**RESUMO CURSO HTML5 E CSS3**

***#DICA***

***@REGRA***

**Estrutura básica:**

***#DICA Utilizar apenas “I + ENTER”***  
<!DOCTYPE html>

<html lang="pt-br">

<head>

<meta charset="UTF-8">

<meta http-equiv="X-UA-Compatible" content="IE=edge">

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

<title>Document</title>

**Favicon:**

<link rel="shortcut icon" href="favicon.ico" type="image/x-icon>

***#DICA TRANSFORMAR IMAGENS EM FAVICON: https://favicon.io/***

***#DICA CRIAR FAVICON: https://www.favicon.cc/***

Chamar arquivo CSS EXTERNO:

<link rel="stylesheet" href="style.css"> nome de arquivo pode ser diferente

***#DICA: Para criar arquivo novo utilizar CTRL e clicar no nome do arquivo.***

</head>

<body>

<h1>Títulos</h1> ... <h6> </h6>

<p>Parágrafos</p>

</body>

**Listas:**

**Ordenadas:**

<ol> ou <ol type="1 ou A ou a ou I ou i">

<li>Lista ordenada</li> (fechamento opcional)

</ol>

Não ordenadas:

<ul> ou <ul type="circle(padrão) ou disc ou square">

<li>Lista não ordenada</li>

</ul>

**Links <a>ancora</a>:**

<p><a href="outrapagina" target="\_self ou \_blank" rel="next ou prev ou external">link<a/></p>

Downloads:

<a href=”arquivos/meulivro.pdf” download=”meulivro.pdf” type=”application/pdf”>

***#DICA: Para saber qual tipo de mídia usar:*** [***https://www.iana.org/assignments/media-types/media-types.xhtml***](https://www.iana.org/assignments/media-types/media-types.xhtml)

**Imagens:**

<img src="arquivo ou link da imagem" alt="descrição da imagem"> #DICA: usar CTRL + Espaço

Imagens dinâmicas:

<picture>

<source media="(max-width: 750px)" srcset="imagens/foto-p.png" type="image/png">

<source media="(max-width: 1050px )" srcset="imagens/foto-m.png" type="image/png">

<img src="imagens/foto-g.png" alt="Imagem flexível"> ***@REGRA "SEMPRE DO MAIOR EMBAIXO PARA O MAIOR EM CIMA"***

</picture>

**Áudio:**

<audio src="midia/happy-mistake.mp3" controls autoplay></audio>

<audio preload="metadata ou none ou auto" autoplay controls(aparece o player) loop(repetir sempre que acabar)>

<source src="midia/guanacast-33.mp3" type="audio/mpeg">

<source src="midia/guanacast-33.ogg" type="audio/ogg">

<source src="midia/guanacast-33.wav" type="audio/wav">

<p>Infelizmente seu navegador não consegue reproduzir áudio. <a href="midia/guanacast-33.mp3">Clique aqui para baixar o arquivo MP3</a></p>

</audio>

*Dentro da tag <audio>, adicionamos vários <source> com formatos diferentes do*

*mesmo áudio. Coloque na parte de cima o seu formato favorito. Os demais só serão*

*carregados caso o de cima falhe.*

**Vídeos:**

<video width="300" controls>

<source src="midia/meu-video-30p.mp4.mp4" type="video/mp4">

<source src="midia/meu-video-30p.mkv.mkv" type="video/mp4">

<source src="midia/meu-video-30p.webm.webm" type="video/webm">

<source src="midia/meu-video-30p.ogv.ogv" type="video/ogg">

<p>Seu navegador não tem compatibilidade com reprodução de vídeos.</p>

</video>

**Vídeos externos (Youtube, Vimeo, etc...):**

<iframe width="560" height="315" src="https://www.youtube.com/embed/FdE5G95B7Mg" title="YouTube video player" frameborder="0" allow="accelerometer; autoplay; clipboard-write; encrypted-media; gyroscope; picture-in-picture; web-share" allowfullscreen></iframe>

***#DICA: Copiar código do próprio YouTube/Vimeo na aba “Compartilhar”***

</body>

</html>

**CSS:**

@charset "UTF-8";

**Externo (arquivo style.css) ou interno dentro do <head> utilizar a tag <style>:**

\* { /\* Configurações globais das CSS \*/

            height: 100%;

