**HTML e CSS**

**Conceitos Básicos:**

📌 Tags: São os elementos principais do HTML, que definem a estrutura da página.  
📌 Atributos: São configurações extras dentro das tags, modificando seu comportamento ou aparência.

**Estrutura básica:**

* <html>: Define o início do documento HTML.
* <head>: Contém metadados, título e links para estilos.
* <title>: Define o título da página (aparece na aba do navegador).
* <body>: Contém todo o conteúdo visível da página.
* <style> Adiciona **CSS (estilos)** diretamente dentro do código HTML.
* <link> é usada para conectar o documento HTML a **recursos externos**.

**Cabeçalhos e Texto:**

* <h1> a <h6>: Definem títulos e subtítulos (h1 é o maior, h6 o menor).
* <p>: Define um parágrafo.
* <br>: Insere uma quebra de linha.
* <hr>: Insere uma linha horizontal.
* <strong>: Deixa o texto em **negrito**.
* <em>: Deixa o texto em *itálico*.
* <mark> é usada para **destacar um trecho de texto** (Marca Texto)
* <del>: **~~Riscado~~**, indica um texto que foi removido.
* <ins>: **Sublinhado**, indica um texto adicionado.
* <sup>: Texto sobrescrito, usado para expoentes ou notas.
* <sub>: Texto subscrito, usado para fórmulas químicas ou índices.
* <code>: Exibe código de forma monoespaçada.
* <pre>: Mantém a formatação original do texto, incluindo espaços e quebras de linha.
* <q>: Indica uma citação curta e geralmente exibe aspas automaticamente.
* <blockquote>: Indica uma **citação longa**, geralmente com recuo.
* <abbr>: Define uma **abreviação**, exibindo o significado ao passar o mouse.

**Listas:**

* <ul>: Cria uma lista não ordenada (com bolinhas).
* <ol>: Cria uma lista ordenada (com números).
* <li>: Define cada item dentro de listas.
* <dt>: Define o termo (título) de uma descrição.
* <dd>: Define a descrição do termo.

**Links e Imagens e áudio:**

* <a href="URL">: Cria um link para outra página.
* <img src="imagem.jpg" alt="Descrição">: Exibe uma imagem.
* <picture>: Permite carregar **diferentes imagens** dependendo do tamanho da tela.

**Tabelas:**

* <table>: Define uma tabela.
* <tr>: Define uma linha da tabela.
* <th>: Define um cabeçalho de coluna.
* <td>: Define uma célula de dados.

**Formulários:**

* <form>: Cria um formulário.
* <input type="text">: Campo de entrada de texto.
* <input type="submit">: Botão de envio do formulário.
* <textarea>: Caixa de texto grande.
* <select> e <option>: Criam menus suspensos.

**Divisões e Organização:**

* <div>: Define uma seção genérica de conteúdo.
* <span>: Usado para estilizar partes específicas do texto.

**Tags Semânticas (HTML5):**

* <header>: Cabeçalho da página ou seção.
* <nav>: Define um menu de navegação.
* <section>: Define uma seção do conteúdo.
* <article>: Define um artigo ou post.
* <aside>: Conteúdo relacionado (ex.: barra lateral).
* <footer>: Rodapé da página.

**Integração de CSS com Html**

No HTML, existem **três formas de implementar CSS1️ CSS Inline (Dentro da Tag)**  
✅ **Usado para estilizar um único elemento**.  
❌ **Não recomendado** para projetos grandes, porque deixa o código bagunçado.

    <p style="color: red; font-size: 20px;">Texto vermelho</p>

📌 **Quando usar?** Para pequenos ajustes rápidos, como estilizar um botão específico.**2️ CSS Interno (Dentro da Tag <style> no <head>)**  
✅ **Melhor que o inline**, pois organiza os estilos dentro do HTML.  
❌ **Ainda não é ideal para projetos grandes**, porque não separa HTML e CSS.

    <head>

        <style>

            p {

                color: blue;

                font-size: 18px;

            }

        </style>

    </head>

    <body>

        <p>Esse texto é azul</p>

    </body>

📌 **Quando usar?** Quando o estilo afeta apenas uma única página.**3️ CSS Externo (Arquivo .css separado)**  
✅ **A melhor prática**, pois separa HTML e CSS, deixando o código mais organizado e reutilizável.

Arquivo: style.css

p {

    color: green;

    font-size: 22px;

}

Arquivo: index.html

  <head>

        <link rel="stylesheet" href="style.css">

    </head>

    <body>

        <p>Esse texto é verde</p>

    </body>

📌 **Quando usar?** **Sempre que possível**, pois facilita a manutenção e reaproveitamento dos estilos.

