escolha um arquivo" com um botão "Procurar" ao lado, que permite ao usuário selecionar um ou mais arquivos para upload; 2) Abaixo do campo, há um botão "Upload" que, quando clicado, envia os arquivos selecionados para "submit.php" usando o método POST; 3) O atributo "accept" limita a seleção de arquivos apenas a imagens.

O uso de **<input type="file">** é essencial para formulários que envolvem a submissão de arquivos, oferecendo uma maneira fácil e direta para os usuários enviarem dados além de texto.

Campo de entrada de datas e horas

Os elementos `<input type="date">`, `<input type="time">`, e `<input type="datetime-local">` são usados para criar interfaces de usuário que facilitam a entrada de datas, horas, ou ambos. Eles fornecem widgets de seleção intuitivos, eliminando a necessidade de formatos de entrada manuais complexos e melhorando a experiência geral do usuário. Veja abaixo estes três elementos:

- **`<input type="date">`**: Permite ao usuário selecionar uma data (ano, mês e dia). Os navegadores geralmente apresentam isso como um calendário pop-up do qual uma data pode ser selecionada.
- **`<input type="time">`**: Permite ao usuário selecionar um horário (horas e minutos, opcionalmente segundos e fração de segundo). A apresentação exata pode variar, mas geralmente inclui algum tipo de interface gráfica para a seleção de horário.
- **`<input type="datetime-local">`**: Combina as funcionalidades dos tipos **date** e **time**, permitindo ao usuário selecionar uma data e um horário (sem considerar fuso horário). Este tipo é particularmente útil para agendamentos ou entradas de eventos que acontecem em uma data e hora específicas.

Veja um exemplo no quadro abaixo:


```
<form action="submit.php" method="post">
  <label for="meeting-date">Data da Reunião:</label>
  <input type="date" id="meeting-date" name="meeting_date"><br><br>


  <label for="meeting-time">Horário da Reunião:</label>
  <input type="time" id="meeting-time" name="meeting_time"><br><br>


  <label for="event-datetime">Data e Horário do Evento:</label>
  <input type="datetime-local" id="event-datetime"
    name="event_datetime"><br><br>

  <input type="submit" value="Enviar">
</form>
```

Veja o resultado no navegador, conforme figura abaixo:

Data da Reunião: 

Horário da Reunião: 

Data e Horário do Evento: 

março de 2024 ▾

D	S	T	Q	Q	S	S
25	26	27	28	29	1	2
3	4	5	6	7	8	9
10	11	12	13	14	15	16
17	18	19	20	21	22	23
24	25	26	27	28	29	30
31	1	2	3	4	5	6

Limpar

↑ ↓

21	24
22	25
23	26
00	27
01	28
02	29
03	30

Hoje

Figura 11 - Exemplo de uso de campos de entrada de datas e horas no navegador.

Descrição: No navegador, este código cria um formulário com três campos: 1) Um campo para "Data da Reunião", que permite ao usuário selecionar uma data; 2) Um campo para "Horário da Reunião", que permite ao usuário selecionar um horário; 3) Um campo para "Data e Horário do Evento", que permite ao usuário selecionar tanto a data quanto o horário. Abaixo dos campos, há um botão "Enviar" que, quando clicado, envia os dados inseridos nos campos para "submit.php" usando o método POST.

Ao enviar o formulário, os valores de **date** e **datetime-local** são transmitidos no formato **YYYY-MM-DD** para datas e **YYYY-MM-DDTHH:MM** (opcionalmente incluindo **:SS** ou **:SS.SSS** para segundos e milissegundos) para data e hora, respectivamente. O **time** é enviado no formato **HH:MM**, possivelmente incluindo segundos e milissegundos.

Esses elementos suportam atributos adicionais como **min** e **max** para restringir as datas ou horários selecionáveis, **step** para definir os incrementos de tempo permitidos (em segundos) para **time** e **datetime-local**, e **required** para torná-los campos obrigatórios no formulário.

Lista suspensa de opções

O elemento **<select>** é utilizado em formulários HTML para criar uma lista suspensa de opções, permitindo que os usuários escolham uma ou, se especificado, várias opções de uma lista predefinida. Um **<select>** é composto por uma ou mais tags **<option>**, cada uma representando uma opção diferente na lista suspensa.