}

body {

background-attachment: fixed ou scroll, ou local, ou inherit; (fixar o background)

main {

  box-shadow: 5px 5px 15px rgba(173, 139, 119, 0.555); (Sombra no cotorno da borda)

display: inline-block; (tudo na mesma linha)

h1 {

text-shadow: 1px 1px 2px rgba(128, 77, 6, 0.637); (Sombra no contorno do texto)

}

p {

text-align:justify; ou text-align: center; ou text-align: right; ou text-align: left

text-indent: 10px; (espaço no começo do parágrafo)

}

        a {

            text-decoration: none, ou underline, etc...;

        }

**Inline:**

<p style="text-align: right"> CSS na linha </p>

**Herança:**

1º lugar: CSS externo 2º lugar: CSS interno/local 3º lugar: CSS inline

**CORES**

****

***#DICA Existem vários sites e serviços que vão te ajudar na escolha da paleta de cores do seu site. A que vai permitir mais opções, na minha opinião é o Adobe Color (disponível em https://color.adobe.com/pt/ ou*** [***https://paletton.com***](https://paletton.com) ***ou coolors.co***

**Representação por nomes**

    <h2 style="background-color: blue; color: white;">Exemplo de Cores</h2>

**Representação por códigos Hexadecimais**: 0 1 2 3 4 5 6 7 8 9 A B C D E F

    <h2 style="background-color: #0000ff; color: #ffffff;">Exemplo de Cores</h2>

**Representação em RGB (Red, Green, Blue)**

    <h2 style="background-color: rgb(0, 0, 255); color: rgb(255, 255, 255);">Exemplo de Cores</h2>

**Representação em HSL(Hue, Saturation, Luminosity)**

    <h2 style="background-color: hsl(240, 100%, 50%); color: hsl(0, 0%, 100%);">Exemplo de Cores</h2>

**PALETA DE CORES**

***#DICA: Utilizar a cor principal do logo do cliente.  
#DICA: Usar de 3 a 5 cores, sem conta o branco e o preto.***

**CORES GRADIENTES**

Pode parecer esquisito no início, mas um gradiente é considerado pelo navegador como se fosse uma imagem, por isso usamos a propriedade background-image na declaração CSS. A função linear-gradient é auto-explicativa e gera um gradiente linear angular. O primeiro parâmetro da função, indica o ângulo de inclinação de 90 graus (*90deg*) e as seguintes indicam as cores do degradê a ser criado. Você pode indicar quantas cores quiser e o navegador vai saber se virar pra gerar seu degradê personalizado. Experimente na sua casa outros valores de ângulo também, incluindo negativos (45deg, -90deg, 25deg,…) e note as diferenças.

body {

    background-image: linear-gradient(90deg, yellow, red);

ou

 background-image: linear-gradient(to bottom ou to top, ou to right, ou to left, #81AECA, #197FBF, #12608F, #1B3647 );

Também é possível gerar os chamados gradientes radiais, que também são meio auto-explicativos. Veja o exemplo:

    background-image: radial-gradient(circle, red, yellow, green);

Você também pode personalizar ainda mais seu degradê colocando uma porcentagem ao lado da cor como red 10%, yellow 40%, green 50%. Experimente!

***#DICA: Para descobrir uma cor que gostou, utilizar a extensão do Google Chrome Colorzilla.***

**FONTES**

color: white; (cor da letra)

font-family: Arial, Helvetica, sans-serif;

font-size: 16px ;

*“ Medidas Absolutas*

*cm, mm, in, px, pt, pc*

*Medidas Relativas*

*em, ex, rem, vw, vh, % ”*

***@Regra: UTILIZAR px e em***

***#DICA: 16px geralmente = 1em***

Nem todas as fontes possuem as opções:  
font-weight: lighter, ou normal, ou bold, ou bolder ou de 100 ~ 900 ;

font-style: italic;

text-decoration: underline;

***#Shorthand font:   
font-style, font-weight, font-size, font-family. Pode utilizar apenas algumas opções, não é necessário usar as 4.***

***font: italic bolder 3em ‘Work Sans’, sans-serif;***

***#DICA: Utilizar o Google Fonts para usar fontes que não estão no computador fonts.google.com***

***#DICA: Utilizar 3 a 4 tipos de fontes no mesmo site é o suficiente.***

**Fontes personalizadas, devem ser baixadas e colocadas dentro do servidor e o código dentro de <style>:**

@font-face {

font-family: ‘Nome que desejar dar a fonte’;

src: url(‘nome da fonte.ttf ou .otf’) format();

font-weight: normal;

font-style: normal;

