HTML e CSS para Iniciantes

# Html, Css e JavaScript para iniciantes

**Linguagens**

HTML – HyperText Markup Language => linguagem de marcação de hipertexto. Arquivo de texto simples em .html.

CSS – Cascading Style Sheets => linguagem de estilo (cor do texto, tamanho do texto, tamanho da imagem, posição do conteúdo, cor do fundo, sombras e mais. Arquito de texto simples terminado em .css. => <style> </style>

JS – linguagem de script/programação. Utilizado para modificar o estilo, criar galerias de imagens, observar as ações do usuário, puxar informações de outros sites, manipular dados e mais. Arquivo de texto simples termiando em .js. => <script> </script>

**Editores de código**

Visual Studio Code => Microsoft, gratuito.

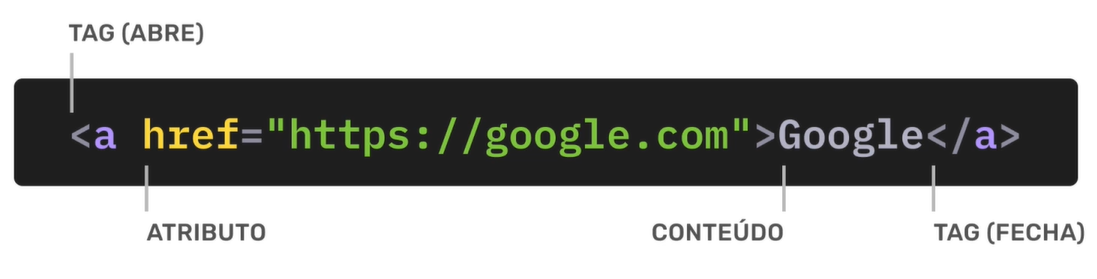
**Browsers**

Responsável por executar e interpretar o HTML, CSS e JavaScript. Existem diferentes browsers (Chrome, Firefox, Safari, Edge etc) e diferentes positivos (Windows, Mac, Desktop, Mobile, SmarTV etc). Obs.: Google Chrome tem uma boa ferramenta para desenvolvedores.

# Html e Css Básico

**HTML**

Documento html é marcado por tags diferentes.



É case insensitive.

Tags servem para inserirmos conteúdos e dá informações ao site sobre o conteúdo.

**Tags mais comuns**

<p> – parágrafo

<h1 a h8> – títulos em diferentes níveis

<a> - marca o link

**Estrutura**

Sempre começa com <! DOCTYPE html>

Tag raiz:

<html>

.....

</html>

Tags essenciais:

<head> </head>

<body></body> => coloca todo o conteúdo

Dentro do Head

<meta> => só abre e dentro dela coloca atributos com valores

* Charset => como os caracteres serão interpretados (essencial para acentos)

<title> => título do site

**Atalhos básicos no VSCode**

* CtrlC e CtrlV
* CtrlZ – voltar
* Ctrl + Shift + Z – volta o que foi desfeito
* Alt e seta – andar documento
* Quando seleciona palavras iguais: aperta ctrl + D e muda tudo.
* Comentários em HTML: <!-- adfadfadf --> => Mais simples: CTRL e barra
* Ctrl = B => tira a barra