Os atributos básicos deste elemento são os seguintes:

- **id:** Uma identificação única para o campo, que pode ser usada para referenciá-lo com CSS ou JavaScript.
- **name:** O nome do campo, que é enviado com o formulário.
- **required:** Indica que o usuário deve selecionar uma opção antes de enviar o formulário.
- **multiple:** Quando presente, permite que o usuário selecione mais de uma opção simultaneamente.
- **size:** Se combinado com **multiple**, define quantas opções são visíveis sem rolar.

Veja um exemplo no quadro abaixo:

```
<form action="submit.php" method="post">
<label for="fruta">Escolha sua fruta favorita:</label>
<select id="fruta" name="fruta_favorita" required>
  <option value="">Selecione uma opção</option>
  <option value="maca">Maçã</option>
  <option value="banana">Banana</option>
  <option value="laranja">Laranja</option>
  <option value="manga">Manga</option>
</select><br><br>
<input type="submit" value="Enviar">
</form>
```

Neste exemplo, o usuário pode escolher sua fruta favorita de uma lista suspensa. A opção inicial é intencionalmente deixada em branco e marcada como não selecionável para encorajar o usuário a fazer uma escolha consciente, ao mesmo tempo em que atende ao requisito do campo ser preenchido.

Veja o resultado no navegador, conforme a figura abaixo:

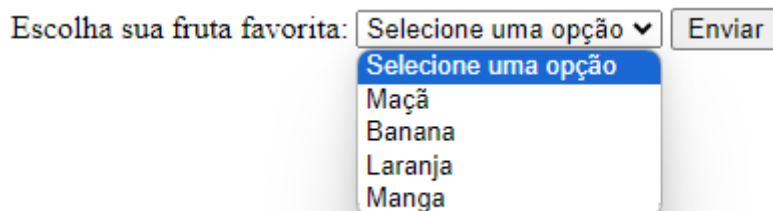


Figura 1 - Exemplo de uso da tag <select> no navegador.

Descrição: No navegador, este código cria um formulário com um menu suspenso: 1) O menu suspenso permite ao usuário selecionar sua fruta favorita entre várias opções, como maçã, banana, laranja e manga; 2) Há um rótulo associado ao menu suspenso indicando "Escolha sua fruta favorita"; 3) Abaixo do menu suspenso, há um botão "Enviar" que, quando clicado, envia a opção selecionada para "submit.php" usando o método POST. O campo é obrigatório, então uma opção deve ser selecionada antes de enviar o formulário.

A tag <optgroup> pode ser usada para agrupar opções relacionadas dentro do <select>, facilitando a organização de listas longas. Ela permite que você crie subseções dentro de uma lista suspensa, cada uma com um rótulo próprio.

Veja no quadro abaixo um exemplo:

```
<form action="submit.php" method="post">
<label for="carro">Escolha um carro:</label>
  <select id="carro" name="carro">
    <optgroup label="Fabricantes Alemães">
      <option value="mercedes">Mercedes</option>
      <option value="audi">Audi</option>
      <option value="bmw">BMW</option>
    </optgroup>
    <optgroup label="Fabricantes Japoneses">
      <option value="toyota">Toyota</option>
      <option value="nissan">Nissan</option>
      <option value="honda">Honda</option>
    </optgroup>
  </select><br><br>
  <input type="submit" value="Enviar">
</form>
```

Neste exemplo, o elemento <select> é usado para criar uma lista suspensa onde o usuário pode escolher uma marca de carro. As marcas são organizadas em dois grupos: "Fabricantes Alemães" e "Fabricantes Japoneses". Cada grupo é criado usando a tag <optgroup>, com o atributo **label** fornecendo o nome do grupo.

Veja o resultado no navegador, conforme a figura abaixo:

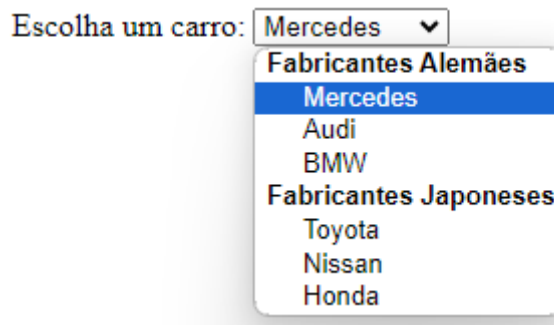


Figura 2 - Exemplo de uso da tag <optgroup> exibido no navegador.