Formatos: opentype (otf), truetype(ttf), embedded-opentype, truetype-aat(Apple Advanced Typography, svg

***#DICA: Utilizar fontes externas baixadas:*** [***https://www.dafont.com/pt/***](https://www.dafont.com/pt/)

**SELETORES PERSONALIZADOS**

***# = id Exemplo: h1#principal {}***

***. = class . Exemplo: .principal {}***

***: = pseudo-class Exemplo: a:hover{}***

***:: = pseudo-element Exemplo:  a::after{}***

***> = children Exemplo: div > p {***

***display: none;***

***}***

***div:hover > p {***

***display:block;***

***color: white;***

***background-color: red;***

***width: 300px;***

***}***

**Criando identidade no HTML:**

<h1 id="principal">Criando Sites com HTML e CSS</h1>

**Para personalizar no arquivo CSS:**

h1#principal {

text-align: center;

}

**Criando classes no HTML:**

h2 class="basico">HTML básico</h2>

**Para personalizar no arquivo CSS:**

.basico{

    color: #3c905e;

}

Obs:. Nesses casos as tags que tem a identidade ou classe, serão formatadas de forma diferente dos outros. É possível utilizar mais de uma classe em uma tag e também envelopar com a tag <span> e usar a classe no meio do texto.

***@REGRA: id só pode ser usado em uma tag.***

***@REGRA: para utilizar a mesma configuração para duas ou mais tags, deve ser usado o class***

**Pseudo-classes “:” e pseudo-elementos “::”**

Uma pseudo-classe CSS é uma palavra-chave adicionada às declarações de um seletor após um sinal de dois pontos e especificam um estado especial de um elemento. Existem várias pseudo-classes para estilos, podemos citar :hover, :visited, :active, :checked, :empty e :focus.

**Pseudo-classes:**

display: none;

} (indica que o p é filho do div e vai esconder os parágrafos do div)

div:hover > p {

            display:block;

            color: white;

            background-color: red;

            width: 300px;

} Indica que quando passar o mouse em cima da div, o parágrafo aparecerá.

        a:visited {

            color: darkred;

        }

        a:hover {

            text-decoration: underline; (quando passar o mouse ficará sublinhado)

        }

        a:active {

            color: yellow;

        }

**Pseudo-elementos:**

  a::after{

            content: '[link]';

            text-decoration: none;

            font-weight: normal;

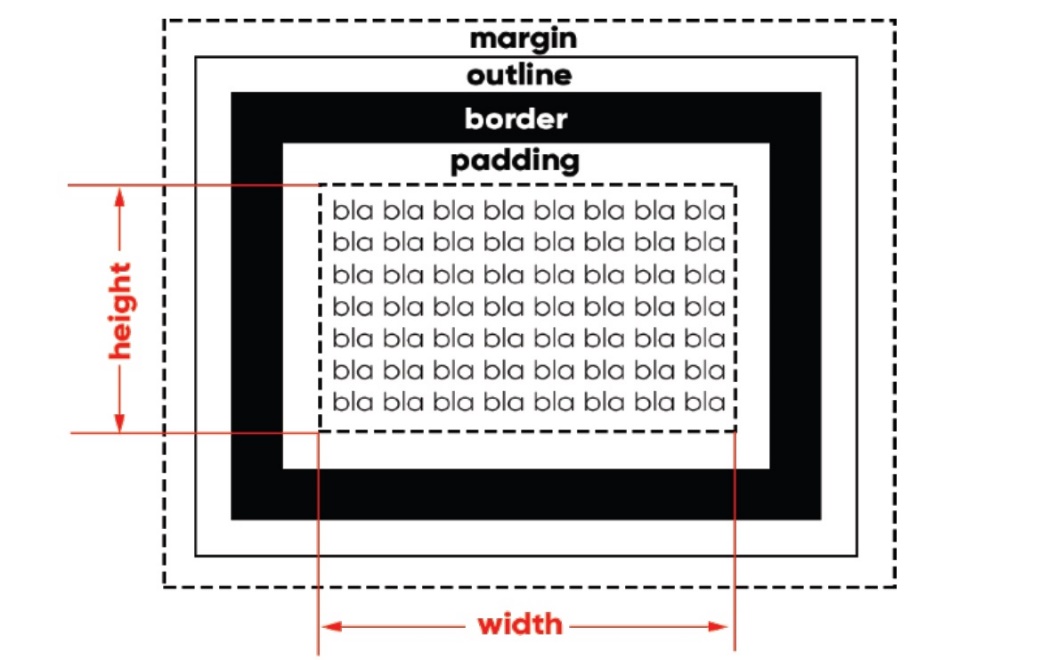
            color: red;

        } Indica que aparecerá o termo ‘[link]’ após o conteúdo.