Ver todos em Ctrl + shift + p e selecionar keyboard shortcuts

**Links Caminhos**

Caminho absoluto: links externos

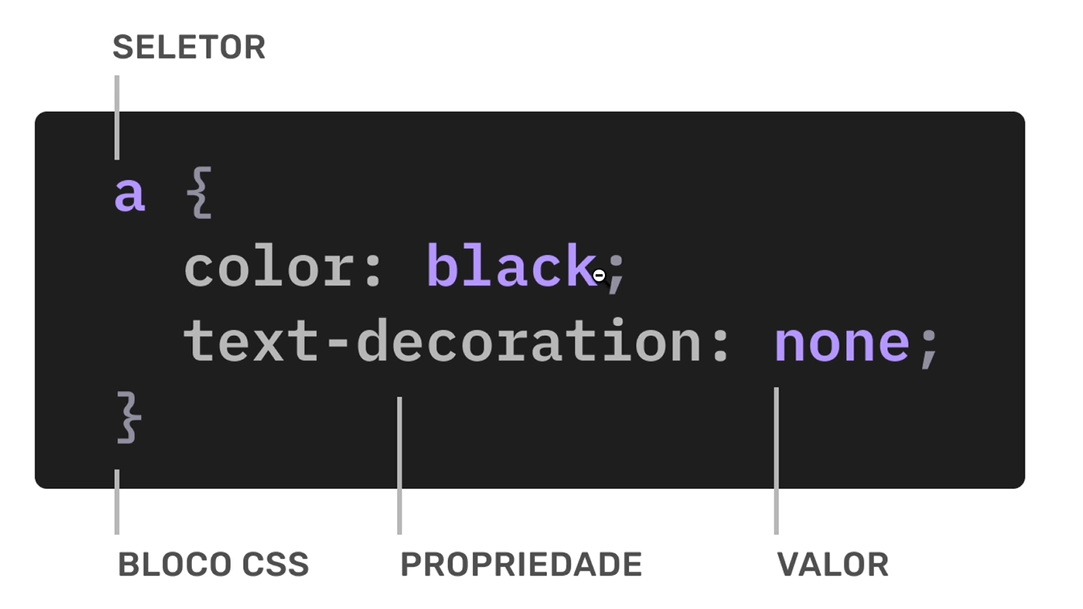
***Caminho relativo: uso arquivos internos do site***

* /index.html => vai no diretório raiz /
* /produtos/bicicletas.html => vai no diretor produtos que está na raiz
* index.html ou ./index.html => sem a barra ou com ./ => vai no arquivo no diretório atual
* ../index.html => procurar o arquivo index.html no diretório anterior ao atual
* ../../index.html => procura o arquivo index.html em dois diretórios anteriores ao atual.

**CSS**

Usar arquivo externo (.css) e linkar com a tag link.

<link rel=”stylesheet” href=”/style.css” />



Seletor => qual element quero selecionar

{ } => propriedades para o seletor

Seletores:

h1, p, a => a vírgula permite selecionar múltiplos elementos

ID e Classe

id => id=”logo” => coloca uma id no elemento

no CSS usa **#logo**

class =>

no CSS usa **.logo**

A diferença entre id e classe é que ela pode ser utilizada para diversos elementos simultaneamente, enquanto id é somente para um elemento.

Mudança de fundo de um elemento: background ou background-color

Forma mais comum de informar cor é pelo hexadecima ou RGB

Para mudar cor da fonte é somente color: ....;

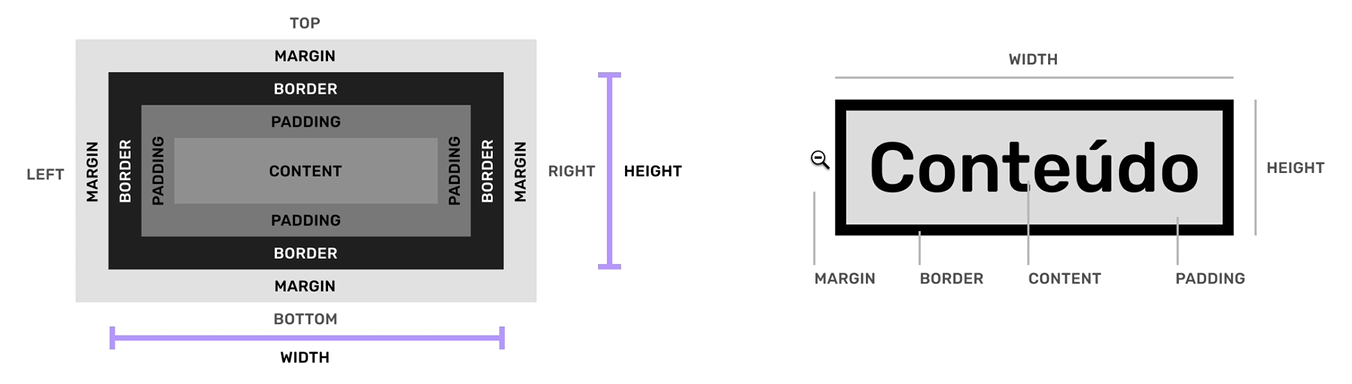
RGB é função (0, 0, 0)

RGBA dá uma função com quatro dígitos em que o último número é a transparência da cor (vai de 0 a 1).

Hexadecimal é caracterizado por números de 0 a 6 e letras de A até F.

**Box model**

Como caixas de comportam na WEB. Necessário diferenciar as caixas para organizar o conteúdo. Modelo de box model define o tamanho das caixas, o tamanho, a largura.