Descrição: No navegador, este código cria um formulário com um menu suspenso: 1) O menu suspenso permite ao usuário escolher um carro de duas seções distintas: "Fabricantes Alemães" e "Fabricantes Japoneses"; 2) Cada seção é identificada por um rótulo, como "Fabricantes Alemães" e "Fabricantes Japoneses"; 3) Sob cada rótulo, há uma lista de opções de carros, como Mercedes, Audi e BMW para fabricantes alemães, e Toyota, Nissan e Honda para fabricantes japoneses.

Usar <optgroup> em elementos <select> é uma prática recomendada para melhorar a experiência do usuário ao lidar com formulários complexos ou com muitas opções de escolha.

Área de texto de múltiplas linhas

O elemento <textarea> é usado para criar uma área de texto de múltiplas linhas, permitindo aos usuários inserir texto livre, como comentários, mensagens ou qualquer outro tipo de conteúdo extenso. Diferente dos <input type="text">, que são projetados para inserção de uma única linha de texto, o <textarea> oferece mais espaço, tornando-o ideal para entradas de texto maiores.

Os atributos básicos deste elemento são os seguintes:

- **id:** Uma identificação única para o campo, que pode ser usada para referenciá-lo com CSS ou JavaScript.
- **name:** O nome do campo, que será enviado com o formulário.
- **rows:** O número de linhas visíveis na área de texto sem a necessidade de rolagem.
- **cols:** O número de colunas que definem a largura da área de texto.
- **placeholder:** Um texto de orientação exibido na área de texto até que o usuário comece a digitar.
- **maxlength:** O número máximo de caracteres que o usuário pode inserir na área de texto.

- **readonly:** Torna a área de texto somente leitura.
- **disabled:** Desabilita a área de texto, impedindo a interação do usuário.
- **required:** Especifica que o usuário deve preencher o campo antes de enviar o formulário.

Veja no quadro abaixo um exemplo:

```
<form action="submit.php" method="post">
  <label for="comentario">Comentário:</label><br>
  <textarea id="comentario" name="comentario" rows="5" cols="33"
    placeholder="Digite seu comentário aqui..." required></
    textarea><br><br>
  <input type="submit" value="Enviar">
</form>
```

Neste exemplo, uma área de texto é criada para permitir que os usuários insiram um comentário. O atributo **placeholder** fornece uma dica sobre o que o usuário deve inserir, enquanto os atributos **rows** e **cols** definem o tamanho da área de texto. O atributo **required** garante que o formulário não possa ser enviado sem que algum texto seja inserido na área de texto.

Veja o resultado no navegador, conforme a figura abaixo:

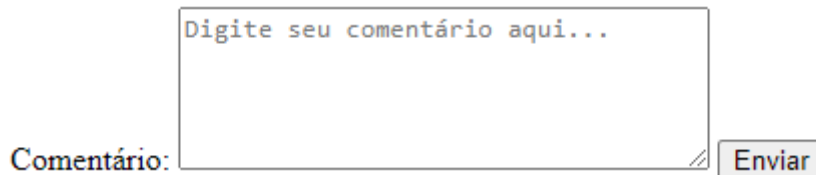


Figura 1 - Exemplo de uso da tag <textarea> no navegador.

Descrição: Esse código cria um formulário com um campo de texto grande onde os usuários podem digitar seus comentários: 1) Um rótulo "Comentário" identifica o campo de texto; 2) O campo de texto é uma área de texto onde os usuários podem escrever seus comentários; Este campo tem um tamanho predefinido de 5 linhas por 33 colunas. Também tem um espaço de placeholder fornece uma orientação sobre o que digitar. Ainda tem um botão "Enviar" que permite que os usuários enviem seus comentários.