**Modelo de Caixas**

De forma simples e objetiva, baseado em um conceito chamado “box model”, a grande maioria dos elementos HTML que temos no nosso site são como caixas. Elas são containers que armazenam conteúdos ou até mesmo outras caixas.

**Anatomia da Caixa:**

****

Exemplos:

    <style>

        h1 {

            width: 300px;

            height: 50px;

            background-color: color;

            border-width: 10px;

            border-style: solid;

            border-color: red;

  padding-top: 10px;

            padding-right: 10px;

            padding-bottom: 10px;

            padding-left: 10px;

***#shorthand: padding: 10px;***

            outline-width: 30px;

            outline-style: solid;

            outline-color: blue;

#***shorthand: outline: 5px dashed color;***

            margin-top: 10px;

margin-right: 10px;

margin-bottom: 10px;

margin-left: 10px;

margin: auto;

***#shorthand: margin: margin-top margin-right margin-bottom margin-left***

***#shorthand: margin: 10px -> Define uma margem igual em todas a direções.***

        }

    </style>

**Tipos de Caixa**

**box-level:**

**div> exemplo </div>**

O elemento HTML Div define uma divisão ou seção em um documento HTML. O elemento div é frequentemente usado como um contêiner para outros elementos, o que facilita na estilização de blocos. Também é muito útil para inclusão de interações com Scripts (JavaScript).

**HTML:**

<div>

Passe o mouse aqui

<p>TEXTO ESCONDIDO...</p>

</div>

**CSS:**

div > p { #representação de child (filho): >

}

Mais exemplos de box-level: <h1>~<h6>, <p>, <main>, <header>, <nav>, <article>, <aside>, <footer>, <form>, <vídeo>

**Inline-level:**

**<span> exemplo </span>**

Mais exemplos de inline-level: <a>, <code>, <small>, <strong>, <em>, <sup~<sub>, <label>, <button>, <input>, <select>

***#DICA Pode ser utilizado “display” para modificar as propriedades iniciais de uma caixa, é possível transformar um inline-level em box-level (display: block;), e vice-versa (display:inline;), ou ele pode ser as duas coisas ao mesmo tempo com display: inline-block;***

**Grouping Tags**

**Header**   
Cria áreas relativas a cabeçalhos. Pode ser o cabeçalho principal de um site ou até mesmo o cabeçalho de uma seção ou artigo. Normalmente inclui títulos <h1> - <h6> e subtítulos. Podem também conter menus de navegação.

**Nav**  
 Define uma área que possui os links de navegação pela estrutura de páginas que vão compor o website. Um <nav> pode estar dentro de um <header>.

**Main**  
 É um agrupador usado para delimitar o conteúdo principal do nosso site. Normalmente concentra as seções, artigos e conteúdos periféricos.

**Section**

Cria seções para sua página. Ela pode conter o conteúdo diretamente no seu corpo ou dividir os conteúdos em artigos com conteúdos específicos. Segundo a documentação oficial da W3C, “uma seção é um agrupamento temático de conteúdos, tipicamente com um cabeçalho”.

**Article**   
Um artigo é um elemento que vai conter um conteúdo que pode ser lido de forma independente e dizem respeito a um mesmo assunto. Podemos usar um <article> para delimitar um post de blog ou fórum, uma notícia, etc.

**Aside**

Delimita um conteúdo periférico e complementar ao conteúdo principal de um artigo ou seção. Normalmente um conteúdo <aside> está posicionado ao lado de um determinado texto ou até mesmo no meio dele, exatamente como fizemos no bloco de texto apresentado anteriormente, falando sobre “MÚLTIPLOS NÍVEIS”.