Conteúdo pode ser textua ou imagem ou qualquer outro elemento

Após o conteúdo tem o padding, a borda, a margin.

A soma do conteúdo + padding + border = width e height

Depois disso tem a margin, que separa os elementos.

Principal unidade da web é pixel, pois os monitores são configurados em pixels.

Monitores típicos: 3840 px de largura por 2160 px de altura.

No CSS o pixel é uma unidade de referência e não representa 1 pixel extado do dispositivo.

**TAG div**

Tag genérica, serve para auxiliar no posicionamento dos elementos na tela.

Estilos do browser:

- todo browser utiliza estilos previamente definidos. Por isso é bom definir todas as medidas de todos os elementos.

- técnicas de Reset (remove os estilos iniciais para manter maior consistência entre os Browsers – ver CSS reset tools – deixa tudo no zero), Reboot e normalize.css (estilos padrões é legal ter, porém mantém consistência entre os browsers),

**Conceito de herança**

Algumas propriedades do CSS são passadas do pai para o filho como uma herança

Ex.: color: inherit (para herdar do elemento pai e tirar configuração própria do CSS).

**Fluxo de layout**

Inline – respeita o fluxo da escrita sem iniciar uma nova linha, não sendo possível definir valores de widht, height, margin etc. É o estilo padrão.

Block – inicia uma nova linha e não permite que outros elementos estejam posicionado sem sua linha. Aceita todas as propriedades do box model. É o estilo inicial de elementos como h1, p, div e outros.

Display: none => remove o element completamente da tela.

inline-block: o elemento continua inline, mas passa a receber as propriedades do box model.

Display grid

Referências boas para documentações:

Mozilla

https://developer.mozilla.org/en-US/docs/Web/HTML/Element

https://developer.mozilla.org/en-US/docs/Web/CSS/Reference

W3C

<https://www.w3.org/TR/>

**IMAGENS**

Tag <img> tem dois atributos

Src = caminho da imagem

Alt = descrição da imagem

Em monitores de alta densidade, as imagens borram, então é preciso deixar somente o tamanho igual à metade da resolução dela, no máximo.

Svg = não é imagem, é código. É fácil, por exemplo, mudar a cor mudando o código. É ótima para não desfocar, pois o tamanhooo é definido pelo programador. SEMPRE PEGAR SVG, pois é melhor.

Png = é melhor, pois não tem fundo.

Jpg – imagens complexas, fotos.

Site interessante para pegar fotos gratuitamente:

Unsplash: <https://unsplash.com/pt-br>

Squoosh: ótimo para processar imagens

<https://squoosh.app/>

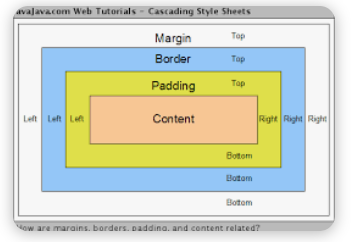
# CSS Posicionamento

margin: 30px 100px 60px 200px => top, right, bottom, left

margin: 30px 100px => top e bottom, right e left

margin: 30px => todas

ou define individualmente todos eles.



**display: grid**

define linhas e colunas na seção.

.grid {

display: grid;

grid-template-columns: 1fr 1fr;

gap: 20px;

}

Guia Origamid:

https://www.origamid.com/projetos/grid/

https://www.origamid.com/projetos/css-grid-layout-guia-completo/

Outros guias:

https://grid.malven.co/

<https://css-tricks.com/snippets/css/complete-guide-grid/>

**Flexbox**

display: flex

Mais uma forma de posicionar elementos da tela. Filhos passam a ter um tamanho flexível e ficam um ao lado do outro.

Flex-wrap: wrap => se elementos não cabem em uma mesma linha, quebra-se para a próxima.

Gap: distância entre elementos

É possível utilizar junto com grid.

Flex:

Flex-grow

Flex-basis

Flex-shrink

Flex:

Position: influencia na ordem dos conteúdos, para tirar do fluxo normal.

# Html e Semântica

adfadf

# CSS Propriedades

adfadf

# Responsivo

asdfads

# Projeto Portfólio

adsfadf

# Ferramentas

adsfadf

# Mais Html e CSS

asdfadf

# Projeto final

asdfadf

# JavaScript Básico

adsfadf

# JavaScript Projeto

asdfadsf

# Produção

adsfadf