Botões

O elemento **<button>** em HTML é usado para criar um botão clicável, que pode executar uma ação quando pressionado. Os botões são elementos essenciais em interfaces web, permitindo aos usuários interagir com a página de maneiras diversas, como enviar formulários, executar scripts JavaScript, ou simplesmente agir como links.

O elemento **<button>** pode ser configurado de três maneiras principais, usando o atributo **type**:

- **submit**: O padrão se nenhum **type** for especificado. Usado para enviar um formulário ao servidor.
- **reset**: Redefine os campos de um formulário para seus valores padrão.
- **button**: Não tem comportamento padrão de envio ou redefinição e é comumente usado para executar um script JavaScript.

Veja no quadro abaixo um exemplo:

```
<form action="submit.php" method="post">
  <button type="submit">Enviar</button>
</form>
```

Veja o resultado no navegador, conforme figura abaixo:



Figura 1 - Exemplo de uso da tag **<button>** no navegador.

Descrição: No navegador, esse código cria um formulário com um botão "Enviar". Este botão é usado para enviar os dados inseridos no formulário para o servidor para processamento posterior.

Este botão, quando colocado dentro de um formulário, envia os dados do formulário ao servidor.

OBS: A diferença entre **<input type="submit">** e **<button type="submit">** reside principalmente na flexibilidade e na capacidade de personalização. Ambos desempenham a função básica de enviar um formulário ao servidor, mas há nuances importantes que podem influenciar

qual escolher dependendo do contexto específico do projeto. Enquanto que o elemento **<input type="submit">** é tradicionalmente usado em formulários HTML para criar um botão de envio, o conteúdo (o texto do botão) é definido pelo atributo **value**. Dessa forma, não é possível incluir HTML ou ícones dentro do botão; ele suporta apenas texto. O elemento **<button>** oferece uma maior flexibilidade. Pode conter conteúdo HTML, permitindo a inclusão de ícones, imagens ou formatação especial dentro do botão.

Agrupando elementos de formulário

Os elementos **<fieldset>** e **<legend>** são usados em conjunto em formulários HTML para agrupar logicamente um conjunto de elementos de formulário e definir um título para esse grupo, respectivamente. Essa combinação não apenas melhora a organização visual dos formulários, mas também aumenta a acessibilidade, facilitando a navegação e compreensão do formulário por usuários que utilizam leitores de tela.

Veja abaixo um exemplo no quadro abaixo:

```
<form action="submit.php" method="post">
  <fieldset>
    <legend>Informações de Contato</legend>
    <label for="nome">Nome:</label>
    <input type="text" id="nome" name="nome"><br><br>

    <label for="email">E-mail:</label>
    <input type="email" id="email" name="email"><br><br>

    <label for="telefone">Telefone:</label>
    <input type="tel" id="telefone" name="telefone"><br><br>
  </fieldset>
  <input type="submit" value="Enviar">
</form>
```

Neste exemplo, o **<fieldset>** é usado para agrupar campos relacionados às informações de contato do usuário, com **<legend>** fornecendo um título claro para o grupo. Isso não apenas melhora a organização visual do formulário, mas também facilita a compreensão dos usuários sobre quais informações estão sendo solicitadas.

Veja o resultado no navegador, conforme figura abaixo:

The image shows a web form with a title 'Informações de Contato' (Contact Information). Inside a rectangular frame, there are three labels with corresponding input fields: 'Nome:' (Name), 'E-mail:', and 'Telefone:' (Telephone). Below the frame is a button labeled 'Enviar' (Send).

Figura 1 - Exemplo de uso da tag <fieldset> no navegador.

Descrição: No navegador, esse código cria um formulário com um conjunto de informações de contato. O formulário tem um campo para o nome, um para o e-mail e um para o telefone. Os campos são agrupados dentro de um elemento **<fieldset>**, que tem uma legenda descrevendo o grupo de campos. Ao clicar no botão "Enviar", os dados inseridos serão submetidos para o servidor.

Finalização da etapa

Caro estudante, chegamos a uma etapa importante!

Compreendemos os conceitos básicos da estrutura de uma página web com HTML.

Com isso, vamos adiante nesta jornada, embora tenham muitos recursos do HTML que não poderemos ver devido ao tempo disponível.

Vamos conhecer os recursos de estilização do CSS para deixar suas páginas mais atraentes?