**Footer**

Cria um rodapé para o site inteiro, seção ou artigo. É um conteúdo que não faz parte diretamente do conteúdo nem é um conteúdo periférico (o que caracterizaria um <aside>), mas possui informações sobre autoria do conteúdo, links adicionais, mapa do site, documentos relacionados. A seguir, vou criar uma proposta de estrutura para um projeto de site. Não tome ela como a única possibilidade de criar o posicionamento de elementos de agrupamento semântico. #

**BACKGROUND**

**Colocando uma imagem de fundo no seu site**

background-image: url(' ');

body {

background-image: url('imagens/wallpaper001.jpg');

}

Com cor sólida:

background-color: skyblue;

Com gradiente: background-image: linear-gradient(to bottom, yellow, red);

Com url: background-image: url('link do local ou site');

***#DICA: Para criar 3 <div> com classe quadrado ao mesmo tempo, digitar div.quadrado\*3, dentro de <style>***

**Definindo um tamanho para imagem:**

background-size: 100px 100px;

Repetição de imagem: (a partir do canto superior esquerdo)

background-repeat: no-repeat; -> sem repetição (apenas 1 imagem)

repeat-x; -> repetição na horizontal

repeat-y; -> repetição na vertical

A imagem se repetirá automaticamente, em seu tamanho padrão;

**Posição da imagem**

background-position: posição horizontal posição vertical

background-position: left top; -> esquerda topo

background-position: left center; -> esquerda centro

background-position: right bottom; -> direita embaixo

background-position: center center; -> centro centro

Obs: vh -> altura da view port (height)

vw -> largura da view port (width)

**Mudando o tamanho da imagem de fundo do site**

background-size: 100% 100%; -> preenche a tela, mas achata e distorce a imagem

countain; -> a imagem é 100% exibida, com laterais vazias as vezes

cover; -> cobre a tela toda mesmo que tenha que cortar a imagem

**background attachment :**

background-attachment: scroll; -> padrão; o fundo rola junto com o conteúdo

fixed; -> fundo fixo à tela, a medida que o conteúdo rola (ótimo combinado com background-size: cover;)

***#Shorthand – background:***

***color > image > repeat > attachment > position > / size > origin > clip***

***Exemplo: background: white url('imagens;wallpaper002.jpg') no-repeat fixed center center / cover;***

***}***

**Centralização vertical de caixas:**

**Na parte de HTML:**

<body>

<section id="container">

<article id="conteúdo">

</article>

</section>

</body>

**Na parte de CSS:**

- O que tá e fora (container) -> position: relative

Ex:

[#container](https://www.youtube.com/hashtag/container) {

position: relative; -> opcional; em CSS já é relative por padrão

height: 500px;

padding: 10px;

background-color: purple;

}

O que tá de dentro (conteúdo) -> position: absolute

Ex:

[#conteúdo](https://www.youtube.com/hashtag/conte%C3%BAdo) {

position: absolute; -> permite usar as propriedades top e left

height: 200px;

width: 400px;

background-color: yellow;

left: 50%; -> o canto top left que ficará centralizado (precisa de transform)

top: 50%;

transform: translate(-50%, -50%); -> translate(mov horizontal, mov vertical).

-> Muda a posição de referência de 'topo/esquerdo' (padrão)

, para uma outra qualquer (ex: centro)

}

***#DICA forma mais fácil de centralizar:***

[***#container***](https://www.youtube.com/hashtag/container) ***{***

***display: flex; -> (possui diversas funções, uma delas é habilitar o manejo do conteúdo) justify-content: center; ->(alinhamento horizontal, dispensando o 'margin: auto')***

***align-items: center; (alinhamento vertical)***

***} -> Essas propriedades são colocadas no elemento-pai (container, nesse exemplo).***

**TABELAS**

**Hierarquia de tabelas simples:**

table (tabela)

table row (tr - linha)

table header (th - cabeçalho)

table data (td - dado)

Ex: criar uma tabela com 4 linhas e 3 dados:

<table> <tr> <td>A1</td> <td>B1</td> <td>C1</td> </tr> <tr> <td>A2</td> <td>B2</td> <td>C2</td> </tr> <tr> <td>A3</td> <td>B3</td> <td>C3</td> </tr> <tr> <td>A4</td> <td>B4</td> <td>C4</td> </tr> </table>

- De forma mais prática podemos escrever: table>tr\*4>td\*3 + Enter

Estilo:

<style>

body {

font-family: Arial, Helvetica, sans-serif;

}

table {

width: 400px;

border-collapse: collapse; -> junta as bordas das células

}

tr.linha {

background-color: lightgray;

}

td {

border: 1px solid black;

padding: 8px;

}

td.dado {

background-color: yellow;

}

</style>

Observações: - W3C: os fechamentos das tags <tr> e <td> não são obrigatórios, porém recomendados; - Se a hierarquia não for respeitada, os elementos vão aparecer antes ou depois da tabela.