No CSS, usamos **propriedades** para estilizar elementos HTML. Aqui estão as principais **categorias** e suas **tags (propriedades CSS):**

**📌 1. Cores e Fundo**

* color → Define a cor do texto. (color: red;)
* background-color → Define a cor do fundo. (background-color: blue;)
* background-image → Define uma imagem de fundo. (background-image: url("fundo.jpg");)
* opacity → Define a transparência. (opacity: 0.5; de 0 a 1)

**📌 2. Texto e Fontes**

* font-size → Tamanho da fonte. (font-size: 16px;)
* font-family → Tipo de fonte. (font-family: Arial, sans-serif;)
* font-weight → Espessura da fonte. (font-weight: bold;)
* text-align → Alinhamento do texto. (text-align: center;)
* text-decoration → Sublinhado, riscado etc. (text-decoration: underline;)
* line-height → Espaçamento entre linhas. (line-height: 1.5;)

**📌 3. Box Model (Espaçamentos e Tamanhos)**

* width → Largura. (width: 300px;)
* height → Altura. (height: 200px;)
* margin → Espaçamento externo. (margin: 10px;)
* padding → Espaçamento interno. (padding: 15px;)
* border → Borda. (border: 2px solid black;)
* box-shadow → Sombra da caixa. (box-shadow: 5px 5px 10px gray;)

**📌 4. Posição e Layout**

* display → Controla a exibição do elemento. (display: flex;)
* position → Define a posição (absolute, relative, fixed, sticky).
* top, left, right, bottom → Ajusta a posição dos elementos.
* z-index → Define a sobreposição de camadas (z-index: 10;).

**📌 5. Flexbox (Alinhamento de Elementos)**

* display: flex; → Ativa o Flexbox.
* justify-content → Alinha no eixo horizontal (center, space-between, space-around).
* align-items → Alinha no eixo vertical (center, flex-start, flex-end).
* flex-wrap → Permite que os itens quebrem linha (wrap, nowrap).

**📌 6. Grid (Layout Responsivo Avançado)**

* display: grid; → Ativa o Grid Layout.
* grid-template-columns → Define colunas. (grid-template-columns: 1fr 1fr;)
* grid-template-rows → Define linhas. (grid-template-rows: 100px 200px;)
* gap → Define espaçamento entre os itens. (gap: 20px;)

**📌  7. Overflow — Estouro**

* **Definição**: Controla o que acontece quando o conteúdo de um elemento excede seu tamanho definido.
* **Propriedade**: overflow
* **Valores principais**:
  + visible: conteúdo excedente é exibido fora do elemento (padrão).
  + hidden: conteúdo excedente é cortado e não visível.
  + scroll: adiciona barras de rolagem sempre, mesmo sem necessidade.
  + auto: adiciona barras de rolagem **somente quando necessário**.
* **Direções específicas**:
  + overflow-x: controla o transbordamento horizontal.
  + overflow-y: controla o transbordamento vertical.

Border-radius: arredonda os cantos no elemento, podendo referenciar os 4 cantos com valores diferentes

Z-index

Responsividade

Pseudo-classes:

Pseudo-elementos:

Tipos de display:

Usada para definir um estado especial de um elemento

selector:pseudo-class {  
  property: value;  
}

* a:
  + hover: mouser sobre o link
  + visited: links já visitados
  + unvisited: links não visitados
  + active:

Unidades de medidas:

Absolutos:

* um
* mm
* in
* px
* pt
* pc

Mobile first

Comportamento de transition

Animações em css puro

Especifidade CSS

Animation

Background-color:  
bac

Midea Query

Flexbox

Ativado com Display: flex;

- Propriedades para o Container Pai

* flex-direction
  + row
  + column
  + row-reverse
  + column-reverse
* flex-wrap
  + nowrap
  + wrap
  + wrap-reverse
* flex-flow
  + Abreviação para as propriedades Direction e wrap
* justify-content
  + center
  + flex-start
  + flex-end
  + space-around
  + space-between
  + space-evenly
* align-items
  + center
  + flex-start
  + flex-end
  + stretch
  + baseline
  + normal
* align-content
  + center
  + flex-start
  + flex-end
  + stretch
  + baseline
  + normal

Propriedades para os itens do container Pai:

* order
* flex-grow
* flex-shrink
* flex-basis
* flex
* align-self

Grid CSS

Transformação

Transição

Animações

**🧱 Propriedades do container (grid)**

* display
* grid-template-columns
* grid-template-rows
* grid-template-areas
* grid-template
* grid-auto-columns
* grid-auto-rows
* grid-auto-flow
* grid
* column-gap
* row-gap
* gap
* justify-content
* align-content
* place-content
* justify-items
* align-items
* place-items

**📦 Propriedades dos itens (filhos do grid)**

* grid-column-start
* grid-column-end
* grid-row-start
* grid-row-end
* grid-column
* grid-row
* grid-area
* justify-self
* align-self
* place-self

DISPLAY:  
**📦 Valores comuns**

* block
* inline
* inline-block
* none

**🧱 Valores de layout moderno**

* flex
* inline-flex
* grid
* inline-grid

**🧩 Valores para layout de tabela**

* table
* inline-table
* table-row
* table-cell
* table-column
* table-caption
* table-row-group
* table-header-group
* table-footer-group
* table-column-group

**🧙‍♂️ Valores para conteúdo interno**

* contents
* list-item
* run-in

**🧪 Valores experimentais ou específicos**

* flow-root
* ruby
* ruby-base
* ruby-text
* ruby-base-container
* ruby-text-container
* sub
* super